FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

FEUP

Operação de *Pick-and-Place* Adaptativa em Ambientes Pouco Estruturados

Paulo Jorge Moreira da Costa

Mestrado Integrado em Engenharia Electrotécnica e de Computadores

Orientador: Paulo José Cerqueira Gomes da Costa (Prof. Doutor)

31 de Julho de 2012

Resumo

Esta dissertação enquadra-se num projeto da PRODUTECH, com o objetivo de satisfazer uma necessidade da empresa Adira S.A.. As máquinas de corte de metal por meio de um laser são um dos produtos desta empresa. Neste projeto é apresentada uma solução para um problema recorrente neste tipo de máquinas: as peças recortadas ficam inclinadas ou caem ficando inacessíveis ao robô que tem como tarefa retirar as peças com o corte finalizado.

De modo a dotar o sistema de flexibilidade na tomada de decisões é necessário equipá-lo com um sistema de visão adequado ao problema. Desta forma, optou-se por um equipamento de visão tridimensional técnica e economicamente acessível: Microsoft Kinect. O dispositivo é estudado com detalhe com o objetivo de compreender o seu modo de funcionamento e sensibilidade a mudanças no meio.

Um sistema baseado em visão computacional, necessita normalmente de uma aplicação base para realizar as operações típicas de calibração. Foi, por isso, desenvolvida uma aplicação recorrendo à plataforma de desenvolvimento Lazarus que oferece a possibilidade de calibrar o Kinect e sobrepor a imagem colorida e de profundidade. É também implementado um método de calibração do referencial mundo necessário à cooperação com um manipulador robótico.

O processamento de dados, resultantes do sistema de visão, é realizado recorrendo a algoritmos de processamento de imagem. As peças recortadas são identificadas de modo a inferir sobre se estas podem ou não ser recolhidas pelo robô. No primeiro caso, calcula-se a inclinação do plano que aproxima a peça identificada, que serve para que robô efetue a pega com o ângulo devido. No segundo caso, o robô não deve realizar a pega, dado o insucesso garantido à partida.

Abstract

This work is part of a project from PRODUTECH in order to meet a business need of the Adira S.A. enterprise. The laser metal cutting machines is one of the products of this company. In this project is presented a solution to a recurring problem in this type of machines: the cut pieces usually get misaligned or fall, becoming inaccessible to the robot that has the task of picking them.

In order to give the system decision-making flexibility is necessary to equip it with a vision system suitable for the problem. The choice was a three-dimensional vision equipment technically efficient and economically affordable: Microsoft Kinect. The device is studied in detail to understand their pros and cons, in order to use in its best operating range.

A system based on computer vision, normally requires an application to perform basic operations like calibration. Therefore, this was carried out using the Lazarus application development platform that offers, besides other functionalities, the possibility to calibrate Kinect and join the depth and color images. It was also implemented a calibration method to the world reference frame to cooperate with a robotic arm.

The data returned from the vision system is computed using image processing technics. The algorithm identifies the parts that correspond to cut pieces. The cut pieces are identified in order to infer whether or not they can be collected by the robot. In the positive scenario, the algorithm performs plane fitting in order to tell the robot the angle of attack to successful pick the piece. In the negative scenario, the robot should not even try to pick the cut piece.

Agradecimentos

Em primeiro lugar, quero agradecer ao meus pais. Sem o seu apoio, sacrifício e amor tudo isto não seria sequer possível.

Quero também deixar um agradecimento especial aos meus avós maternos por todo o amor e simpatia distribuídos naqueles almoços tão agradáveis ao fim de semana.

À minha irmã Célia por me dar sempre um sorriso em alturas mais complicadas, e estar sempre disposta a ajudar à sua maneira.

Como é evidente, deixo um especial agradecimento aos meus amigos de faculdade. Sem eles seriam insuportáveis muitos dos momentos complicados vividos no decorrer deste curso. São muitos e bons futuros engenheiros, e por isso não os posso citar a todos. Mas vocês sabem...

Aos colegas de laboratório: Germano, Luís e Marcos por me receberem de braços abertos, estarem sempre dispostos a ajudar e tornarem todo o trabalho muito mais alegre.

Ao meu orientador Paulo José Cerqueira Gomes da Costa pela sua disponibilidade em ajudar naqueles momentos em que o material parece não ter razão.

Por fim, à minha namorada Catarina, que me acompanhou em todos os cinco anos do curso e que esteve sempre, e incondicionalmente, do meu lado.

Um obrigado a todos!

Paulo Jorge Moreira da Costa

"A fábrica do futuro só terá dois empregados: um homem e um cão.
O homem estará lá para alimentar o cão.
O cão estará lá para impedir o homem de tocar no equipamento."

Warren G. Bennis

Conteúdo

Lista de Figuras

Lista de Tabelas

Abreviaturas e Símbolos

2D	Duas dimensões
3D	Três dimensões
CCD	Charge-coupled device
CMOS	Complementary metal-oxide semiconductor
IR	Infrared
IBVS	Image based visual servoing
LIDAR	Light detection and ranging
OpenCV	Open source computer vision
PBVS	Position based visual servoing
RGB	Red green blue
RGB-D	Red green blue - Depth
RMSD	Root mean square deviation
SDK	Software development kit
SVD	Singular value decomposition
TOF	Time-of-flight

Capítulo 1

Introdução

Neste capítulo é exposta a motivação inerente a este projeto, e ainda a estrutura do documento.

1.1 Motivação

A constante evolução tecnológica permite que os robôs estejam cada vez mais preparados para interagir adaptativamente com o mundo que os rodeia. Um estudo detalhado desta sensorização é, por isso, fundamental para aumentar as funcionalidades dos robôs.

No contexto desta dissertação é dada uma especial atenção ao domínio industrial. As exigências são elevadas neste ramo dominado pela forte concorrência e na constante necessidade de redução de custos, pelo que os robôs podem desempenhar um papel fundamental na obtenção destes objetivos.

A precisão e flexibilidade na execução das tarefas por parte de um robô está diretamente associado à precisão dos seus sensores e ao uso devido das suas funcionalidades. Para além disso, também existe um compromisso na escolha dos diferentes produtos que o mercado tem para oferecer. Desta forma, revela-se fundamental perceber as vantagens e desvantagens de cada um desses produtos, de modo a que a atuação, juntamente com a sensorização, tenha um desempenho tão elevado quanto possível.

Ao aumentar as funcionalidades de um robô permite-se uma menor organização do mundo que o rodeia. Caso contrário, passariam a existir elevados custos temporais e monetários, que a médio e longo prazo podem ser cruciais. Ao deixar que um robô se adapte a um meio pouco estruturado, este torna-se mais independente e multifacetado na realização das suas tarefas, diminuindo - ou até mesmo suprindo - os custos anteriormente mencionados.

Para além de tudo isto, a crescente substituição do 2D pelo 3D na visão computacional tem vindo a tornar os componentes de visão tridimensionais economicamente mais acessíveis. Este salto tecnológico, conseguido com uma mais fácil inclusão da noção de profundidade, abre portas a implementações mais robustas, fiáveis e precisas de algoritmos de perceção.

1.2 Estrutura do Documento

Para além da introdução, este documento contém mais 5 capítulos.

No capitulo 2 é feita uma contextualização e descrição do problema. São também apresentados os objetivos a cumprir, bem como o ambiente de trabalho onde foram realizados os testes.

A revisão bibliográfica é apresentada no capítulo 3, recorrendo à apresentação de tecnologias de visão computacional semelhantes. Aqui, os diferentes tipos de tecnologias são subdivididos em função da sua principal caraterística.

A apresentação teórica base do sistema de visão implementado encontra-se no capítulo 4. São também dados a conhecer os resultados dos testes de performance efetuados e os resultados do mapeamento 3D obtido com recurso ao Microsoft Kinect.

No capítulo 5 são relatadas as principais funcionalidades da aplicação desenvolvida em Lazarus. O algoritmo responsável pelo processamento de imagem implementado em Matlab é também aqui apresentado.

Por fim, a conclusão que inclui comentários ao trabalho desenvolvido, melhorias e trabalho futuro proposto encontra-se no capítulo 6.

Capítulo 2

Problema

Neste capítulo é feita a contextualização do problema, apresentada a descrição e, por fim, explicados os objetivos que se pretendem obter com a concretização do projeto.

2.1 Contexto

Esta dissertação insere-se num projeto do PRODUTECH - Pólo das Tecnologias de Produção. Esta entidade é uma rede articulada de fornecedores de tecnologias de produção capazes de responder aos desafios e aos requisitos de competitividade e sustentabilidade da indústria transformadora. É com soluções inovadoras, flexíveis, integradas e competitivas que a empresa Adira S.A. se associa a esta rede. O projeto abordado nesta dissertação enquadra-se numa necessidade especificada pela empresa. [1]

2.2 Descrição

As máquinas de corte laser são um dos produtos da empresa Adira S.A.. Estas máquinas cortam chapas de diferentes espessuras com elevada precisão, fiabilidade e performance.

Os cortes são feitos por um robô cartesiano, munido de um laser. A chapa assenta numa estrutura de suporte constituída por uma matriz de triângulos, figura 2.1. Assim, de modo a facilitar o corte desta superfície, cria-se um vazio entre ela e o fundo do suporte - deixando-a apoiada nos vértices superiores do triângulo. No entanto, este sistema de apoio está na origem do problema: após o corte de uma peça, esta pode cair ou ficar inclinada, impossibilitando ou dificultando, respetivamente, a remoção da mesma, figura 2.2. Este problema ocorre de uma forma difícil de prever à priori, porque depende apenas do número de apoios que a peça tem no momento do corte.

Quando a tarefa de remoção das peças recortadas está atribuída a um robô, é necessário desenvolver um sistema de perceção que lhe permita tomar decisões acertadas autonomamente. Com isto, pretende-se minimizar as falhas na execução daquela tarefa por parte do robô, impedindo que este ataque peças que estão caídas. Pretende-se também possibilitar ao robô a pega de peças inclinadas com a devida orientação.

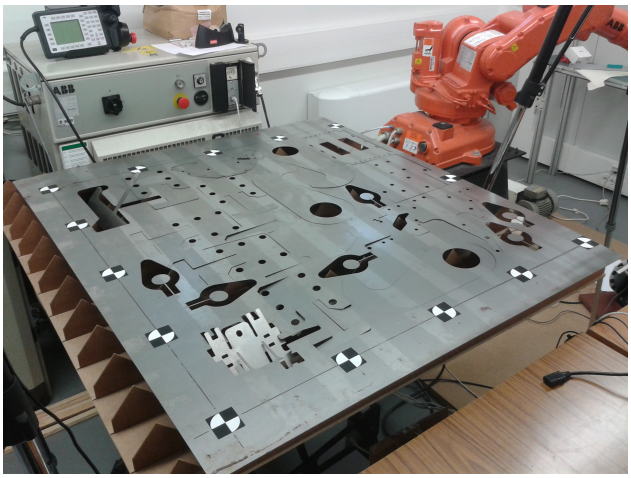
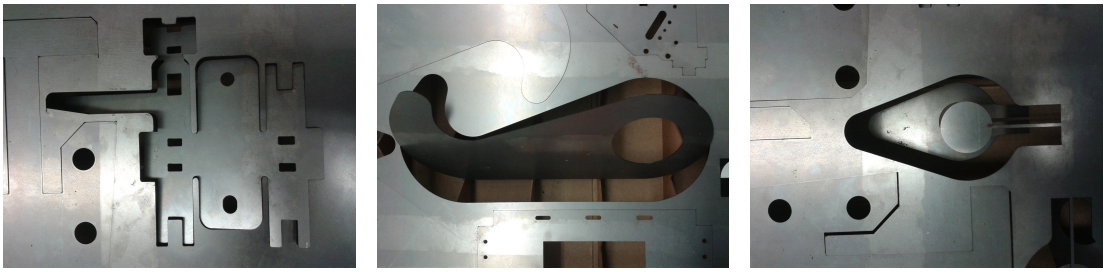


Figura 2.1: Chapa fornecida pela empresa Adira, sobre o suporte de madeira

Resumindo, o problema centra-se na perceção que o robô tem da sua área de trabalho, impedindo que este realize tarefas que à partida são impossíveis de serem concretizadas.



(a) Peça ligeiramente inclinada (b) Peça muito inclinada (c) Peça caída

Figura 2.2: Ocorrências possíveis após o corte da peça

2.3 Objetivos

A resolução do problema não passa por seguir uma arquitetura específica e, por isso, a composição e disposição do hardware é um objetivo na resolução do problema enunciado. Pretende-se ainda desenvolver uma aplicação multiplataforma para tratamento de dados de visão, responsáveis pela tomada de decisão.

Assim sendo, torna-se fundamental criar uma arquitetura passível de ser inserida num ambiente industrial quer do ponto de vista de *hardware*, quer de *software*. Desta forma, são escolhidas tecnologias de visão tridimensional que, por meio de processamento de imagem, funcionarão em cooperação com um robô. A aquisição e processamento de dados permitirá a classificação das diferentes peças em função da possibilidade de serem retiradas pelo robô. A aplicação deverá possibilitar os diferentes tipos de calibração necessários a um funcionamento robusto, mesmo

com possíveis alterações do meio. Pretende-se também possibilitar a visualização dos dados tridimensionais de forma apelativa e de fácil compreensão. Na realidade, a aplicação terá uma base funcional passível de ser usada em múltiplos projetos que recorram a processamento de imagem 3D.

É necessário comparar diferentes soluções tecnológicas, de modo a que a entidade interessada aceite o projeto como viável, não só em termos de engenharia mas também economicamente. Por isso, neste projeto serão comparadas duas tecnologias de visão de funcionamento e caraterísticas semelhantes, mas de preços completamente distintos.

2.4 Testes em Laboratório

De modo a simular o problema em ambiente laboratorial foi construído um suporte para a chapa, com a matriz de triângulos, em madeira com forma e tamanhos equivalentes ao existente na empresa Adira, figura 2.3. Foram fornecidas, pela empresa, chapas com peças recortadas para servirem como teste do algoritmo. A chapa carateriza-se por ser fina ($1mm$) e por ter uma grande densidade de peças numa área de, aproximadamente, $1m^2$.

O hardware escolhido para visualização foi o Kinect. Este foi colocado numa barra de madeira apoiada em tripés a uma altura de $1,30m$ da chapa, encarando-a ligeiramente na diagonal. A justificação desta arquitetura é explicada no capítulo 4.

Figura 2.3: Chapa fornecida pela empresa Adira

Capítulo 3

Revisão Bibliográfica

Nesta capítulo é apresentado o estado da arte das tecnologias de visão computacional. Dada a natureza do projeto, as referências bibliográfica são maioritariamente direcionadas para visão tridimensional, devido ao facto de ser este o tipo de tecnologia a ser usado para a concretização prática deste projeto.

Uma das mais importantes tarefas de um sistema autónomo consiste na extração de informação do meio envolvente. Isto só é possível fazendo medições usando os diversos sensores que equipam o robô. Posteriormente, é preciso extrair e tratar a informação relevante de modo ao robô entender o meio que o rodeia - perceção.

Os sensores que equipam o robô podem ser de diversos tipos, podendo estes serem classificados de diferentes formas [2]. No que diz respeito à sua função, estes podem ser:

- **Propriocetivos** - Medem valores internos ao robô (velocidade, bateria, ângulos das juntas);
- **Exteriocetivos** - Adquirem informação do meio envolvente do robô e os dados têm de ser interpretados de modo a serem percebidos pelo robô (distâncias, intensidade luminosa).

De acordo com a explicação anterior, e do que são os objetivos desta dissertação, serão abordados apenas os sensores extereocetivos.

Convém também mencionar outra componente importante na classificação dos sensores e que é também levada em conta neste relatório. Os sensores são também classificados pelo modo como interagem com o meio. Desta forma, a classificação anteriormente mencionada pode ainda ser subdividida em:

- **Passivos** - Medem diretamente a energia do meio que é adquirida pelo sensor. Exemplos deste tipo de sensores são os termómetros, microfones e câmaras CMOS.
- **Ativos** - Emitem energia no meio e medem a sua correspondente reação. Exemplos deste tipo de sensores são os sensores ultra-sónicos (medem a reflexão das ondas sonoras) e *laser rangefinders* (medem reflexão dos feixes luminosos).

No que diz respeito a esta última classificação, são apresentados neste capítulo sensores de ambos os tipos.

3.1 Mapeamento Ativo Baseado em Triangulação

Este tipo de sensores usam propriedades geométricas nas suas medições de modo a inferir sobre a distância aos objetos. No seu modo mais simples, os sensores baseados em triangulação são ativos, porque estes emitem um feixe de luz (ponto, linha, textura...) no ambiente que se pretende visualizar. A reflexão desse feixe é capturado por um recetor que, em função de variáveis do ambiente conhecidas, permite tirar conclusões acerca da distância do objeto. [2]

3.1.1 Luz Estruturada

O sistema de visão por luz estruturada é largamente utilizado na indústria pela sua simplicidade e robustez na deteção de defeitos. Como pode ser visto na figura 3.1, uma configuração exemplo deste tipo de tecnologia é constituída por uma câmara CCD ou CMOS e um emissor, que projeta no ambiente um feixe de luz de padrão e estrutura conhecidos (luz estruturada). Neste caso particular, o recetor mede a posição da reflexão ao longo de dois eixos ortogonais, podendo ser assim classificado como sensor 2D.

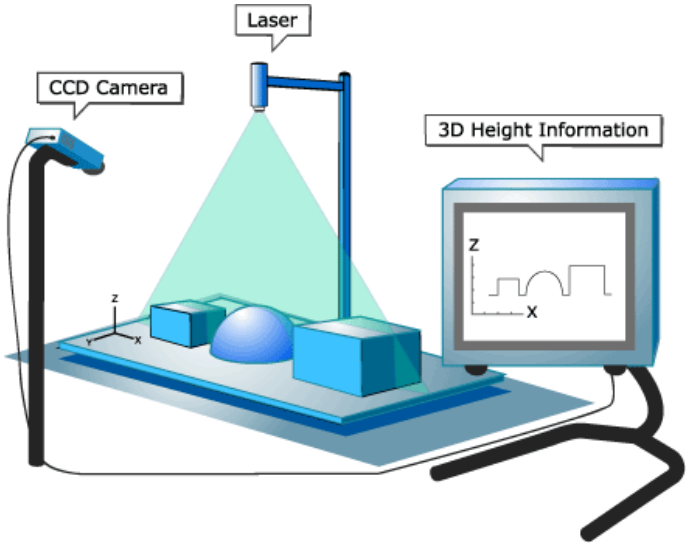


Figura 3.1: Configuração exemplo de um sistema de luz estruturada

3.1.2 Tecnologias da PrimeSense

A PrimeSense [3] é uma empresa israelita, fundada em 2005, que ficou conhecida com a implementação da tecnologia que equipa o Microsft Kinect [4]. Atualmente esta empresa é responsável pela criação de diversos equipamentos, que funcionam com base num mesmo princípio, figura 3.3.

Este tipo de sensores têm vindo a tornar-se importante na comunidade científica, nomeadamente no domínio de imagem tridimensional. As suas caraterísticas rapidamente atraíram a atenção dos investigadores para a implementação de sistemas de mapeamento e modelação 3D. Este

Figura 3.2: Kinect com o hardware assinalado

tipo de sensor é uma alternativa atrativa aos *laser rangefinders* para aplicações na área de *indoor mapping*, vigilância, robótica e mesmo indústria forense. [5]

Atualmente existem no mercado quatro produtos deste tipo no mercado, figura 3.3:

1. **PSDK reference** - Primeiro dispositivo criado por esta empresa;
2. **Microsoft Kinect** - Surgiu como acessório de jogos para a consola Microsoft Xbox 360;
3. **Asus Xtion Pro** - Em tudo idêntico ao Kinect, mas para o uso em computadores;
4. **Asus Xtion** - Igual ao Asus Xtion Pro, mas sem câmara colorida RGB.

(a) PSDK reference (b) Microsoft Kinect

(c) Asus Xtion Live (d) Asus Xtion

Figura 3.3: Dispositivos de mapeamento 3D desenvolvidos pela PrimeSense

Apesar de serem todos eles baseados no mesmo princípio de funcionamento, existem alguns detalhes que os distinguem [6]. Na tabela 3.1 é feita a comparação entre os dispositivos mais idênticos e também mais acessíveis no mercado nacional: Microsoft Kinect e Asus Xtion Live [7]. Os outros dois aqui apresentados são descartados à partida pelas seguintes razões: o PSDK reference não está atualmente disponível no mercado pois foi posteriormente substituído pelos outros modelos apresentados; o Asus Xtion tem a limitação de não ter câmara colorida o que limita as capacidades de visão computacional.

Para este projeto foi utilizado o Microsoft Kinect por ser aquele mais acessível no mercado dado a sua associação ao mundo dos videojogos. Na realidade, para as exigências deste projeto, as diferenças existentes entre ambos os dispositivos comparados na tabela 3.1 não são suficientes para se optar por um, em detrimento de outro. De salientar que o preço de ambos é muito semelhante, rondando os 150 euros.

Dispositivo	Vantagens	Desvantagens
Kinect	•Boa qualidade dos drivers de fábrica •Estável com vários modelos de *hardware* •Motor de inclinação controlado remotamente •Mais acessível no mercado (mais popular)	•Menos compacto e maior •Mais pesado •Necessita de alimentação externa •Atinge no máximo 30fps
Xtion Live	•Mais compacto e pequeno •Mais leve •Alimentação por USB •Consegue 60fps a uma resolução de 320×240	•Menos acessível no mercado (menos popular) •Menor qualidade dos drivers de fábrica •Não funciona com alguns controladores USB •Sem motor de inclinação

Tabela 3.1: Tabela comparativa do Microsoft Kinect e Asus Xtion Live

Considerando tudo isto, são aqui apresentadas as caraterísticas base do dispositivo usado no desenvolvimento deste projeto: Micrososft Kinect. Ficando a explicação detalhada do seu modo de funcionamento reservada para o capítulo 4.

Este dispositivo de mapeamento 3D é constituído por um projetor infravermelhos, uma câmara de infravermelhos, uma câmara RGB, dois microfones e um motor que permite variar a inclinação do mesmo sobre o eixo horizontal. O projetor e a câmara de infravermelhos em conjunto servem para triangular pontos no espaço, em que o resultando é uma nuvem de pontos num espaço 3D. A câmara RGB pode ser usada para dar informação acerca da cor e textura. Estes componentes, quando calibrados, resultam numa nuvem de pontos colorida.

Outra forma de analisar o Kinect consiste em avaliar as suas funcionalidade como sensor de medida, em que este retorna três medições: imagem infravermelhos, imagem RGB e imagem de profundidade. [9]

- **Imagem infravermelhos** - Resolução de 640×480 pixeis, 11 *bits* de sensibilidade, com um ângulo de visão de $57^\circ \times 45^\circ$. A câmara de infravermelhos é utilizada para observar e descodificar a projeção do padrão de infravermelhos para triangular o meio 3D;
- **Imagem RGB** - Resolução de 640×480 pixeis, 8 *bits* de resolução VGA, com um ângulo de visão de $63^\circ \times 50^\circ$. A câmara necessita de calibração;

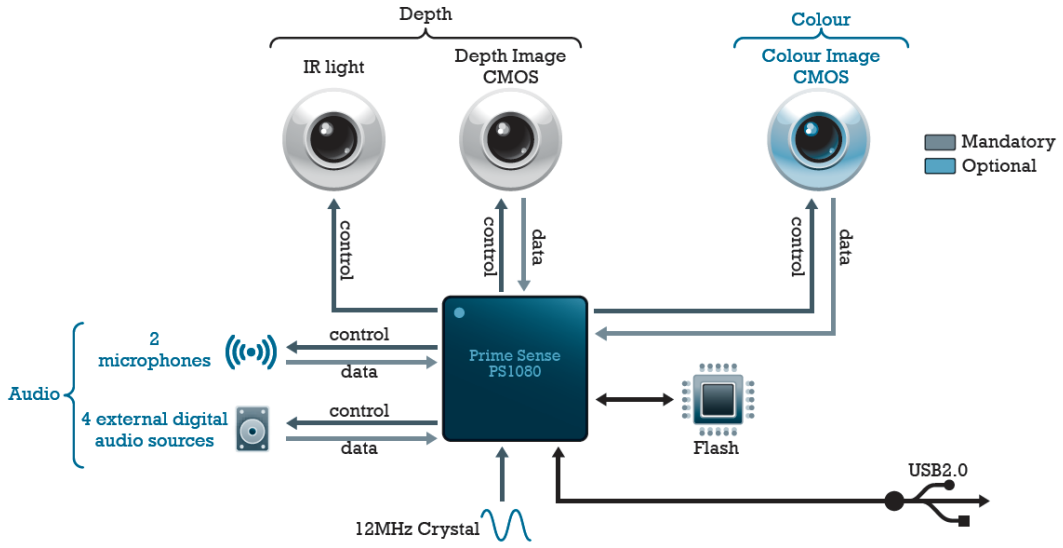


Figura 3.4: Diagrama representativo da arquitetura do Kinect [8]

- **Imagem de profundidade** - A imagem de profundidade é construída por triangulação entre o projetor e a imagem infravermelhos. Esta imagem de profundidade é retornada com maior precisão (menores valores de quantização) para valores mais próximos da câmara.

O Kinect é um dispositivo rápido e preciso dentro dos seus limites, pois é capaz de capturar simultaneamente a profundidade e imagem colorida a uma taxa de 30 *fps*. Esta integração resulta numa nuvem de pontos que contém cerca de 300 mil pontos em cada *frame*. [5]

3.2 Mapeamento Passivo com Sensores Visuais

O mundo 3D em visão é normalmente projetado em 2D, perdendo informação de profundidade. Assim, tirar conclusões acerca da distância ao ambiente é normalmente difícil quando se recorre às câmaras mais comuns 2D. Num caso limite, poderia inferir-se acerca da profundidade de um determinado objeto analisando o seu tamanho, cor ou reflexão luminosa. No entanto, é particularmente difícil para um robô tirar estas conclusões e, por isso, sem estes dados uma simples imagem não é suficiente para extrair informação tridimensional. [2]

3.2.1 Visão Estereoscópica

Visão estereoscópica é um sistema de visão, semelhante ao possuído pelos humanos, que permite extrair informação tridimensional de duas câmaras, baseadas em dois pontos de vista diferentes e que estão normalmente separados por uma distância fixa, conhecida e com campo visual em comum, figura 3.5. Desta forma, é possível construir mapas de profundidade que permitem inferir sobre a distância dos objetos ao sistema de visão.

Figura 3.5: Exemplos de arquiteturas de visão esteroscópica

Este tipo de sistema de visão computacional usa duas variáveis importantes para inferir sobre profundidade: distância entre as câmaras b e disparidade $u_l - u_r$, figura 3.6. Desta forma, é possível verificar que a distância de profundidade é inversamente proporcional à disparidade, isto é, para objetos mais próximos da câmara teremos uma disparidade maior.

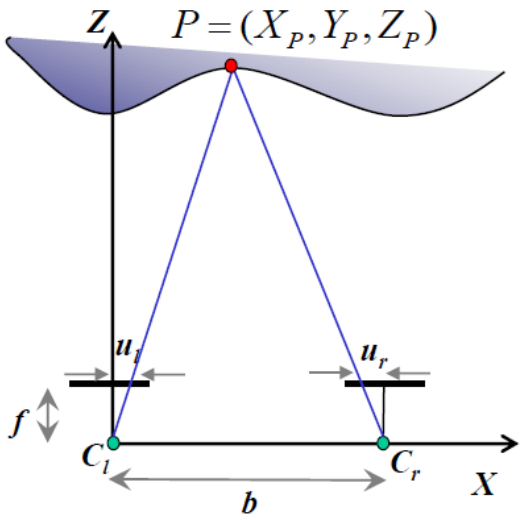
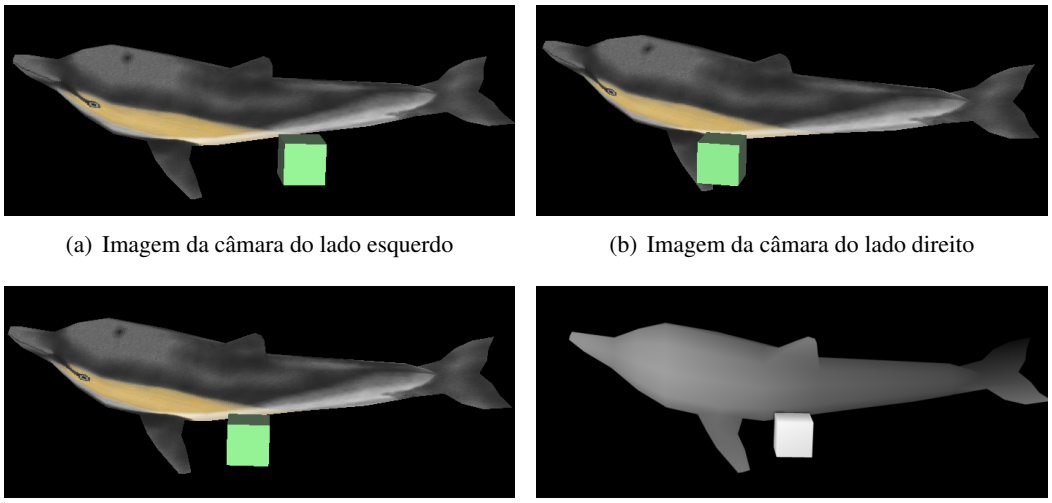


Figura 3.6: Diagrama simplificado representativo de visão estereoscópica

Por outro lado, a disparidade é proporcional à distância entre as câmaras, ou seja, para uma dada disparidade, a precisão da estimação de profundidade aumenta com o aumento da distância entre câmaras [2, 10]. No entanto, à medida que b aumenta alguns objetos podem aparecer apenas numa câmara, impossibilitando assim a visão estereoscópica desse objeto.

A informação tridimensional é conseguida através da correspondência das duas imagens, isto é, achando em cada uma das imagens, ao longo das linhas epipolares uma característica comum às duas, como por exemplo, o cubo na figura 3.7. Assim é possível calcular a disparidade pelo método acima mencionado e posteriormente inferir sobre a profundidade, figura 3.7(d).



(a) Imagem da câmara do lado esquerdo

(b) Imagem da câmara do lado direito

(c) Resultado da correspondência das duas imagens

(d) Mapa de profundidade

Figura 3.7: Exemplo de visão esteroscópica

3.3 Mapeamento Ativo por *Time-of-Flight*

Existem sensores que extraem informação de distância por meio de propagação de uma onda sonora ou eletromagnética. É sabido que a distância percorrida por uma onda deste tipo é dado por:

$$d = vt \tag{3.1}$$

De forma simples, pela equação 3.1, verifica-se que sabendo a velocidade da onda v e calculando o seu tempo de ida e volta t é possível calcular de forma direta a distância por ela percorrida d. A qualidade deste tipo de sensores depende normalmente da:

- incerteza na determinação do tempo exato de chegada do sinal refletido;
- precisão na medição do tempo de voo;
- dispersão do sinal emitido;
- interação da onda com o objeto (por exemplo, absorção pela superfície);
- variação da velocidade de propagação;
- velocidade do robô ou objeto alvo;

3.3.1 Sensor Ultra-sónico

Um sensor-ultra sónico, figura 3.8, transmite ondas de pressão de forma a medir o tempo que a onda demora a refletir e a ser adquirida pelo recetor. A distância ao objeto que causa a reflexão é calculada tendo por base a equação 3.1, considerando apenas metade da viagem feita pela onda e velocidade próxima da velocidade do som (dependendo da qualidade do meio).

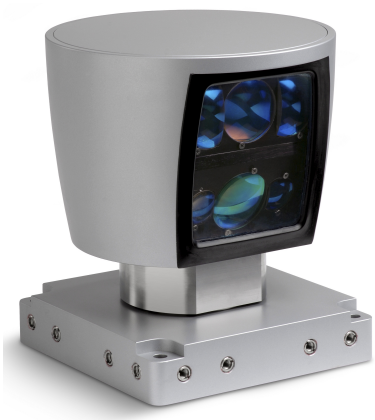
$$d = \frac{vt}{2} \tag{3.2}$$

Este tipo de sensor é normalmente utilizado para deteção ou prevenção de colisões e para cálculo de distâncias. Os sensores ultra-sónicos têm a vantagem de detetar superfícies transparentes, que normalmente oferecem problemas em sistemas de visão.

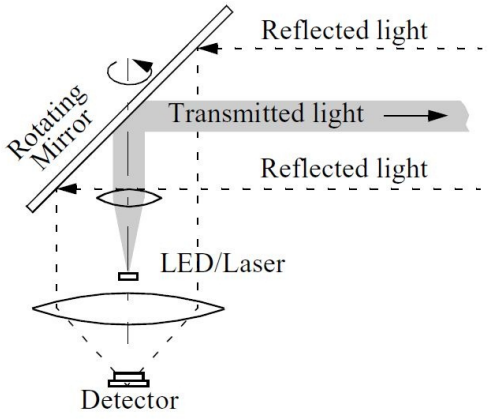
Figura 3.8: Sensor de distância ultra-sónico, Parallax's PING)))

3.3.1.1 *Laser Rangefinders*

Os sensores *laser rangefinders*, figura 3.9, apresentam resultados melhores quando comparados com os ultra-sónicos devido ao facto de estes usarem ondas eletromagnéticas, em vez de ondas sonoras. Este tipo de sensores estão providos de um transmissor que ilumina o objeto por meio de um feixe (por exemplo, laser) e um recetor, sendo por isso normalmente conhecidos por sensores LIDAR (*LIght Detection And Ranging*). Estes dispositivos mapeiam o ambiente baseando-se no tempo de ida e volta do feixe luminoso. Um *laser rangefinder* adquire dados num plano e estão normalmente associados a um mecanismo mecânico com um espelho rotativo que projeta vários feixes em todo o meio envolvente, figura 3.9(b).



(a) Velodyne HDL-64E



(b) Esquemático do modo de funcionamento

Figura 3.9: Esquemático e exemplo de uma tecnologia *Laser rangefinder*

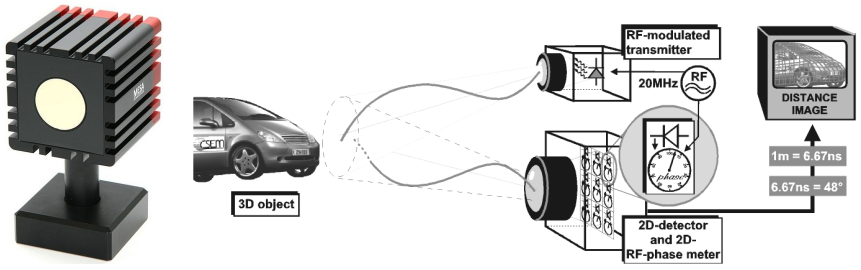
De seguida são apresentadas caraterísticas gerais deste tipo de sensores [10, 11, 2]:

- Resultados muito densos;
- Grande resolução (milhões de pontos);
- Muito precisos (inferir ao *mm*);
- Preço na ordem dos milhares de euros.

Os sensores LIDAR têm sido especialmente preferidos para controlo de veículos autónomos, desde carros [12] a helicópteros [13, 14].

3.3.2 Câmaras *Time-of-Flight*

Este tipo de sensores emitem um pulso de luz para o objeto e medem o intervalo de tempo entre a emissão, reflexão e receção do mesmo sinal. Como a velocidade da luz é conhecida com precisão, a medida da distância é calculada diretamente com o cálculo da equação 3.2. Para obter informação 3D, algumas câmaras TOF usam um *scanner* mecânico que faz mover o laser pelo meio. O meio é mapeado combinando sequencialmente todas as distâncias calculadas. No entanto, este tipo de *scanners* são caros, volumosos e acumulam erros devido a vibrações. Um possibilidade mais fiável de mapeamento 3D consiste em iluminar o ambiente por meio de um cone de luz modulada em vez de um único feixe, e adquirir esta iluminação em conjunto com uma câmara 2D. Desta forma, cada pixel contém informação acerca da intensidade do meio e profundidade [15]. A figura 3.10(b) ilustra este último tipo de mapeamento 3D (sem recurso a um *scanner*). Com este tipo de arquitetura o meio é capturado de uma só vez, isto é, para cada *frame* é atualizada a distância ao meio, estando este refrescamento dependente da frequência com que é projetada a luz nos objetos no seu campo de visão. [10, 11]



(a) SR4000 (b) Esquemático do modo de funcionamento

Figura 3.10: Esquemático e exemplo de uma câmara *Time-of-flight*

Existem várias marcas e modelos no mercado para este tipo de sensores. No entando, todas elas apresentam algumas caraterísticas em comum [10, 11, 16] :

- Compactas e fáceis de manusear;
- Resolução baixa (176×144);
- Muito rápidas (100 *fps*);
- Resultados com muito ruído;

- Preço na ordem dos milhares de euros.

As câmaras TOF são aplicadas nos mais diversos ramos da engenharia. Convém no entanto salientar o uso deste tipo de sensores no ramo automóvel [17, 18] e em situações de interação homem-máquina como, por exemplo, videojogos [19].

Capítulo 4

Sistema Baseado em Visão

Este projeto centra-se maioritariamente na resolução de um problema por meio de visão computacional. Desta forma, neste capítulo o projeto é contextualizado teoricamente, e são também apresentados alguns resultados práticos do sistema de visão que equipa a plataforma de resolução do problema.

4.1 Modelo de Câmara

Neste capítulo é feita uma primeira abordagem teórica daquilo que é essencial num sistema baseado em visão.

4.1.1 Parametrização

Os parâmetros do modelo de uma câmara determinam a relação matemática da transformação dos pontos 3D (mundo) em pontos 2D (imagem) [20]. Estes dividem-se normalmente em dois grupos:

- **Parâmetros intrínsecos:** representam as características ópticas e geométricas internas da câmara como a distância focal f, factores de escala (α_x, α_y), posição em pixeis da projeção ortogonal do centro ótico no plano de projeção (c_x, c_y) e as imperfeições geométricas introduzidas pelo sistema sensor da câmara s;
- **Parâmetros extrínsecos:** fornecem a posição e orientação da câmara em causa, definidas através de uma matriz de rotação $\mathbf{R}$ e um vector de translação $\mathbf{t}$, em relação a um certo sistema de coordenadas 3D global (mundo).

Este tipo de parametrização é normalmente explicada recorrendo ao modelo de câmara *pinhole*. Este tipo de câmara consiste numa caixa opaca com um pequeno orifício por onde a luz consegue penetrar.

Quando a luz é captada pela câmara atravessa o orifício e a imagem do meio é projetada no plano oposto ao orifício de forma invertida, figura 4.1. Para fazer variar o tamanho da imagem

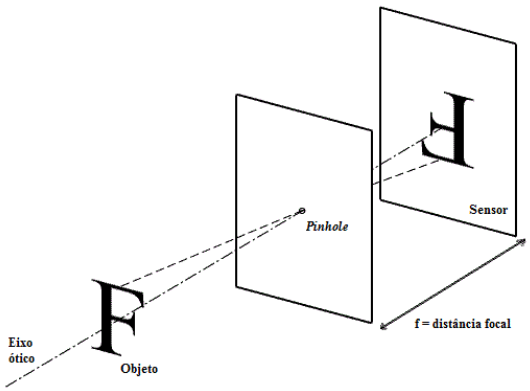
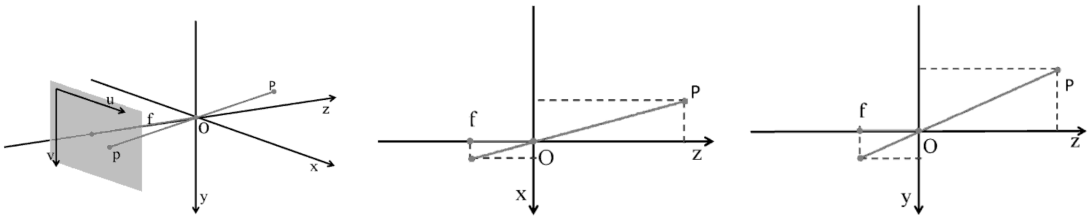


Figura 4.1: Ilustração do princípio de funcionamento do modelo de câmara *pinhole*

projetada em relação ao objeto real, faz-se variar a distância entre o plano que contém o buraco e o plano onde é projetada a imagem - distância focal f.

Quanto mais pequeno for o orifício, mais nítida será a imagem. No entanto, esta será também mais escura, dado que menos luz atravessa a câmara. A escolha do tamanho do orifício é, portanto, um compromisso entre a nitidez e a intensidade luminosa da imagem.



(a) (b) (c)

Figura 4.2: Projeção: a) Ponto P é projetado para o pixel p; b) secção horizontal; c) secção vertical

A projeção ilustrada na figura 4.2 é uma descrição geométrica da relação entre os pontos do mundo e as que são obtidas na imagem por aplicação do modelo de câmara *pinhole*, com centro de projeção O. Desta forma, e por aplicação do teorema de similaridade de triângulos, obtém-se o seguinte:

$$\frac{u}{f} = \frac{x_w}{z_w} \tag{4.1}$$

$$\frac{v}{f} = \frac{y_w}{z_w} \tag{4.2}$$

Como o centro do sensor de imagem não coincide normalmente com o eixo ótico da câmara, tem de ser realizada uma translação em cada eixo de c_x e c_y. Por outro lado, a imagem é normalmente retangular o que implica distâncias focais diferentes para cada eixo. Desta forma, considere-se as novas distâncias focais f_x e f_y para o eixo horizontal e vertical, respetivamente. Assim sendo,

as equações 4.1 e 4.2 são reformuladas de forma a incluírem estas alterações:

$$\frac{u}{f_x} = \frac{x_w}{z_w} + c_x \tag{4.3}$$

$$\frac{v}{f_y} = \frac{y_w}{z_w} + c_y \tag{4.4}$$

Em conjunto, as equações 4.3 e 4.4 definem as condições de formação de uma imagem: transformação associada à projeção de pontos 3D do mundo, de coordenadas (x_w, y_w, z_w), em pontos 2D da imagem, de coordenadas (u, v). A equação 4.5 compacta o modelo até aqui apresentado, que inclui os parâmetros intrínsecos e extrínsecos da câmara.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x.\alpha_x & s & c_x \\ 0 & f_y.\alpha_y & c_y \\ 0 & 0 & 1 \end{bmatrix} . \Big[\mathbf{R}|\mathbf{t}\Big] . \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = \mathbf{A} . \Big[\mathbf{R}|\mathbf{t}\Big] . \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{4.5}$$

A matriz **A** é normalmente apelidada de matriz câmara, dado que contém todos os seus parâmetros internos de arquitetura ótica.

4.1.2 Distorção

A aproximação considerada no modelo de câmara descrito pela equação 4.5 não tem em conta as distorções introduzidas pelo sistema ótico da câmara (nomeadamente, pelas lentes), o que não é compatível com aplicações nas quais é requerida elevada precisão. Como a modelação exata das lentes é complexa, normalmente utiliza-se um modelo empírico para a distorção imposta pelas mesmas, considerando-se geralmente dois tipos de distorção: radial e tangencial. A distorção radial representa um desvio da posição desejada na direção do raio (segmento de reta entre o centro da imagem e a posição correta) e manifesta-se na forma de "barril". Por outro lado, distorção tangencial representa um desvio na direção tangencial do dito ponto, e ocorre porque as lentes não estão perfeitamente paralelas com o plano de imagem. Na figura 4.3, verifica-se que a posição assinalada como distorcida sofre destes dois tipos de distorção.

O modelo de Brown-Conrady [21, 22] incluiu distorção radial e tangencial e é o modelo mais utilizado na remoção da distorção, equações 4.6 a 4.9.

$$x_d = \frac{u - c_x}{f_x} \tag{4.6}$$

$$y_d = \frac{v - c_y}{f_y} \tag{4.7}$$

$$x_u = x_d(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + 2p_1 x_d y_d + p_2(r^2 + 2x_d^2) \tag{4.8}$$

$$x_u = x_d(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + p_2(r^2 + 2y_d^2) + 2p_1 x_d y_d \tag{4.9}$$

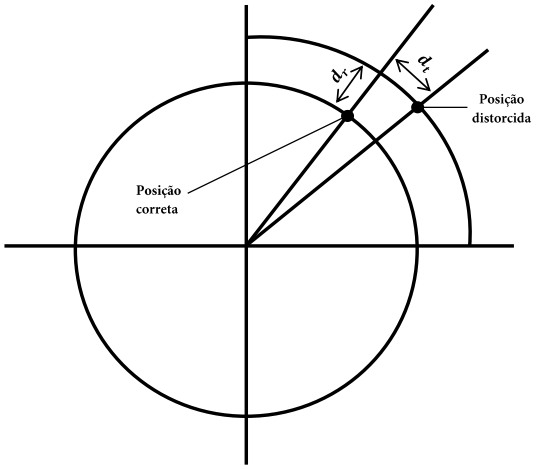


Figura 4.3: Representação de dois tipos de distorção: radial (d_r) e tangencial (d_t)

As variáveis (x_d, y_d) representam as coordenadas distorcidas; (x_u, y_u) representam os novos valores das coordenadas não destorcidas; k_1, k_2, k_3 são os coeficientes de distorção radial; p_1, p_2 são os coeficientes de tangencial e, por fim, $r^2 = x_d^2 + y_d^2$.

Este algoritmo permite mapear as coordenadas para novas posições. Na realidade, ao aplicar a equação 4.10 o valor de determinada coordenada está a ser movido para um nova posição. [23]

$$mapa_{xy}(u,v) = (x_u f_x + c_x, y_u f_x + c_y) \tag{4.10}$$

Na situação exemplo da figura 4.3, a aplicação do mapa 4.10 permitiria passar o ponto assinalado como distorcido para a posição assinalada como correta. Os pontos em redor seriam também transladados para as suas posições corretas, pois a distorção nunca afeta apenas um ponto isolado. Evidentemente, a precisão desta translação depende dos coeficientes de distorção. Assim, após aplicação a todos os pontos que constituem a imagem, a distorção é atenuada ou mesmo anulada.

4.1.3 Calibração

No domínio da visão computacional entende-se por calibração, como o processo para a determinação da relação matemática que, para o respetivo sistema, transforma corretamente os pontos 3D no mundo em pontos 2D na câmara, e vice-versa. Neste contexto, a calibração de uma dada câmara é o processo de determinação dos parâmetros que caraterizam a arquitetura ótica (parâmetros intrínsecos e de distorção), e a atitude (orientação e posição) da câmara relativa a um certo sistema de coordenadas do mundo (parâmetros extrínsecos). [20]

Os métodos de calibração são usualmente classificados em dois grandes grupos:

- **Calibração tradicional**: Estes métodos associam pontos da imagem 2D com pontos 3D bem conhecidos no mundo, requerendo assim a aquisição de imagens de um objeto de calibração, cuja geometria 3D é perfeitamente conhecida. Este tipo de abordagem necessita normalmente de uma arquitetura de calibração complexa.
- **Auto-calibração**: Estes métodos associam, entre imagens sucessivas, características da cena ou do objecto a reconstruir, tal como a correspondência de pontos, de tal forma que não requerem a utilização de objetos de calibração. Apesar de esta técnica ser bastante flexível, nem sempre se obtêm resultados fiáveis dada a quantidade de parâmetros a estimar.

Zhang [20, 24] propôs um método flexível, robusto e de custos monetário e computacional baixos. Estas caraterísticas verificam-se devido ao facto de a sua técnica necessitar apenas de um padrão planar em algumas orientações distintas. O padrão pode ser impresso em impressoras convencionais e afixado a uma estrutura planar rígida. A orientação relativa entre a câmara e o objeto planar pode ser feita movendo a câmara ou o plano. O movimento relativo envolvido nas imagens de calibração usadas não necessita de ser conhecido. Durante o processo de aquisição das imagens a usar na calibração, as características internas da câmara não podem ser alteradas. Desta forma, apenas os parâmetros extrínsecos são alterados quando o padrão de calibração é reposicionado. De forma resumida o algoritmo de Zhang consiste no seguinte:

1. Imprimir um padrão e afixá-lo a um superfície plana;
2. Capturar imagens do modelo planar em diferentes orientações fazendo mover o plano ou a câmara;
3. Detetar os pontos chave da imagem;
4. Estimar os cinco parâmetros intrínsecos e todos os parâmetros extrínsecos;
5. Estimar os coeficientes de distorção;
6. Refinar todos os parâmetros por minimização;

Este tipo de abordagem enquadra-se entre uma calibração tradicional e auto-calibração dado que neste método se usa informação 2D (padrão planar impresso) em vez de 3D, o que a torna mais flexível que a técnica tradicional. Comparando-a com a auto-calibração, este método é consideravelmente mais robusto, pois está menos sujeito às mudanças do meio, por exemplo.

4.2 Microsoft Kinect

No capítulo 3 foram apresentadas, de uma forma sumária, as caraterísticas principais do Microsoft Kinect, dada a sua contextualização no estado da arte atual. Aqui, este dispositivo é abordado com muito mais detalhe, dando a conhecer a sua história e modo de funcionamento, através de teoria e de testes práticos realizados durante a realização deste projeto. De forma a relembrar as principais caraterísticas deste dispositivo é apresentada na figura 3.4 a interligação dos componentes que o constituem.

4.2.1 História

O Microsoft Kinect foi lançado no final do ano 2010 e as suas características tornaram este dispositivo de visão computacional largamente aceite pela comunidade científica. Aquela que mais se evidencia é a relação qualidade/preço que o torna especialmente atrativo para as instituições académicas e é, por isso, um dispositivo que se enquadra no estado da arte atual.

A crescente utilização do Kinect em trabalhos científicos impulsionou o surgimento de uma comunidade que se dedica a partilhar drivers, algoritmos e testes de performance. Uma solução que à partida era fechada - do ponto de vista do software - rapidamente se tornou liberalizada dado o trabalho de toda a comunidade na construção de drivers *open source*. Ainda em 2010, várias entidades, interessadas em usar este produto nos seus trabalhos projetaram e publicaram esses drivers com o intuito de possibilitar a todos o uso deste equipamento para programação de algoritmos de visão tridimensional. Exemplos dessas entidades são a OpenKinect, responsável pela criação dos drivers libfreenect [25] e a PrimeSense, criadora dos drivers OpenNI [26].

Resultado disto são os inúmeros projetos publicados com recurso a esta tecnologia, que atualmente abrange quase todas as áreas científicas em que a visão desempenha um papel importante. Apercebendo-se do impacto que o Kinect alcançou na comunidade científica, a Microsoft lançou SDKs (*software development kits*) no verão de 2011, para promover a programação do Kinect no ambiente Microsoft Windows.

4.2.2 Cálculo de Profundidade

Tratando-se de um dispositivo ainda no ativo e rentável no mundo dos jogos, a Microsoft avançou com pouca informação acerca do seu modo de funcionamento. No entanto, a curiosidade daqueles que usam este dispositivo fez com que surgissem inúmeros artigos que explicam o seu funcionamento. A explicação tem normalmente por base o desmantelamento do Kinect de modo a perceber a construção e constituição do dispositivo, e ainda um conjunto de testes para compreender o funcionamento do seu conjunto no cálculo de profundidade.

4.2.2.1 Princípio de Funcionamento

A explicação mais aceite pela comunidade para o princípio de funcionamento no cálculo de profundidade, consiste naquela que trata o Kinect como um dispositivo que tem por base a triangulação e a projeção de pontos na gama do infravermelho. A projeção de pontos tem um padrão conhecido, como demonstra a figura 4.4, e recorre a um elemento difusor e difrativo. [27]

Aquando da construção do Kinect, uma série de imagens de referência são adquiridas a diferentes distâncias e guardadas no dispositivo. Para as primeiras aquisições a câmara de infravermelhos observa o meio e calcula, para cada ponto projetado, a triangulação entre a imagem virtual (padrão guardado a uma distância conhecida) e o padrão observado. Nas aquisições seguintes, este calcula a triangulação entre a frame atual e a frame anterior, figura 4.5. Na realidade, o Kinect calcula a profundidade com base na deformação do seu padrão conhecido induzida pelo meio, em que cada ponto da imagem de infravermelhos tem um ponto do mundo associado.

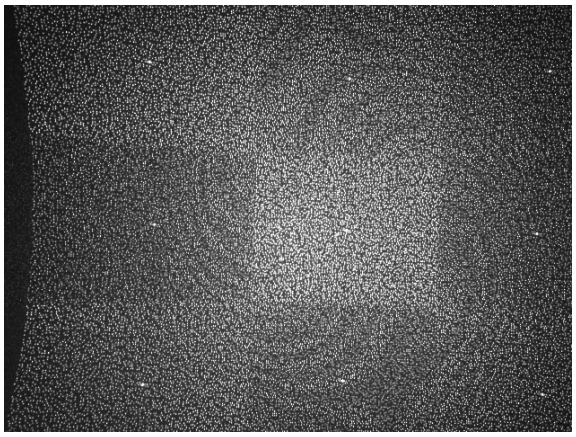


Figura 4.4: Imagem infravermelhos do padrão projetado pelo Kinect

Os pontos projetados pelo Kinect variam de tamanho e forma consoante a distância e orientação do objeto ao sensor. O feixe emitido pode ser encarado como tendo uma forma cónica (em que o vértice coincide com o sensor). Este facto, faz com que o tamanho do ponto projetado seja tanto maior, quanto maior for a distância a que ele se encontro do emissor. A forma do feixe a uma dada distância varia com a orientação do objeto, dado que a projeção de um feixe luminoso afeta naturalmente a forma do ponto, tendendo para uma forma mais ou menos oval consoante a inclinação do objeto. O facto do aumento de tamanho dos pontos ser proporcional à distância leva a uma perda de precisão, dado que a triangulação será feita sobre pontos maiores e, por isso, o cálculo de profundidade será mais ambíguo e incerto para distâncias maiores.

O sensor de infravermelhos do Kinect tem ainda a particularidade de possuir um elemento ótico astigmático - possui diferentes focagens para orientações distintas. Esta facto faz com que sejam adquiridos pontos não só com tamanho distinto (como explicando anteriormente), mas também com orientação diferente. Isto acontece mesmo mantendo a orientação do objeto em relação ao sensor, figura 4.6.

Considerando todas estas caraterísticas, a própria PrimeSense divide o funcionamento no cálculo de profundidade pelo dispositivo em três regiões distintas, como se vê na figura 4.7. A primeira região permite obter alta precisão para objetos entre $0,8$ e $1,2m$, a segunda região tem média precisão para distâncias entre $1,2$ e $2,0m$ e, por fim, a terceira região tem baixa precisão para objetos que se encontrem entre $2,0$ e $3,5m$. O Kinect deteta objetos acima dos $3,5m$, mas com grandes erros de medição que podem ultrapassar os $10cm$.

A triangulação é calculada através da correlação entre frames. Desta forma, um desvio entre os dois padrões na direção de xOy, implica uma distância proporcional de profundidade em z.

Na realidade, a existência de um *offset* de 8 pixeis entre a imagem de infravermelhos e a de profundidade (criada a partir da infravermelhos) remete para uma janela de correlação de 9×9 ou 9×7, que não varia com a profundidade nem entre diferentes aparelhos. [27]

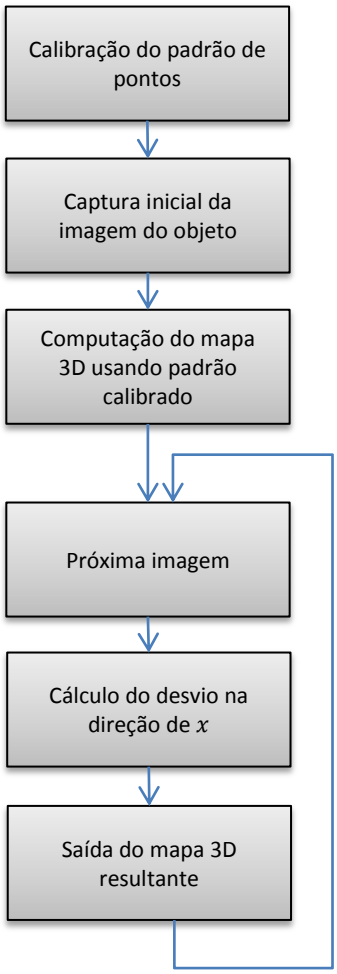


Figura 4.5: Fluxograma do funcionamento do Kinect, baseado na patente da PrimeSense

4.2.2.2 Cálculo por Disparidade

Um sistema estereoscópico comum é calibrado de forma às imagens serem paralelas e terem linhas horizontais correspondentes. Um sistema deste tipo é dado por:

$$z = \frac{bf}{d} \tag{4.11}$$

onde z corresponde à profundidade em metros, b a distância entre câmaras, f a distância focal em pixeis e d a disparidade também em pixeis [27]. Numa situação de disparidade zero a profundidade é infinita, o que corresponde a uma situação em que os raios de cada câmara se encontram paralelos. Tal pode corresponder a um caso em que as imagens não estão sobrepostas e, por isso, não é possível inferir sobre a profundidade. No entanto, no Kinect, a disparidade não está normalizada desta forma, isto é, uma disparidade nula não corresponde a uma profundidade infinita. No

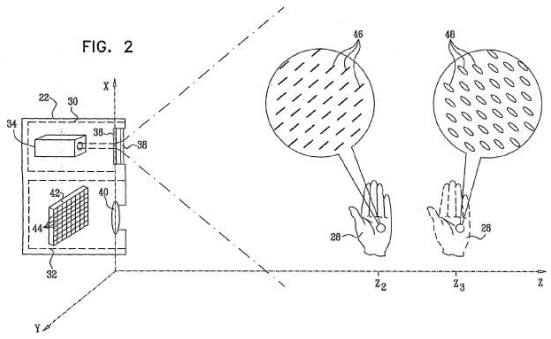


Figura 4.6: Variação da forma e orientação dos pontos em função da distância, original da patente da PrimeSense

caso do Kinect, a disparidade é calculada da seguinte forma:

$$d = \frac{1}{8}(d_{off} - K_d) \tag{4.12}$$

onde d é a disparidade normalizada, K_d a disparidade retornada pela Kinect e o d_{off} um valor de *offset* particular a cada Kinect. O coeficiente $\frac{1}{8}$ resulta da conversão de unidades de $[pixel]$ para $[\frac{pixel}{8}]$.

Posto isto, o cálculo da profundidade do Kinect pode ser visto da seguinte forma:

$$z = \frac{bf}{\frac{1}{8}(d_{off} - K_d)} \tag{4.13}$$

em que b é aproximadamente $0.075m$ (distância entre o projetor e câmara infravermelhos) e d_{off} tipicamente ronda os 1090.

4.2.2.3 Cálculo por Aproximação Matemática

Alternativamente, o cálculo de profundidade pode ser feito com recurso a uma função matemática. Esta função funciona como conversor dos dados diretamente retornados do Kinect para unidades SI (metros). Ao longo do tempo foram surgindo diferentes funções com a intenção de que esta conversão se aproximasse cada vez mais da realidade. Um modelo proposto para a calibração trata-se da equação 4.14, apresentada por Stephane Magnenat [28].

$$p_m = 0,1236\tan(\frac{p_k}{2842,5} + 1,1863) \tag{4.14}$$

Em que p_m corresponde ao valor em metros e p_k ao valor diretamente retornado pelo Kinect. Ao contrário do método anterior, esta conversão não usa parâmetros intrínsecos do dispositivo, o que permite obter informação da profundidade em unidades SI sem o conhecimento dos dados calculados pela calibração.

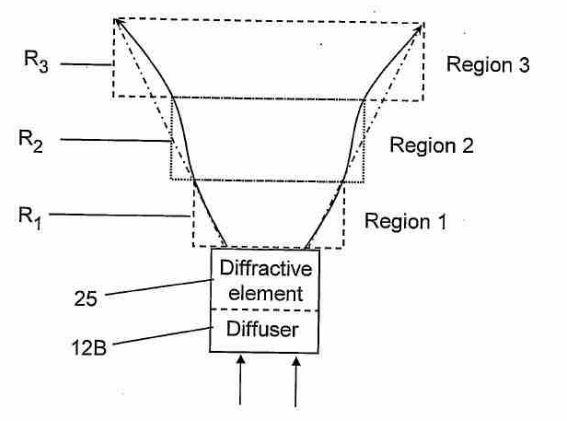


Figura 4.7: As três regiões distintas de funcionamento, original da patente da PrimeSense

O gráfico da figura 4.8 permite verificar que o valor máximo retornado pelo Kinect, numa situação de inexistência de erro de medição, está abaixo dos 1100. Este valor vai também de encontro ao *offset* utilizado pelo método anterior por forma a não obtermos uma disparidade zero.

4.2.3 Algoritmo de Calibração

Obter informação simultânea acerca da cor e profundidade do meio é algo muito desejado em visão computacional. Para que tal possa acontecer são necessários, no mínimo, dois sensores. O Kinect possui a configuração básica: um par de sensores (um de infravermelhos responsável pelo cálculo de profundidade, e um de cor) afixados a uma estrutura rígida.

Para a grande maioria das aplicações que usam a imagem colorida e a de profundidade em conjunto é absolutamente crucial conhecer os parâmetros de calibração de ambas as câmaras. Aqui, incluem-se os parâmetros intrínsecos de cada câmara e a rotação e translação relativa das duas câmaras. A calibração de uma única câmara colorida é algo já extensivamente estudado, mas a calibração conjunta da imagem colorida e a de profundidade oferece novos desafios:

- Pontos chave para a calibração, como os cantos de uma estrutura plana, são normalmente indistinguíveis de outros pontos na imagem de profundidade;
- Apesar de a descontinuidade poder ser facilmente observada na imagem em profundidade, os pontos nas fronteiras são normalmente incertos e não confiáveis;

O método de calibração utilizado neste projeto tem por base a calibração independente de cada uma das câmaras, sendo posteriormente calculada a atitude relativa entre elas. O resultado permite juntar, numa só imagem, a informação acerca da profundidade e cor. Alternativamente, poderia ser utilizado um método em que as câmaras não são calibradas de forma independente, mas sim como um todo. Desta forma é possível usar as medições de um sensor para melhorar a calibração do outro. No entanto, estes métodos são mais complexos e os resultados, ainda assim, são semelhantes aos obtidos pelo primeiro procedimento enunciado.

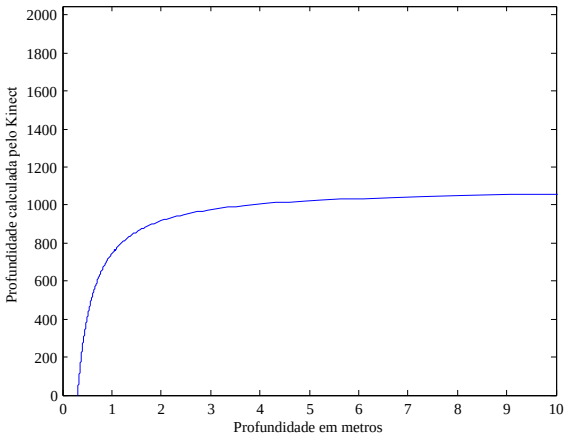


Figura 4.8: Representação dos valores em metros gerados pela equação 4.14

O método de calibração de Zhang encontra-se implementado em linguagem C++ na biblioteca pública de processamento de imagem OpenCV (*open source computer vision*). Esta biblioteca tem grande aceitação na comunidade científica e é, por isso, largamente utilizada em projetos de visão computacional. De forma a enquadrar-se com as caraterísticas do projeto, o algoritmo foi implementado invocando as funções do OpenCV de FreePascal, de modo a possibilitar a sua execução numa aplicação desenvolvida em Lazarus.

Na figura 4.9 podem ser vistos, de forma resumida, os passos principais do algoritmo usado na calibração do Kinect. O algoritmo divide-se em duas partes:

1. Ciclo de aquisição, processamento de imagem e organização dos dados;
2. Cálculo dos parâmetros intrínsecos e extrínsecos.

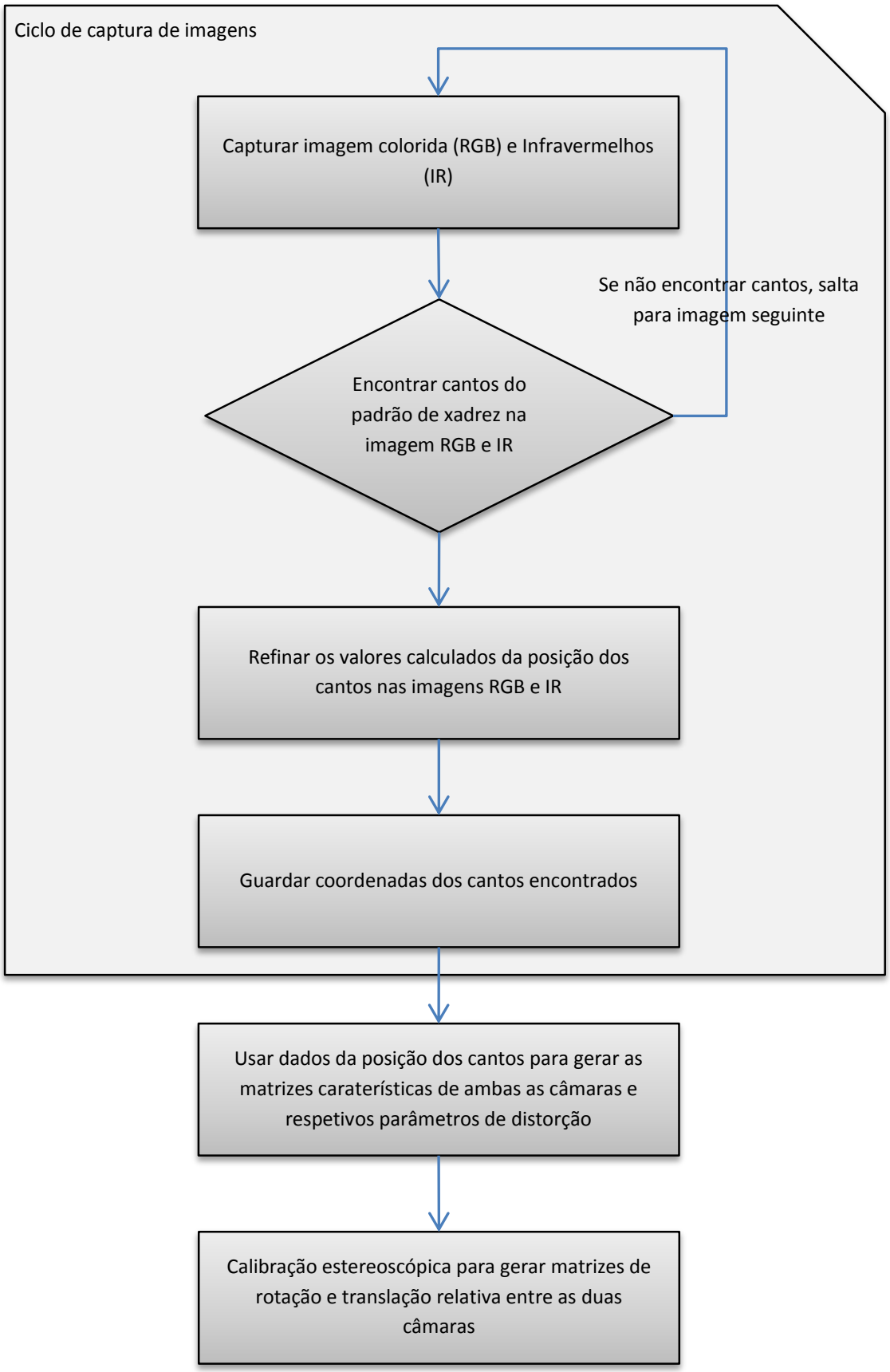
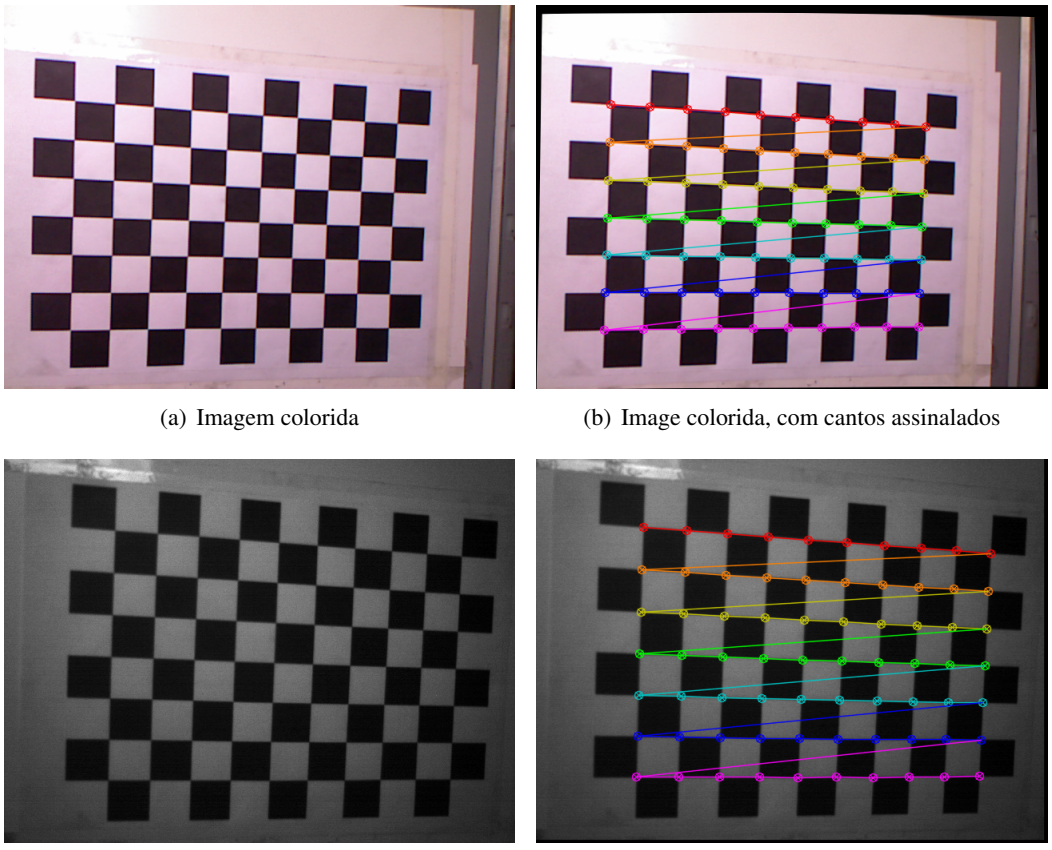


Figura 4.9: Representação resumida do algoritmo de calibração do Kinect

Numa primeira fase, cada par de imagens capturado é processado com o objetivo de encontrar a posição dos cantos interiores que constituem o padrão do xadrez, isto é, os pontos centrais que resultam da interseção de 4 quadrados. O padrão de xadrez utilizado na calibração tem $10 \times 7 = 70$ pontos desse tipo, e o algoritmo de pesquisa só funcionará se todos os cantos forem encontrados. Posteriormente, as coordenadas na imagem de todos os cantos são refinadas, realizando uma nova pesquisa na primeira região detetada e atualizando, assim, a posição inicialmente encontrada, tabela 4.1. Para efeitos de visualização, o algoritmo implementado permite que os parâmetros de distorção calculados para cada par de imagens seja aplicado no mesmo par, figura 4.10. Mas na realidade, este tipo de processamento é realizado ciclicamente para todas as imagens capturadas, e todos os conjuntos de coordenadas dos cantos são guardados. O objetivo é chegar ao fim do algoritmo com um conjunto de parâmetros robusto, independentemente da orientação com que a câmara encara o mundo.



(a) Imagem colorida

(b) Image colorida, com cantos assinalados

(c) Imagem infravermelhos

(d) Imagem infravermelhos, com cantos assinalados

Figura 4.10: Amostra do conjunto de pares imagens usado no algoritmo de calibração

Numa segunda fase, é realizada a parametrização do modelo de câmara e calculada a rotação e posição relativa de ambos os sensores. Este processamento é concretizado recorrendo ao algoritmo de Zhang, tendo como base de cálculo todos os pontos guardados na etapa anterior. Aqui, o algoritmo computa sequencialmente os parâmetros de cada sensor, dando origem à matriz caraterística de cada câmara: colorida e infravermelhos, usando para cada uma, as coordenadas dos

Posições Iniciais		**Posições Refinadas**	
Ponto u	Ponto v	Ponto u	Ponto v
169.4582214	140.6264954	168.5822449	140.3236542
218.4303741	140.3722992	217.6098633	140.5870819
264.5852661	142.4075012	264.2110291	140.6059265
309.5411072	138.4974213	309.0335693	139.3108063
352.6125488	140.3268585	352.1502075	139.3183289
392.3797913	138.3362427	392.4596252	138.5177917
432.0238953	138.0047913	432.0507507	137.7920685
468.9493103	137.9847717	468.9306030	138.2051086
502.2992554	138.1404266	505.5789490	137.3484955

Tabela 4.1: Amostras das coordenadas iniciais e refinadas dos cantos do xadrez

pontos da imagem correspondente. Os parâmetros à saída incluem as distâncias focais, os centros óticos e os coeficientes de distorção. Todos os dados até aqui calculados servem, posteriormente, para calcular os parâmetros extrínsecos por meio de uma calibração estereoscópica, que à saída inclui a matriz de rotação e translação relativas.

De notar que é necessário dar a conhecer ao algoritmo o tamanho do lado dos quadrados, em unidades SI, para que o algoritmo seja capaz de mapear o tamanho do pixel no mundo. Desta forma, os parâmetros extrínsecos da calibração são apresentados na mesma unidade passada ao algoritmo, enquanto que os intrínsecos têm o pixel como unidade. Durante todo o processo foi usado o metro como unidade de medida.

Na figura 4.10 é apresentada uma amostra do algoritmo de calibração em ação. Nas figuras à esquerda, 4.10(a) e 4.10(c), são representadas as imagens diretamente adquiridas pelo Kinect, que servem para calibração. As figuras à direita, 4.10(b) e 4.10(d), representam o resultado da pesquisa dos cantos do xadrez, com as posições devidamente assinaladas. Para além disso, estas imagens apresentam já o resultado da remoção da distorção. Note-se que existe uma margem de cor preta, resultante do mapeamento que reposiciona os pixeis. O método de mapeamento é explicado na subsecção 4.1.2.

Para a calibração do Kinect utilizado no projeto foram capturadas 40 pares de imagens, de modo a existir um bom número de amostras para o cálculo dos valores finais. Para a aquisição do conjunto de pares de imagens e por imposição do algoritmo de Zhang, faz-se variar a orientação do plano que contém o xadrez. Na prática, a translação também é alterada. No entanto, deve ser sempre mantida uma distância mínima entre o xadrez e o Kinect, de modo a conseguir a melhor qualidade de imagem possível na zona do xadrez. Recorde-se que a calibração é realizada apenas na zona do xadrez, pelo que também é importante que este ocupe uma grande área da imagem, de forma a alargar a calibração a todo o sensor.

No subcapítulo 4.1 é apresentado o modelo de câmara e explicada a parametrização, distorção e calibração. A calibração tem como objetivo dar significado numérico a todas as variáveis que constituem o modelo. Devido a variações que ocorrem na altura da sua construção estes valores variam de dispositivo para dispositivo, o que torna a calibração dependente deles. No entanto,

os valores retornados pela calibração são normalmente estáticos, desde que não se faça variar a configuração das câmaras; por exemplo, alterando as suas posições relativas (novos parâmetros extrínsecos) ou colocando lentes (novos parâmetros intrínsecos e de distorção).

Relembremos a forma como são, normalmente, organizados os parâmetros relativos aos sensores.

$$D = \begin{bmatrix} k_1 \\ k_2 \\ p_1 \\ p_2 \\ k_3 \end{bmatrix}, \quad A = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{4.15}$$

Em que o vetor D contém os coeficientes de distorção radial k e tangencial p. A matriz câmara é constituída pelas distâncias focais f e e centro ótico c. Quando foi apresentado o modelo de câmara na secção 4.1, foram introduzidos mais parâmetros, nomeadamente os fatores de escala α e de imperfeição geométrica s. O algoritmo de calibração implementado na aplicação com recurso à biblioteca OpenCV ignora esses fatores, pelo que para a matriz câmara de ambos os sensores deverá ser apenas considerado aquilo que foi apresentado em 4.15. A calibração do Kinect utilizada neste projeto produziu os resultados apresentados de seguida.

$$D_{IR} = \begin{bmatrix} -1.10792249 \times 10^{-1} \\ 3.41987759 \times 10^{-1} \\ -5.17484406 \times 10^{-3} \\ 2.03531492 \times 10^{-3} \\ 0.00000000 \times 10^{+1} \end{bmatrix}, \quad D_{RGB} = \begin{bmatrix} -1.24741876 \times 10^{-2} \\ 6.60263300 \times 10^{-2} \\ -4.54839040 \times 10^{-3} \\ 5.47013013 \times 10^{-3} \\ 0.00000000 \times 10^{+1} \end{bmatrix}$$

$$A_{IR} = \begin{bmatrix} 5.83422791 \times 10^{+2} & 0.00000000 \times 10^{+1} & 3.28588226 \times 10^{+2} \\ 0.00000000 \times 10^{+1} & 5.88072937 \times 10^{+1} & 2.39609024 \times 10^{+2} \\ 0.00000000 \times 10^{+1} & 0.00000000 \times 10^{+1} & 1.00000000 \times 10^{+1} \end{bmatrix}$$

$$A_{RGB} = \begin{bmatrix} 5.20303711 \times 10^{+2} & 0.00000000 \times 10^{+1} & 3.12870697 \times 10^{+2} \\ 0.00000000 \times 10^{+1} & 5.24107422 \times 10^{+2} & 2.57529602 \times 10^{+2} \\ 0.00000000 \times 10^{+1} & 0.00000000 \times 10^{+1} & 1.00000000 \times 10^{+1} \end{bmatrix}$$

A análise dos parâmetros intrínsecos e dos coeficientes de distorção, resultantes da calibração efetuada, permite-nos conhecer melhor as caraterísticas óticas de cada sensor. Interpretando os vetores de distorção, verifica-se que ambas as câmaras apresentam mais distorção radial do que tangencial. E, analisando as matrizes da câmara colorida e infravermelhos, constata-se que a distância focal no eixo v é maior do que a do eixo u, devido ao facto de a imagem de saída ser retangular, aspeto esse já explicado na subsecção 4.1.1. Para além disso, é possível afirmar que o centro ótico da câmara infravermelhos está, aproximadamente, em $(329, 240)$ e o da câmara

colorida em $(313,258)$. Estes valores contrastam com o que seria de esperar teoricamente dada a resolução das câmaras ser 640×480 e, por isso, achando o centro da frame de imagem, ter-se-ia $\frac{(640,480)}{2} = (320,240)$.

$$R = \begin{bmatrix} 9.99975562 \times 10^{-1} & 6.98029296 \times 10^{-3} & 3.22546024 \times 10^{-4} \\ -6.97962474 \times 10^{-3} & 9.99973595 \times 10^{-1} & -2.02806247 \times 10^{-3} \\ -3.36693978 \times 10^{-4} & 2.02576164 \times 10^{-3} & 9.99997914 \times 10^{-1} \end{bmatrix}$$

$$T = \begin{bmatrix} -2.49013081 \times 10^{-2} \\ 8.12286278 \times 10^{-4} \\ 1.28835626 \times 10^{-3} \end{bmatrix}$$

Analisando os parâmetros extrínsecos, verifica-se que ambas as câmaras apresentam praticamente a mesma atitude. Tal facto resulta da matriz de rotação ser composta por valores muito próximos de uma matriz identidade, o que representa inexistência de rotação nos três eixos que compõem os seus referenciais. Os valores de translação evidenciam também outra caraterística de construção do Kinect, pois o valor x do vetor de transformação, em módulo, é muito próximo de $2,5cm$, que é precisamente a distância física que separa os sensores. Por outro lado, os valores em y e z são muito baixos, comprovando que existe apenas translação significativa no eixo x. Todos estes resultados são esperados, pois por inspeção do disposição percebe-se o alinhamento dos sensores. Portanto, considerar estas pequenas variações é o que distingue a precisão dos resultados adquiridos pelo Kinect.

Ao assumir um referencial para cada sensor, é possível perceber qual a câmara referência utilizada nos cálculos dos valores de calibração, retornados pelos métodos da biblioteca OpenCV. De facto, o valor negativo em x evidencia que a referência para os cálculos destes parâmetros extrínsecos é o referencial da câmara de infravermelhos, figura 4.11.

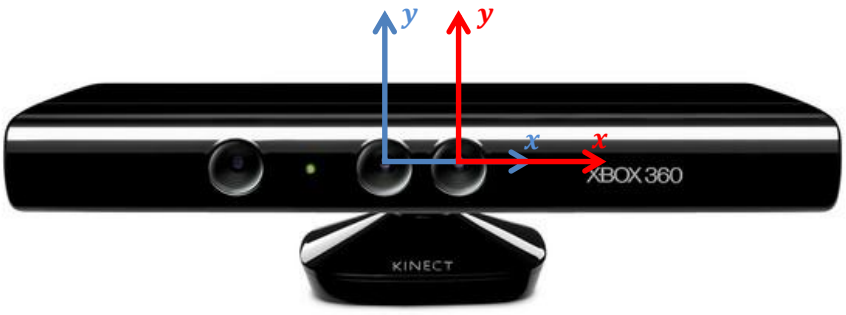


Figura 4.11: Representação dos referenciais da câmara RGB (azul) e IR (vermelho)

Em métodos de calibração mais simplistas, os parâmetros extrínsecos são aproximados tratando a matriz de rotação como uma matriz identidade, e atribuindo o valor de $-0,025$ à componente x do vetor de translação, correspondente à distância entre sensores de $2,5cm$. No caso dos

valores calculados pelo algoritmo diferirem muito destes valores médios, estes servem também para perceber se algo de errado ocorreu durante a calibração.

A qualidade da calibração está diretamente relacionada com a qualidade das imagens capturadas pelo Kinect. Para obter os melhores resultados devem ser seguidas algumas técnicas:

1. Manter o Kinect estático;
2. Fixar o padrão de calibração a um tripé;
3. Utilizar condições de luminosidade adequadas:
    - Realizar todo o processo em ambiente *indoor*;
    - Lâmpadas de halogéneo;
    - Luz solar moderada;
    - Não permitir incidência de luz direta nas câmaras ou no padrão de calibração;
    - Obstruir o emissor de infravermelhos;
4. Fazer rodar o padrão em relação ao Kinect;
5. Ocupar a imagem, o máximo possível, com o padrão de calibração;
6. Capturar um número elevado de pares de imagens (colorida e infravermelhos).

A fixação do padrão de calibração evita oscilações do mesmo, reduzindo a possibilidade de arrastamento nas imagens capturadas. A obstrução do emissor infravermelhos impede a projeção dos pontos de luz provenientes do Kinect. Tal é aconselhável porque os feixes projetados pelo Kinect introduzem irregularidades na imagem capturada pela câmara de infravermelhos. Isto é visível na figura 4.12, onde se verifica que os lados dos quadrados que compõem o xadrez apresentam um aspeto pouco retilíneo, resultante da forma circular dos pontos projetados. Evidentemente, isto afeta a precisão na altura do processamento de imagem para a pesquisa dos cantos.

Os melhores resultados conseguidos com a obstrução do emissor de infravermelhos podem ser constatados comparando a mesma figura 4.12 com a figura 4.10(d). No entanto, ao obstruir o emissor do Kinect, existe a necessidade de realizar a calibração num ambiente em que a luz do meio contenha radiação na gama do infravermelho, de forma a ser captada pelo sensor desse mesmo tipo.

A biblioteca contém um método de cálculo do erro de reprojeção para avaliar os resultados obtidos na calibração. Este valor deve ser o mais próximo possível de zero. O erro é calculado pela norma que separa os pontos obtidos pela reprojeção usando os parâmetros de calibração obtidos, e os pontos resultantes da pesquisa dos cantos (distância euclidiana). O erro final resulta da média aritmética dos erros de todas as imagens de calibração. Utilizando a metodologia de calibração apresentada anteriormente neste subcapítulo, obtém-se facilmente um erro de projeção inferior a 1 pixel.

4.2.4 Coordenadas no Referencial Câmara em Metros

Até esta fase, apenas a distância em profundidade é convertida para metros. Por razões evidentes, interessa também obter as medidas dos outros dois eixos em unidades SI. O método de

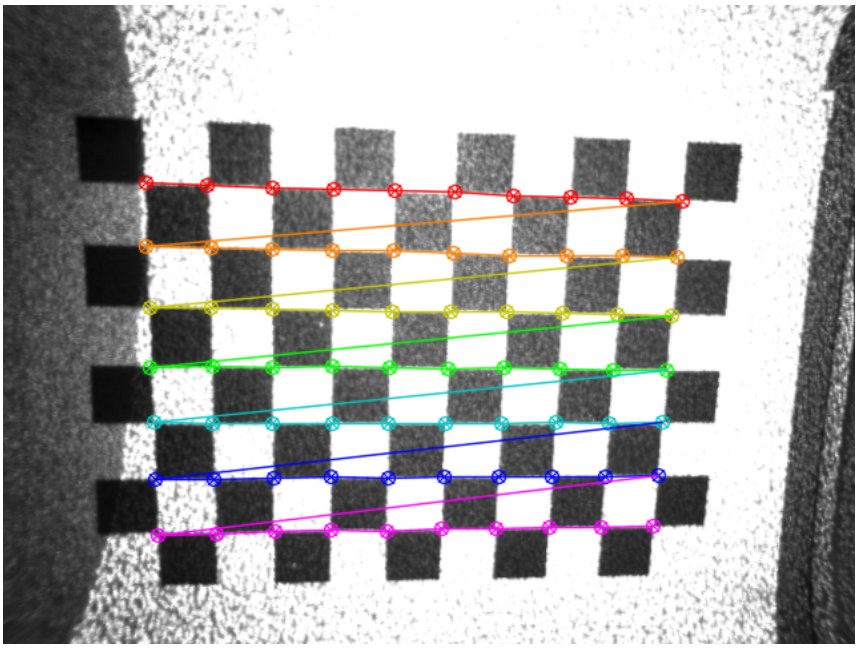


Figura 4.12: Imagem infravermelhos capturada, sem obstrução do emissor de infravermelhos

conversão mais utilizado consiste num modelo de aproximação, em que a distância em metros no plano xOy é proporcional à profundidade e inversamente proporcional à distância focal, equação 4.16. Este modelo é tanto melhor quanto mais pequenas forem as variações em profundidade, dada a sua direta dependência da distância medida em z. A origem das coordenadas no referencial câmara são centradas na imagem, recorrendo aos parâmetros de calibração do centro do sensor da câmara de profundidade (c_x, c_y). [29]

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} \frac{(u-c_x)\times p_m(u,v)}{f_x} \\ \frac{(v-c_y)\times p_m(u,v)}{f_y} \\ p_m(u,v) \end{bmatrix} \tag{4.16}$$

Em que $p_m(u,v)$ representa a profundidade em metros na posição (u,v) da imagem de profundidade. Este valor em unidades SI é calculado pelo método de aproximação matemática apresentado anteriormente, equação 4.14.

Na figura 4.13 é apresentado o referencial tridimensional do Kinect e o método de cálculo de profundidade. Note-se que a profundidade calculada não é a distância euclidiana do objeto ao centro do sensor, ou seja. Isto permite determinar a coordenada z como igualdade direta ao valor de profundidade retornado pelo Kinect. Desta forma, o valor de profundidade deve ser constante ao longo de um plano paralelo ao plano de imagem. A distância euclidiana pode, no entanto, ser facilmente obtida recorrendo a teorema de Pitágoras.

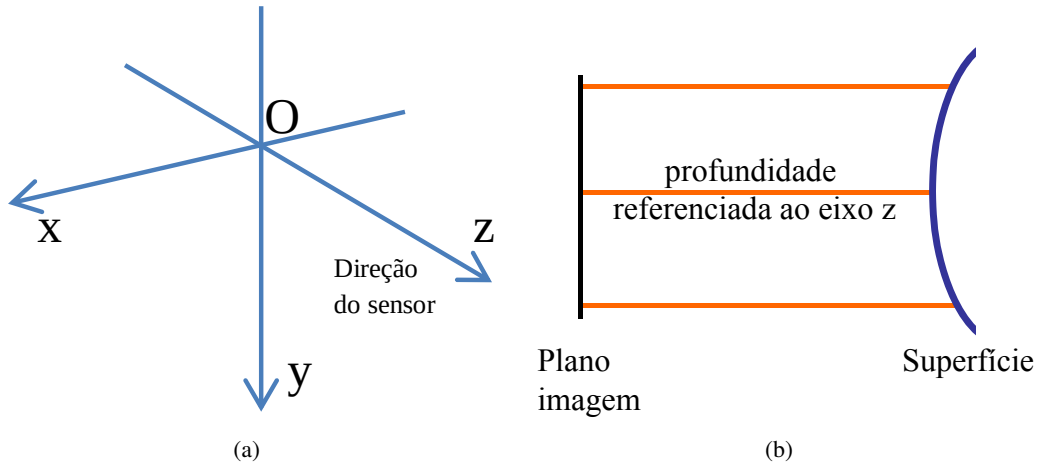


Figura 4.13: Representação das coordenadas câmara: a) Referencial câmara; b) Profundidade ao longo do eixo eixo z

4.2.5 Coordenadas no Referencial Mundo em Metros

Em robótica é normalmente vantajoso, ou até mesmo necessário, tratar os pontos do mundo em função de um referencial geral. Até aqui, os pontos avistados pela câmara têm coordenadas impostas pelo referencial da mesma. Isto cria um problema, dado que, os pontos do mundo têm coordenadas diferentes sempre que se faz mover a câmara. Para além disso, em tarefas desempenhadas por mais do que um sistema em cooperação, é fundamental existir um referencial comum a todos os sistemas. Por exemplo, no caso deste projeto, o facto de se ter um referencial mundo comum à câmara e ao robô faz com que uma determinada coordenada mapeie o mesmo ponto no mundo para ambos os sistemas. Este entendimento mantém-se mesmo que as suas posições e/ou orientações sejam alteradas.

A configuração tipo em que a câmara encara a área de trabalho é apresentada na figura 4.14. Comparando a posição e orientação de ambos os referenciais, verifica-se que existe uma distância entre as suas origens, e que os eixos que os constituem estão rodados.

Assim sendo, de modo a referenciar os pontos da câmara, R_C, em função do referencial mundo, R_M, é necessário aplicar-lhes uma translação e rotação. Uma transformação homogénea deste tipo pode ser realizada fazendo mover inicialmente os pontos de R_C para R_M e, posteriormente, aplicar-lhes uma rotação de modo a ficarem alinhados com os eixos que constituem R_M, equação 4.17.

$$\mathbf{p_M} = (\mathbf{p_C} - \mathbf{t}) \times \mathbf{R} \Leftrightarrow \begin{bmatrix} x_M \\ y_M \\ z_M \end{bmatrix}^{\mathrm{T}} = \left(\begin{bmatrix} x_C \\ y_C \\ z_C \end{bmatrix}^{\mathrm{T}} - \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}^{\mathrm{T}} \right) \times \begin{bmatrix} r_{1,1} & r_{1,2} & r_{1,3} \\ r_{2,1} & r_{2,2} & r_{2,3} \\ r_{3,1} & r_{3,2} & r_{3,3} \end{bmatrix} \tag{4.17}$$

O algoritmo de Kabsch calcula a matriz ótima de rotação que minimiza o RMSD (*root mean squared deviation*) entre dois conjuntos de pontos, e tem por base o cálculo do SVD (*singular value decomposition*) da matriz de covariância dos pares de pontos. Este algoritmo é normalmente

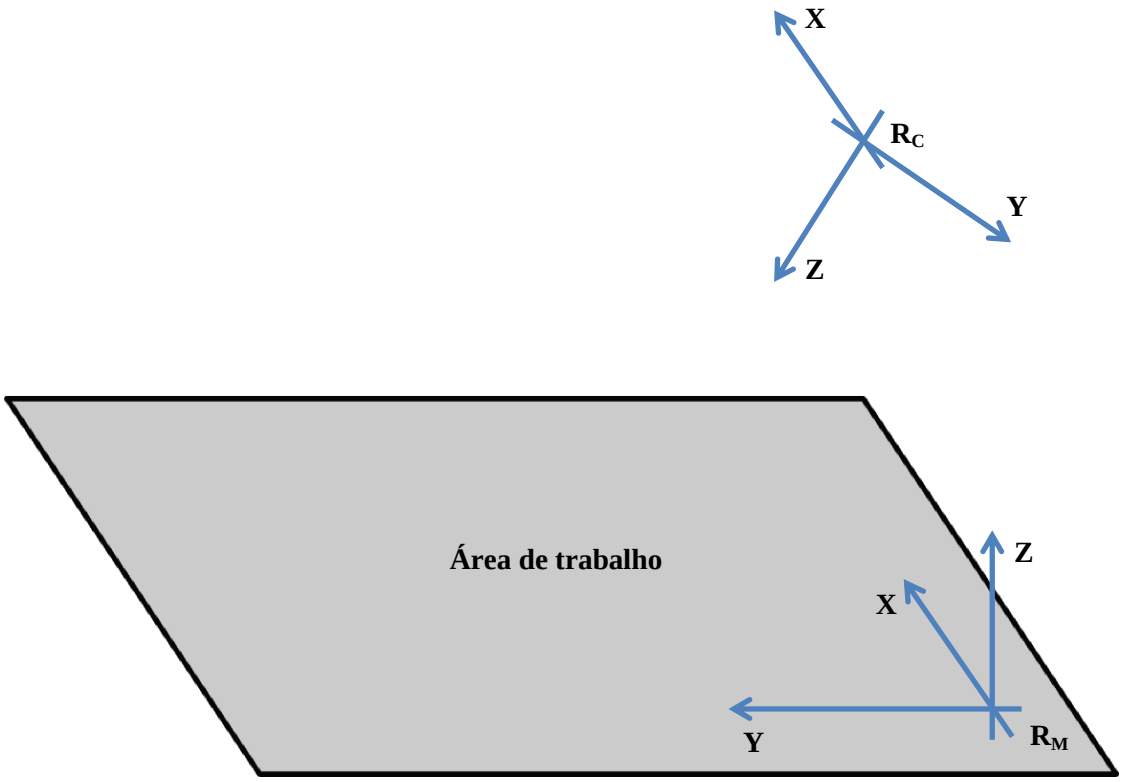


Figura 4.14: Configuração dos referenciais da área de trabalho e câmara

utilizado em química para comparação de duas estruturas da mesma molécula para decidir sobre a sua similaridade. No contexto deste projeto, este algoritmo permite calcular a matriz de rotação que transforma os pontos do referencial R_C em R_M. Para tal, é necessário construir dois conjuntos de pontos, isto é, duas matrizes $N \times 3$, em que cada ponto do conjunto do referencial R_C tem um ponto associado no referencial R_M.

A computação deste algoritmo devolve apenas a matriz ótima de rotação. Por inspeção da equação 4.17 verifica-se que a correta transformação dos pontos necessita de uma translação. Desta forma, é necessário calcular a distância entre os dois referenciais por outro método. Na realidade, a translação é facilmente calculada, bastando para tal verificar que coordenadas em R_C tem o ponto que se pretende como origem do referencial R_M. Os valores dessas coordenadas constituem o vetor translação $\mathbf{t}$.

Este processo é implementado na aplicação desenvolvida. São assinaladas posições na chapa com posições conhecidas e bem definidas. O utilizador seleciona essas posições e são guardadas as coordenadas no referencial câmara correspondentes. Assim se constroem as duas matrizes que constituem o conjunto de pares de pontos. No final, o algoritmo calcula a matriz $\mathbf{R}$, ficando apto a apresentar qualquer ponto em coordenadas mundo computando a equação 4.17, em que $\mathbf{t}$ é calculado pelo método enunciado no parágrafo anterior.

Finalmente, como os pontos no mundo resultam de uma transformação algébrica dos pontos do referencial câmara, a unidade SI do referencial inicial mantém-se.

4.2.6 Testes de Performance

Com o objetivo de compreender melhor o funcionamento e limitações do Kinect, foram realizados e analisados alguns testes em condições específicas. Estes testes foram ajustados às necessidades do problema.

Inicialmente é importante perceber a evolução da precisão do cálculo de profundidade do Kinect com o aumento da distância. Com isto pretende-se perceber em que zona de funcionamento deve ser colocado o dispositivo, de forma a obter os melhores resultados.

Para além disso, o facto de se estar a analisar uma chapa com recortes torna crucial perceber como é que este dispositivo se comporta no cálculo de profundidade nas fronteiras do material - entenda-se por fronteiras, mudanças súbitas de profundidade. Note-se que é uma situação que ocorre com frequência, dado que as peças, ao deslocarem-se da sua posição inicial, traduzem-se em mais situações fronteira.

Finalmente, é também relevante entender o comportamento do Kinect no cálculo de distâncias a um objeto exclusivamente plano. Devido ao facto de a chapa de trabalho ser maioritariamente plana, pretende-se avaliar o comportamento do dispositivo em situações deste tipo.

Estes testes foram implementados na aplicação desenvolvida. A ideia é permitir que futuros projetos, que usem câmaras de profundidade possam realizar testes de *performance*, com a intenção de se conhecer melhor o funcionamento do dispositivo.

4.2.6.1 Evolução da Precisão com o Aumento da Distância

Como apresentado anteriormente neste capítulo, o erro de cálculo de profundidade aumenta proporcionalmente à distância a que se encontra o objeto do Kinect. Este facto deve-se a caraterísticas inerentes à construção e método de cálculo do Kinect. De maneira a conseguir tirar proveito da melhor gama de funcionamento do dispositivo, é necessário perceber a evolução e respetiva magnitude do erro. Para isso, foi realizado um simples teste que consistiu na colocação de um objeto plano em frente ao Kinect e respetiva extração da medida de profundidade do dispositivo ao centro do plano. Estes testes foram realizados dentro de um edifício com condições normais de luminosidade artificial e solar.

Na aplicação desenvolvida, o utilizador clica na imagem de profundidade, nos quatro cantos correspondentes ao plano, e a aplicação retorna a profundidade ao centro da forma delimitada pelas posições introduzidos pelo utilizador. Este processo foi repetido a cada incremento de $0.25m$ de distância, entre o objeto e o Kinect, desde o momento em que este consegue construir um mapa de profundidade até ao ponto em que deixa de o conseguir fazer. Na tabela 4.2 são apresentados os resultados obtidos.

Representando os resultados obtidos da tabela 4.2, verifica-se que a forma do gráfico da figura 4.15 é muito semelhante ao da figura 4.8, que representa a função matemática que converte

Mundo	<0,5	0,50	0,75	1,0	1,25	1,50	1,75	2,00
Kinect	n.d	0,4977	0,7489	0,9989	1,2465	1,4985	1,75544	1,9944
Mundo	2,25	2,50	2,75	3,00	3,25	3,50	3,75	4,00
Kinect	2,2509	2,5087	2,7661	3,0289	3,2536	3,5143	3,7792	4,0403
Mundo	4,25	4,50	4,75	5,00	5,25	5,50	5,75	6,00
Kinect	4,2870	4,5658	4,8833	5,0959	5,3279	5,5819	5,8614	6,1704

Tabela 4.2: Tabela comparativa das medidas reais com a profundidade calculada pelo Kinect

os dados adquiridos pelo Kinect para metros. Este resultado é esperado e demonstra também um possível método de encontrar uma função que converta os dados do Kinect para metros - colecionar uma série de dados com distâncias conhecidas e respetivos valores de profundidade calculados e, por fim, encontrar uma função que aproxime a evolução das distâncias obtidas em função da medida real.

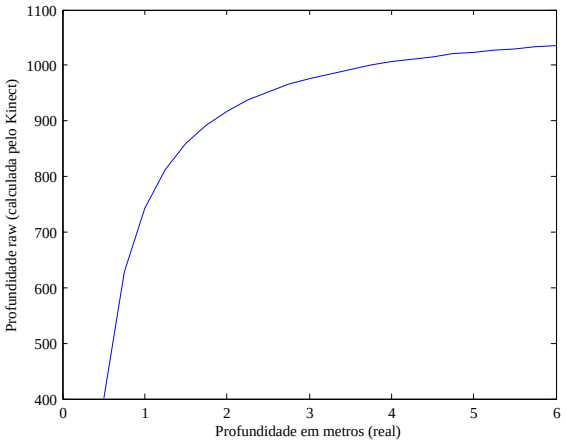
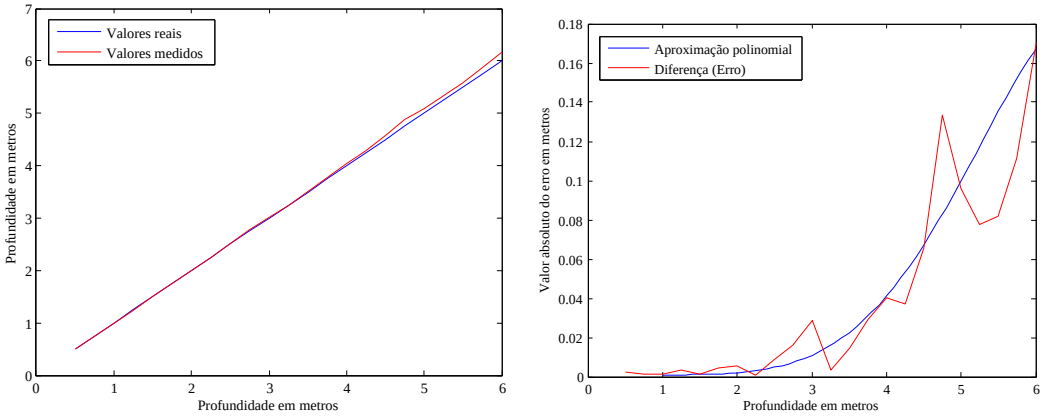


Figura 4.15: Representação gráfica dos valores calculados pelo Kinect da tabela 4.2

De acordo com o que é apresentado na tabela 4.2, o Kinect não é capaz de calcular distâncias para objetos a uma profundidade inferior a $0.50m$. Além do mais, para valores superiores a $6m$ este vai perdendo gradualmente informação até ficar completamente impossibilitado de calcular distâncias. O aumento do erro em função do aumento da distância do Kinect ao objeto em questão é evidente na figura 4.16. Note-se que existe irregularidade nas medições efetuadas, dado que o erro não sobe constantemente. No entanto, é muito difícil garantir que o dispositivo esteja exatamente na distância a que se pretende - e essa incerteza cresce com o aumento da distância ao objeto. Para além disso, é também muito complicado garantir que este está posicionado de forma perfeitamente paralela ao plano à sua frente. No entanto, por aproximação por uma função polinomial de grau 6, percebe-se a tendência do erro com o aumento de profundidade.

Em particular, verifica-se que o erro tem um crescimento mais evidente junto dos $3m$ de profundidade. De notar que este dado vai de encontro ao apresentado anteriormente, aquando da explicação da divisão do Kinect em 3 zonas de diferente precisão, onde foi dito que a terceira e



(a) (b)

Figura 4.16: Comparação entre as distâncias reais e medidas: a) Diferença direta das medições em valor absoluto; b) Comparação com aproximação polinomial

última zona acaba nos $3.5m$. Os resultados obtidos mostram que, de facto, junto desse valor a dinâmica do erro é especialmente abrupta, tendendo para valores que podem exceder os $10cm$. É por isto que muitos autores consideram que o Kinect consegue apenas medir distâncias até $3m$, aproximadamente, pois este a partir deste valor, não oferece precisão suficiente para ser usado na maioria das aplicações minimamente exigentes. Outros consideram a gama de medição do Kinect até ao ponto em que este deixa de conseguir medir por completo, independentemente do erro. Nestes casos, os valores podem atingir cerca de $10m$.

4.2.6.2 Erro em Situações Fronteira

É sabido que o Kinect tem dificuldades em calcular distâncias em situações fronteira, isto é, uma situação em que existe uma mudança súbita de profundidade. Com o intuito de perceber também a dinâmica do erro no cálculo de profundidade neste tipo de situações, foi realizado um teste em que tal foi provocado. A análise consistiu em colocar um plano com cerca de $10cm$ de profundidade encostado a uma parede perfeitamente plana. Foram realizadas 50 medições consecutivas, a uma distância de $1,25m$, ao longo de uma linha que atravessa toda a imagem de profundidade. A figura 4.17 ilustra o teste realizado, em que o fundo cinza representa a parede, a zona azul o plano com espessura que induz mudanças abruptas de profundidade e, por fim, a linha a vermelho representa a zona onde foram realizadas as medições.

Com este teste, a medição gerada é apenas de uma dimensão, pois é realizado numa imagem ao longo do seu comprimento, em apenas um pixel de altura. Assim, numa imagem com tamanho 480×640, a saída terá no máximo 1×640 dados. A escolha da linha é feita pelo utilizador na aplicação desenvolvida e no teste apresentado, a linha desenhada ocupa apenas 436 posições de comprimento, pelo que apenas nesses valores existem dados de profundidade, figura 4.18.

A figura 4.18 apresenta uma amostra e a média de todas as medições efetuadas. Aqui, pretende-se mostrar a consistência das medições, sendo já possível verificar oscilações nas zonas onde existe

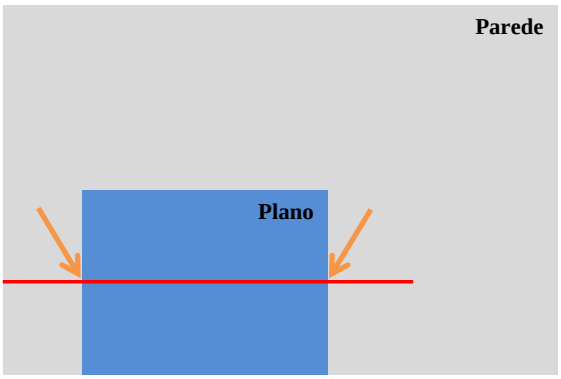
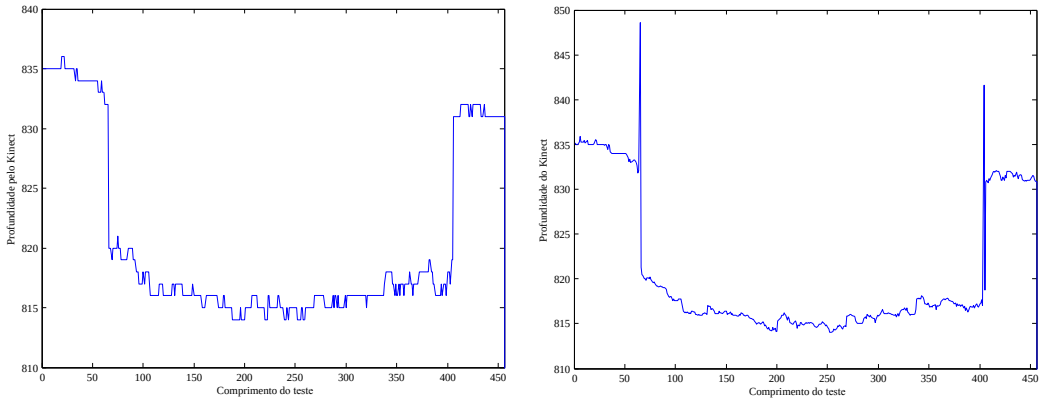


Figura 4.17: Ilustração do teste realizado para medições em situações fronteira, com zonas de interesse assinaladas

mudança abrupta de profundidade. Na figura 4.18(a) podemos confirmar as medidas atrás mencionadas. Por observação do gráfico verifica-se que a região com profundidade mais elevada tem o valor de, aproximadamente, $835 \Rightarrow 1,3584m$. Enquanto que a região caraterizada por uma profundidade inferior tem um valor de, aproximadamente, $815 \Rightarrow 1,2601m$. A conversão para metros, é realizada mais uma vez com recurso à equação 4.14. Desta forma, confirma-se a profundidade do plano colocado à frente da parede de $\approx 10cm$, resultante da subtração das duas distâncias; e também a distância do Kinect ao plano de $\approx 1,25m$, resultante da distância não nula mais pequena.



(a) Amostra de uma das 50 medições

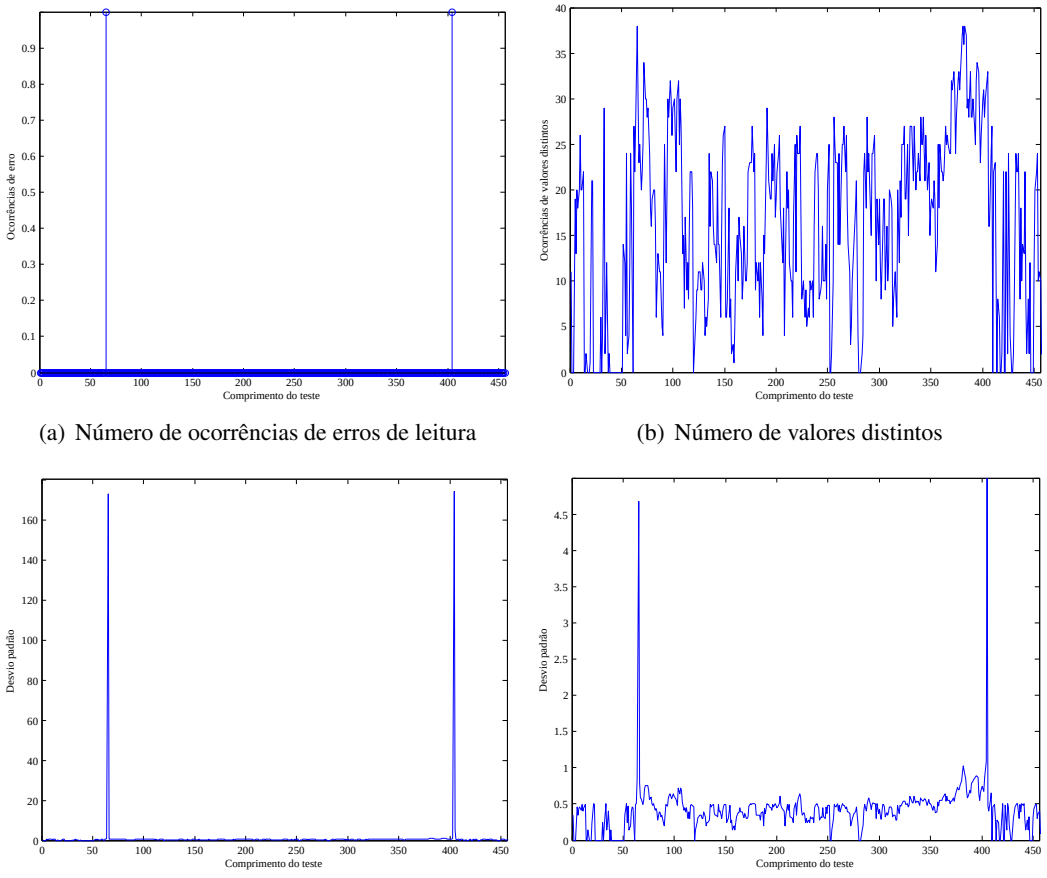
(b) Média das 50 medições

Figura 4.18: Gráficos representativos das distâncias calculadas no teste de situações fronteira

Em situações fronteira o Kinect pode-se comportar de três maneiras distintas para um determinado ponto no mundo:

- Não consegue medir e, por isso, retorna o valor de 2047;
- Consegue medir mas os resultados são, na sua maioria, consecutivamente diferentes;
- Uma mistura das situações anteriores, em que oscila entre medições sucedidas e falhadas;

Desta forma, é importante perceber com que frequência é que esta alternância de valores acontece e, mais importante que isso, que magnitude tem a diferença entre medições para um determinado ponto. Para a primeira situação, o gráfico da figura 4.19(b) demonstra que existem variações ao longo de toda a linha de teste. No entanto, é também possível verificar que os dois máximos do gráfico encontram-se coincidentes com as fronteiras do plano. Por isso, conclui-se que a existências de fronteiras induz um maior número de varições nas medições do Kinect. Para a segunda situação, o gráfico da figura 4.19(c) evidencia que, para além de ocorrerem variações nas medições retornadas pelo Kinect, estas variam com maior magnitude. No entanto, testando o número de vezes que ocorrem erros de leitura, verifica-se, pelo gráfico da figura 4.19(a), que curiosamente em duas situações (uma em cada fronteira) o Kinect não foi capaz de calcular a profundidade. Como o valor de retorno é elevado comparado com um valor médio de leitura para este caso, convém ignorar estas leituras erróneas com a intenção de tornar o desvio padrão mais compreensível e verdadeiro. Tal é apresentado no gráfico da figura 4.19(d), onde ainda assim se verifica que as variações são maiores em magnitude nas fronteiras, mesmo ignorando os casos em que o Kinect não é capaz de efetuar o cálculo da distância.



(a) Número de ocorrências de erros de leitura

(b) Número de valores distintos

(c) Desvio padrão

(d) Desvio padrão, ignorando erros de leitura

Figura 4.19: Gráficos representativos da variação no cálculo de profundidade nas 50 medições efetuadas

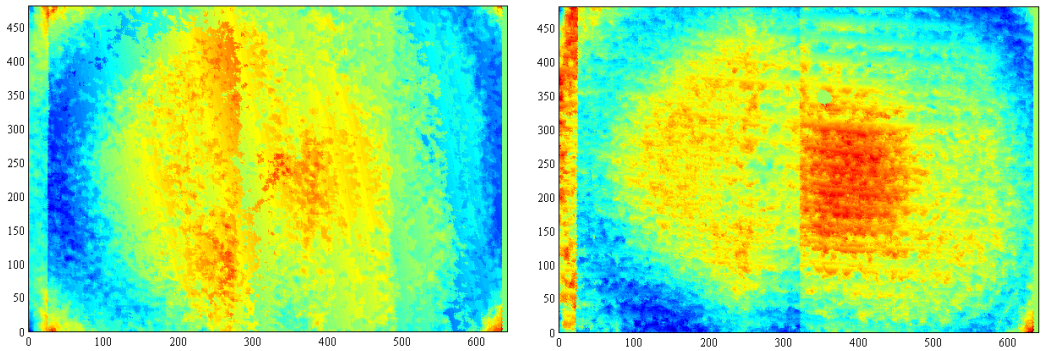
Em suma, o Kinect é sensível à existência de fronteiras no material a ser mapeado. Esta sensibilidade, manifesta-se por uma maior oscilação no cálculo de profundidade quer em frequência, quer em magnitude; podendo em alguns casos nem ser capaz de efetuar o cálculo, originando falhas de medição que resultam em valores de profundidade nulo.

4.2.6.3 Profundidade em Planos Perfeitos

Este teste consiste na determinação da consistência de medições em profundidade ao longo de todo o seu ângulo de visão. Isto é realizado ocupando toda a área de mapeamento do Kinect com um plano perfeito e extrair todos os 640×480 valores resultantes da medição efetuada. Não é realizada conversão para unidades SI nem qualquer tipo de calibração, de modo aos resultados dependerem apenas da performance do Kinect. Este teste foi repetido a várias distâncias por forma a perceber, não só a possível existência de erro de medição, mas também a sua dinâmica em função da profundidade ao plano. Como explicado anteriormente, é extremamente complicado perceber a orientação do Kinect em relação ao plano para o qual ele aponta. Desta forma, as medições foram efetuadas colocando o Kinect minimamente alinhado com o plano da parede e no tratamento dos dados resultantes da medição é efetuada uma aproximação por um plano. Posteriormente, verifica-se a orientação do plano aproximado relativamente a um plano referência, por forma a aplicar-lhe uma rotação de modo a anular o desalinhamento entre eles. Finalmente, o plano é centrado em zero para facilitar a análise de resultados. Este algoritmo foi desenvolvido em Matlab, e para facilitar a compreensão dos resultados recorreu-se a representação por *mesh*. Este tipo de apresentação é normalmente usado para dados tridimensionais, caraterizando-se por colorir de diferentes cores, valores de magnitudes diferentes. De modo a facilitar leitura dos dados estes são apenas apresentados em 2D, mantendo no entanto as cores originais da *mesh* e, por isso, a informação tridimensional não é perdida, figura 4.20.

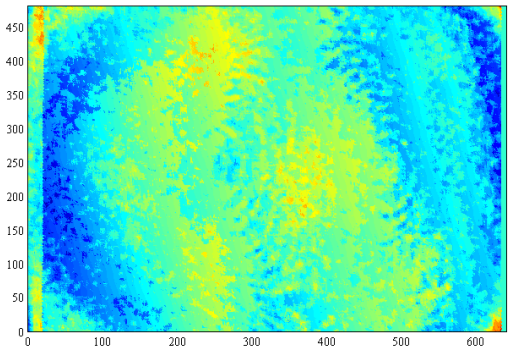
Analisando todas as imagens da figura 4.20 é possível identificar diferentes cores em cada imagem resultante do mapeamento feito pelo Kinect. Isto não vai de encontro ao que seria de esperar no mapeamento de um plano, ou seja, uma cor uniforme ao longo de todo o seu espaço bidimensional. Para além disso, verifica-se um comportamento variável em função da distância. No entanto, este apresenta uma certa uniformidade na maneira como as cores estão distribuídas nas imagens, isto é, as cores estão distribuídas como circunferências concêntricas. Desta, forma, é possível concluir que o Kinect mapeia os planos com alguma concavidade e convexidade. Para análise das imagens resultantes da representação dos dados em *mesh* é preciso entender que os valores em profundidade são representados numa gama de cores frias a cores quentes. Desta forma, os valores atribuídos com cor azulada representam distâncias mais curtas e os que apresentam uma cor avermelhada representam um profundidade mais elevada. Com este pressuposto, constata-se que entre as medições realizadas, o Kinect mapeia o plano com uma forma concava para distâncias até $1,019m$, figuras 4.20(a) a 4.20(d). A partir da medição efetuada a uma distância de $1,407m$ os planos resultantes apresentam uma forma convexa, figuras 4.20(e) e 4.20(f).

Em suma, o mapeamento de planos em profundidade realizado pelo Kinect apresenta uma resposta que, para distâncias crescentes, carateriza-se por ser inicialmente concava, passando a

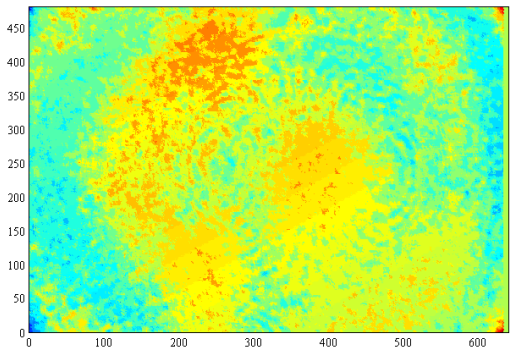


(a) $0,611m$

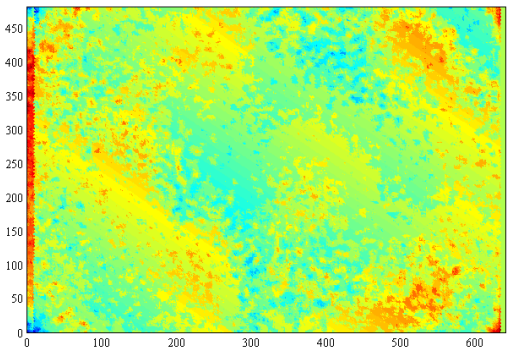
(b) $0,816m$



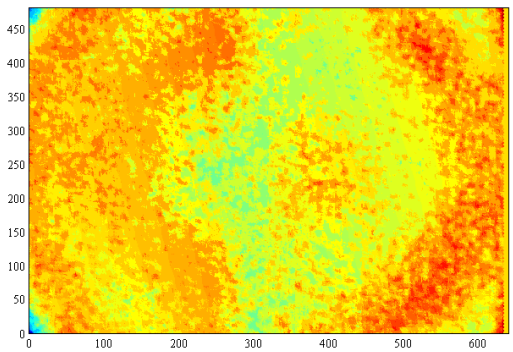
(c) $0,860m$



(d) $1,019m$



(e) $1,407m$



(f) $1,669m$

Figura 4.20: Mapa de profundidade de um plano perfeito a diferentes distâncias: cores quentes → mais distante, cores frias → mais próximo

convexa para uma distância entre $1,019m$ e $1,407m$. É por isso legítimo afirmar que algures nesta gama o dispositivo é capaz de mapear corretamente um plano, sem apresentar este tipo de deformações. De relembrar que a arquitetura implementada para a concretização deste projeto, incluiu fixar o Kinect a uma altura em que toda a chapa lhe fosse visível. Essa altura é muito próxima do $1,30m$, enquadrando-se na gama anteriormente mencionada. Portanto, é de prever que a chapa seja mapeada com uma forma muito próxima de um plano, podendo, no entanto, apresentar uma ligeira convexidade dada a proximidade desta altura de valores onde este tipo de deformação se faz notar.

4.2.7 Profundidade e Cor em Simultâneo

A importância da calibração num sistema deste género está na possibilidade de ter disponível informação sobre profundidade e cor real do meio numa só imagem. Até aqui esta informação era separada em duas imagens:

- a imagem de profundidade é normalmente representada com recurso a uma escala de cores em função da distância a que se encontra o objeto, figura 4.22(a).
- a imagem colorida apresenta informação em função da cor do mundo (câmara dita normal), figura 4.22(b).

Com recurso aos parâmetros extrapolados pela calibração, é possível conseguir-se numa única imagem (para cada pixel) informação proveniente de ambos os sensores, nomeadamente, cor (RGB) e distância tridimensional (x, y e z) em unidades SI.

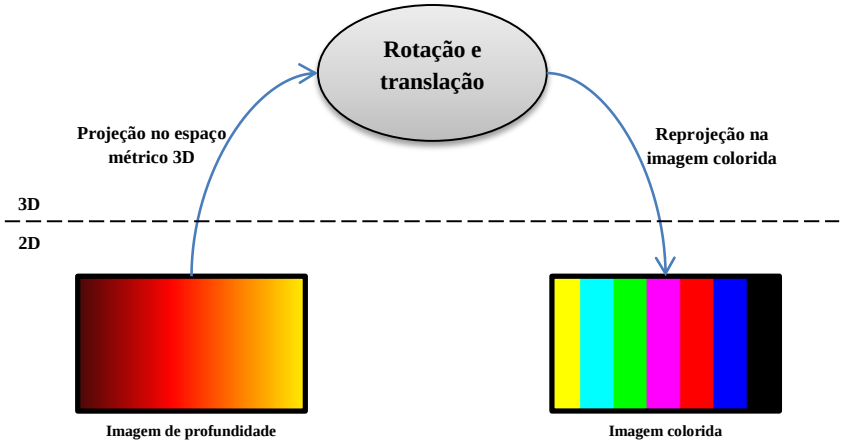


Figura 4.21: Diagrama ilustrativo do método da coloração da imagem de profundidade com cores reais do meio

A figura 4.21 pretende retratar os passos necessários para atingir esse objetivo. A projeção no espaço métrico 3D foi já explicada na subsecção 4.2.4. Reescrevendo a equação 4.16 temos:

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} (u^{IR} - c_x^{IR}) \times p_m(u,v)/f_x^{IR} \\ (v^{IR} - c_y^{IR}) \times p_m(u,v)/f_y^{IR} \\ p_m(u,v) \end{bmatrix} \tag{4.18}$$

Neste ponto, as coordenadas estão referenciadas à câmara de infravermelhos (IR). Estas têm de ser rodadas e transladadas por forma a ficarem orientadas e alinhadas com a câmara colorida (RGB). Para tal, são usados os parâmetros extrínsecos da calibração (matrizes de rotação **R**, e

vetor de translação **T**), equação 4.19. Com este cálculo matricial, as coordenadas transformadas estão ainda no espaço métrico 3D.

$$\begin{bmatrix} x'_c \\ y'_c \\ z'_c \end{bmatrix} = \mathbf{R} \times \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} + \mathbf{T} \tag{4.19}$$

De modo a adicionar a informação de cor é necessário reprojetar as coordenadas tridimensionais no espaço bidimensional. Desta vez, a projeção é realizada na imagem colorida, pelo que têm de ser usados os parâmetros intrínsecos desta câmara, como descrito na equação 4.20.

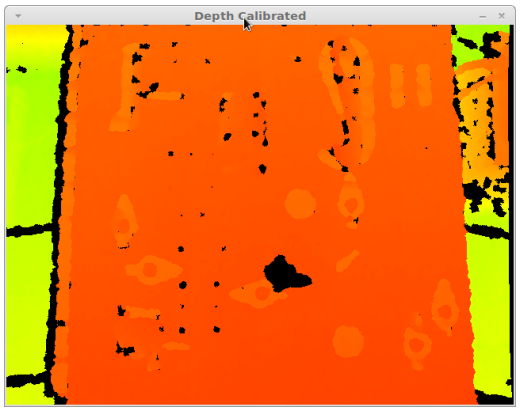
$$\begin{bmatrix} u^{RGB} \\ v^{RGB} \end{bmatrix} = \begin{bmatrix} (x'_c \times f_x^{RGB}/z'_c) + c_x^{RGB} \\ (y'_c \times f_y^{RGB}/z'_c) + c_y^{RGB} \end{bmatrix} \tag{4.20}$$

4.2.7.1 Profundidade como Imagem Colorida (2D)

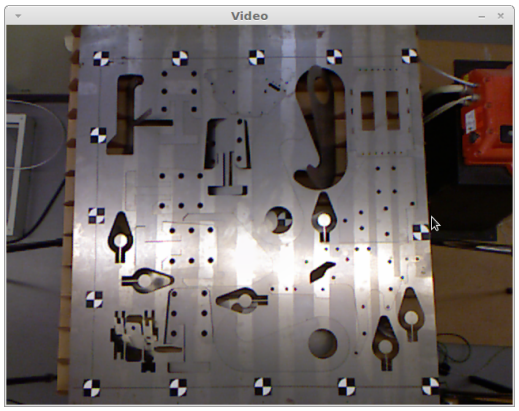
Desta projeção resulta uma imagem classificada na literatura como RGB-D (RGB-*Depth*), figura 4.22(c). Esta imagem tem a caraterística de em formato bidimensional conter informação acerca da distância estando, no entanto, colorida com de acordo com o que existe no meio observado. Desta forma, temos numa única imagem o conjunto de toda a informação relevante que um dispositivo como o Kinect pode oferecer. No entanto, este método de apresentação dos dados tem a desvantagem de necessitar que seja especificado o pixel do qual se pretende obter informação acerca da profundidade. Pois, ao contrário da imagem de profundidade, esta está colorida de acordo com as cores que existem no meio, e não de acordo com a distância a que os objetos se encontram do Kinect. Na realidade, esta conjugação dos dados de profundidade e cor, tem a particularidade de permitir uma mais fácil visualização das zonas em que ocorre erro de medição.

Na figura 4.22 pode visualizar-se o conjunto da informação disponível, com recurso a um único dispositivo. Analisando a figura resultante do mapeamento verifica-se que as ocorrências de erro de leitura, assinaladas a cor preta, representadas na imagem de profundidade estão coincidentes com as da imagem de profundidade. De relembrar que as reprojeções do método de mapeamento recorrem a parâmetros de calibração. Desta forma, a correta sobreposição da informação das duas imagens (profundidade e cor) depende única e simplesmente da calibração. A imagem mapeada, é por isso, uma boa forma de testar os parâmetros de calibração.

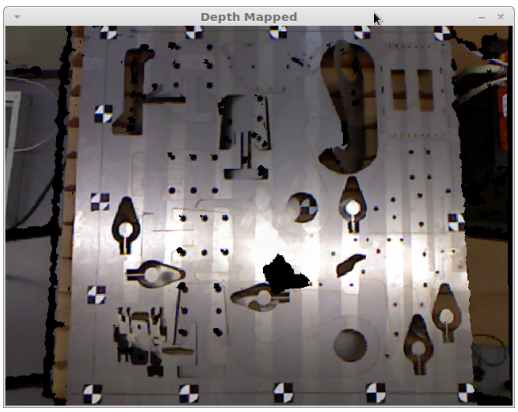
Note-se que a imagem mapeada não poderia resultar da sobreposição direta das imagens de profundidade e colorida, ou seja, sem realizar as operações ilustradas na figura 4.21. Apesar de se ter verificado que os sensores IR e RGB estão praticamente alinhados (aquando da calibração dos parâmetros extrínsecos), estes estão separados por $2,5cm$, o que faz com que, cada um encare o meio de modo ligeiramente diferente. Para além disso, os sensores têm caraterísticas distintas, como por exemplo, o ângulo de visão, que na câmara de infravermelhos é mais curto que na câmara colorida. Isto pode ser facilmente constatado, comparando a região superior e lateral direita das figuras 4.22(a) e 4.22(b), onde se verifica que a imagem colorida abrange mais área de trabalho do que a de profundidade. Desta forma, a imagem resultante do mapeamento é dominada pelas



(a) Imagem de profundidade



(b) Imagem colorida



(c) Imagem resultante do mapeamento da imagem em profundidade com cores da imagem colorida

Figura 4.22: Conjunto de imagens que constituem o mapeamento da imagem de profundidade

caraterísticas mais restritas da imagem de profundidade. Por isso, a imagem mapeada apresenta exatamente a mesma área de trabalho que a imagem proveniente do sensor de infravermelhos, mas colorida de acordo com a luminosidade do meio, comparar figura 4.22(a) com figura 4.22(c).

4.2.7.2 Profundidade como Nuvem de Pontos (3D)

Como foi dito anteriormente, neste subcapítulo, o resultado do mapeamento da imagem de profundidade com cores da imagem RGB, tem a desvantagem de não transparecer a profundidade de todos os pixeis. Com o intuito de ultrapassar esse defeito, opta-se normalmente por apresentar o resultado com uma nuvem de pontos, figura 4.23. O princípio é o mesmo do método anterior, ilustrado na figura 4.21, e explicado matematicamente nas equações 4.18-4.20.

Na aplicação desenvolvida foi acrescentada esta funcionalidade, recorrendo à biblioteca 3D *opensource* GLScene, para o processamento gráfico. Inicialmente é necessário escalar os pontos em coordenadas tridimensionais, de modo a serem compreendidos pelos referenciais do *software*. Posto isto, tem-se os pontos no espaço referenciados à posição do dispositivo no espaço, mas sem

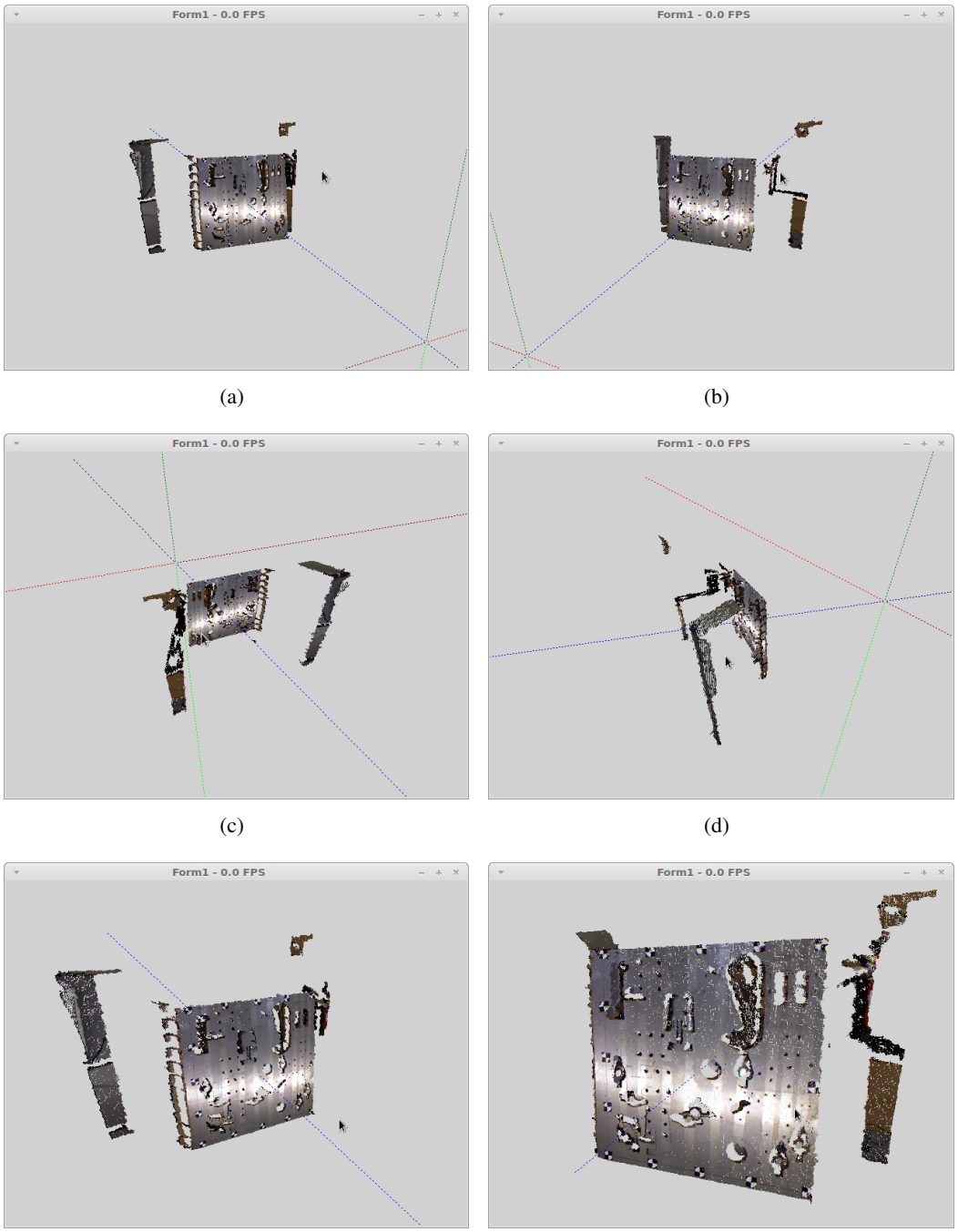


Figura 4.23: Conjunto de seis perspetivas distintas da nuvem de pontos gerada

cor atribuída. Aqui, é mais uma vez necessário ter em conta os parâmetros intrínsecos e extrínsecos das câmaras, para fazer corresponder corretamente ao pixel do mundo a sua cor respetiva.

Este é o ponto base para a grande maioria das aplicações que recorrem a visão computacional 3D. Nesta fase tem-se todas as imagens possíveis de ser adquiridas pelo Kinect, nas mais diversas formas. Estabelece-se assim, uma aplicação com os requisitos mínimos para o sucesso no cumpri-

mentos dos objetivos de um projeto dependente de visão que, resumidamente incluem a aquisição, calibração e apresentação dos dados.

4.2.7.3 Erros na Sobreposição de Imagens de Profundidade e Cor

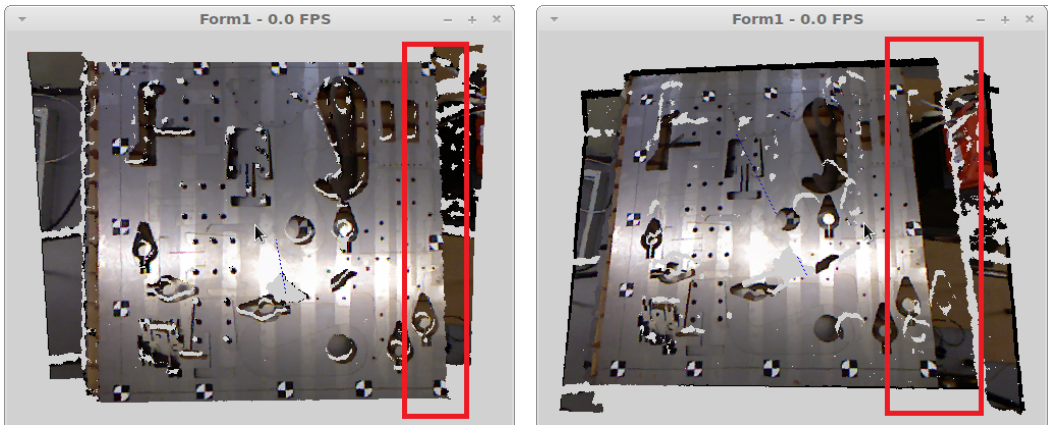
As imagens de profundidade e colorida têm origem em sensores distintos, separados horizontalmente, o que origina discrepâncias na altura do mapeamento colorido da imagem de profundidade. Esse desvio ocorre quando os parâmetros extrínsecos da calibração não são aplicados no espaço tridimensional. A figura 4.24 pretende demonstrar as diferenças originadas quando se ignora a rotação e translação - que servem para alinhar os referenciais de ambos os sensores. Por forma a facilitar a análise das figuras foram assinaladas as regiões onde as diferenças se manifestam com maior intensidade. Ao comparar a figura 4.24(a) com a figura 4.24(b) verifica-se que existe uma ligeira diferença no mapeamento da profundidade. O desvio é percetível apenas no eixo horizontal, o que é concordante com os parâmetros extrínsecos. Recorde-se que, como foi visto na subsecção 4.2.3, apenas a componente x do vetor de translação tem significado. Pode-se concluir que os resultados do algoritmo de mapeamento são melhorados quando se considera a atitude relativa dos sensores. Em particular, note-se que na zona demarcada à esquerda da figura 4.24(b) os valores de profundidade sobrepõem um pouco a imagem colorida. Estes casos, apesar de menos percetíveis, ocorrem em toda a área da imagem nos casos de erro de leitura de profundidade. Ao comparar a figura 4.24(b) nessa mesma região, verifica-se que tal sobreposição não acontece. O desalinhamento das imagens de profundidade e cor induz erros importantes, porque um pixel com determinada cor não tem associada a posição tridimensional correta.

(a) Sobreposição correta: com rotação e translação

(b) Sobreposição errada: sem rotação nem translação

Figura 4.24: Zonas assinaladas para comparação dos resultados obtidos por mapeamento

A figura 4.25 ilustra o erro na sobreposição direta da imagem de profundidade com a imagem colorida. Pretende-se com isto perceber também até que ponto o ângulo de visão é diferente nas duas câmaras. A zona assinalada na figura 4.25(b) mostra que existe uma parte da chapa que foi mapeada com cores que não lhe correspondem. O facto de os resultados serem apresentados em nuvens de pontos tem a vantagem de se perceber onde termina a chapa, por estar a uma profundidade diferente do resto do meio em seu redor. Isto acontece porque, como dito anteriormente neste capítulo, o sensor RGB tem um ângulo de visão maior do que o sensor IR. Assim, ao sobrepor as imagens a informação colorida tem de ser comprimida por forma a encaixar com a imagem em profundidade. Isto, dá origem a que os valores fiquem mal sobrepostos. Analisando mais especificamente estes resultados, verifica-se que existe diferença no ângulo de visão tanto na horizontal como na vertical. No entanto, na figura 4.25(b) é mais evidente o desvio da componente horizonta. Sabendo que o Kinect se encontra a uma distância de $1,30m$ da chapa, e que esta tem cerca de $1,20m$ de lado, pode chegar-se à conclusão que, com estas distâncias, o erro de sobreposição direta é cerca de $20cm$ no eixo horizontal. Evidentemente, este erro torna inviável a maior parte das aplicações baseadas em visão computacional. É, por isso, fundamental considerar todas as caraterísticas internas e externas dos sensores de modo a conseguir-se um correto mapeamento, como aquele representado na figura 4.25(a).



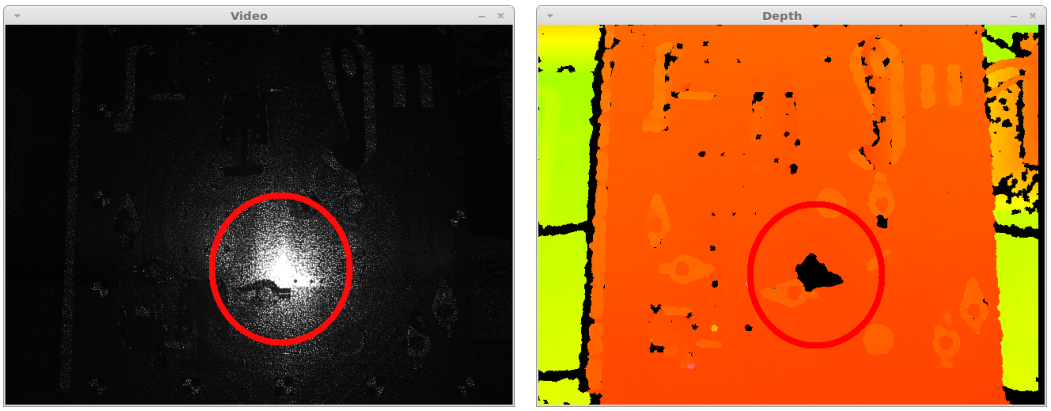
(a) Aplicação do algoritmo de reprojeções (b) Sobreposição direta de imagens

Figura 4.25: Zonas assinaladas para comparação dos resultados obtidos por construção de nuvem de pontos

4.2.8 Limitações no Ângulo de Incidência do Sensor de Profundidade

O material da chapa é brilhante e altamente refletor o que prejudica a performance do Kinect em determinadas situações. Na figura 4.26 é representada a consequência de uma incidência na chapa quase perpendicular dos feixes emitidos (posição 1 da figura 4.27). Pela imagem infravermelhos do padrão emitido na chapa é possível constatar uma grande concentração de luz numa região central da superfície, figura 4.26(a). Comparando com o mapa de profundidade gerado verifica-se que isto resulta em erro de medição - mapeamento com cor preta nessa região,

figura 4.26(b). Obviamente esta situação é problemática, pois dá origem a uma zona "cega"na área de trabalho.



(a) Imagem infravermelhos (b) Imagem de profundidade resultante

Figura 4.26: Erro na projeção direta do padrão de infravermelhos numa superfície metálica

Este erro de leitura é tanto maior, quanto mais direta for a projeção dos feixes infravermelhos emitidos pelo Kinect. Portanto, uma possível solução, passa por não permitir que tal aconteça, fazendo mover o Kinect para uma posição em que este encare a chapa com uma orientação mais diagonal (posição 2 da figura 4.27). No entanto, esta solução tem a desvantagem de o dispositivo necessitar de estar a uma maior distância da área de trabalho, entrando em regimes de menor precisão. Para além disso, a diferença entre a zona mais próxima e a menos próxima da chapa é aumentada, o que induz gamas de precisão distintas ao longo da área de trabalho. Para além de tudo isto, o aumento da diagonalidade diminui a noção de altura das peças inclinadas. Podendo no limite, a inclinação de determinada peça obstruir uma que esteja na sua retaguarda.

Por fim, existe ainda a possibilidade de colocar o Kinect numa posição elevada, concêntrico com a área de trabalho. A distância deve ser suficientemente grande para que o padrão por ele emitido disperse na chapa de tal forma que a mancha de erro não se manifeste (posição 3 da figura 4.27. Neste caso, mais uma vez, corre-se o risco de se entrar em regimes de pouca precisão do Kinect, influenciado os resultados. Quando se coloca o sistema de visão nesta posição influencia-se também a cooperação com um robô. Note-se que o dispositivo fica obstruído pelo manipulador quando este realiza operações nas peças recortadas. Seria necessário restringir a captura do cenário apenas em situações em que o robô não está a trabalhar sobre a chapa.

Como ilustrado no capítulo 2 a posição escolhida do Kinect no decorrer deste projeto é equivalente à posição 1 assinalada na figura 4.27. A justificação desta escolha deve-se a restrições logísticas do laboratório. Para além disso é também a posição que representa o meio-termo entre precisão de resultados e tamanho da mancha de erro.

Relembrando a inclusão em ambiente industrial deste projeto e considerando as soluções atualmente existentes na empresa Adira [1], verifica-se que o corte é realizado com a chapa parada, e

[1] Suposição feita com base em vídeos demonstrativos no site oficial da empresa

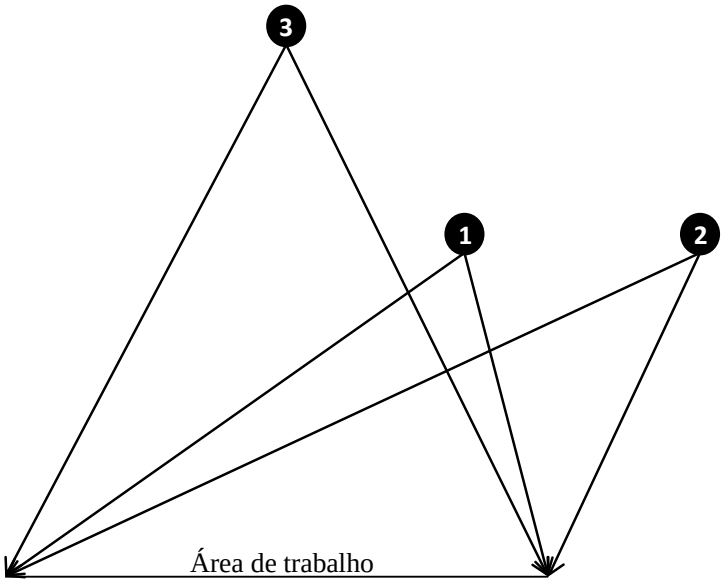


Figura 4.27: Ilustração das diferentes posições com que o Kinect pode encarar a área de trabalho

que após os cortes estarem finalizados esta move-se para uma outra zona. A solução para a atenuação deste tipo de erro de leitura tem de ser considerada em função das exigências do local. Como essas exigências ainda não existem foram considerados todos estes casos possíveis com o intuito de facilitar a escolha da posição do dispositivo de visão computacional.

Contudo, no caso de a posição escolhida resultar sempre em erro sistemático de leitura numa determinada zona, existe ainda a possibilidade de o ignorar temporariamente. Pois, como a chapa se move ao longo do tempo, a zona que está agora inacessível, passará a estar visível após o movimento da chapa.

Capítulo 5

Implementação e Resultados

Neste capítulo é apresentada a implementação do sistema de visão que carateriza este projeto, onde é relatada a construção da aplicação que põe em prática a teoria apresentada anteriormente. Pretende-se classificar as peças recortadas em função da magnitude e direção de inclinação das mesmas, de modo a que o sistema seja capaz de tomar a decisão de recolher a peça recorrendo a um manipulador robótico. Em caso afirmativo este deve calcular o ângulo e posição que a ferramenta de recolha deve exibir.

5.1 Aplicação Desenvolvida

Num sistema baseado em visão computacional existem necessidades comuns. A imediata visualização das imagens adquiridas, a possibilidade de calibração e a visualização do resultado do processamento são bons exemplos desse tipo de requisitos. Com isto, pretende-se usar a aplicação desenvolvida para simplificar estes processos - para qualquer que seja o objetivo do sistema onde o dispositivo de visão tridimensional vá funcionar.

Esta aplicação tem também como objetivo ser multiplataforma. O Lazarus é um ambiente de desenvolvimento livre, que recorre ao FreePascal como linguagem de programação e que permite a compilação na maioria dos sistemas operativos comuns, onde se incluem o Windows, Linux e MacOS. Estas caraterísticas aumentam a portabilidade da aplicação entre os diversos sistemas de controlo.

Esta aplicação serve apenas para a configuração do sistema de visão e representa uma parte do *software* desenvolvido. Isto porque o processamento responsável pela classificação das peças está implementado em Matlab, cujo algoritmo é apresentado na subsecção 5.2.

5.1.1 Interface

No desenvolvimento da aplicação teve-se o cuidado de permitir um fácil controlo da mesma, evitando a existência de submenus. Com isto, pretende-se que a aplicação seja de fácil aprendizagem e uso. Para além da janela principal, apresentada na figura 5.1, existem apenas mais três que têm como função realizar testes de performance ao dispositivo e configurar diferentes calibrações.

A razão para não incluir estas funcionalidades na janela inicial deve-se ao facto de estas serem realizadas apenas pontualmente. Esta divisão em módulos permite também eliminar ou adicionar funcionalidades, sem existir necessidade de modificar as caraterísticas principais desta aplicação.

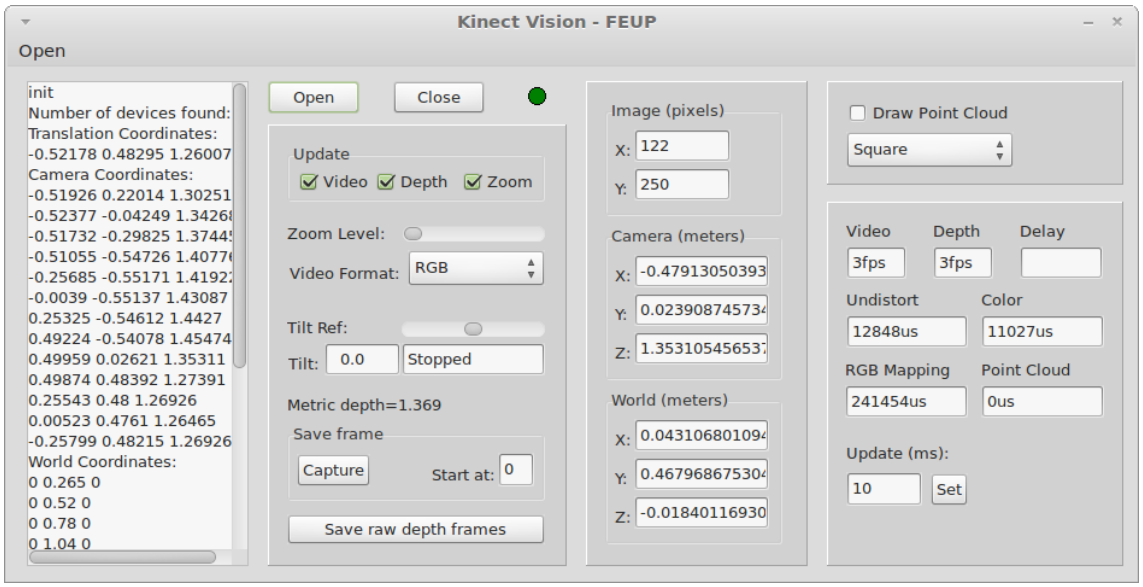


Figura 5.1: Interface do menu principal da aplicação desenvolvida em Lazarus

5.1.2 Funcionalidades

Esta aplicação oferece um grande número de funcionalidades muito úteis em visão computacional, enumeradas abaixo.

1. Visualizar imagem vídeo (colorida) nos modos: *a*) RGB *b*) Bayer *c*) IR
2. Visualizar imagem de profundidade;
3. Fazer zoom de determinada imagem;
4. Capturar e guardar imagens de vídeo em formato PNG;
5. Capturar imagens de profundidade em formato *raw*;
6. Variar o ângulo de inclinação do dispositivo;
7. Calibrar o dispositivo e guardar os dados em formato XML;
8. Carregar dados de calibração em formato XML;
9. Apresentar imagens calibradas e com distorção eliminada;
10. Definir o referencial mundo;
11. Apresentar resultados nos referenciais: *a*) imagem *b*) câmara *c*) mundo
12. Desenhar nuvem de pontos com recurso a: *a*) quadrados *b*) triângulos
13. Controlar os tempos das diversas tarefas;
14. Variar taxa de atualização do dispositivo;
15. Realizar testes de performance: *a*) situações fronteira *b*) distância a determinados pontos

Desta lista, as funcionalidade mais importantes e imprescindíveis são aquelas que têm como base os cálculos e parametrizações matemáticas apresentadas no capítulo 4. As que caraterizam uma aplicação base e que respondem aos requisitos mínimos deste tipo de aplicações são a capacidade de calibrar e de aplicar os seus resultados, de forma a aumentar a precisão do sistema. Para além disso, é também importante que haja a possibilidade de ter os diversos pontos referenciados ao mundo, pois permite uma fácil cooperação do sistema de visão com um outro que dele dependa. Por fim, é também fundamental ter a possibilidade de visualizar *online* os diversos tipo de imagens que um dispositivo como o Kinect oferece.

Neste ponto, com o desenvolvimento desta aplicação, tem-se um sistema base de perceção, e a partir daqui o processamento dos dados é ditado pelas necessidades do sistema.

5.1.3 Limitações

A aplicação oferece a possibilidade de verificar os tempos de execução de determinadas tarefas. E nesta fase é possível perceber que o desenho da nuvem de pontos é demasiado demorado (cerca de 2 segundos). A otimização deste processo é obrigatória no caso de existir necessidade de ter esta tarefa constantemente ativa, pois limita a taxa de atualização de tal maneira que pode tornar a aplicação inviável para o projeto.

5.2 Algoritmo de Classificação

A aplicação desenvolvida em Lazarus permite a aquisição, tratamento e exportação dos dados adquiridos pelo Kinect. Com vista a facilitar a apresentação dos dados, nesta dissertação foi escolhido o Matlab como plataforma de desenvolvimento para o processamento de imagem. Este permite apresentar os resultados 3D de forma extremamente intuitiva, facilitando a sua compreensão.

A extração dos dados pela aplicação Lazarus é feita apenas sobre a imagem de profundidade. O algoritmo desenvolvido só recorre a este tipo de imagem, pois a informação de cor não é relevante para a concretização do projeto. Recorde-se que o objetivo se centra na classificação das peças, como sendo ou não passíveis de recolha por parte de um robô. A informação em profundidade é suficiente para dotar o sistema de poder de decisão. A imagem colorida poderia, no entanto, ser usada para informação auxiliar.

5.2.1 Pré-processamento

O estudo do Kinect permitiu perceber algumas limitações no cálculo de profundidade em sistemas com situações fronteira, tal como explicado na subsecção 4.2.6. A chapa com as peças recortadas carateriza-se por possuir inúmeras situações deste género.

5.2.1.1 Minimização de Medições Falsas

De forma a minimizar o erro e aumentar a confiança dos dados adquiridos diretamente pelo Kinect, são capturadas três *frames* consecutivas de toda a área de trabalho. Estas imagens são exportadas pela aplicação tendo em conta a calibração e distorção do sensor. Cada ficheiro possui $640 \times 480 = 307200$ pontos, que dizem respeito à resolução da câmara de infravermelhos.

Como explicado na subsecção 4.2.2, o cálculo de profundidade é realizado pela comparação de padrões de infravermelhos projetados em *frames* consecutivas. Assim sendo, admita-se a seguinte situação para um determinado ponto no mundo: na primeira *frame* ocorre erro de leitura, e na segunda o dispositivo retorna um determinado valor de profundidade. Como o método de cálculo de profundidade é baseado na comparação entre consecutivos padrões projetados, o valor resultante da correlação destas *frames* será pouco preciso devido ao erro de leitura ocorrido na primeira.

Assim, existe uma componente de pré-processamento com a intenção de aumentar a fiabilidade dos dados recebidos. O método consiste em considerar apenas pontos onde não tenham ocorrido erros de leitura em nenhuma das três *frames* consecutivas. Por outro lado, naqueles pontos onde não ocorreram erros de medição é realizado o cálculo da mediana - de forma a escolher apenas um valor para a *frame* a ser processada. O cálculo da mediana garante que o valor resultante tem menos probabilidade de ser influenciado por uma medida claramente errada, o que não é garantido pelo cálculo da média. Obviamente, isto aumenta o número de pontos com profundidade indeterminada na imagem resultante do pré-processamento. No entanto, temos mais garantias que as profundidades que foram devidamente calculadas estão mais perto da realidade.

Na tabela 5.1 é apresentado o número de ocorrências de erro de leitura em cada uma da imagens de profundidade exportadas e na imagem resultante do cálculo da mediana.

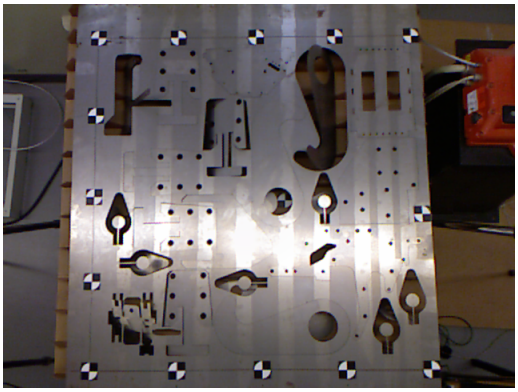
frame 1	*frame* 2	*frame* 3	mediana
18684	18185	18181	23108

Tabela 5.1: Número de ocorrências de erro de leitura em três *frames* consecutivas

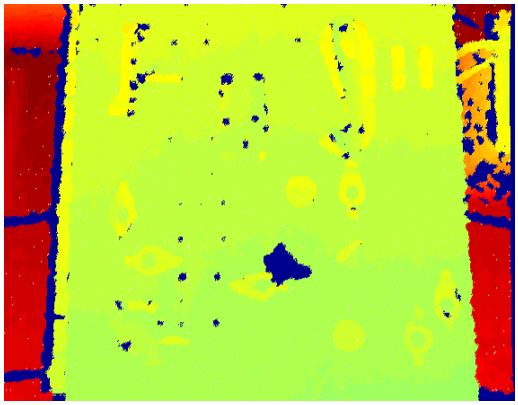
Analisando esses valores percebe-se que o aumento percentual do número de valores intencionalmente ignorados é: $\frac{3 \times 23108}{18684+18185+18181} = 1,259$. Tal corresponde a um aumento de aproximadamente 26% quando comparado com o número médio de erros de leitura ocorridos nas três *frames* adquiridas. Comparando o valor resultante da mediana com o total de pontos que constitui a imagem de profundidade $\frac{23108}{307200} = 0,075$, verifica-se que estes representam apenas $7,5\%$ do tamanho da imagem. Ou seja, a quantidade de valores indeterminados é relativamente pequena face ao ganho no grau de confiança e precisão dos resultados obtidos com o cálculo da mediana.

A quantidade de valores de profundidade excluídos pode ser visualiza na figura 5.2. Para a compreensão dos mapas de profundidade, a cor de um ponto é atribuída em função da magnitude do seu valor. As cores variam do azul escuro (cor fria) a vermelho (cor quente), sendo que esta última representa valores de maior magnitude. Por outras palavras, em relação ao referencial do Kinect, os objetos mais perto deste são assinalados com cores frias, e vice-versa. Os pontos

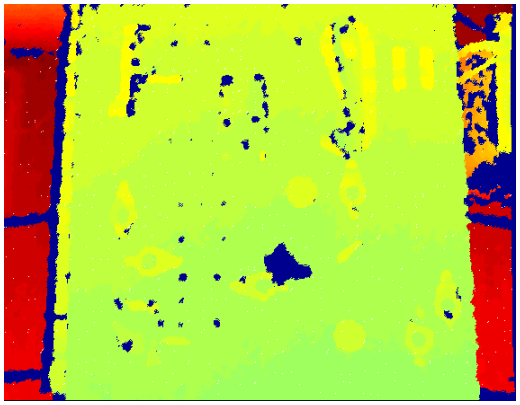
nas imagens de profundidade correspondentes a leituras indeterminadas são assinaladas com cor azul escura, correspondente a uma profundidade zero. Note-se que para estes casos tem de ser atribuído um valor impossível de ser retornado pelo Kinect. Uma distância zero nunca acontece, pois corresponderia a algo coincidente com o sensor.



(a) Imagem colorida



(b) Projeção de uma amostra



(c) Projeção da mediana

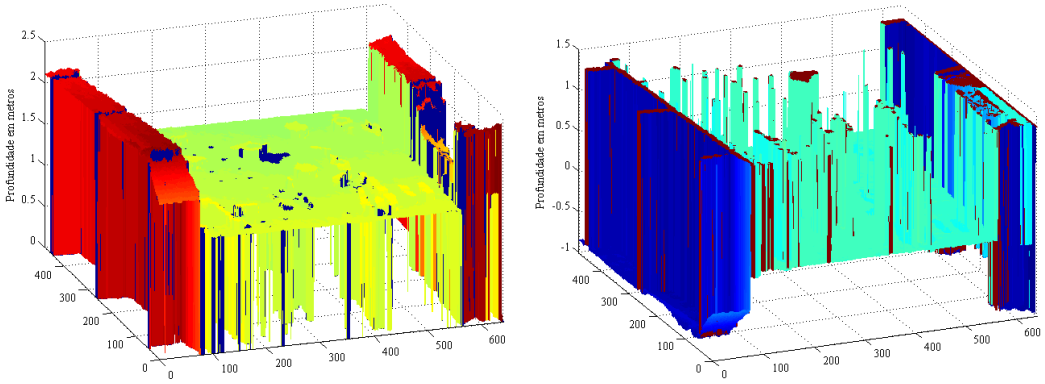
Figura 5.2: Comparação dos mapas obtidos entre uma amostra das três *frames* e respetiva mediana

A figura 5.2(a) serve para situar as imagens de profundidade no mundo. Como seria de esperar, comparando as figuras 5.2(b) e 5.2(c), a maior ocorrência de valores excluídos ocorre em zonas fronteira e onde existiam já algumas dificuldades por parte do Kinect em calcular a distância a esse ponto. Esse facto é comprovado pelo aumento da área das zonas representadas a azul na figura da amostra. As situações novas de dados incertos que aparecem na imagem resultante da mediana - mas não na amostra - dizem respeito a situações de ocorrência de erros de leitura nas outras duas *frames*, que não são aqui representadas.

O facto de nesta fase ser usado o referencial câmara é evidente em qualquer uma das imagens 3D da figura 5.2. Isto é especialmente notório na alteração linear da cor na zona da chapa, onde esta varia de um tom esverdeado para um amarelado. Uma variação deste tipo é justificada pela orientação ligeiramente inclinada do Kinect face ao plano da chapa. Isto vai de encontro ao enunciado anteriormente na subsecção 4.2.5, e apresentado na figura 4.14.

5.2.1.2 Transformação para Coordenadas Mundo

Para entender melhor a área de trabalho é preferível utilizar coordenadas no referencial mundo. Para tal, é realizado um processo de cálculo de transformação homógena que converte os pontos em coordenadas câmara para o referencial mundo. Todo este processo foi apresentado na subsecção 4.2.5 e representado na figura 4.14. O resultado desta calibração é apresentado na figura 5.3 onde se pode verificar que o plano da chapa, em coordenadas mundo, está centrado em profundidade zero.



(a) Referencial câmara (b) Referencial mundo

Figura 5.3: Comparação do mapeamento 3D nos referencias câmara e mundo

Por comparação dos mapeamentos representados nas figuras 5.3(a) e 5.3(b) verifica-se que o resultado ainda não é o esperado. Isto porque a transformação moveu os valores de profundidade nula correspondentes a leituras indeterminadas no referencial câmara ($z = 0m$), para valores de profundidade elevada no referencial mundo ($z > 1m$). De forma a explicar o sucedido considere $\mathbf{p_0}$ como um vetor de coordenadas nulas. Aplicando-lhe a transformação homogénea de transformação do referencial câmara para o referencial mundo temos:

$$\left(\mathbf{p_0}^{\mathrm{T}} - \mathbf{t}^{\mathrm{T}}\right) \times \mathbf{R} = \begin{bmatrix} 0.5001 & 0.3309 & 1.3303 \end{bmatrix} \tag{5.1}$$

Em que, os valores $\mathbf{R}$ e $\mathbf{t}$, calculados pelo método apresentado em 4.2.5, valem:

$$\mathbf{R} = \begin{bmatrix} 0.9999 & -0.0031 & -0.0157 \\ -0.0049 & -0.9927 & -0.1207 \\ -0.0152 & 0.1208 & -0.9926 \end{bmatrix}, \quad \mathbf{t} = \begin{bmatrix} -0.4782 \\ 0.4915 \\ 1.2880 \end{bmatrix}$$

Como se verifica pela equação 5.1, a componente z de um valor que corresponda a uma leitura indeterminada no referencial inicial passa a valer $1,3303m$ no referencial mundo.

A partir deste ponto as posições no mundo serão classificadas como *inliers* ou *outliers*, consoante estes pertençam ou não à área de trabalho. Desta forma, as discrepâncias introduzidas pela transformação de coordenadas torna necessário uma nova etapa de pré-processamento, que classifique como *inlier*, para o eixo z, apenas valores cujas distâncias sejam inferiores à anteriormente calculada.

5.2.1.3 Delimitação da Área de Trabalho

O Kinect está posicionado de tal forma que, para visualizar toda a área da chapa, este capta também nas zonas laterais informação que não lhe é relevante. Neste caso, estas posições correspondem a valores de distância elevados que correspondem ao solo e a outros objetos também sem significado. Por isso, existirá uma fase no algoritmo de processamento onde essas zonas são eliminadas.

Como o referencial do mundo foi escolhido de forma ao plano xOy ficar coincidente com a chapa, fica assim simples definir a área de trabalho neste plano. Para isso, basta definir os valores mínimos e máximos que x e y podem tomar, escolhendo-os de acordo com as dimensões da chapa.

Assim, em função da origem do referencial dá-se um valor máximo ao eixo x e y equivalente a $1m$. Para além disso, sabemos que para evitar mais valores sem significado, podemos restringir o valor mínimo para o eixo z do referencial mundo, pois a altura entre o fundo do suporte e o topo dos triângulos é conhecida e vale $10cm$. Assim sendo, em teoria, nunca deveriam ser retornados pelo Kinect valores inferiores a $-0,010$. No entanto, mesmo com a tentativa de minimizar o erro com recurso à mediana, isso acontece ainda que muito esporadicamente. Tipicamente, cerca de 5 pontos apresentam valores fora dos limites considerados.

Assim sendo, temos a área de trabalho delimitada nos três eixos do referencial mundo da seguinte forma:

- $\mathbf{x} \in [0;\ 1,05]$
- $\mathbf{y} \in [0;\ 1,05]$
- $\mathbf{z} \in [-0,1;\ 1,33)$

Os resultados das restrições em cada eixo podem ser vistos na figura 5.4(a). O eixo dos z apresenta agora valores numa gama muito mais pequena, facilitando a compressão dos dados. A figura 5.4(b) representa apenas a projeção em xOy do mapeamento 3D. Nesta figura consegue-se já tirar algumas conclusões, nomeadamente:

1. A área de interesse está delimitada à área da chapa e as posições que não lhe pertencem são mapeadas com o valor mínimo de z, apresentando, por isso, cor azul escura;
2. Zonas da imagem que apresentem cor azulada indicam que o dispositivo vê o fundo do suporte, demonstrado uma situação de inexistência de peça nessa região;
3. Zonas da imagem que apresentam variação uniforme de cor remetem para situações de peças inclinadas;
4. A chapa não é mapeada de forma perfeitamente plana, pois a parte central da mesma apresenta uma cor mais quente, o que é indicador de convexidade.

A delimitação enunciada no ponto 1 foi conseguida estabelecendo os limites para x, y e z anteriormente apresentados. A convexidade da chapa mencionada no ponto 4 está de acordo com os resultados apresentados na subsecção 4.2.6. O Kinect está a um altura próxima do $1,30m$ que se enquadra na gama de funcionamento onde a convexidade se evidencia. Para além disso a configuração matriz de madeira mais chapa assenta numa banca de suporte, que ocupa apenas uma pequena área central daquela configuração. Com o peso da chapa a madeira cede um pouco nos cantos, contribuindo assim para o aumento da convexidade.

5.2.2 Processamento

Nesta fase têm-se apenas informação referente ao que se encontra na área da chapa. Agora, recorrendo a processamento de imagem, é necessário extrair as peças inclinadas para ser possível classificá-las em função da sua inclinação.

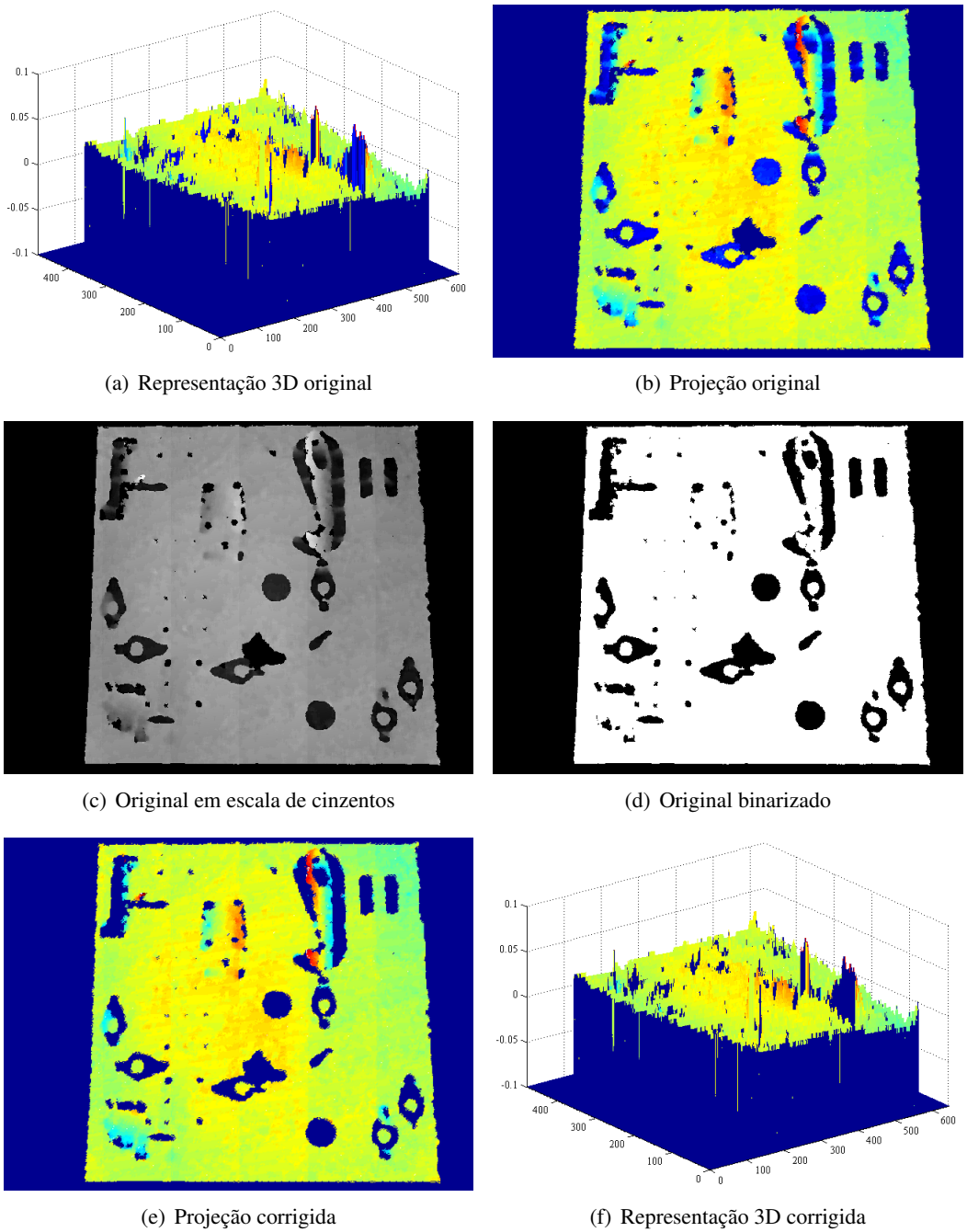
5.2.2.1 Remoção de Valores Ambíguos

A projeção representada na figura 5.4(b) apresenta valores mapeados a cor azul que se aproximam muito do valor mínimo de z. Estes valores confundem-se entre posições da peça ou do fundo do suporte. De forma a obter apenas dados relevantes para a identificação das peças, é conveniente removê-los, incluindo no conjunto dos *inliers* valores que se referem apenas a material da chapa.

Isto é conseguido dividindo o processamento em duas etapas: primeiro, convertendo a imagem de projeção para escala de cinzentos; e em segundo, aplicando-lhe uma binarização. O valor de *threshold* usado na segunda etapa controla o que se quer classificar como *inlier*. Os resultados obtidos, figura 5.4(c), foram conseguidos escolhendo para valor de *threshold* o valor resultante da aplicação do método de Otsu - que calcula o valor ótimo baseado na forma do histograma da imagem.

A imagem resultante da binarização, figura 5.4(d), funciona como uma imagem classificadora para os pontos presentes na projeção original: os valores a branco representam a posição dos *inliers*; os valores a preto representam a posição dos *outliers*. A aplicação dessa máscara na projeção 5.4(b), resulta na imagem 5.4(e). Por comparação dessas figuras verifica-se que foi eliminada a ambiguidade na gama do azul, representativa de valores muito próximos do limite mínimo admitido para profundidade no referencial mundo. Posteriormente os resultados classificados como *inliers* foram reprojetados em 3D, onde as alterações induzidas pelo algoritmo são menos evidentes, figura 5.4(f).

A figura 5.4 no seu todo representa o processo sequencial de transformações na obtenção de uma imagem tridimensional que contém apenas dados fiáveis. A imagem final deste processo serve agora como base para a extração das zonas referentes a peças e sua consequente classificação, em função da inclinação das mesmas.



(a) Representação 3D original

(b) Projeção original

(c) Original em escala de cinzentos

(d) Original binarizado

(e) Projeção corrigida

(f) Representação 3D corrigida

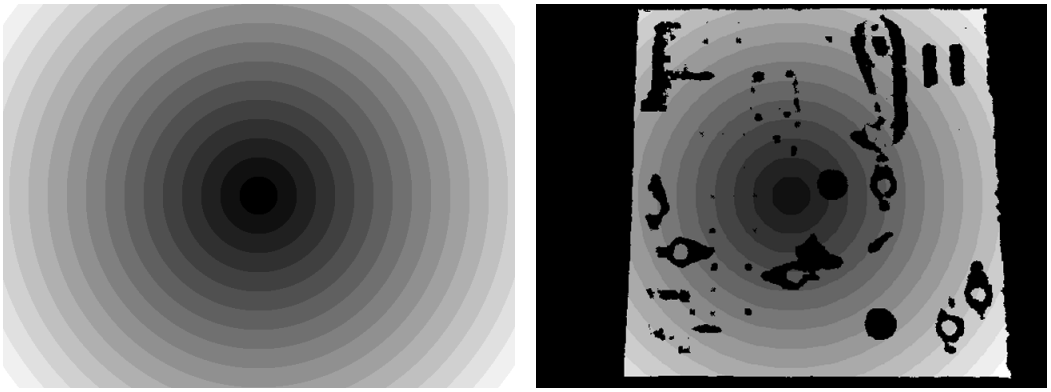
Figura 5.4: Representação do resultado das diferentes fase do algoritmo na remoção de *outliers*

5.2.2.2 Modelo da Não-Linearidade do Plano

Como foi já enunciado neste capítulo, o mapeamento em profundidade da chapa não é feito como um plano perfeito, devido a caraterísticas intrínsecas do sistema de visão, e também às condições do meio físico onde a mesma se encontra apoiada. Desta forma, foi necessário arranjar um modelo da sua superfície, de modo a ser possível identificar as peças recortadas e que estão inclinadas. O princípio de funcionamento centra-se em achar o plano correspondente à profundi-

dade zero da chapa, isto é, onde todas as peças estariam no caso de nenhuma delas inclinar após o corte. Assim sendo, as peças recortadas que não se encontram inclinadas não são identificadas pelo algoritmo, pois como se está apenas a usar informação de profundidade, estas confundem-se com o plano total da chapa.

Com este pressuposto, aproxima-se esta não linearidade no mapeamento da chapa por meio de círculos concêntricos de raio progressivamente maior. Com isto criam-se diversas zonas delimitadas pela diferença da área de círculos consecutivos, figura 5.5(a). A imagem resultante funciona como uma máscara que, quando aplicada a uma determinada imagem, agrupa o conjunto de pixeis que se encontram na mesma área circular, figura 5.5(b). Depois de sobreposta esta informação, é realizado o cálculo do valor médio de profundidade para cada zona. Neste cálculo são excluídos os valores que representam leituras indeterminadas (coloridos a preto), por forma a evitar interferências de valores de profundidade irreais.



(a) Máscara (b) Máscara aplicada

Figura 5.5: Aproximação da não linearidade da superfície por círculos de diferente raio

A figura 5.6 apresenta os valores de profundidade médios para as diferentes zonas delimitadas pela máscara. Estes compravam a convexidade do plano, devido ao facto do gráfico representativo dos valores médios obtidos apresentar uma evolução maioritariamente decrescente.

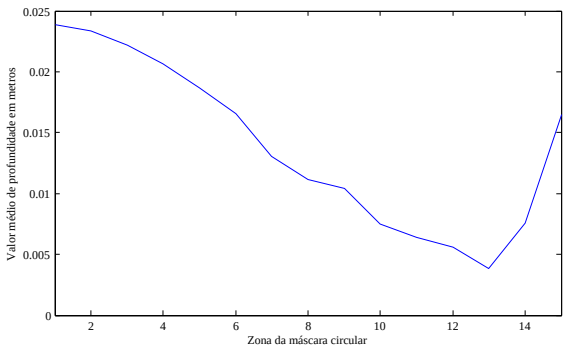


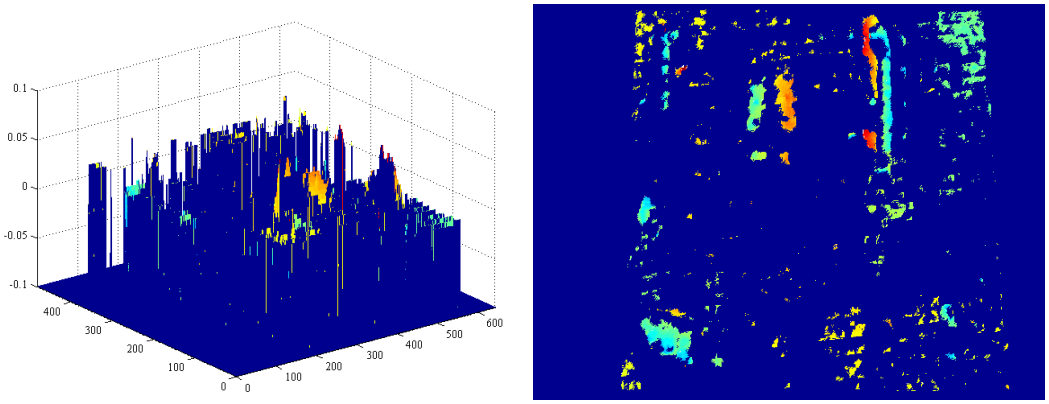
Figura 5.6: Resultado com extração do plano zero

Existe no entanto uma subida dos valores referentes às zonas finais, que não chegam a constituir áreas completamente circulares. Este valor está também de acordo com uma caraterística intrínseca do Kinect, pois é conhecido que a profundidade nos cantos não é calculada com a mesma precisão.

Esta modelização do plano da chapa torna o algoritmo adaptativo. Note-se que este pode ser realizado qualquer que seja a configuração da chapa ou da inclinação das peças que a constituem. Alternativamente, e para obtenção de resultados mais precisos, este cálculo por aproximação pode ser realizado com um plano sem recortes ou peças inclinadas. Desta forma, o cálculo da média será mais consistente pelo simples motivo de existirem mais valores disponíveis, e nenhum deles apresentar inclinação.

5.2.2.3 Extração do Plano

Removendo estes valores médios, com um *threshold* de altura de $0,5cm$, obtém-se os resultados representados na figura 5.7. É possível verificar que com a aplicação do método enunciado, grande parte da imagem fica reduzida a valores que remetem para peças que apresentam uma determinada inclinação.

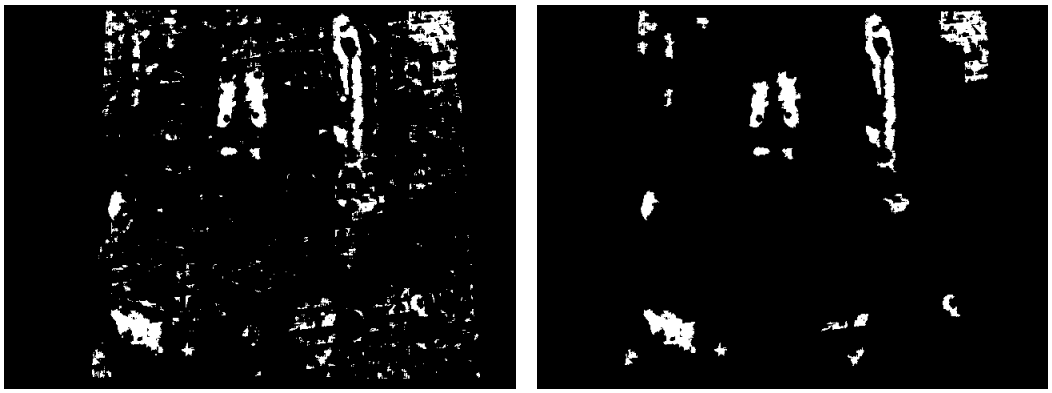


(a) Representação 3D (b) Projeção

Figura 5.7: Resultado com extração do plano zero

Com a projeção da figura 5.7(b) é possível perceber que existem manchas demasiado pequenas que não necessitam de ser consideradas. Estas podem ser interpretadas como ruído dado que, na sua maioria, resultam de oscilações nas medições. Para além disso, posteriormente é realizada a aproximação por um plano dos pontos que pertencem a uma determinada mancha, cujo cálculo é mais preciso para um elevado número de pontos. Desta forma, optou-se por não considerar aquelas que são constituídas por menos de 150 pixeis no referencial imagem. O resultado deste processo é apresentado na figura 5.8.

De modo a analisar o ângulo de inclinação de um conjunto de pontos foi implementado um algoritmo que aproxima esse mesmo conjunto por um plano. O algoritmo recorre ao cálculo do SVD que permite devolver o vetor normal ao plano interpelado. O vetor normal vem normalizado



(a) Todos os valores (b) Maiores que o tamanho mínimo especificado

Figura 5.8: Comparação das imagens depois da remoção das manchas pequenas

e referente ao sistema de coordenadas do conjunto de pontos submetido ao algoritmo. Desta forma, para analisar o ângulo de inclinação calcula-se diretamente o ângulo entre esse vetor normal e o versor do referencial mundo. Esse mesmo vetor contém também informação sobre a direção, no plano da chapa, da direção de inclinação. Para isso, basta analisar as componentes x e y do vetor que funcionam como a projeção no plano xOy (coincidente com a chapa). Com isto, consegue-se obter toda a informação necessária acerca da inclinação de determinada peça recortada: ângulo e direção da inclinação. O conjunto desta informação, em associação a um modelo de decisão, permite inferir sobre a possibilidade da peça ser retirada por um robô, por exemplo.

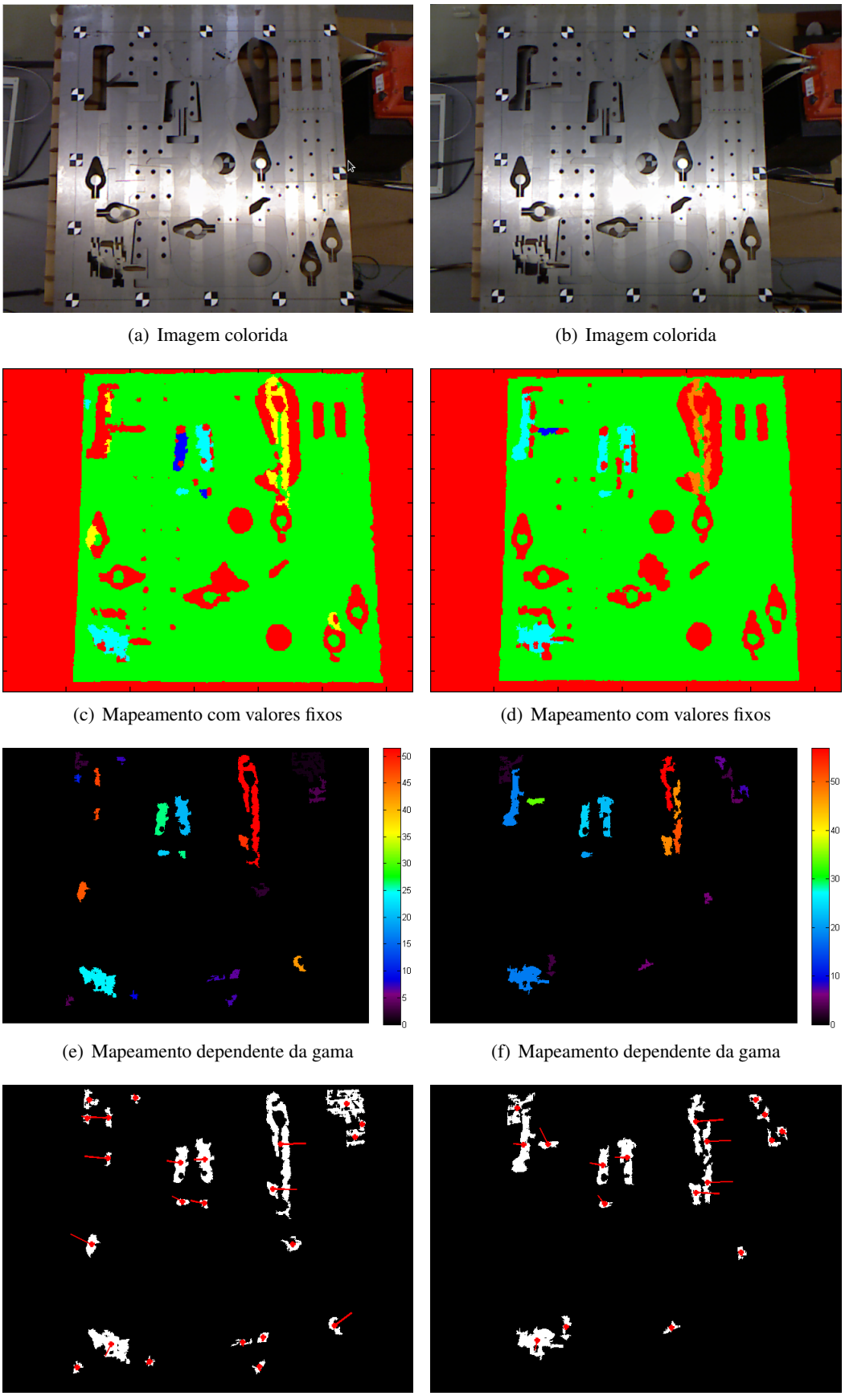
5.2.3 Resultados

Até aqui foi apresentada toda a sequência de processamento referente a uma única configuração da chapa. De modo a comprovar a consistência do algoritmo foi realizado um novo teste:

- as peças foram colocadas na posição normal (perfeitamente alinhadas) e deixadas cair novamente ao acaso, movendo ligeiramente a chapa para que estas não ficassem na mesma posição que o teste anterior;
- o Kinect foi mudado de posição e orientação, para ser necessária uma nova calibração do referencial mundo.

Na figura 5.9 são apresentados os resultados do algoritmo para as duas situações, onde a diferente orientação das peças pode ser constatada comparando as figuras 5.9(a) e 5.9(b).

De forma a apresentar os dados de diferentes formas são apresentados dois casos de mapeamento: um fixo em que cada cor corresponde a uma gama especificada, figuras 5.9(c) e 5.9(d); e outro em que a cor é mapeada em função de uma gama de valores que depende da diferença entre a inclinação máxima e mínima, figuras 5.9(e) e 5.9(f). Por fim, é apresentada numa só imagem a informação acerca da magnitude e direção da inclinação com recurso a semi-retas de cor vermelha, que têm início no centróide da mancha. O tamanho da semi-reta é proporcional à magnitude e a sua orientação reflete a direção de inclinação no plano da chapa, figuras 5.9(g) e 5.9(h).



(a) Imagem colorida (b) Imagem colorida

(c) Mapeamento com valores fixos (d) Mapeamento com valores fixos

(e) Mapeamento dependente da gama (f) Mapeamento dependente da gama

(g) Representação da magnitude e direção (h) Representação da magnitude e direção

Figura 5.9: Resultados finais obtidos em dois testes separados na vertical: um teste ilustrado nas imagens do lado esquerdo; o outro nas imagens do lado direito

Analisando as figuras 5.9(a) e 5.9(b), verifica-se que os dois casos apresentam peças com inclinação idêntica. No entanto, a informação apresentada nas restantes figuras comprova que o algoritmo é capaz de medir essas pequenas diferenças. Isto permite concluir que o mesmo é sensível a pequenas variações do meio.

Ainda comparando os dois casos nota-se que existem situações que ocorrem recorrentemente, nomeadamente:

1. Manchas que não representam peças;
2. Manchas distintas que representam a mesma peça;
3. Manchas que representam a mesma peça, mas com ângulo e/ou direção diferentes.

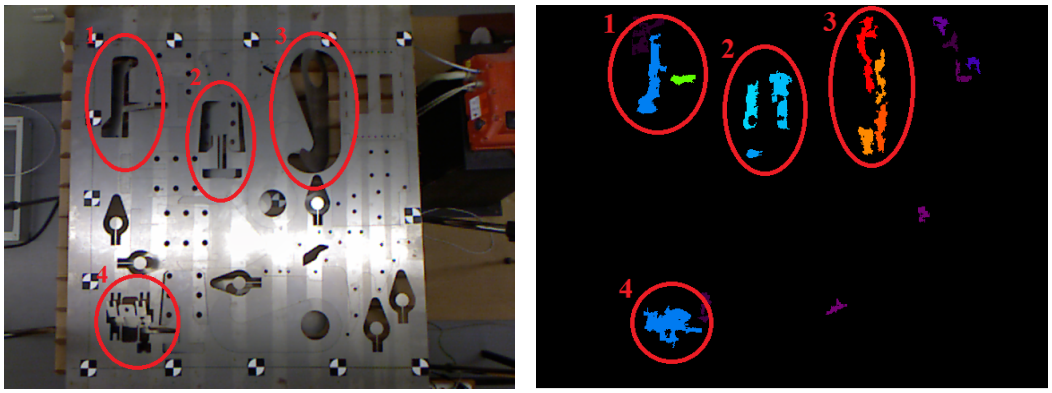
A situação 1 ocorre especialmente nos cantos da imagem. Independentemente da zona de área de trabalho, isto acontece devido ao *threshold* escolhido não incluir o desvio padrão das medições efetuadas em algumas zonas da aproximação por áreas circulares. Isto surge também como consequência de se escolher o mais pequeno valor possível para *threshold*, de forma a não remover informação importante para a posterior aproximação por um plano. Por outro lado, a inexistência de peça numa determinada zona, em que o suporte de triângulos fique visível, induz em erro o algoritmo. Recorde-se que o número deste tipo de ocorrências é minimizado na altura de binarização da imagem de projeção dos *inliers*, para eliminação de ambiguidades na zona de profundidade mínima admitida.

O ponto 2 é explicado pela remoção do plano de referência (profundidade zero) que também elimina pontos correspondentes a peças. Na maioria dos casos, estas diferentes manchas apresentam resultados de magnitude e orientação de inclinação muito próximos, o que é indicador de se tratar da mesma peça.

No entanto, como enunciado no ponto 3, ocorrem algumas exceções. Existem, de facto, algumas situações em que o resultado difere um pouco para manchas representativas da mesma peça. Estas situações são mais frequentes se as manchas forem pequenas ou representarem peças muito inclinadas. A primeira situação explica-se pela menor eficiência do algoritmo de aproximação por um plano, e a segunda por dificuldades na leitura por parte do Kinect.

Para enquadrar os resultados com a realidade, foram realizadas medições para o segundo caso de teste (grupo de imagens do lado direito da figura 5.9) com recurso a um aristo. Note-se que o aristo mede apenas num eixo, pelo que são apresentados apenas os resultados onde foi possível calcular o ângulo de inclinação com recurso a este aparelho. Por outro lado, torna-se complicado medir situações em que as peças estão inclinadas e abaixo do plano da chapa, onde o aristo não consegue entrar. Desta forma, a título de exemplo foram realizadas duas medições nas peças assinaladas na figura 5.10, cujos resultados são apresentados na tabela 5.2.

Estes resultados demonstram as situações recorrentes apontadas anteriormente. A peça 1 apresenta uma menor consistência nos resultados da inclinação. Por outro lado, verifica-se que para as peças 2, 3 e 4 os valores calculados estão muito próximos da inclinação medida, para todas as manchas representativas da mesma peça.



(a) Imagem original RGB (b) Imagem resultante do algoritmo

Figura 5.10: Comparação das imagens depois da remoção das manchas pequenas

Peça	1		2			3				4
Medido	29		23			52				19
Aplicação	33,93	18,19	24,15	22,48	20,11	57,01	51,42	47,46	47,15	17,14

Tabela 5.2: Inclinação medida e calculada nos casos assinalados (valores em graus)

Até aqui o agrupamento das manchas referentes a cada peça tem sido feito visualmente, isto é, cruzando a imagem RGB representativa do meio com as imagens resultantes do algoritmo. Evidentemente, é importante que este tipo de cruzamento seja feito de modo automático sem intervenção humana. Na secção 6.2 é apresentada uma solução de implementação que vai de encontro às caraterísticas deste tipo de trabalho.

Capítulo 6

Conclusão

Este capitulo sumariza as principais conclusões obtidas e, por fim, é feito um levantamento de possíveis trabalhos futuros no âmbito deste e outros projetos relacionados.

6.1 Comentários ao Trabalho Realizado

O Microsoft Kinect é cada vez mais usado em projetos de visão computacional. Os testes realizados permitiram perceber a dinâmica de funcionamento do dispositivo. Para além disso, consolidou-se o conhecimento teórico do modo de funcionamento dos dispositivos da PrimeSense. A confiança aumenta quando se conhece o que o dispositivo é capaz de fazer e tentou-se tirar o máximo proveito das suas caraterísticas. Esta análise à sua performance incrementa o conhecimento das suas possibilidades face a outros dispositivos concorrentes mais conhecidos e usados até à data.

Os sistemas baseados em visão têm muitas caraterísticas em comum que necessitam de serem simplificadas, para que o tempo perdido nesse tipo de pré-processamento seja o menor possível. A aplicação desenvolvida em Lazarus serve para isso mesmo. É a base para todos os projetos que recorram ao Kinect, e em teoria a outras tecnologias da PrimeSense. Funcionalidades importantes como a remoção de distorção e calibração foram testadas e os resultados foram muito positivos. Isto ficou comprovado com o uso da aplicação do início ao fim do processamento sem intervenção de aplicações externas. Com isto conclui-se que a aplicação é autónoma para o efeito que foi implementada.

No contexto do problema em concreto a resolver constatou-se que a combinação do *software* e *hardware* utilizado devolve resultados muito próximos da realidade, com o correto cálculo da magnitude e orientação das peças que apresentam inclinação. Fica apenas em falta o teste do conjunto do sistema implementado em ambiente industrial. No entanto, as soluções apresentadas são moldáveis ao ponto daquela combinação funcionar em meios mais exigentes, quando comparados com o de laboratório.

6.2 Trabalho Futuro

Como desenvolvimento futuro seria interessante juntar todo o processamento de imagem à aplicação desenvolvida em Lazarus. A adição do algoritmo de processamento de imagem deve, no entanto, ser feito de forma modular, de modo a que a sua alteração ou remoção não afete as funcionalidades base. Assim, a aplicação manteria a sua principal função mas com uma extensão para um determinado problema.

Mais direcionado para o objetivo deste projeto, seria muito útil recorrer aos ficheiros DXF utilizados nas máquinas de corte a laser, que indicam onde são realizados os cortes. A utilização desta mesma informação por parte da aplicação facilitaria o mapeamento das peças de determinada configuração da chapa. Assim, a componente de processamento de imagem seria mais leve, sem decremento da precisão. Aliás, o mais provável seria um aumento de qualidade no cálculo da posição de determinada peça, dado que neste tipo de ficheiros consta toda a informação geométrica da mesma. Numa situação deste tipo, bastaria mapear as posições do ficheiro DXF com as coordenadas mundo da imagem de profundidade processada pela aplicação. Depois, seria evidente se, numa determinada posição, existe ou não uma peça. Em caso afirmativo, recorrer-se-ia aos resultados do algoritmo para inferir sobre a sua inclinação, de forma a ser posteriormente recolhida por um robô. A inclusão da informação dos ficheiros DXF permitem um cruzamento entre a realidade e as manchas resultantes do algoritmo, conferindo à aplicação final a componente de decisão automática mencionada no capítulo anterior.

É sabido que o ambiente industrial deste tipo de máquinas é exigente. Para além disso, as aplicações funcionam ininterruptamente, durante longas horas. Estas caraterísticas teriam de ser tidas em conta na altura de optar pelo Kinect como solução de visão computacional. Neste caso, seria conveniente alterar a sua estrutura externa de forma a torná-lo mais robusto. Por exemplo, acrescentando uma componente de ventilação ou proteção para as lentes. Nesta última opção, o papel da calibração é mais um vez fundamental, pois as suas caraterísticas óticas mudariam consideravelmente quando comparadas com os valores *standard*.

Referências

[1] Produtech. http://www.produtech.org.

[2] Roland Siegwart e Illah R. Nourbakhsh. *Introduction to Autonomous Mobile Robots*. Bradford Company, Scituate, MA, USA, 2004.

[3] PrimeSense. Primesense official site. http://www.primesense.com.

[4] Microsoft. Xbox 360 kinect. http://www.xbox.com/kinect/.

[5] K. Khoshelham. Accuracy analysis of kinect depth data. Em *ITC Faculty of Geo-information Science and Earth Observation, University of Twente*, 2011.

[6] iPiSoft. Asus xtion vs ms kinect comparison. http://wiki.ipisoft.com/ASUS_Xtion_vs_MS_Kinect_Comparison.

[7] Asus. Asus xtion pro live. http://www.asus.com/Multimedia/Motion_Sensor/Xtion_PRO_LIVE/.

[8] Journalist. The teardown. *Engineering Technology*, 6(3):94 –95, april 2011.

[9] Jan Smisek, Michal Jancosek, e Tomas Pajdla. 3d with kinect. Em *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, páginas 1154 – 1160, nov. 2011.

[10] Xin Wang Boris Lenseigne, Maja Rudinac. 3d vision. Relatório técnico, TU Delft, 2011.

[11] D. Scaramuzza R. Siegwart. Perception - sensors. Relatório técnico, ETH Zurich, 2011.

[12] Google. Google self-driving car - part 1: Outside. http://www.youtube.com/watch?v=e_lGPRIRG3Y.

[13] Byron Spice. Researchers help develop full-size autonomous helicopter. Relatório técnico, Carnegie Mellon, 2010.

[14] Olivia Koski. In a first, full-sized robo-copter flies with no human help. Wired, 2010.

[15] Peter Seitz Robert Lange. Seeing distances – a fast time-of-flight 3d camera. *Sensor Review*, Vol. 3:212–217, 2000.

[16] SwissRanger. Sr-4000 datasheet. Relatório técnico, Mesa Imaging, 2011.

[17] S. Hsu, S. Acharya, A. Rafii, e R. New. Performance of a time-of-flight range camera for intelligent vehicle safety applications.

[18] Ulfig Brockherde Hosticka Mengel Listl Elkhalili, Schrey. A 64x8 pixel 3-d cmos time-of flight image sensor for car safety applications. Relatório técnico, Ecole Polytechnique Federale de Lausanne, 2006.

[19] Paul Wootton. Out of control gaming. http://www.popsci.com/gear-gadgets/article/2008-05/out-control-gaming.

[20] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(11):1330–1334, Novembro 2000.

[21] D. C. Brown. Decentering Distortion of Lenses. *Photometric Engineering*, 32(3):444–462, 1966.

[22] OpenCV. Camera calibration with opencv. http://opencv.itseez.com/doc/tutorials/calib3d/camera_calibration/camera_calibration.html.

[23] OpenCV. Camera calibration and 3d reconstruction. http://opencv.itseez.com/trunk/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html.

[24] Zhengyou Zhang. Flexible camera calibration by viewing a plane from unknown orientations. Em *in ICCV*, páginas 666–673, 1999.

[25] OpenKinect. libfreenect drivers. http://openkinect.org/wiki/Main_Page.

[26] PrimeSense. Openni drivers. http://openni.org/.

[27] ROS. Technical description of kinect calibration. http://www.ros.org/wiki/kinect_calibration/technical.

[28] Stephane Magnenat. Raw depth to meters.

[29] Nassir Navab Victor Castaneda. Time-of-flight and kinect imaging. Relatório técnico, technische universität münchen, 2011.