FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Sistema *Projection Mapping* Adequado a Robótica Móvel e Cenas Complexas

Hugo José Magalhães Costa

Mestrado Integrado em Engenharia Electrotécnica e de Computadores

Orientador: Prof. Doutor Armando Jorge Miranda de Sousa

Co-orientador: Doutor Germano Manuel Correia dos Santos Veiga

28 de Julho de 2015

U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

MIEEC - MESTRADO INTEGRADO EM ENGENHARIA ELETROTÉCNICA E DE COMPUTADORES | **2014/2015**

A Dissertação intitulada

"Sistema Projection Mapping Adequado a Robótica Móvel e Cenas Complexas"

foi aprovada em provas realizadas em 23-07-2015

o júri

Presidente Professora Doutora Maria Margarida de Amorim Ferreira
Professora Auxiliar do Departamento de Engenharia Eletrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto

Professor Doutor Artur José Carneiro Pereira
Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro

Professor Doutor Armando Jorge Miranda de Sousa
Professor Auxiliar do Departamento de Engenharia Eletrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto

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

Autor - Hugo José Magalhães Costa

Faculdade de Engenharia da Universidade do Porto

Resumo

Robótica móvel será predominante na abordagem de muitas necessidades industriais e sociais futuras. No entanto, para que tais robôs possam operar em cenários da vida real cada vez mais complexos e dinâmicos, alguns desafios importantes devem ser abordados em primeiro lugar. Várias tarefas tais como modelagem 3D de ambiente, planeamento de tarefas, auto-localização, e Interação Humano-Robô (HRI) são áreas em desenvolvimento constante.

Relativamente à comunicação da máquina (ou robô) para o humano, a técnica *Projection Mapping* mostra uma imagem corretamente em quase qualquer cena alvo. Isto significa que cada superfície pode transformar-se num meio de comunicação mais imersivo onde é possível acrescentar informação sem que o trabalhador humano tenha de mudar de contexto de trabalho (olhar para um sistema externo).

No que toca à comunicação do humano para a máquina, é necessário utilizar sensores específicos que tirem proveito de uma interação dentro do contexto de trabalho, facilitando assim a comunicação bidirecional entre homem e máquina (diretamente no ambiente de trabalho do humano).

Estas ideias são particularmente apelativas para comunicação entre robôs e humanos, principalmente em robôs colaborativos. Como os robôs preenchem cada vez mais papéis na sociedade atual, realizando tarefas mais complexas e socialmente mais desafiadoras, é importante definir uma interação capaz de abranger todas as necessidades exigidos pelas tarefas ou de atender a todas as restrições impostas pela ambiente.

Esta dissertação propõe uma arquitetura para um sistema genérico de HRI baseado em *projection mapping* e um apontador laser com duas frequências, bem como a implementação e validação num manipulador robótico (ABB IRB140), no robô do projeto CARLoS e no AGV (*Automated guided vehicle*) jarvis.

O ambiente real é recriado em preto no simulador gazebo através de ficheiros CAD e *.STL* disponibilizados e adicionados os objetos 3D e/ou ficheiros de imagens que o utilizador pretenda visualizar. O mundo é modelado em preto para não aparecer na imagem a ser projetada mas causar as devidas oclusões nos objetos.

As transformações da imagem são conseguidas através da colocação de uma câmara no mundo virtual, com as caraterísticas do projetor real, na posição real. Este processo resulta em projeções corretas sobre quase qualquer superfície, evitando engenharia dispendiosa na sua implementação.

A interação humano para máquina é feita através de um apontador laser de duas frequências de funcionamento configuráveis (atualmente, clique direito 2.5Hz e clique esquerdo 3.5Hz) e permite ao utilizador selecionar diretamente no ambiente através de caixas de diálogo projetadas quais os objetos a visualizar.

Todo o sistema é desenvolvido com a framework de ROS proporcionando abstração e modularidade.

Abstract

Mobile robotics will be prevalent in addressing many future industrial and social needs. However, in order for such robots to operate in increasingly complex and dynamic real life scenarios, important challenges must be tackled first. Several tasks like 3D environment modeling, task planning, self-localization, and Human-Robot Interaction (HRI) are areas of expertise mobile robotics is set to overcome in the current literature.

When speaking about the communication between machine (or robot) and the human, projection mapping can be used to project an image correctly over almost all surfaces. This means that every surface can become an immersive display where all sort of visual information can be portrayed. In the other direction, human-machine, projection mapping can be coupled with other sensors to allow a truly interactive experience inside the scope of the work. It's for these reasons that this technique has generated interest in the community and in the industry. This ideas are particularly useful when speaking about collaborative robots. As robots fill more and more roles in current society, performing more complex and more socially challenging tasks, it is important to define an interaction capable of filling all the needs demanded by the tasks or to answer all the restrictions imposed by the environment. This thesis proposes generic architecture features for a HRI with Projection Mapping application with a 2-frequency laser pointer, as well as an implementation on a validation setup (ABB IRB140), the CARLoS robot, and in the AGV Jarvis.

The environment is mapped in black in Gazebo through CAD and .STL files and the objects the user wants to visualize are added. This is done this way to create the effect of object occlusions.

Image transformations are managed by placing a camera in the virtual world with the characteristics of the real life projector. This process outputs accurate projections over almost any surface, avoiding the need for costly engineering in its implementation.

Human-Machine interaction is done through a two-frequency laser pointer, both configurable (nowadays is set to 2.5 Hz and 3.5 Hz), and allows the user to select directly in the environment through dialog windows projected beforehand. All the system is develop with the help of ROS allowing abstraction and modularity.

Agradecimentos

A realização deste trabalho só foi possível graças a todas as pessoas e entidades que, direta ou indiretamente, contribuíram para o meu percurso pessoal e académico, tendo-me ajudado a superar este desafio. Às seguintes pessoas gostaria de deixar um agradecimento especial:

Ao meu orientador Prof. Dr. Armando Jorge Miranda de Sousa pelo seu apoio, constante disponibilidade, simpatia demonstrada, conhecimento transmitido e pelo incentivo prestado, que foram fundamentais para a realização desta dissertação.

Ao meu co-orientador Dr. Germano Manuel Correia dos Santos Veiga pela sua disponibilidade e apoio prestado durante o decorrer da tese.

Aos investigadores e técnicos do laboratório de robótica que estiveram sempre prontos a ajudar no que fosse preciso.

Aos meus amigos, que me acompanharam ao longo deste percurso nos bons e em especial nos maus momentos e que ficaram para sempre.

Por fim, agradeço à minha família, em especial aos meus pais, irmão e avós por todo o apoio que me deram e continuam a dar mesmo nos momentos de maior dificuldade.

Hugo Costa

“ There is no substitute for hard work”

Thomas A. Edison

Conteúdo

Lista de Figuras

Lista de Tabelas

Abreviaturas

2D	Duas dimensões
3D	Três dimensões
AGV	*Automated guided vehicle*
AR	Realidade aumentada - *Augmented reality*
CAD	Desenho Assistido por Computador - *Computer Aided Design*
DART	Animação dinâmica e *Toolkit* de Robótica - *Dynamic Animation and Robotics Toolkit*
DMX	Multiplexador digital
DoF	Graus de liberdade - *Degrees of Freedom*
EU	União Europeia
FOV	Ângulo de visão - *Field Of View*
GUI	Interface gráfica - *Graphical User Interface*
HRI	Interação humano robô
HSV	Tonalidade Saturação Valor - *Hue Saturation Value*
I&D	Investigação e Desenvolvimento
I&DT	Investigação e desenvolvimento técnico
ODE	*Open Dynamics Engine*
RGB	Vermelho Verde Azul - *Red Green Blue*
RGBD	Vermelho Verde Azul Distância - *Red Green Blue Depth*
ROS	*Robot Operating System*
PMEs SMEs	Pequenas e médias empresas
px	Píxel
TF	Transformada

Capítulo 1

Introdução

1.1 Contexto

A industria está repleta de manipuladores robóticos fixos que realizam tarefas árduas e repetitivas numa área de trabalho limitada e praticamente sem interação direta com utilizadores. Este tipo de configuração é particularmente útil em linhas de produção em série dado que cada robô tem uma área de trabalho limitada onde realiza a sua função sempre que um novo componente chega. No entanto, no caso de áreas de construção onde os robôs têm de ser movimentados e necessitam de interagir com o utilizador quer para receber ordens de trabalho, quer para o assistir na tarefa em questão, não existe nenhuma solução comercial.

Um robô móvel necessita de se localizar no espaço, ser capaz de planear as suas trajetórias e executá-las sem colidir com o meio nem com os seres humanos presentes e comunicar de forma bidirecional com os seus utilizadores. A interação entre humanos e robôs, principalmente manipuladores robóticos, ainda é uma área relativamente recente, mas em rápido crescimento. No início, os braços robóticos eram colocados dentro de jaulas do tamanho do seu volume de trabalho onde estes realizavam as suas tarefas sem qualquer tipo de interação, parando de forma imediata sempre que esse espaço fosse invadido. Posteriormente, algumas partes da jaula foram retiradas e substituídas por barreiras de sensores infravermelhos, mas o princípio de funcionamento continuava o mesmo, podendo em alguns dos casos existir interação em zonas e tempos predefinidos, segurança por segregação, ou seja, existe um espaço dentro do volume de trabalho do robô ao qual é permitido ao utilizador aceder durante determinado tempo. Mais tarde, o volume de trabalho que até aqui era considerado rígido passou a ser flexível permitindo uma interação mais imergente, através da criação de dois volumes de trabalho distintos, um no qual o utilizado pode permanecer e outro considerado crítico que força a paragem do robô sempre que invadido. Este tipo de interação implica a instrumentação do robô e ou do meio para perceção dos humanos que o rodeiam podendo ser aplicada por exemplo a robôs que seguem humanos e que param sempre que sejam tocados. No entanto, interação não implica necessariamente contacto, pois esta pode ser conseguida através da projeção de informação ou trabalho colaborativo.

1.2 Motivação

A interação entre humanos e robôs é uma área em constante desenvolvimento e dedicada a compreender e criar sistemas robóticos capazes de interagir com seres humanos [8].

Vídeo-projetores são uma solução económica e *off-the-shelf* para interface entre homem-máquina. Isto aliado a um robô bem localizado e com conhecimento do meio envolvente apresenta-se como uma solução muito promissora de interação entre humanos e máquinas.

Atualmente quando um trabalhador necessita de informação para realizar o seu trabalho tem de consultar um computador, que nem sempre se encontra perto do local de trabalho, ou papeis. Esta tarefa pode ser facilitada com recurso a um robô com um projetor, o qual poderia projetar diretamente na área de trabalho a informação requisitada que pode ser desde uma simples imagem, uma lista de procedimentos ou mesmo a projeção de dados de ficheiros CAD. Este sistema por si só implementa comunicação uni-direcional do robô para o utilizador, sendo necessário adicionar sensores tais como uma câmara para detetar movimentos do utilizador ou identificar um ponto laser para existir interação no sentido oposto.

Este sistema pode ser utilizado, por exemplo, para projeção das posições de soldadura de *studs*, para a fixação de linhas elétricas e canalizações, entre outros. É também possível ajudar o utilizador na realizações de reparações projetando no local de trabalho as intra-estruturas existentes, permitindo a sua localização imediata.

Este processo permite aumentar o rendimento do trabalho permitindo um acesso rápido e preciso à informação necessária.

1.3 Objetivos

Esta dissertação tem como objetivo o desenvolvimento de um sistema de interface homem-máquina bidirecional baseado em *projection mapping* para ambientes industriais. O sistema deve ser capaz de projetar imagens não distorcidas em qualquer superfície independentemente da posição em que o robô se encontre. Deve também ser capaz de implementar um sistema de comunicação humano para robô por intermédio de um apontador laser por forma a permitir ao utilizador escolher o que pretende visualizar.

Para atingir este objetivo será necessário realizar uma calibração precisa entre a câmara, o projetor e o referencial do mundo, bem como uma representação tridimensional do ambiente com todos os objetos necessários. A aplicação desenvolvida deve também ser capaz de projetar uma lista de objetos e identificar qual deles é selecionado pelo apontador laser. O sistema deverá utilizar ROS [9].

No final, o sistema desenvolvido será integrado no projeto CARLoS [6].

1.4 Estrutura do Documento

O presente documento encontra-se estruturado em mais 4 capítulos.

No capítulo 2, revisão bibliográfica, são apresentados os conceitos base, definições e metodologias de calibração de sistemas de visão e projeção. É também apresentado o conceito de *projection mapping* e metodologias de aplicação para superfícies planas e não planas. No fim é exposto o tema HRI (interação humano-robô).

No capítulo 3 são apresentados os projetos CARLoS [6] e Clarissa [10] e as seus requisitos. É também exposta uma solução para um sistema genérico para HRI com *projection mapping*.

No capítulo 4 é apresentada a solução implementada e a metodologia de calibração do sistema. Em seguida são expostos os resultados obtidos no âmbito de *projection mapping* e de interface com o robô.

Por fim, no capítulo 5 são apresentadas e discutidas as conclusões do trabalho e propostas algumas soluções que podem ser implementadas para melhorar o presente projeto.

Capítulo 2

Revisão Bibliográfica

Neste capítulo é apresentada revisão bibliográfica sobre sistemas de visão, sistemas de projeção, *projection mapping* e HRI (Interação humano-robô).

Os sistemas de visão são abordados na secção 2.1 onde é apresentado o modelo *pin-hole* e metodologias de calibração de parâmetros intrínsecos e extrínsecos. Em seguida, na secção 2.2 são apresentados sistemas de projeção bem como as tecnologias associadas.

O conceito de *projection mapping* é exposto na secção 2.3 onde é definido e apresentadas metodologias de projeção em superfícies planas e não planas, finalizando na apresentação de *software* comercial e gratuitos existentes.

Por fim, será apresentado o conceito de HRI e métricas associadas ao *design* e implementação de interfaces na secção 2.4.

2.1 Sistema de Visão

Uma câmara é um sistema ótico para captura imagens de luz do espetro visível ou outras porções do espetro eletromagnético.

É constituída por uma lente que a converge e foca para um meio sensível à luz, atualmente foto sensores eletrónicos (CCD ou CMOS), antigamente películas fotográficas, por um sistema que regula a quantidade e tempo de exposição, designado de obturador. Dependendo da câmara em questão estes parâmetros podem ser manuais ou automáticos e podem assumir diferentes gamas de valores tornando a câmara mais genérica ou mais específica para determinadas situações.

2.1.1 Modelo *Pin-hole*

Um modelo de câmara é uma função que mapeia os pontos tridimensionais dos objetos do mundo num plano bidimensional designado de plano de imagem. Este modelo tem por objetivo ser o mais próximo possível das caraterísticas da câmara real.

O modelo *pinhole* define que a abertura da câmara é modelada por um ponto infinitesimal e não considera a existência de lente. Desta forma, as distorções introduzidas pela lente não são

contempladas, implicando que a validade do sistema depende diretamente da qualidade da câmara em utilização.

O modelo consiste num ponto e num plano designados respetivamente de centro da câmara e plano de imagem. De uma forma simplista, o mapeamento de objetos 3D no plano de imagem resulta da interceção entre o plano e a reta que interliga o centro da câmara e o objeto no mundo, como se pode ver na figura 2.1.

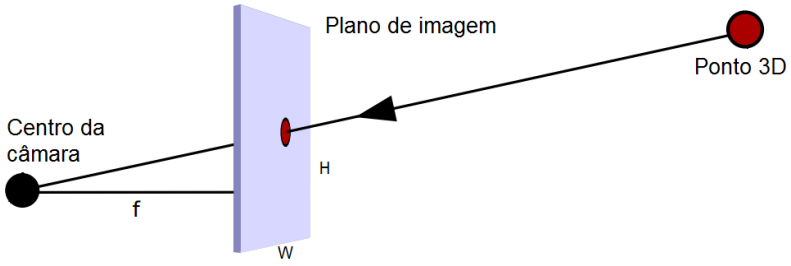


Figura 2.1: Modelo *Pin-hole* [1]

O plano de imagem tem a dimensão da resolução da câmara e num caso ideal a distância f, designada de distância focal, é dependente da abertura da câmara (fov).

$$f = \frac{W/2}{tan(fov/2)} \tag{2.1}$$

Considerando o referencial da origem da câmara ($W \times H$) paralelo ao referencial do plano de imagem com origem em $(0,0,f)^T$, como se pode observar na figura 2.2, verifica-se que a projeção de um ponto 3D é dada por:

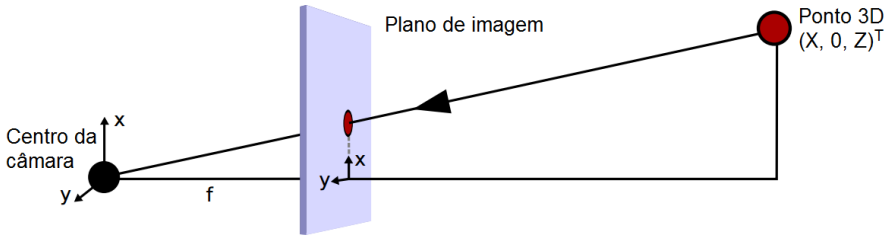


Figura 2.2: Projeção de um ponto 3D no plano de imagem [1]

$$\begin{bmatrix} x \\ y \end{bmatrix} \rightarrow f \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix} \tag{2.2}$$

A equação 2.2 é designada de projeção de perspetiva e apenas é válida para a premissa proposta. No caso de os referenciais se encontrarem em posições e ou orientações diferentes, a

posição do píxel pode ser definida por:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{2.3}$$

Onde f_x e f_y representam as distâncias focais e c_x e c_y as coordenadas do píxel central [1].

2.1.2 Calibração

Calibrar uma câmara corresponde a encontrar os parâmetros do modelo que melhor retratem as suas caraterísticas reais. Quanto mais precisa for esta calibração melhor será a interpretação das imagens e consequente interação com a mesma.

A calibração passa por duas fases. Inicialmente são calculadas as caraterísticas intrínsecas: distâncias focais, centro da imagem e parâmetros de distorção da lente que podem ou não ser considerados para o modelo dependendo do caso em estudo. Em segundo, calculam-se as caraterísticas extrínsecas que relacionam o referencial da câmara com o do mundo, ou seja, a matriz homogénea de transformação entre os dois referenciais.

Estes parâmetros podem ser calculados resolvendo o sistema de equações 2.6 [2].

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{2.4}$$

$$t = \begin{bmatrix} t_1 & t_2 & t_3 \end{bmatrix}^T \tag{2.5}$$

$$b = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R & t \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.6}$$

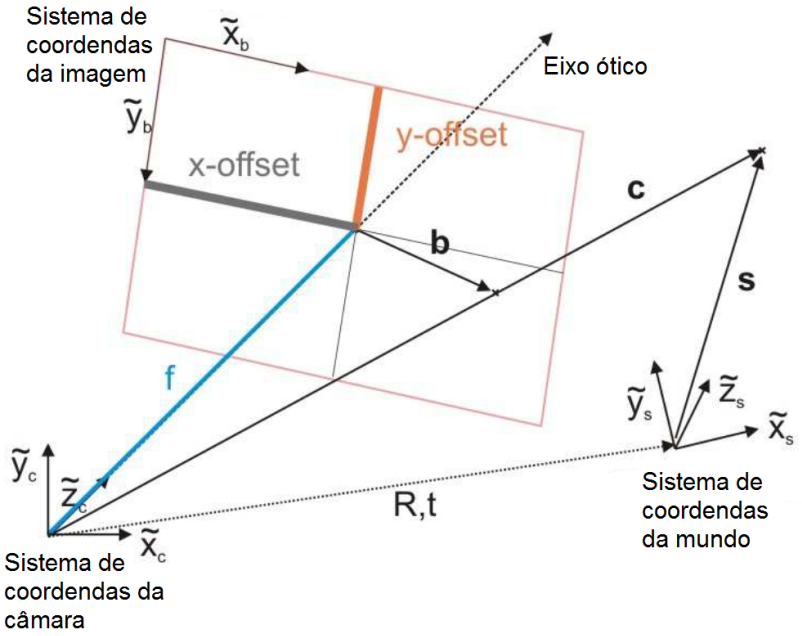


Figura 2.3: Modelo *pinhole* e caraterísticas intrínsecas e extrínsecas [2]

2.2 Sistema de Projeção

Um projetor de vídeo é um sistema capaz de projetar imagens em superfícies. Inicialmente desenhados para utilizações em cinema e apresentações onde as telas de projeção são planas ou ligeiramente arredondas são também utilizados em superfícies complexas para criação de ambientes envolventes e tridimensionais.

O sistema tem uma fonte de luz muito intensa e um mecanismo que reproduz a imagem que em seguida é ampliada e focada pela lente para ser projetada. A regulação de brilho da imagem é conseguida através do ajuste de luminosidade da fonte de luz. Devido ao seu propósito inicial e visto que continua a ser o que tem mais utilizadores, a grande maioria dos projetores possui sensores internos que determinam a sua posição para ajuste da imagem de saída.

O princípio de funcionamento é em tudo semelhante ao de uma câmara mas de forma oposta, em vez de adquirir imagens do meio, é capaz de as projetar nele. Desta forma, o modelo apresentado em 2.1.1 também é válido para o projetor.

2.2.1 Tecnologias

O princípio de funcionamento global é muito semelhante entre as diversas tecnologias. De uma forma geral, os projetores possuem um fonte de luz, um sistema de geração de imagem e uma lente para ajuste de imagem, sendo que as maiores diferenças encontram-se na forma de reprodução da imagem e no tipo de fonte de luz.

Listam-se algumas tecnologias existentes:

- **CRT (*Cathode ray tube*)** - Utiliza um ou três tubos de raios catódicos de alto brilho, na versão preto e branco e cores respetivamente, como meio de geração de imagem. O sistema envolve uma calibração precisa por forma às três imagens ficarem corretamente alinhadas;

- **LCD (*Liquid-crystal display*)** - A fonte de luz é decomposta em três componentes, vermelho, azul e verde, e encaminhada para painéis independentes que permitem a sua passagem com mais ou menos intensidade píxel a píxel. Por fim, as três componentes são sobrepostas e projetadas;

- **DLP (*Digital Light Processing*)** - Utiliza um ou três chips DMD (*Digital Mirror Device*), constituídos por pequenos espelhos, um para cada píxel, que oscilam por forma a refletir a quantidade de luz necessária e assim recriar a imagem. Na versão de um chip a luz atravessa um filtro de cor RGB e a imagem é projetada uma componente de cada vez; na versão de 3 utiliza um prisma para juntar as três componentes [11];

- **LED (*Light-emitting diode*)** - Utiliza uma das tecnologias acima descritas como forma de criação de imagem substituindo a fonte de luz por LED, visto esta ser mais económica e duradoura.

2.3 *Projection Maping*

Projection mapping, também conhecido como *video mapping*, é uma técnica de projeção que em vez de utilizar superfícies planas é capaz de mapear imagens em cenários complexos e convertê-los em ecrãs interativos. Esta técnica baseia-se na utilização de um ou vários vídeo-projetores a trabalhar em conjunto com um *software* que após adquirir uma representação 3D da cena mapeia as imagens [12].

Atualmente, esta técnica é muito utilizada por artistas, agências de publicidade e industria de jogos [13], no entanto, já começa a ser utilizada em outras áreas tais como a medicina [14], educação [15] e industria automóvel [16].

Embora o termo *projection mapping* seja relativamente recente, a sua história data de 1957 quando Morton Heilig construiu uma máquina chamada Sensorama, apresentada na figura 2.4. Foi desenvolvida como uma experiência cinematográfica para todos os sentidos, a imagem era projetada na frente e nos lados recriando um ambiente 3D estereoscópico, tinha som estéreo, vibração no acento e um soprador de ar apontado à face do espetador [17, 18, 19].

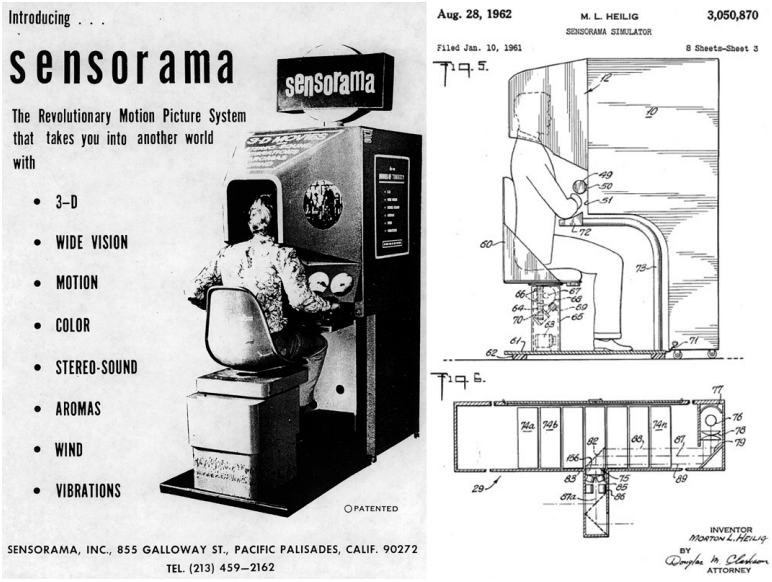


Figura 2.4: Sensorama [3]

A projeção sobre ecrãs não planos apenas apareceu pela primeira vez em 1969 na *Haunted Mansion* na Disneyland. Durante o percurso são apresentadas uma série de ilusões óticas incluindo 5 bustos cantores, apresentados na figura 2.5. Estes, cantavam a música do percurso e as caras dos cantores originais eram projetadas neles [4].

Figura 2.5: Haunted Mansion Singing Busts [4]

Em 1980 Michael Naimark [20] apresentou a obra *Displacements* [21], uma instalação cinematográfica imersiva. Foi montado o cenário de uma sala e uma câmara rotativa foi colocada no centro da sala filmando a interação entre dois atores e o meio. Em seguida a sala foi pintada de branco e câmara substituída por um projetor, resultando em *projection mapping* rotativo.

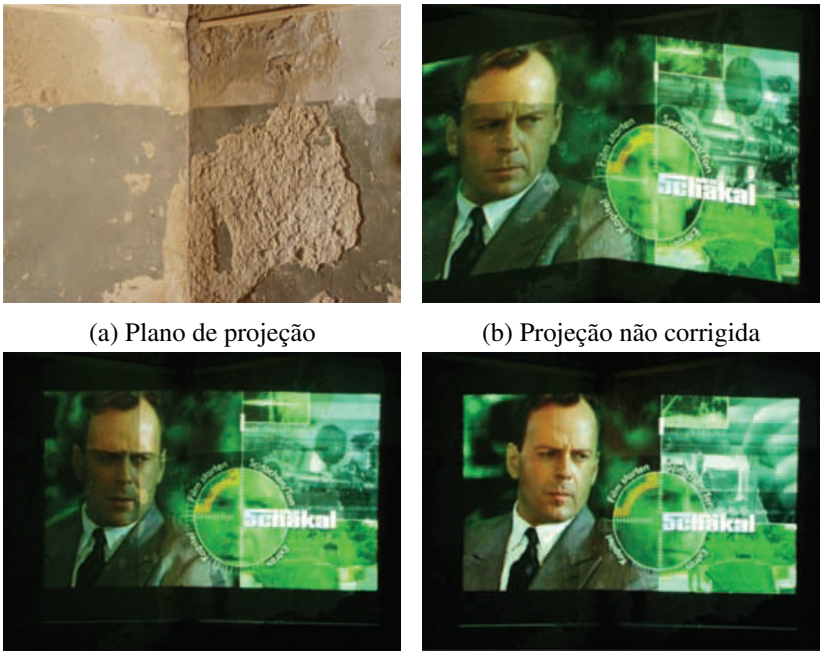
A maior evolução do *projection mapping* ocorreu quando começou a ser investigado academicamente. Um grupo de investigadores constituído por Ramesh Raskar, Greg Welch, Matt Cutts, Adam Lake, Lev Stesin and Henry Fuchs, apresentaram um projeto denominado de *The Office of the Future* [22] onde qualquer superfície pode ser utilizada como um ecrã.

> "The basic idea is to use real-time computer vision techniques to dynamically extract per-pixel depth and reflectance information for the visible surfaces in the office including walls, furniture, objects, and people, and then to either project images on the surfaces, render images of the surfaces, or interpret changes in the surfaces." [22]

Esta ideia potencia o uso de qualquer superfície como ecrã permitindo reuniões virtuais em tamanho real, sendo também possível interagir com aplicações de realidade aumentada com um sistema de seguimento de objetos e pessoas [22, 23].

Com a evolução dos vídeo-projetores e a sua redução de peso e dimensões, a mesma equipa desenvolveu um sistema para assistir na inventariação de armazéns. Nesta situação cada objeto teria uma *tag wireless* como identificador, sendo apenas necessário ao utilizador apontar o projetor para os objetos em questão para identificar de forma gráfica o objeto pretendido. Desta forma torna-se possível encontrar rapidamente itens e ao mesmo tempo atualizar a sua informação se necessário [24].

Mais tarde Oliver Bimber e Ramesh Raskar exploraram a área de projeção em pinturas e cortinas desenvolvendo algoritmos de deformação geométrica e correção de cor, apresentado na figura 2.6. Este avanço permitiu a projeção de imagens com baixa distorção em formato e cor em qualquer superfície [5, 25].



(a) Plano de projeção

(b) Projeção não corrigida

(c) Projeção com correção geométrica

(d) Projeção final com correção de geometria e cor[5]

Figura 2.6: Projeção de imagem sobre um canto

2.3.1 Técnicas de *projection mapping*

Normalmente, um projetor é utilizado para projetar imagens em ecrãs planos e são posicionados à frente dos mesmos por forma a evitar distorções na imagem. No entanto, recorrendo a técnicas de *projection mapping* é possível projetar corretamente uma imagem em quase todas as superfícies permitindo a criação de realidade aumentada e cenários imersivos.

Se se projetar uma imagem numa superfície não plana esta apenas vai estar correta de um ponto de vista e distorcida de todos os outros. Esse ponto de vista é a posição do projetor (P), no entanto, se se quiser que a imagem apareça correta do ponto de vista do utilizador (T) é necessário calcular a transformação geométrica entre este e o projetor para qualquer ponto no espaço (V), considerando que ambos são modelizados como modelos de câmara *pin-hole*.

Designando o projetor pelo seu centro de projeção P_c, e o seu plano de imagem por P_R, as coordenadas do píxel m_p resultam da interceção entre do vetor P_cM com o plano de imagem P_R, onde M é resultado da interceção de TV com a superfície D, apresentado na figura 2.7a. Esta transformação permite converter pixel a pixel do ponte de vista do utilizador para o do projetor. Desta forma é possível *renderizar* uma imagem em dois passos, apresentados na figura 2.7b:

1. Calcular a projeção da imagem na superfície do ponto de vista do utilizador;

2. Calcular a imagem anterior do ponto de vista do projetor.

Este método, no primeiro passo realiza a conversão tridimensional das coordenadas do mundo para bidimensional, imagem, do ponto de vista do utilizador. Este passo é conseguido por interceção do vetor TV com o plano de imagem m_T, pixel a pixel. No segundo passo é feito o mapeamento inverso da imagem para o ponto de vista do projetor, ou seja, os pixeis do plano de imagem m_T são convertidos para o plano de imagem m_p. Esta conversão é realizada diretamente sabendo a matriz homogénea de transformação entre as duas posições, utilizador e projetor.

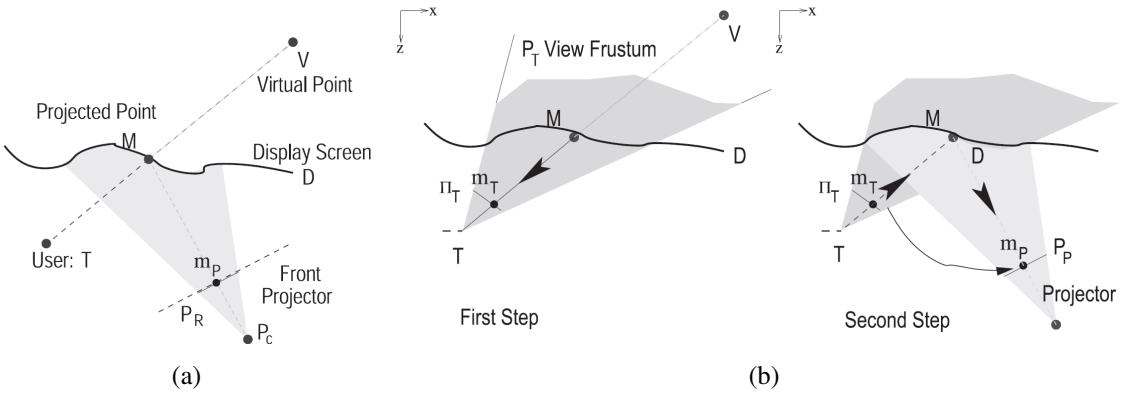


Figura 2.7: (a) Relação geométrica entre componentes; (b) Processo de *renderização* [5]

2.3.1.1 Homografia

Na área da visão por computador, duas imagens da mesma superfície plana estão relacionados por uma homografia, assumindo modelos *pinhole* para as câmaras. Uma homografia é uma transformação entre duas perspetivas diferentes que permite não remover este efeito mas também recriá-lo principalmente quando é preciso realizar projeções nesse mesmo plano e não é possível colocar o projetor solidário com o plano [26]

> "Um mapeamento de $P^2 \rightarrow P^2$ é uma projeção se e só se existir uma matriz H_{3x3} tal que para qualquer ponto em P^2 representado pelo vetor x, o seu ponto projetado será Hx." [27]

Isto significa que para calcular a homografia que transforma cada píxel m_T para o correspondente m_P apenas é necessário resolver a seguinte equação para pelo menos 3 pontos conhecidos:

$$m_p \cong H_{3x3} m_T \tag{2.7}$$

$$\begin{bmatrix} m_{Px} \\ m_{Py} \\ 1 \end{bmatrix} \cong \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \begin{bmatrix} m_{Tx} \\ m_{Ty} \\ 1 \end{bmatrix} \tag{2.8}$$

2.3.1.2 Projeção em superfícies planas

Se o projetor não estiver posicionado ortogonalmente ao plano de projeção a imagem final vai aparecer distorcida com efeito *keystone* de acordo com o ângulo. Atualmente, a maioria dos projetores tem um acelerómetro e giroscópio incorporados para poderem determinar a sua posição, e desta forma corrigir a deformação, dentro de uns determinados limites, assumindo que o plano de projeção é vertical e frontal.

Nos casos em que a correção efetuada pelo projetor não é suficiente ou se se pretender visualizar objetos tridimensionais do ponto de vista do utilizador é necessário aplicar técnicas de *projection mapping*. Por forma a utilizar o algoritmo apresentado, numa primeira etapa é necessário obter uma representação tridimensional do ambiente. O método mais utilizado consiste em fazer uma representação em rede polinomial (*mesh*), ou seja, um conjunto de vértices tridimensionais interligados entre si. Mas, este tipo de representação tem um compromisso entre qualidade e custo, isto é, quanto maior for o número de pontos maior fidelidade com a realidade, no entanto maior será o poder de cálculo e espaço de memória necessários. Mas, como neste caso a superfície de projeção é plana um elevado numero de pontos não aumenta a definição da cena porque estão todos contidos no mesmo plano.

No caso particular de o projetor estar ortogonal ao plano, a imagem projetada já se encontra representada corretamente podendo apenas ser necessário deslocá-la para coincidir com o ponto de vista do utilizador.

No entanto, se o projetor se encontrar numa posição oblíqua a imagem tem de ser processada por forma a aparecer corretamente. Isto, pode ser alcançado com o método dos dois passos apresentado no início da secção. Inicialmente calcula-se a posição de V no plano de imagem do utilizador e em seguida converte-se para o plano de imagem do projetor para esta ser exibida. Pode observar-se que a imagem resultante deste processo esta relacionada por uma homografia entre os dois pontos de vista do mesmo plano, descrita na secção 2.3.1.1 [5].

É importante notar que a matriz de homografia entre o projetor e o plano não é alterada enquanto as suas posições se mantiverem.

Este método pode ser mais eficiente se as duas passagens pela imagem fossem comprimidas numa só, poupando imenso tempo de processamento, especialmente se o trabalho em questão envolver altas resoluções e ou elevados *frame rates*. Isto pode ser obtido criando uma matriz de projeção *3x4* $\mathrm{H}\tilde{\mathrm{P}}_T$. $\tilde{\mathrm{P}}_T$ é a matriz de projeção resultante da pirâmide *view frustum* criada por T e os quatro cantos da imagem gerada pelo projetor na superfície. Para tal é então necessário criar uma versão de *4x4* das matrizes H e $\tilde{\mathrm{P}}_T$, denominadas de H_{4x4} e P_T. P_T transforma coordenadas 3D homogéneas em coordenadas 3D homogéneas normalizadas e H_{4x4} para transformar estas em coordenadas de píxel do projetor, mantendo os valores de distância intactos.

$$H_{4x4} = \begin{bmatrix} h_{11} & h_{12} & 0 & h_{13} \\ h_{21} & h_{22} & 0 & h_{23} \\ 0 & 0 & 1 & 0 \\ h_{31} & h_{32} & 0 & 1 \end{bmatrix} \tag{2.9}$$

$$P'_T = (H_{4x4} P_T) \tag{2.10}$$

$$\begin{bmatrix} m_{Px} \\ m_{Py} \\ m_{Pz} \\ 1 \end{bmatrix} \cong H_{4x4} P_T [V, 1]^T \tag{2.11}$$

Verifica-se assim que com uma única passagem pela imagem é possível fazer a sua renderização do ponto de vista do utilizador para o ponto de vista do projetor, e ao mesmo tempo evitando reamostragem de artefactos [5].

2.3.1.3 Projeção em superfícies não planas

Em superfícies não planas e caso se pretenda projetar imagens de objetos tridimensionais do ponto de vista do utilizador não é possível unificar os dois passos. Isso apenas é possível se a imagem a ser projetada só contiver objetos bidimensionais e estiverem sobre a superfície, tais como posições de furações e *studs*.

A aplicação do método das duas passagens implica que na primeira etapa a imagem seja mapeada no plano do ponto de vista do utilizador e na segunda para o projetor. Por forma a obter bons resultados é necessário garantir que as calibrações do sistema estejam corretas e o ambiente mapeado com a precisão requerida.

A criação de uma representação 3D do mundo pode ser feita de diversas formas, uma delas passa por adquirir uma nuvem de pontos do ambiente com recurso a sensores de distância baseados em luz estruturada ou tempo de voo, conjuntos de câmara-projetor, entre outros. No caso da maioria dos sensores de distância os valores adquiridos diretamente, por outro lado, no *setup* câmara-projetor é necessário projetar padrões de luz estruturada para se poder calcular as distâncias por intermédio de triangulação dos píxeis correspondentes. É de notar que a resolução dos padrões e a exatidão da calibração entre a câmara e o projetor, transformada e caraterísticas intrínsecas de cada elemento, irão limitar a definição das medidas geradas.

Por fim tem de se definir o referencial do mundo e a sua transformação em relação ao projetor, que caso seja o mesmo utilizado no mapeamento já é conhecido, e em relação ao utilizador.

As calibrações e a aquisição tridimensional do ambiente proporcionam todas as transformações geométricas entre os objetos em cena incluindo a posição do projetor e do utilizador que é constantemente monitorizada. Sempre que este muda de posição a imagem é *renderizada* segundo o método apresentado.

2.3.2 Ferramentas relacionadas

2.3.2.1 Gazebo - ROS

ROS (*Robot Operating System*) é um meta sistema operativo *open-source* dedicado ao desenvolvimento de *software* para sistemas robóticos.

Atualmente, esta *framework* é utilizada por diversos grupos de investigação robótica incluindo a FEUP dada a sua simplicidade e capacidade de repartir problemas de grande dimensão e complexidade em problemas mais pequenos e fáceis de resolver. Este feito é conseguido pelo facto de ROS implementar uma política baseada em publicadores e subscritores e serviços, com os seguinte componentes fundamentais:

- **Nós** - aplicação com a capacidade de subscrever e publicar mensagens em tópicos;
- **Mensagens** - forma pela qual os nós comunicação, estas podem ser standard ou específicas criadas pelo utilizador;

- **Tópicos** - "meio de transmissão"de mensagens entre nós que pode ser subscrito por vários nós;
- **Serviços** - semelhante ao modelo publicador/subscritor com a diferença de ser de um para um e a comunicação existir nos dois sentidos.

Com recurso a esta estrutura é possível dividir o projeto em várias camadas diferentes, criar abstração de *hardware*, separar o controlo de baixo nível do de alto nível, executar *software* em diferentes computadores e utilizar diversas linguagens de programação. ROS possui também ferramentas para guardar dados quer de simulações quer de ambientes reais e utilizá-los mais tarde sem a necessidade de montar todo o *setup* associado aos testes [9].

Gazebo é um simulador gratuito de alta fidelidade, inicialmente desenvolvido para simulação de robôs em ambientes exteriores mas também com grande desempenho para ambientes realistas interiores.

De entre todas as suas caraterísticas destacam-se:

- **Motor de física** - Suporta múltiplos motores de física de elevado desempenho destacando-se: ODE, Bulet, Simbody e DART;
- **Motor gráfico** - Providencia renderizaçãod e imagem de elevado realismo incluindo texturas e sombras devido ao motor gráfico OGRE;
- **Sensores** - Geração de dados de sensores simulados com e sem ruído tais como câmaras 2D e 3D, lasers *range finders*, sensores de contacto, força, entre outros;
- ***Plugins*** - Permite o desenvolvimento de *plugins* específicos para robôs, sensores e ambiente;
- **Modelos de robôs** - É possível utilizar os modelos já disponibilizados pelo simulador ou criá-los de raiz;
- ***TCP/IP Transport*** - Permite executar simulações em servidores remotos e comunicar com as mesmas através de comunicação por sockets usando *Protobufs* da Google;
- **Command Line Tools** - Permite a interação com a simulação através da linha de comandos;
- **ROS** - Comunicação com ROS.

Integrando o Gazebo com ROS é possível atingir os objetivos propostos, mapear uma imagem numa superfície complexa e convertê-la para o ponto de vista do projetor.

2.3.2.2 Software de *projection mapping*

Projection mapping está a despertar cada vez mais interesse, principalmente por parte de artistas como uma nova forma de exprimirem o seu trabalho.

Assim, uma série de software foram e estão a ser desenvolvidos, uns gratuitos e ou *open-source* e outros comerciais, para diferentes sistemas operativos.

VPT (VideoProjectionTool)

VPT é um software multiuso *realtime* para OSX e Windows desenvolvido pela HC Gilje.

É utilizado em projeção de imagens e vídeo em superfícies complexas para sistemas de múltiplos projetores, combinando cenas gravadas e ao vivo, podendo também integrar sensores ou sistemas de *tracking* para criar ambientes interativos [28].

TouchDesigner

TouchDesigner é uma plataforma de desenvolvimento gratuita e de uso não comercial para Windows para criação de projetos de imagem e vídeo.

Permite a criação de sistemas de vídeo interativos, *projection mapping*, efeitos de música visuais, com ligação a bases de dados e Internet, sensores e alguns tipos de iluminação. Tem também um motor de imagem 3D que permite a integração de vários projetores e modelização de objetos reais e texturas [29].

HeavyM

HeavyM é um sistema de *projection mapping* gratuito e pronto a utilizar para Windows e OSX.

Permite a criação de animações visuais e o mapeamento das mesmas para superfícies complexas. O software incluiu interação em tempo real com a cena através do rato do computador, animações e efeitos prontos a utilizar e áudio reativo. HeavyM tem uma comunidade onde os utilizadores podem partilhar o seu trabalho, expor e tirar dúvidas facilitando a aprendizagem mesmo para quem não possua bases nesta área [30].

Painting With Light

Painting With Light é um sistema de *projection mapping open source* para Windows, OSX e Linux.

Este software tem um princípio de funcionamento semelhante ao Paint, Gimp ou Adobe Photoshop, no entanto, o utilizador em vez de desenhar em superficies 2D, com recurso a um projetor, o desenho é feito em objetos tridimensionais reais [31].

Outros

A tabela 2.1 resume algumas das principais caraterísticas dos softwares acima mencionados e refere mais alguns.

Tabela 2.1: Softwares de *projection mapping*

	Gratuito	***Open Source***	**Windows**	**OSX**	**Linux**
VPT	Sim	Não	Sim	Sim	Não
TouchDesigner	Sim, para uso não comercial	Não	Sim	No	Não
HeavyM	Sim	Sim, para a comunidade	Sim	Sim	Não
Painting With Light	Sim	Sim	Sim	Sim	Sim
FacadeSignage	Não	Não	Sim	Não	Não
MXWendler	Não	Não	Sim	Sim	Não
MWM – Multi Window Mapper	Não	Não	Sim	Sim	Não

2.4 Interação Humano-Robô

Interação humano-robô (HRI) tem sido alvo de muita atenção a nível académico, em laboratórios de investigação, em empresas de tecnologia e por todos os meios de comunicação [32].

Dedica-se ao entendimento, *design* e avaliação de sistemas humano-robô para uso por humanos ou colaborativo. Existem dois tipos de interação distinguíveis [8]. O primeiro, a interação remota onde o robô e o humano estão separados espacialmente ou até temporalmente, e a segunda, a interação de proximidade onde o operador e o robô são colocados no mesmo espaço.

Robôs estão a preencher cada vez mais papeis na sociedade de hoje em dia, desde automação de fábricas, serviços, aplicações médicas ou até entretenimento [33]. Inicialmente, eram usados em tarefas repetitivas onde toda a informação era disponibilizada pelo humano à priori. Agora, estão cada vez mais envolvidos em cenários complexos e menos estruturados, e as atividades incluem interação com as pessoas necessárias para completar as tarefa designadas. É importante

perceber como funciona esta interação, as métricas comuns, onde se pode intervir, e como desenhar e implementar um sistema capaz de realizar as necessidades interativas requeridas pela tarefa, ou pelo ambiente.

A primeira vez que HRI foi introduzida pelo Isaac Asimov foram criadas três leis [34]:

1. *"A robot may not injure a human being or, through inaction, allow a human being to come to harm";*

2. *"A robot must obey orders given it by human beings except where such orders would conflict with the First Law";*

3. *"A robot must protect its own existence as long as such protection does not conflict with the First or Second Law".*

HRI possui diversos desafios. Por exemplo, *Multi-Modal Perception* é a perceção *real-time* e lidar com incertezas na aquisição vinda dos sensores. Outro desafio é o *Design and Human Factors*: O *design* do robô é um aspeto importante de HRI. A personificação, forma e nível de antropomorfismo, simplicidade e complexidade são áreas de principal relevância. Dentro destas categorias gerais, é útil decidir se a HRI se aplicará a uma aplicação que requeira mobilidade, manipulação física ou interação social. Interação entre humanos e robôs está presentes em toda a robótica, mesmo em robôs autónomos, e portanto, é necessário entender como desenhar uma interface que satisfaça as condições impostas para a tarefa.

Podem ser estudados cinco atributos de interação [8].

O primeiro, a autonomia, é o tempo que o robô pode ser negligenciado ou a tolerância a negligência do robô. Um sistema com um alto nível de autonomia é um que possa ser negligenciado por um longo período do tempo sem haver interação. Sheridan e Verplank [35], definiram uma escala de autonomia para um *designer* de HRI ter em atenção durante o projeto de uma interface:

1. O computador não oferece qualquer assistência; o operador faz tudo. - *Computer offers no assistance; human does it all.*

2. O computador disponibiliza um conjunto completo de ações. - *Computer offers a complete set of action alternatives.*

3. O computador restringe as alternativas apenas a algumas opções. - *Computer narrows the selection down to a few choices.*

4. O computador apenas indica uma opção. - *Computer suggests a single action.*

5. O computador executa a ação, se o operador aprovar. - *Computer executes that action if human approves.*

6. O computador apenas permite ao operador a tomada de decisão num espaço de tempo pré-determinado, antes da execução da ação de forma automática. - *Computer allows the human limited time to veto before automatic execution.*

7. O computador executa a ação de uma forma automática e informa o operador da sua execução. - *Computer executes automatically then necessarily informs the human.*

8. O computador apenas informa o operador da execução automática, se o operador o pedir. - *Computer informs human after automatic execution only if human asks.*

9. O computador apenas informa o operador da execução da ação apenas se este decidir que o deve fazer. - *Computer informs human after automatic execution only if it decides too.*

10. O computador decide e age autonomamente independentemente das ordens do utilizador. - *Computer decides everything and acts autonomously, ignoring the human.*

O robô ser apto para colaboração ponto a ponto depende também da sua capacidade de funcionar autonomamente quando necessário, daí ser mais difícil atingir esse tipo de colaboração do que programar o robô para ser completamente autónomo [35].

Um segundo componente, é a maneira como a informação é trocada entre os dois pontos da interação (humano-robô) a qual pode ser feita de diversas maneiras. Por exemplo, recorrendo a *displays* (GUIs) ou interfaces de realidade aumentada (AR) [36], gestos, voz ou interação física (lasers, botões, outros). A eficiência desta interação é dada pelo tempo que as instruções demoram a ser dadas ao robô, pela carga de trabalho da interação, pela quantidade de dados gerados para entender o estado do robô produzido pela interação, e a quantidade de entendimento da situação mútuo entre o robô e o utilizador.

O terceiro componente é a maneira como o robô interage com a sua equipa, que papel assume: "O robô é um colega, um assistente, um escravo. Reporta a outro robô, a um humano ou é completamente independente?" [33].

O quarto componente é a forma como o robô se adapta à tarefa, como é que ele aprende e como é que ele é ensinado. Este componente não é muito referido na literatura, pois o objetivo de HRI é minimizar o treino que o operador tem que receber para estar apto a realizar a tarefa cooperativa.

O componente final lida com a forma da tarefa, ou seja, perceber como a tarefa deve ser feita e como esta a ser feita corretamente. Estes processos incluem *goal-directed task analyses* e *cognitive work analyses* [33].

2.5 Conclusões

Neste capítulo foi abordado o modelo *pin-hole* para modelação de câmaras e projetores bem como o tema de calibração de parâmetros intrínsecos e extrínsecos.

Foi apresentado o conceito *projection mapping* e as suas utilidades no mundo atual bem como técnicas de projeção em superfícies planas e não planas de objetos do ponto de vista do utilizador. Foram também enumerados alguns software dedicados existentes e as suas principais caraterísticas e proposta a implementação destas técnicas com recurso a um simulador gráfico 3D.

Por foi definido conceito de HRI, as suas métricas e desafios associados a esta interação que cada vez é mais presente na sociedade atual.

Capítulo 3

Análise de requisitos no âmbito dos projetos de estudo

Neste capítulo é apresentado o projeto CARLoS na secção 3.1 e o projeto Clarissa na secção 3.2 dos quais é descrito o seu propósito e objetivos a ser atingidos. Na secção 3.3 é feito um resumo dos requisitos de ambos.

Por fim, na secção 3.4 é proposto um sistema genérico para implementação de HRI baseada em *projection mapping* e na secção 3.5 as suas limitações e algumas formas de as solucionar como trabalho futuro.

3.1 Projeto CARLoS

3.1.1 O projeto

> " O projeto CARLoS tem como objetivo aplicar tecnologias recente e inovadoras em robótica móvel cooperativa num cenário industrial de construção naval. O robô será construído de forma modular utilizando tecnologia *off-the-shelf*. O protótipo será demonstrado como um robô cooperativo para realização de operações *fit-out* dentro se estruturas de navios. Atualmente ainda não existe nenhuma solução autónoma para a realização destas tarefas." [6]

A utilização de robôs em grandes espaços fabris embora levante desafios ao nível da mobilidade, segurança e fiabilidade apresenta-se como uma boa solução para resolver tarefas árduas e repetitivas. Os estaleiros navais, no caso da construção de super-estruturas de navios, apresentam-se como um bom exemplo, onde a realização de tarefas *fit-out* que exigem trabalho manual duro e repetitivo estão sempre presentes. Dado que ainda não existe nenhuma solução para automatizar estes procedimentos nestes ambientes complexos, *the CARLoS project* tem como objetivo produzir um protótipo de um robô capaz de trabalhar em equipa para operações *fit-out* dentro de super-estruturas de navios.

Algumas das operações especificas que o projeto deverá cumprir:

- Navegação semi-autónoma dentro de estruturas de navios;
- Realizar soldadura de *studs* completamente autónomo com base em ficheiros CAD;
- Marcação de informação proveniente de ficheiros CAD para ajudar trabalhadores em operações *fit-out*;
- Robô fácil de programar e controlar por trabalhadores do estaleiro de construção naval.

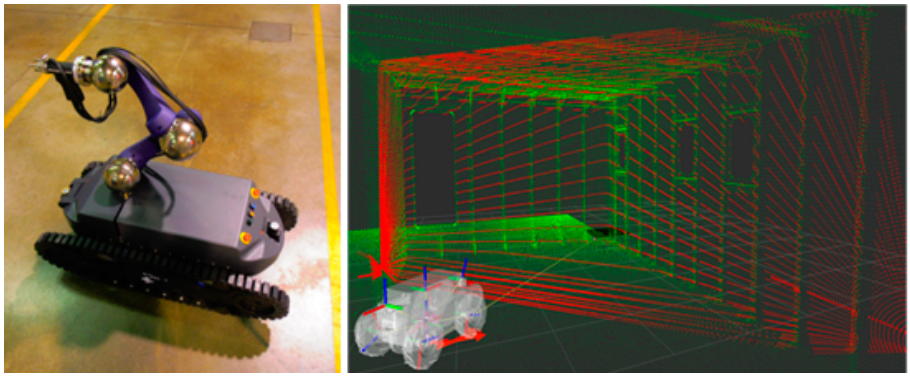


Figura 3.1: Robô CARLoS com pistola de soldadura (esquerda); Simulações numa representação 3D de um bloco de navio (direita) [6]

Durante os primeiros 9 meses de projeto a especificação funcional foi cumprida, o cenário 3D de testes foi selecionado e os principais componentes do robô escolhidos, montados e parcialmente testados como se pode observar na figura 3.1.

A arquitetura global de software foi definida consistindo em 3 camadas, primitivas, habilidades e tarefas. Algumas versões iniciais de *software* de navegação e execução de processos foram desenvolvidas e testadas apresentando resultados positivos.

Figura 3.2: Testes de soldadura com o manipulador [6]

Este projeto tem como objetivo automatizar a soldadura de *studs* por intermédio de um manipulador robótico como o da figura 3.2 e visualização de informação relevante proveniente de

ficheiros CAD projetada diretamente na área de trabalho. Desta forma é possível aumentar a produtividade, competitividade do negócio e minimizar as falhas [6].

3.1.1.1 Impacto na sociedade

Este projeto tem como objetivo aumentar a competitividade das empresas de construção naval por intermédio da automação de alguns processos morosos e repetitivos.

Além disso, será um passo em frente no que toca a gestão de processos. A este respeito, irá facilitar a implementação de processo inovadores em estaleiros navais mas também em outras áreas baseadas em construções metálicas, como é exemplo a construção civil.

Com o desenvolvimento do projeto *CARLoS* é também expectável que este contribua para fortalecer a posição de mercado das PMEs Europeias do ramo da tecnologia e eletrónica para aplicações industriais [6].

3.1.1.2 Consórcio

"THE CARLoS PROJECT"é apoiado pela comissão Europeia que juntou 4 PMEs e 4 I&DT de 4 países diferentes [6].

- AIMEN Technology Centre;
- AALBORG UNIVERSITET;
- INESC TEC - Instituto de Engenharia de Sistemas e Computadores;
- Robotnik Automation SLL;
- C.A.T. Progetti SRL;
- Deltamatic S.A.;
- ATEIN NAVAL S.A.;
- Astilleros José Valiña S.A.

3.1.2 Inter-relação com a dissertação

Esta dissertação tem como objetivo implementar um sistema de interação bi-direcional entre o utilizador e um robô móvel que possui um braço robótico com um vídeo-projetor e uma câmara.

O vídeo-projetor será utilizado para projetar a informação necessária utilizando técnicas de *projection mapping* e, em conjunto com a câmara, um sistema de comunicação bidirecional por identificação de um ponteiro laser.

O sistema utilizará ficheiros CAD como fonte de informação do ambiente envolvente e dados a serem projetados, tais como posição de *studs*, suportes para infraestruturas e criação de cenários virtuais de realidade aumentada para validação de projeto. Todos estes dados serão selecionados pelo utilizador podendo ser vistos em separado.

3.2 Projeto Clarissa

O projeto CLARiSSA [10], apresentado na figura 3.3, faz parte do projeto *SMERobotics*. A missão deste projeto é desenvolver um robô colaborativo habilitado para aplicações de montagem em ambientes industriais, mais concretamente, de fabricação de aço. O demonstrador CLARiSSA foi apresentado pela primeira vez na feira decorrida em 2014 (feira AUTOMATICA 2014). Este projeto é desenvolvido em cooperação com a Norfer, com a SARKKIS e com o INESC Porto.

(a)

(b)

Figura 3.3: CLARiSSA na Automatica 2014 (poster + *dual-arm*)

No momento de apresentação, o robô fez bastante sucesso devido a apresentar diversas inovações tais como planeamento automático de trajetórias para soldadura com braço robótico *"dual-arm"*, adaptação a variantes de peças por possuir interpretação CAD, HRI para controlo de qualidade e para completar informação em falta, melhoria continua de desempenho. Estas características permitem uma adaptação rápida e simples a novas variantes de produtos e permite também trabalhar com peças de diversos tamanhos sem reajustar o robô.

3.3 Requisitos dos projetos

Para cumprir com os objetivos dos trabalhos foram definidos os seguintes requisitos:

Projeto CARLoS

- Importar informação relativa ao ambiente e objetos;
- Identificar e classificar objetos dos ficheiros importados;
- Projeção de informação, imagens, sem distorções em superfícies não planas;
- As imagens projetadas devem manter-se estáticas independentemente da posição e orientação do projetor;
- Calibração do sistema projetor-robô-mundo;
- Interação (HRI) com o robô por intermédio de um ponteiro laser;
- Utilização da *framework* ROS;
- Comunicação com o robô móvel para aquisição de localização.

Projeto do clarissa

- Utilização de um manipulador robótico de dois braços, um para segurar nas peças e outra para soldar;
- Projeção de informação sobre os objetos de trabalho (vigas metálicas);
- O utilizador define os parâmetros de soldadura;
- Criação de uma interface gráfica para interação com o utilizador;
- Uso do *software* Eyeshot da devDept [37].

3.4 Sistema genérico para HRI baseada em *projection mapping*

Os projetos apresentados possuem requisitos muito semelhantes e assim sendo foi definida um arquitetura genérica capaz de os cumprir, a qual será depois implementada tendo em consideração o projeto em causa.

O conceito HRI (*Human-Robot interaction*), tal como o termo indica, é a existência de comunicação uni ou bidirecional entre um utilizador e um sistema robótico. Para atingir este objetivo a aplicação deve ser capaz de recolher informação do meio relativa a ordens do utilizador e localização, processar, analisar e agir em conformidade e por fim comunicar de volta através de *projection mapping* fechando a malha de comunicação.

Estas etapas devem ser executadas de forma sequencial, inicialmente a aplicação recolhe informações do meio relativas aos comandos do utilizador, comunica com o sistema robótico para saber a sua localização 6DoF e consequente posição do projetor no mundo e importa os ficheiros de dados associados ao ambiente em questão. Estes ficheiros são uma representação tridimensional do mundo que é constituído por tudo que rodeia o robô, desde as paredes e infraestruturas locais até aos objetos que o populam desde mesas e cadeiras em ambientes de escritório a outros robôs e obstáculos em meio industrial. É neste ambiente que o utilizador deve ser capaz de estabelecer comunicação com o sistema robótico podendo esta acontecer de inúmeras maneiras tais como: apontador laser, luvas instrumentadas, marcadores, sistemas de reconhecimento de voz e ou gestos, controladores genéricos (*joystick*, comandos da *Wii* ou *Playstation*), um ecrã tátil ou reconhecimento de toque no meio, até um rato de computador. De entre todas estas opões a solução adotada deve ter em conta o meio em questão, por exemplo se houver muita poluição sonora a utilização de comandos de voz pode não ser viável por outro lado gestos ou um ponteiro laser não são influenciados.

Em seguida, todos estes dados, que dependendo do tipo de interação podem ser imagens, nuvens de pontos, ficheiros de som, entre outros, têm de ser filtrados e processados por forma a identificar o comando certo. Estes despoletam ações internas para seleção de informação, de entre a importada dos ficheiros, a ser exibida ao utilizador a qual é convertida em imagens e processada de acordo com a localização do robô com algoritmos de *projection mapping* para se adaptar a superfície de projeção. A esta imagem podem ser adicionas caixas de diálogo para interação com o utilizador.

Por fim, a imagem gerada é projetada no meio e a interface gráfica atualizada com os novos dados, fechando a malha de comunicação desde o pedido até à sua execução e exibição.

Esta conceção é apresentada na figura 3.4.

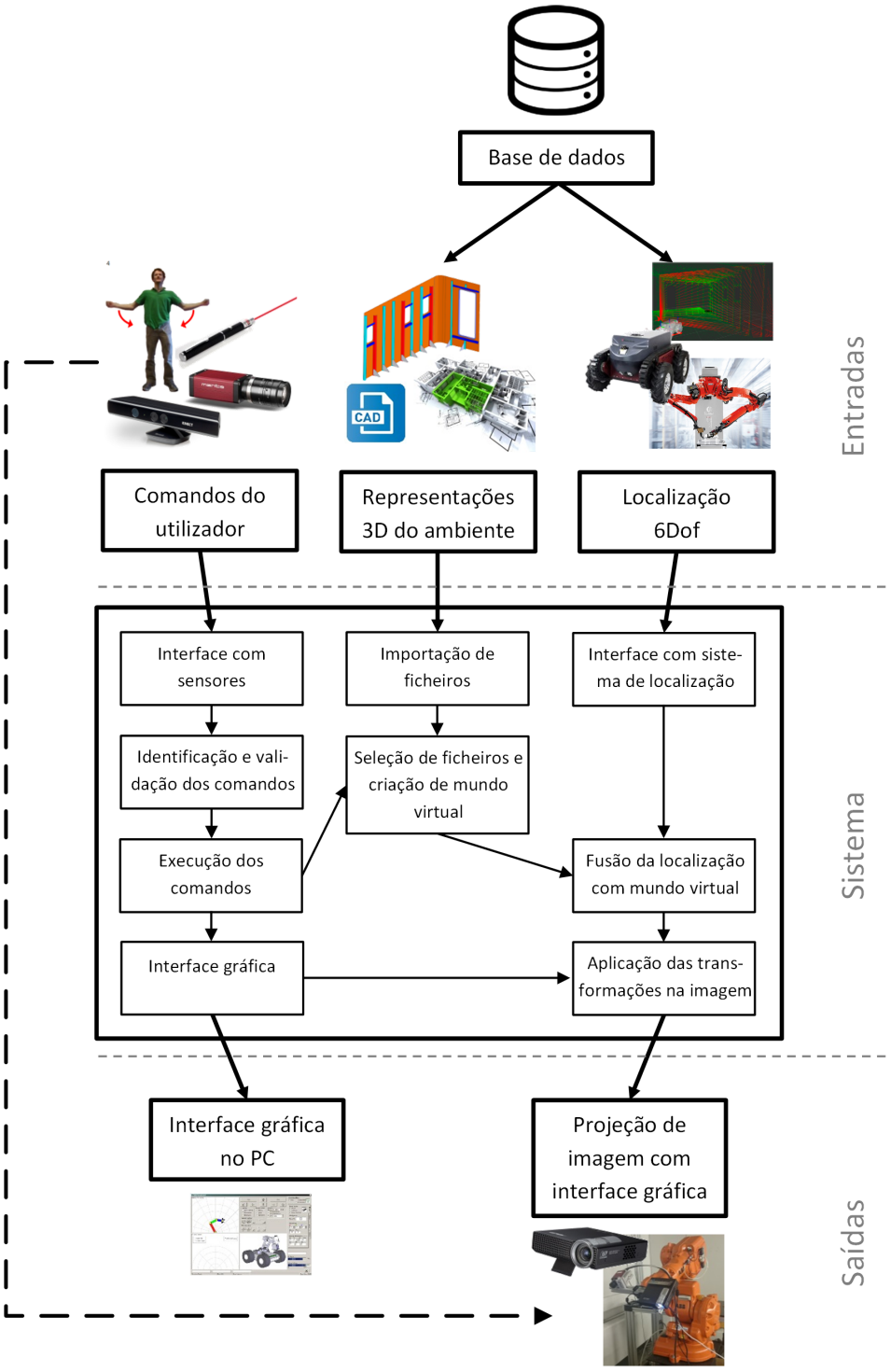


Figura 3.4: Arquitetura genérica

3.5 Validação e aumento da precisão da localização

O sistema desenhado tem como função principal implementar comunicação bidirecional entre o utilizador e o sistema robótico em causa. No sentido humano robô por intermédio de comandos que podem ir desde um simples rato de computador a reconhecimento de gestos, no outro sentido por intermédio de *projection mapping*.

As transformações aplicadas pelo sistema às imagens são em função do seu conhecimento do ambiente e da localização do projetor no mesmo, implicando que o meio deve estar de acordo com o importado pela aplicação e a posição do projetor deve ser precisa. Estas duas implicações levantam dois problemas não triviais, primeiro, é necessário garantir que o robô está bem localizado e segundo, que o ambiente se encontra de acordo com o do sistema. Caso qualquer uma das situações não se verifique a imagem gerada não vai ficar de acordo com o meio de projeção. Num primeiro caso porque o local de projeção não é o correto, num segundo porque está diferente como é exemplo a existência de uma nova janela ou uma parede que mudou de sítio e não foi atualizado no sistema.

Uma forma de combater estes problemas passa por ter meios de adquirir dados do ambiente e compará-los com dados teóricos.

Recorrendo a um sensor de distância tal como um *kinect*, um *Asus Xition*, um *Struct*, ou outro, obtém-se uma nuvem de pontos do local que em seguida pode ser comparada com o modelo teórico 3D, da sua posição no mundo, para verificação de correspondência. O mesmo é possível utilizando uma câmara associa ao projetor e usar este para projeção de padrões de luz estruturada e reconstrução 3D. Com este método é possível não só validar a localização mas também averiguar alterações do meio.

O processo pode também ser realizado com apenas uma câmara, no entanto a sua complexidade aumenta drasticamente. Neste caso é necessário renderizar o que a câmara estaria a visualizar num mundo teórico e compará-lo com a realidade, verificando alterações de estruturas, brilhos intensos entre outros aspetos.

Desta forma, é possível averiguar se o sistema se encontra em condições de realizar *projection mapping* na superfície de forma correta.

3.6 Conclusões

Neste capítulo foi apresentado o projeto CARLoS, o qual tem como objetivo aplicar tecnologias atuais e inovadoras em robótica móvel cooperativa em ambientes industriais, nomeadamente estaleiros de construção naval. O protótipo produzido deve ser capaz de realizar operações *fit-out* autonomamente dentro de super-estruturas de navios e cooperar e interagir com os restantes operadores.

Este projeto irá aumentar a competitividade das empresas de construção naval e fortalecer a posição de mercado das empresas a nível Europeu. Tem também o propósito de substituir operadores em funções árduas e perigosas.

O projeto Clarissa, tem como objetivo desenvolver um robô colaborativo habilitado para aplicações de montagens em ambientes industriais. Este, utiliza um manipulador robótico de dois braços para colocação de peças metálicas e soldadura.

Em seguida, foram apresentados os requisitos globais dos dois projetos, constatou-se que ambos necessitam de um sistema de HRI baseado em *projection mapping* como forma de facilitar e aumentar a eficiência na transmissão de informação visto esta ser apresentada sobre a área de trabalho.

Por fim, é exposta uma arquitetura do sistema genérico baseado em *projection mapping* que contempla a interação do utilizador com o meio e com o robô quer direta quer indiretamente, um sistema robótico, um conjunto de sensores e um projetor para interação com o meio.

Capítulo 4

Sistema Proposto

Neste capítulo é descrito o estado atual do projeto CARLoS à data de início deste trabalho na secção 4.1. Na secção 4.2 é apresentada a arquitetura do sistema implementado seguida de uma descrição pormenorizada das suas funcionalidades em 4.3 e limitações associadas em 4.4.

Na secção 4.5 são expostas as metodologias de calibração utilizadas para o sistema, câmara, projetor e transformações entre referenciais.

Por fim, na secção 4.6 são apresentados e discutidos os resultados obtidos.

4.1 Ponto de partida

À data de início do trabalho, o projeto CARLoS já se encontrava em fase de desenvolvimento, a maioria do *hardware* já tinha sido escolhida e testada, a estrutura de *software* estava definida (baseada em ROS) e uma pequena parte desenvolvida e testada, incluindo uns testes de *projection mapping* com o simulador gazebo.

Foi definido que o sistema de HRI baseado em *projection mapping* tem de utilizar ROS por forma a ser integrado com *software* existente e poder comunicar facilmente com os restante módulos do sistema (nós). Foi selecionado o simulador Gazebo visto ser compatível com ROS e apresentar as caraterísticas de processamento gráfico necessárias para realização de *projection mapping*. Como forma de comunicação do utilizador para o robô definiu-se a utilização de um apontador laser que pisque a duas frequências que representam clique direito e clique esquerdo para abertura de menu (clique direito) e escolha de opção (clique esquerdo).

4.2 Apresentação da arquitetura da solução atual

Com a definição dos objetivos para o desenvolvimento do projeto e obrigatoriedade na utilização de ROS, a estrutura do sistema genérico apresentada em 3.4 foi ajustada e otimizada para o projeto em causa mantendo a arquitetura global, exposta na figura 4.1.

O utilizador interage com o sistema através de um ponteiro laser que utiliza para selecionar os objetos que pretende visualizar ou mudar de posição. Por sua vez, o robô interage com o humano por intermédio de *projection mapping* conseguido por simulação 3D do ambiente real no gazebo.

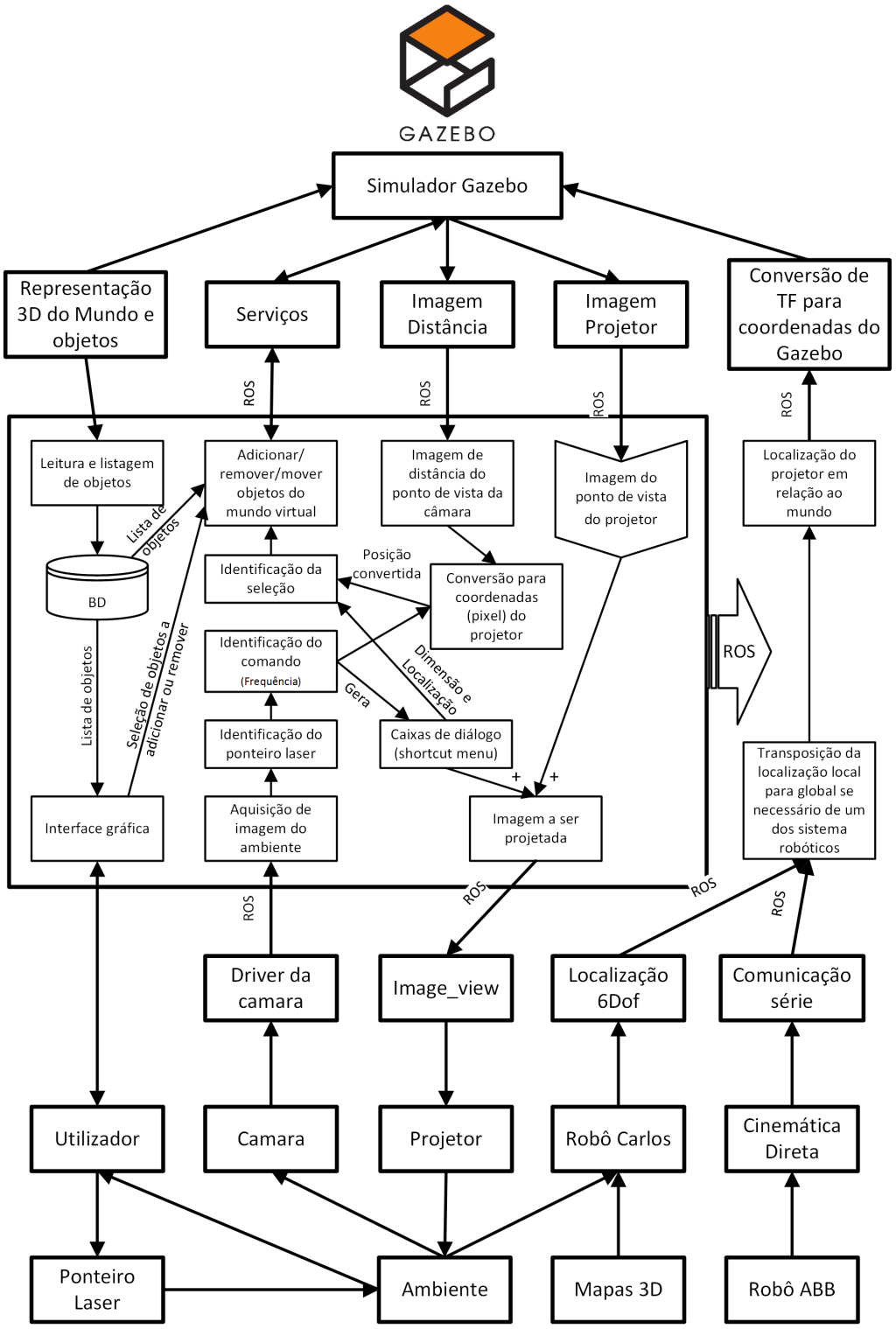


Figura 4.1: Arquitetura do sistema desenvolvido

O sistema tem também uma interface gráfica no pc que permite calibrar a identificação do ponteiro laser e importar os ficheiros de representações tridimensionais do mundo e dos objetos.

4.3 Sistema implementado

O sistema desenvolvido implementa HRI através de *projection mapping*, um apontador laser e uma interface gráfica no computador que permite importar, adicionar e exportar objetos. Foi desenvolvido um suporte ajustável para o projetor e câmara em calha técnica com a finalidade de manter a transformação entre ambos fixa e a mobilidade do *setup* entre diversos robôs. O projetor é utilizado para projeção de imagem sobre o ambiente e a câmara com a finalidade de identificação do ponteiro laser e calibração das transformações homogéneas entre os diversos referenciais.

A interface gráfica desenvolvida para o computador tem como propósito apenas ser utilizada na inicialização do sistema, para importação de todos os objetos necessários para o meio de trabalho em questão e calibração dos parâmetros de identificação do projetor. É também possível selecionar quais os objetos que se pretendem visualizar.

O mapeamento das imagens é conseguido através da recriação de um mundo virtual em preto (para este não aparecer na imagem projetada) com todos os objetos que o utilizador pretenda ver projetados no meio (instalação elétrica, posições de soldadura, tubagens, etc), com o mesmo referencial do sistema de localização do robô. O conjunto projetor-câmara é modelado por uma câmara e *kinect* virtual e são colocados na posição correspondente à real. Desta forma, a imagem adquirida pela câmara virtual que corresponde ao projetor e possui todas as transformações necessárias. O *kinect* retorna uma matriz de distâncias que é utilizada para identificação da posição tridimensional do apontador laser.

O apontador laser tem duas frequências de funcionamento, 2.5Hz e 3.5Hz que representam clique direito e esquerdo respetivamente, as quais são detetadas pelo sistema através de processamento digital de imagem. Com clique direito, o sistema adiciona à imagem projetada a lista de objetos importados, da qual o utilizador pode escolher os que pretende visualizar através clique esquerdo sobre o mesmo. Com clique esquerdo sobre um objeto projetado, o utilizador pode movê-lo para a posição desejada.

O projeto foi desenvolvido usando a *framework* ROS para comunicaçao entre os diversos módulos e a IDE Qt [38] visto ser compatível e possibilitar a criação de interfaces gráficas do tipo *drag and drop*.

O sistema implementado é constituído pelos seguintes blocos:

- **Interface gráfica no pc** - Calibração dos parâmetros de identificação do laser, importação dos mapas e objetos tridimensionais, adição e remoção dos mesmos no mundo virtual;
- **Gazebo** - Simulador gráfico 3D para *projection mapping* e identificação da posição 3D do laser;
- **Posição do projetor e câmara no mundo** - Comunicação com o robô para aquisição da sua localização e cálculo da posição do projetor e da câmara;

- **Identificação do ponteiro laser** - Aquisição de imagem e identificação da frequência e posição do laser no referencial da câmara;

- **Identificação de comandos** - Conversão da posição do apontador laser para o referencial do projetor e identificação da caixa de diálogo selecionada. Conversão da posição do apontador laser para o referencial do mundo, identificação do objeto selecionado e da sua nova posição (*drag and drop*);

- **Imagem projetada** - Fusão da imagem gerada pelo gazebo com as caixas de diálogo dos objetos.

4.3.1 Interface gráfica no pc

O sistema possui uma interface gráfica, apresentada na figura 4.2, permitindo ao utilizador importar ficheiros, comunicar com o simulador para adição e remoção de objetos e calibrar os parâmetros de identificação do apontador laser.

Os ficheiros são representações tridimensionais do mapa/ambiente e dos objetos que o constituem, sendo estes escolhidos pelo utilizador tendo em conta a área de trabalho do robô e as tarefas que este vai executar. Caso o(s) objeto(s) a importar tenham o mesmo referencial do sistema de localização do robô não é necessário preencher os campos de posição e orientação (estes são considerados zero), se tal não se verificar, o utilizador pode defini-los manualmente.

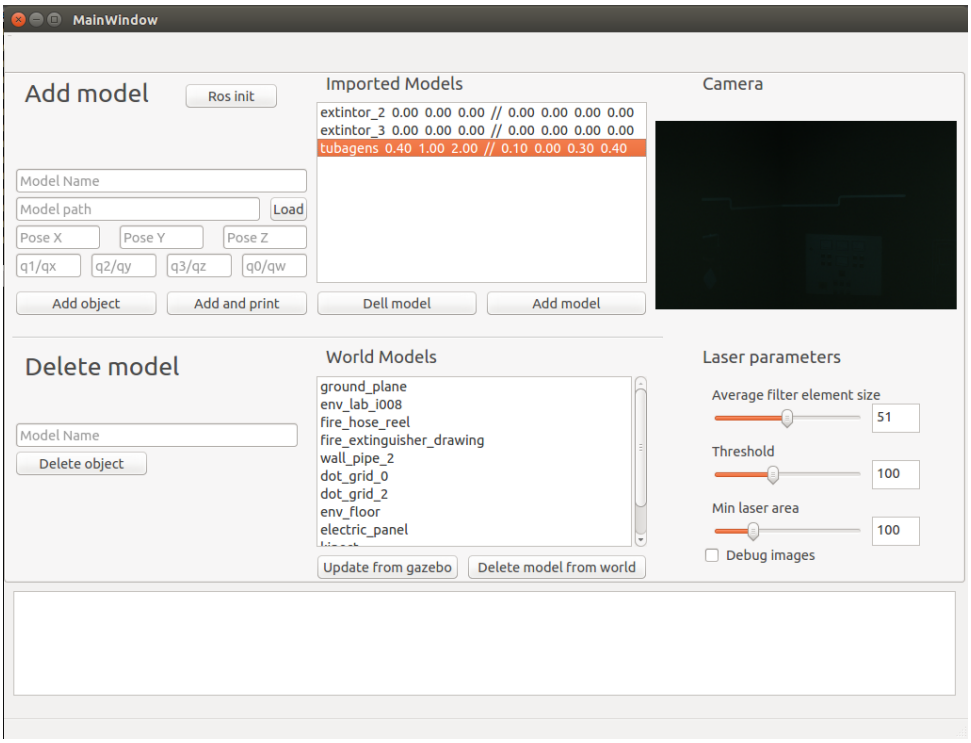


Figura 4.2: Interface gráfica

A interface gráfica tem duas listas de objetos, a superior onde são mostrados os objetos importados (*buffer*) e a inferior onde estão listados os que estão no mundo virtual. Ambas as listas são atualizadas sempre que existir alguma mudança (efetuada a partir desta interface gráfica) e do apontador laser. O sistema comunica com o simulador (gazebo) através de serviços de ROS, adicionando (da lista superior) ou removendo objetos (da lista inferior) de acordo com o que o utilizador pretende visualizar no momento. É também possível importar objetos na interface gráfica do gazebo, sendo necessário dar a conhecer à aplicação a sua existência. Para tal apenas é preciso clicar no botão *"Update from gazebo"* para o sistema efetuar um pedido ao simulador da lista dos objetos presentes.

No canto superior direito encontra-se a imagem adquirida pela câmara real e a posição do laser (círculo azul), sendo por vezes necessário calibrar os parâmetros para identificação do ponteiro laser. Este processo é realizado por intermédio de três barras deslizantes: dimensão do elemento do filtro de média, valor de *threshold*, área mínima do apontador laser. Por forma a tornar este processo mais rápido, é possível visualizar imagens intermédias do algoritmo de identificação (explicado em 4.3.4) selecionando a caixa *Debug images*. Estas não são mostradas em tempo de execução com o intuito de reduzir o tempo total do algoritmo e cumprir as metas temporais.

4.3.2 Gazebo

Gazebo é o simulador gráfico e físico tridimensional no qual é representado o mundo virtual de acordo com o que o utilizador pretende visualizar. Este mundo é acedido por intermédio de uma câmara e um *kinect* que simulam o projetor e a câmara reais respetivamente.

Em primeiro lugar, é importado para o mundo uma representação tridimensional do ambiente a preto (paredes, chão e teto), em segundo são colocados os objetos pretendidos. A representação do ambiente é feita a preto para que não "apareça" na imagem projetada, visto que preto é ausência de luz.

Em seguida são adicionadas a câmara e *kinect* virtual que têm as mesmas caraterísticas intrínsecas dos dispositivos reais correspondestes e nas mesmas posições. Desta forma, a imagem obtida na câmara virtual corresponde à do ponto de vista do projetor no mundo real, já com todas as transformações de perspetivas. O *kinect* virtual retorna uma matriz de distâncias que é utilizada para calcular a posição 3D do ponto laser no mundo e consequente transformação das coordenadas do referencial da câmara para o do projetor.

Dado que o sistema utiliza um simulador para recriar as transformações necessárias à imagem, o mundo virtual deve ser o mais idêntico possível ao real e ter o mesmo referencial do sistema de localização do robô, caso contrário é necessário calcular a posição o robô no novo mundo.

4.3.3 Posição do projetor e câmara no mundo

Projection mapping através de um ambiente simulado implica que a localização da câmara no mundo virtual seja o mais próxima possível da posição do projetor no mundo real e que as caraterísticas intrínsecas sejam idênticas. O mesmo é aplicável para a câmara real que é simulada

com um *kinect*. Para tal, todas as transformadas de referenciais devem estar bem definidas por forma a reduzir o cálculo necessário, o mundo virtual e o mapa de localização do robô devem ter o mesmo referencial, ou haver uma transformação conhecida entre ambos. Estas transformações são definidas em ROS como TF [39];

Por forma a que o conjunto câmara-projetor se mantenha estático entre si e em relação ao ponto de apoio no robô, foi desenvolvido um suporte rígido para a sua fixação, apresentado na figura 4.3. Além disso, permite uma fácil e rápida mudança entre robôs evitando algumas calibrações.

A posição do projetor é obtida pela soma das TF (transformações), que no caso do robô do projeto CARLoS são:

- **Mundo - robô** - TF proveniente da localização do robô;
- **Robô - base do braço robótico** - TF estática;
- **Base do braço robótico - *end effector*** - TF cinemática direta;
- ***End effector* - câmara** - TF estática;
- **Câmara - projetor** - TF estática.

no caso do braço robótico da ABB presente no laboratório I-110 da FEUP:

- **Mundo - base do braço robótico** - TF estática;
- **Base do braço robótico - *end effector*** - TF cinemática direta;
- ***End effector* - câmara** - TF estática;
- **Câmara - projetor** - TF estática.

no caso do AGV jarvis:

- **Mundo - Robô** - TF proveniente da localização do robô;
- **Robô - Apoio do suporte câmara-projetor** - TF estática;
- **Apoio do suporte câmara-projetor - câmara** - TF estática;
- **Câmara - projetor** - TF estática.

As TF são publicadas individualmente e a framework de ROS encarrega-se do cálculo das duas finais, mundo-projetor e mapa-mundo, as quais são convertidas para coordenadas de gazebo e enviadas para o mesmo.

Figura 4.3: Suporte sistema câmara-projetor

4.3.4 Identificação do ponto laser

O ponteiro laser é uma das formas de o utilizador interagir com o sistema após a criação do mundo virtual e importar todos os ficheiros de representações 3D dos objetos.

Numa fase inicial, foi implementada uma metodologia baseada em HSV (Tonalidade Saturação Valor), a imagem era decomposta nas três componentes e aplicado um *threshold* mínimo e máximo a cada uma individualmente, obtendo-se três imagens binárias. Em seguida era feito um *AND* bit a bit das três componentes e uma pesquisa pelo objeto que mais se aproximasse de um círculo e estivesse dentro das dimensões definidas. No entanto, esta metodologia necessitava de muitas calibrações de parâmetros e era muito dependente do meio, levando à identificação de falsos positivos.

A segunda metodologia implementada é baseada em remoção do fundo, binarização e identificação de objetos, apresentada na figura 4.4.

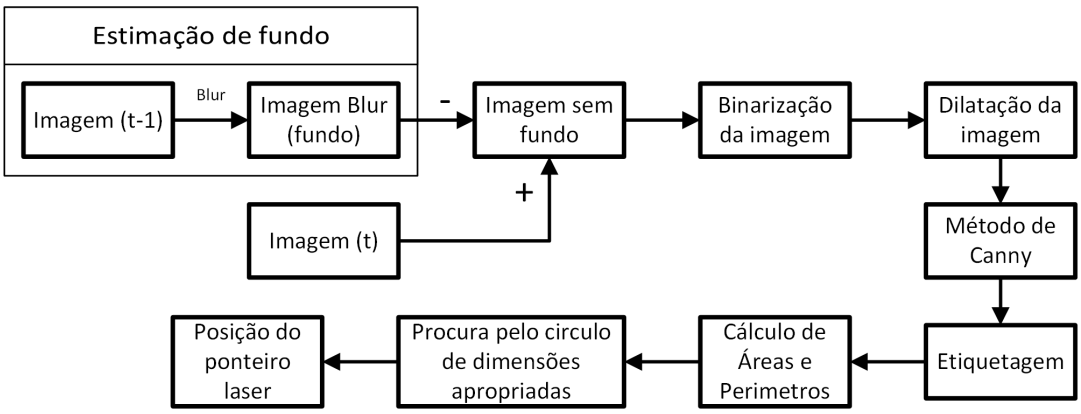
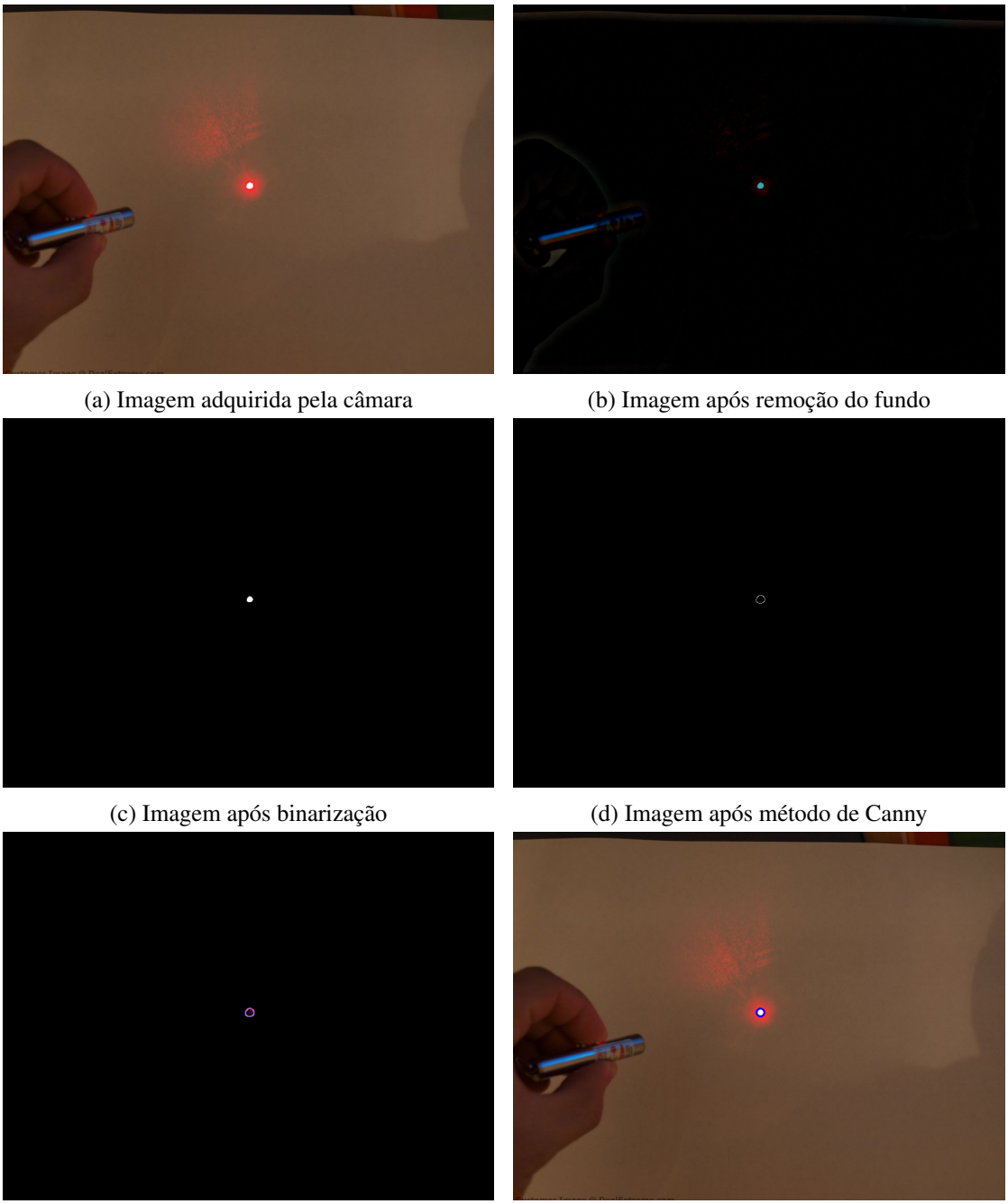


Figura 4.4: Processo de identificação do ponteiro laser

Na primeira etapa é estimado o fundo através de um filtro de média de grandes dimensões, da imagem $t-1$, cujo tamanho do elemento do filtro é definido na interface gráfica para se adaptar à resolução da câmara e dimensão do ponto laser. Em seguida, o fundo é removido através de uma subtração bit a bit entre a imagem atual e o fundo estimado da imagem anterior evidenciando as diferenças entre as duas, principalmente as mudanças bruscas de tonalidade tais como o ponto laser, com se pode observar na figura 4.5b. Esta operação é feita com a imagem anterior dado que por norma a interação é realizada com o robô parado e o que se altera na imagem é a posição do laser e a sua presença ou não (piscar), tornando o sistema mais robusto. A imagem é passada para preto e branco com *threshold* fixo definido pelo utilizador, figura 4.5c, aplica-se a operação morfológica de dilatação para remoção de ruído e o método de *canny* para deteção de orlas, figura 4.5d. Os objetos presentes são etiquetados e medidas as suas áreas e perímetros. De entre estes, é eleito o que apresentar uma área superior e o mais próxima de um valor mínimo configurável (na interface gráfica) e a relação área perímetro da equação 4.1 entre $1 \pm 20\%$.

$$rel = \frac{\sqrt{area/\pi}}{perimetro/(2 * \pi)} \tag{4.1}$$

Por fim, o centro do laser é identificado na imagem 4.5f por intermédio de um círculo azul, atualizada a sua posição e retornada a presença ou não do apontador.



(a) Imagem adquirida pela câmara

(b) Imagem após remoção do fundo

(c) Imagem após binarização

(d) Imagem após método de Canny

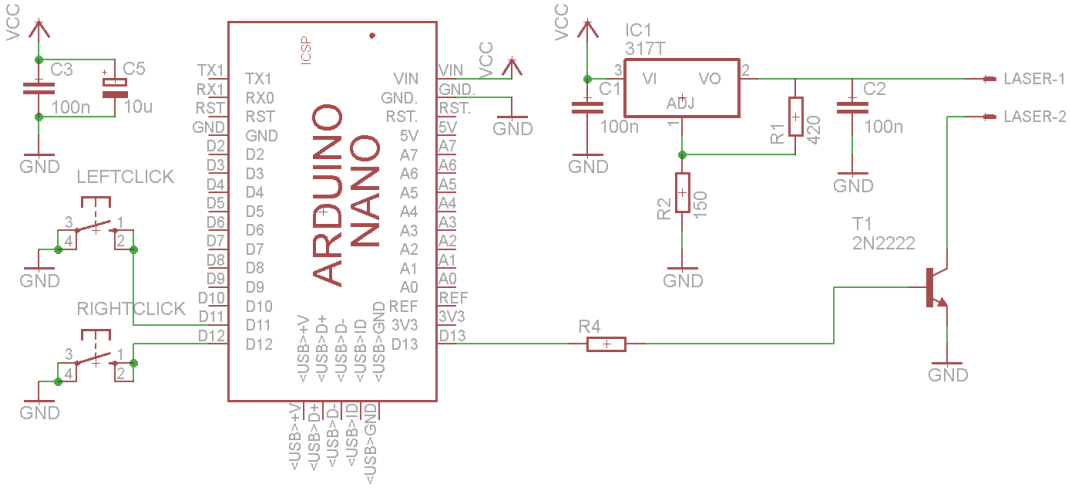
(e) Etiquetagem dos objetos

(f) Posição do ponteiro laser

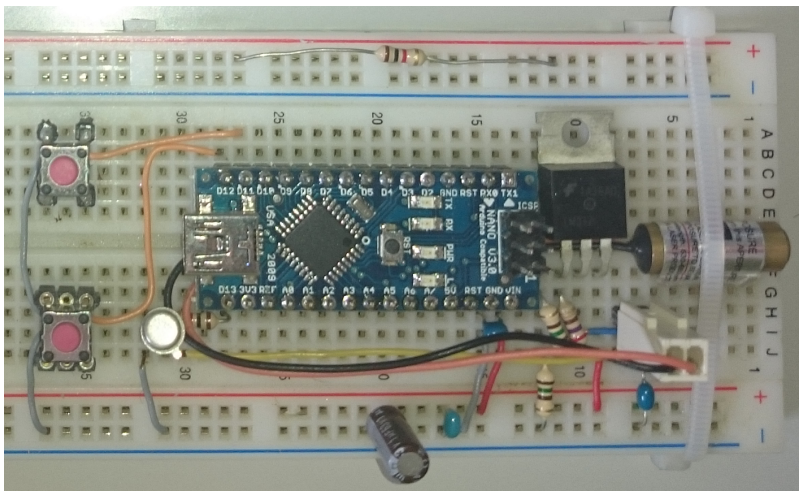
Figura 4.5: Identificação da posição do ponteiro laser

4.3.5 Identificação de comandos

Por forma a tornar a interação do utilizador mais intuitiva, o ponteiro laser tem dois botões que fazem com que este pisque a frequências diferentes, simulando o clique direito (2.5 Hz) e esquerdo (3.5 Hz) de um rato de computador. Este módulo foi prototipado numa *breadboard* com um Arduino Nano [40], como se pode visualizar na figura 4.6. O botão direito tem a funcionalidade de abrir e fechar as janelas de diálogo e o esquerdo de selecionar.



(a) Esquema elétrico



(b) Montagem do circuito em *breadboard*

Figura 4.6: Apontador laser com duas frequências

O sub sistema de identificação de comandos recebe a posição e presença do laser e a matriz de distâncias proveniente do simulador (*kinect*) e implementa o algoritmo apresentado na figura 4.7, proporcionando duas grandes funcionalidades, adição de objetos à cena e alteração da posição dos mesmos.

Numa primeira etapa é identificada a frequência a que o laser pisca através da média de tempos entre flancos descendentes na sua deteção. Em seguida, é validada se durante esse processo a posição do laser não variar acima de 10% da largura da imagem, por forma a evitar a deteção de falsos positivos.

A adição de objetos à cena é realizada em dois passos, inicialmente com clique direito é gerada uma lista de caixas de diálogo, cada uma com o nome da lista do *buffer* de objetos importados (Lista superior na GUI), e adicionada à imagem a ser projetada. Em seguida, o utilizador pode escolher o objeto que pretende visualizar apontando para a caixa de diálogo correspondente e clicando no botão esquerdo. No entanto, as coordenadas da posição do laser estão no referencial da câmara e as das caixas de dialogo no do projetor, tornando-se necessário converter a posição do ponto laser para o referencial do projetor. Para tal, a posição em píxeis do ponto laser é convertida para coordenadas 3D com origem no referencial da câmara, a componente em Z ($Pos_{3D}.Z$, sentido do eixo ótico) é obtida da matriz de distâncias do *kinect*, e as restantes de acordo com a posição do píxel em relação ao centro da imagem.

O cálculo da posição tridimensional segundo o referencial da câmara é realizado de acordo com as seguintes equações:

$$Pos_{3D}.X = Pos_{3D}.Z * \frac{PosLaser_x - CameraC_x}{cameraF_x} \tag{4.2}$$

$$Pos_{3D}.Y = Pos_{3D}.Z * \frac{PosLaser_y - CameraC_y}{cameraF_y} \tag{4.3}$$

onde $CameraC_x$, $CameraC_y$, $cameraF_x$ e $cameraF_y$ são os valores da matriz de caraterísticas intrínsecas da câmara.

A conversão das coordenadas para o referencial do projetor é realizado de acordo com as seguintes equações:

$$PosProjetor = PI * PE * Pos_{3D} \tag{4.4}$$

$$\begin{bmatrix} PosProjetor.x \\ PosProjetor.y \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} r_{00} & r_{01} & r_{02} & h_0 \\ r_{10} & r_{11} & r_{12} & h_1 \\ r_{20} & r_{21} & r_{22} & h_2 \end{bmatrix} * \begin{bmatrix} Pos_{3D}.X \\ Pos_{3D}.Y \\ Pos_{3D}.Z \\ 1 \end{bmatrix} \tag{4.5}$$

onde, PI e PE são as matrizes de caraterísticas intrínsecas e extrínsecas, em relação ao referencial da câmara, do projetor.

Na imagem projetada é desenhado um círculo vermelho na posição calculada como forma de *feedback* para o utilizador e de validação de posição. Em seguida, é verificado se esta posição está contida na área de alguma caixa de diálogo. Se tal se verificar, as caixas de diálogo são apagadas da imagem e enviados comandos para o gazebo por intermédio de serviços ROS para apagar os objetos presentes que façam parte da lista e adicionar o selecionado. Caso as caixas de diálogo já

estejam visíveis e o utilizador fizer clique direito, estas são removidas à imagem e o sistema volta ao estado inicial.

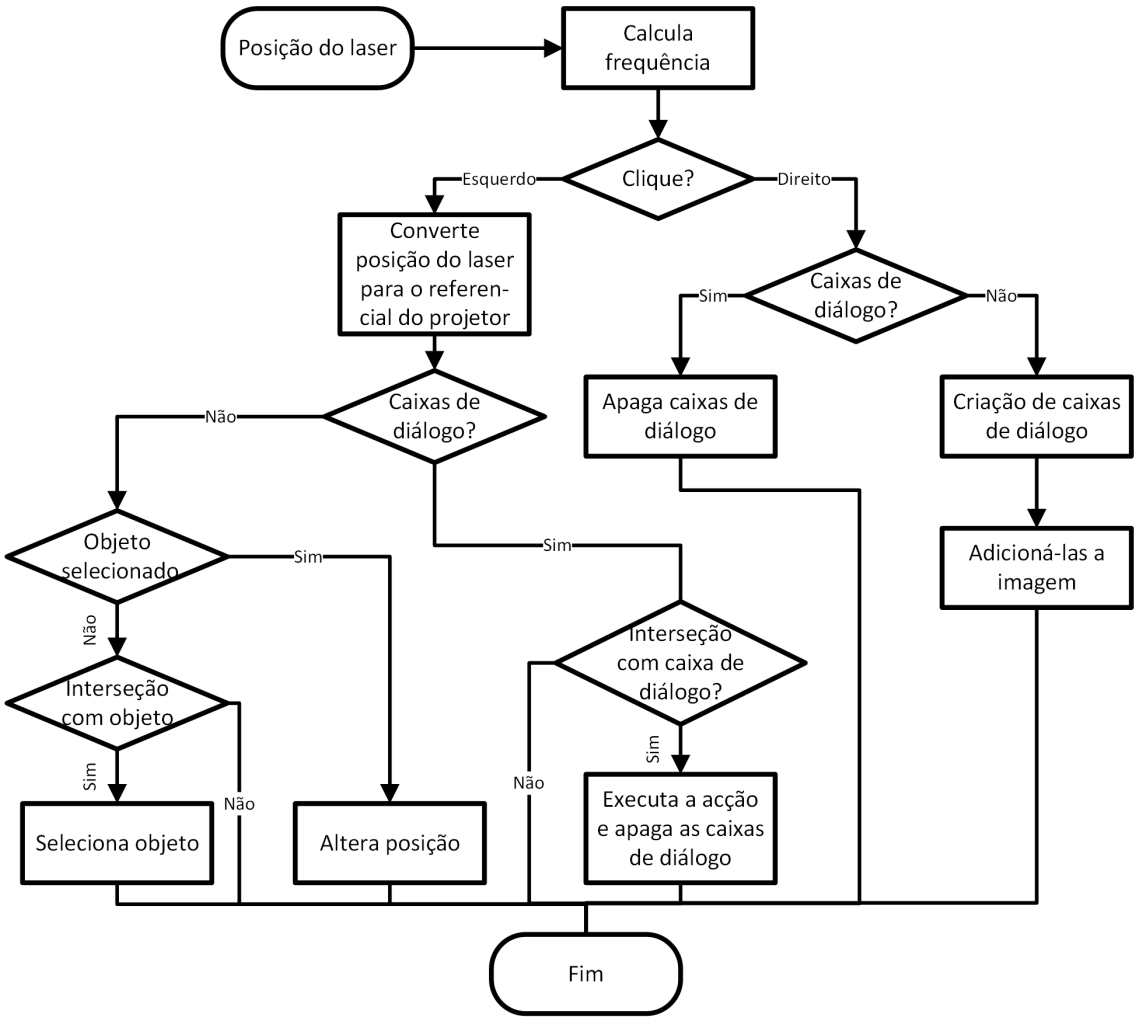


Figura 4.7: Processo de identificação de comandos

A movimentação de objetos sobre as paredes da cena é conseguida através da seleção do objeto pretendido (clique esquerdo) e identificação da nova posição com um novo clique (esquerdo). Este processo é semelhante ao de adição de objetos, no entanto, é verificado se o ponto laser está sobre um objeto da cena tridimensional através da conversão da sua posição 2D para 3D em relação à origem do mundo. Esta conversão é feita com TF, a posição do laser em relação à câmara (obtidas nas equações 4.2 e 4.3) é publicada numa TF e subscrita a TF que relaciona a origem do mundo com a posição do laser. Em seguida, a posição é ajustada para que coincida sobre a parede, de acordo com um dos seguintes conjuntos de equações por forma a minimizar o erro.

$$roll = atan2(2 \times (q0 \times q1 + q2 \times q3), 1 - (2 \times (q1^2 + q2^2))) \tag{4.6}$$

Se $-\pi/4 \leq roll \leq \pi/4$ ou $3\pi/4 \leq roll \leq \pi$ ou $-\pi \leq roll \leq -3\pi/4$

$$\begin{bmatrix} PosObj_{3D}.X = PosLaser_{3D}.X \\ PosObj_{3D}.Y = tan(roll) \times (PosLaser_{3D}.X - Obj.X) + Obj.Y \\ PosObj_{3D}.Z = PosLaser_{3D}.Z \end{bmatrix} \tag{4.7}$$

Senão:

$$\begin{bmatrix} PosObj_{3D}.X = (PosLaser_{3D}.Y - Obj.y)/tan(roll) + Obj.x \\ PosObj_{3D}.Y = PosLaser_{3D}.Y \\ PosObj_{3D}.Z = PosLaser_{3D}.Z \end{bmatrix} \tag{4.8}$$

Em seguida, é identificada a nova posição e comunicado ao gazebo por intermédio de serviços ROS as alterações pretendidas para o objeto em causa, as quais têm efeito imediato e o utilizador pode observar o resultado, procedendo a novas alterações se necessário.

4.3.6 Imagem projetada

A imagem a ser projetada no ambiente real é resultado da simulação 3D com o correto posicionamento de todos os elementos (objetos, câmara e *kinect*) com a sobreposição ou não da interface para o utilizador. Esta é constituída por caixas de diálogo retangulares com o nome do objeto centrado. O circulo vermelho indica a posição do laser nas coordenados do projetor.

A imagem projetada, figura 4.8 e 4.9, vai ser tão precisa quanto mais precisa for a localização do robô, a fidelidade do ambiente simulado em relação ao real e a correspondência das caraterísticas da câmara e *kinect* simulados às reais.

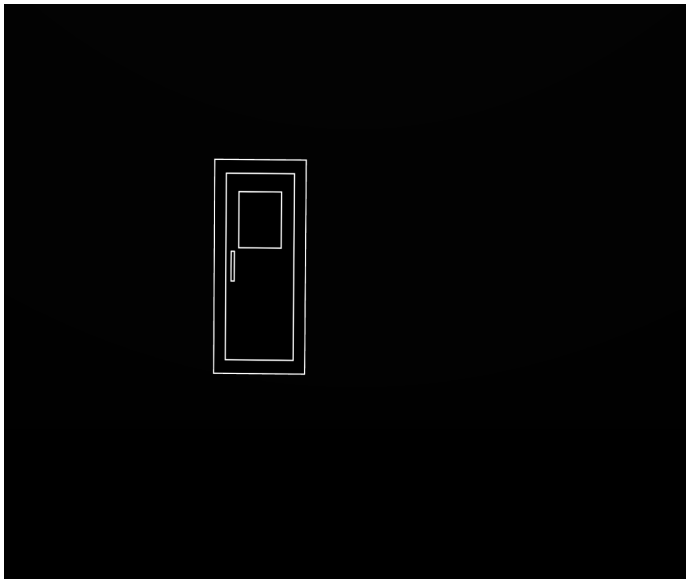


Figura 4.8: Imagem para projeção retornada pelo gazebo

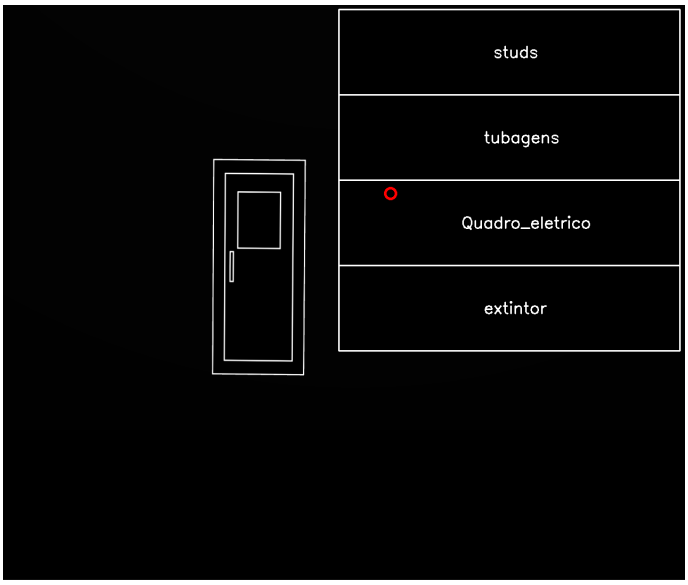


Figura 4.9: Imagem para projeção com caixas de diálogo

4.4 Limitações do sistema

Uma das grandes limitações é o facto de o sistema estar dependente da localização externa e consequentemente da sua precisão. Dado que o sistema assenta na localização para gerar a imagem a ser projetada por intermédio de um simulador, no caso de o erro de localização ser considerável, o resultado final não vai estar de acordo com a superfície de projeção. Mais ainda, a conversão das coordenadas do ponteiro laser da câmara para o projetor e para o referencial do mundo depende também diretamente dessa localização, podendo implicar uma má identificação de comandos.

Por forma a tornar o sistema mais robusto, este poderia fazer uma leitura tridimensional da área de visão do projetor e fundir estes dados com os da localização recebida, validando a posição e aumentando a sua precisão.

Outra limitação é o ponteiro laser não funcionar em ambientes espelhados ou transparentes, e em locais onde a luminosidade for muito elevada.

Este problema poderia ser resolvido integrando diversas formas de interação tais como identificação de gestos e movimentos corporais, reconhecimento de voz entre outros. Desta forma, o utilizador poderia escolher o que melhor se adaptasse ao ambiente em causa.

4.5 Calibrações do Sistema

Para o sistema funcionar corretamente é necessário calcular as matrizes de caraterísticas intrínsecas e extrínsecas da câmara e projetor e todas as transformações estáticas.

Sendo relevante extrair informação métrica das imagens, é necessário passar pelo processo de calibração de câmaras e projetor. Esta calibração descreve o processo de encontrar parâmetros das câmaras e projetores de forma a realizar transformações entre as coordenadas do mundo e imagens e vice-versa. Calcula também certas propriedades da lente tal que seja possível trabalhar com elas como se tivessem sido tiradas por um modelo de câmara *pinhole* perfeito. Os parâmetros intrínsecos estão relacionados com as características internas (ponto e distâncias focais) enquanto que os extrínsecos relacionam as transformações necessárias (no caso do *setup* de validação, são as transformações câmara-padrão xadrez e câmara-projetor).

Para calibração das restantes transformações homogéneas, o suporte câmara-projetor foi fixado a uma braço robótico da ABB, como se pode ver na figura 4.3 e colocado um padrão de xadrez horizontal no seu volume de trabalho. Inicialmente é definida a transformada entre o padrão de xadrez e a base do robô, da base do robô para o *end-effector* (por cinemática direta), da câmara para o padrão de xadrez ficando apenas em falta a do *end-effector* para a câmara.

A transformação entre a câmara e o projetor é obtida por projeção de padrões de luz estruturada sobre um padrão de xadrez impresso e triangulação de pontos entre a imagem projetada e adquirida.

4.5.1 Calibração dos parâmetros intrínsecos da câmara

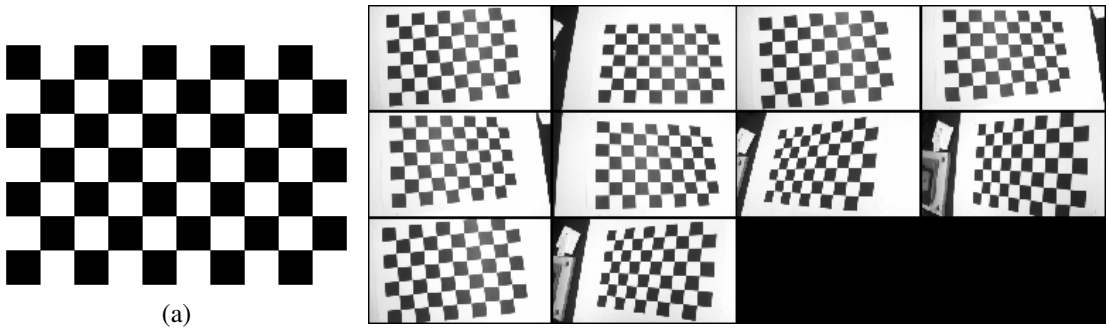
As câmaras não são dispositivos perfeitos e, como tal, necessitam de ser calibradas para sistemas de aquisição e processamento de imagem. Neste caso em concreto, a calibração permite que as suas caraterísticas sejam reproduzidas o mais fielmente num ambiente virtual.

A calibração de parâmetros intrínsecos tem por objetivo determinar as caraterísticas internas da câmara, o píxel central da imagem (C_x, C_y), a distância focal (f_x, f_y) e a distorção introduzida pela lente (k_1 k_2 p_1 p_2 k_3). Num caso ideal, o píxel central seria exatamente metade da resolução da imagem nas duas componentes (x, y) e as distâncias focais seriam iguais, dado que o referencial do centro da câmara seria paralelo ao plano de imagem e determinadas por:

$$f_x = f_y = \frac{C_x}{tan(fov/2)} \tag{4.9}$$

Num caso real, os parâmetros podem ser determinados resolvendo o sistema de equações 2.6. Para tal, foi utilizada a *toolbox* de calibração de câmaras para Matlab [41] que permite o cálculo das caraterísticas intrínsecas e extrínsecas da câmara bem como o erro associado.

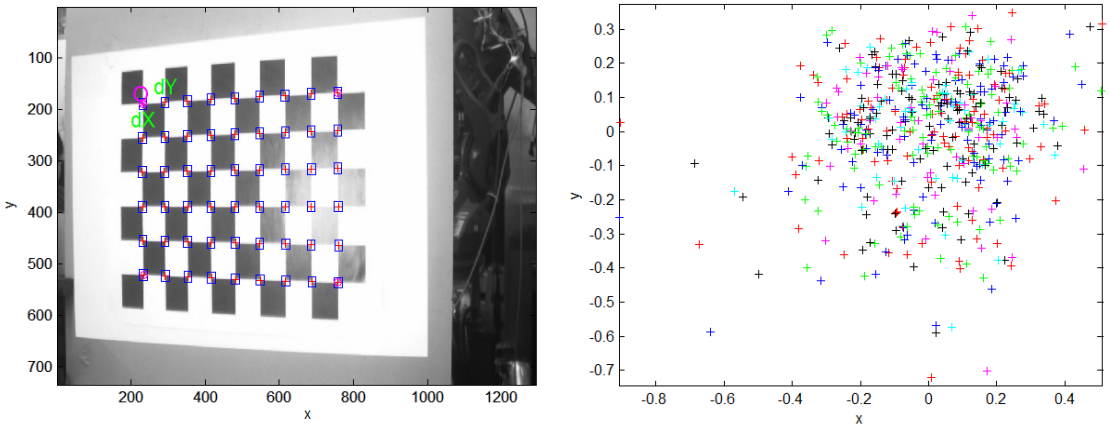
A calibração passa por obter várias imagens de um padrão de xadrez, de dimensões conhecidas, em várias posições e perspetivas. As fotos foram tiradas com a câmara Manta G-95/C, com uma resolução de 1292x734 e lente de 25mm.



(a) (b)

Figura 4.10: (a) Padrão de xadrez 9x6, (b) Imagens utilizadas para calibração

Em seguida foram identificados os cantos e calculadas as caraterísticas intrínsecas. O erro de reprojeção associado a estas medidas é inferior a 1 píxel, como se pode observar na figura 4.11b. Este erro representa a diferença entre a posição do ponto identificado no padrão e a sua projeção correspondente, proporcionando uma boa estimativa da precisão dos parâmetros encontrados.



(a) (b)

Figura 4.11: (a) Identificação dos cantos do padrão, (b) Erro de reprojeção

Quanto mais próximo de zero, mais exata é a calibração. No caso de se obterem valores de erro de reprojeção elevados em alguns pontos, é possível identificar a imagem de origem, removê-la e proceder a uma nova calibração com as restantes.

Obtiveram-se os seguintes resultados:

Tabela 4.1: Parâmetros intrínsecos de calibração da câmara

	Valores	**Erro Associado**
Distância focal (f_c)	[1525.72360, 1523.58147] px	[4.65717, 4.27532] px
Ponto central (C_x, C_y)	[664.94735, 361.32823] px	[4.60629, 3.96935] px
Distorção (kc)	[-0.20860, 0.23070, 0.00220, 0.00027, 0.00000]	[0.01334, 0.10986, 0.00052, 0.00058, 0.00000]
Skew (ângulo entre o eixo x e y)	[0.00000] $\rightarrow 90.0^{o}$	[0.00000] $\rightarrow 0.0^{o}$

4.5.2 Calibração dos parâmetros intrínsecos do projetor e da transformação para a câmara

Os projetores, tal como as câmaras necessitam de ser calibrados para poderem ser utilizados em sistemas de câmara-projetor. Para tal é necessário determinar os parâmetros intrínsecos, por um processo idêntico ao da câmara e a deslocação e rotação entre estes.

Os parâmetros foram obtidos com recurso ao *software* de calibração [42, 43] desenvolvido pela *Brown University School of Engineering*. Este sistema utiliza o projetor para projetar padrões de luz estruturada sobre um padrão de calibração 4.10a e a câmara para adquirir estas imagens, ilustrado na figura 4.12. Este processo é repetido para diversas posições e perspetivas.

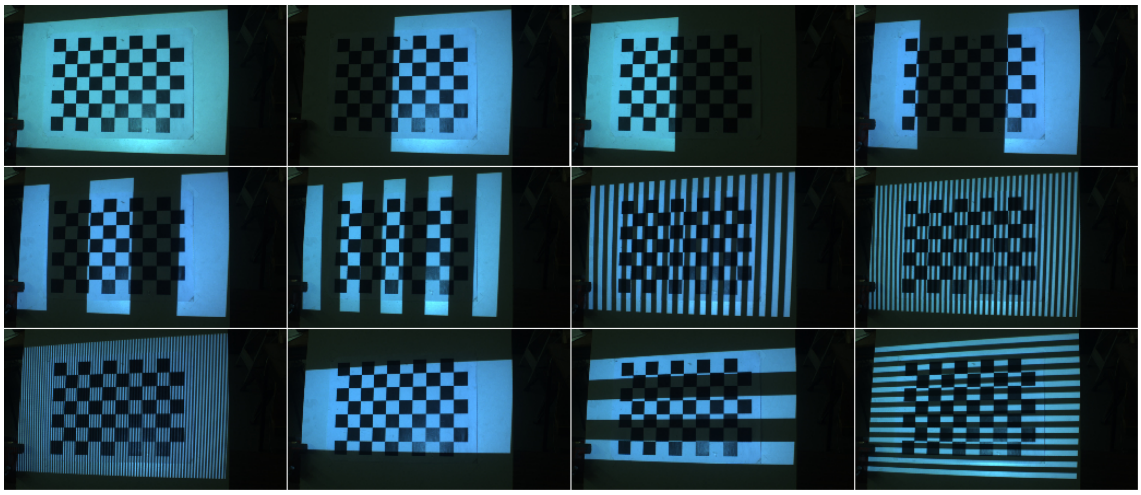


Figura 4.12: Projeção de luz estruturada sobre um padrão de calibração

A metodologia implementada utiliza homografias locais em vez de globais por imagem para obter precisão ao sub-píxel, obtendo-se os seguintes resultados para as caraterísticas intrínsecas:

Tabela 4.2: Parâmetros intrínsecos de calibração do projetor

	Valores	**Erro Associado**
Distância focal (f_c)	[407.8939, 861.1495] px	[0.12342, 0.12342] px
Ponto central (C_x, C_y)	[642.7833, 418.3529] px	[0.12342, 0.12342] px
Distorção (kc)	[-0.00134, -0.10030, -0.00071, -0.00341, 0.00000]	[não disponível]

Para os parâmetros extrínsecos, transformação entre a câmara e o projetor da figura 4.13:

$$R = \begin{bmatrix} 0.97663 & -0.01396 & -0.21445 \\ 0.01416 & 0.99989 & -0.00062 \\ 0.21443 & -0.00243 & 0.97673 \end{bmatrix} \tag{4.10}$$

$$T = \begin{bmatrix} 0.29298 \\ -0.01450 \\ -0.06229 \end{bmatrix} \pm 0.00019 \tag{4.11}$$

$$H_{Projetor}^{Câmara} = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \tag{4.12}$$

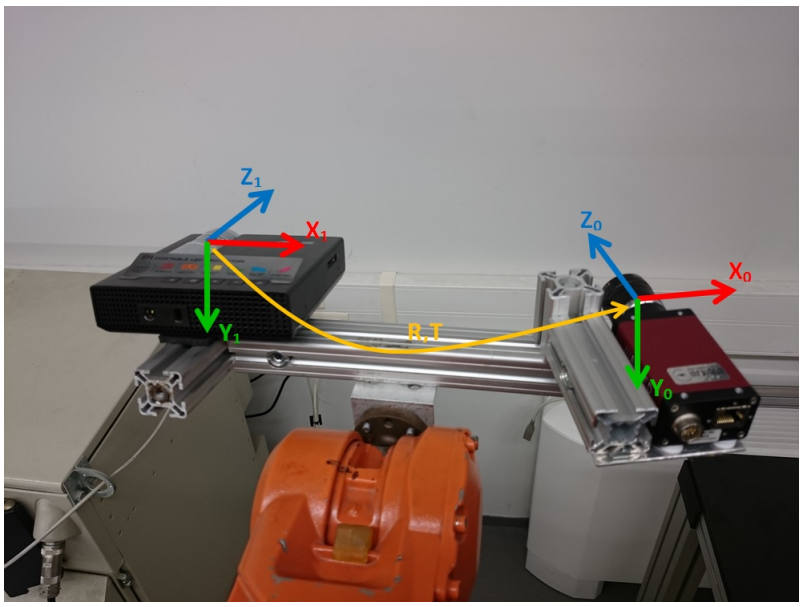


Figura 4.13: Parâmetros extrínsecos

4.5.3 Cálculo das transformação homogéneas

Para o cálculo da transformação em falta, suporte-câmara, foi utilizado um braço robótico onde foi fixado o suporte câmara-projetor. Desta forma, é possível saber com elevada precisão a posição do *end-effector* em relação à base do robô.

Para determinar a transformação em falta definiu-se a seguinte estratégia:

1. Colocação de um padrão de calibração dentro do volume de trabalho do braço robótico;
2. Calcular a transformação homogénea entre o padrão e a base do robô;
   (a) Calibrar uma ferramenta;
   (b) Com a ferramenta identificar três cantos do padrão de calibração (superior esquerda, superior direita, e inferior esquerda) e guardar os valores de translação e quaterniões em cada posição;
   (c) Calcular a transformação homogénea
3. Calcular a transformação homogénea entre a câmara e o padrão de calibração e guardar os valores de translação e quaterniões em cada posição;
4. Calcular a transformada homogénea restante.

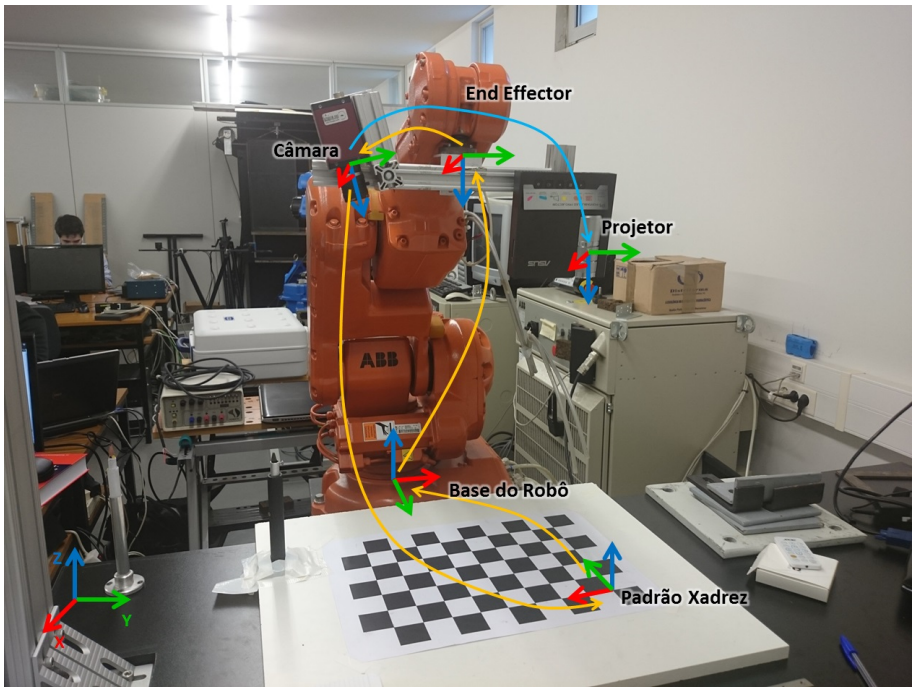
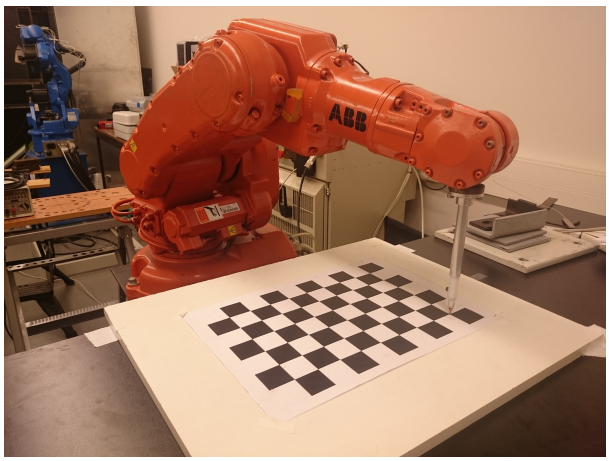


Figura 4.14: Transformações entre os referenciais do sistema

4.5.3.1 Transformação entre o padrão de calibração e a base do robô

Inicialmente é colocada uma ferramenta, de preferência pontiaguda tal como a da figura 4.15a, e calibrada. Esta calibração serve para o manipulador robótico determinar as dimensões da ferramenta e retornar o valor da posição e orientação em função dela. Consiste em colocar a extremidade da ferramenta na mesma posição mas em ângulos, de preferência até em quadrantes diferentes para se obter uma maior precisão.

(a) Ferramenta utilizada para aquisição dos três pontos no padrão



(b) Aquisição do primeiro ponto

Figura 4.15: Transformação do referencial padrão de calibração para base do robô

Por cinemática direta obtém-se a localização tridimensional de cada um dos pontos em relação ao referencial do robô (base), apresentados na tabela 4.3

Tabela 4.3: Posição dos três cantos do padrão

	Origem	**Baixo**	**Direita**
X (m)	-0.1829	-0.1758	0.1158
Y (m)	0.7345	0.5482	0.7474
Z (m)	0.1994	0.1984	0.1998
q0	0.00019	0.00023	0.00022
q1	0.70160	0.70156	0.70159
q2	-0.71255	-0.71259	-0.71257
q3	0.00464	0.00465	0.00468

Os pontos adquiridos representam o referencial do padrão de calibração (origem, eixo X, eixo Y e consequentemente eixo Z) segundo o referencial do robô. Ou seja, estes pontos têm correspondência direta para os do referencial do robô $(0,\ 0)$ $(|Baixo - Origem|, 0)$ $(0,\ |Direita - Origem|)$.

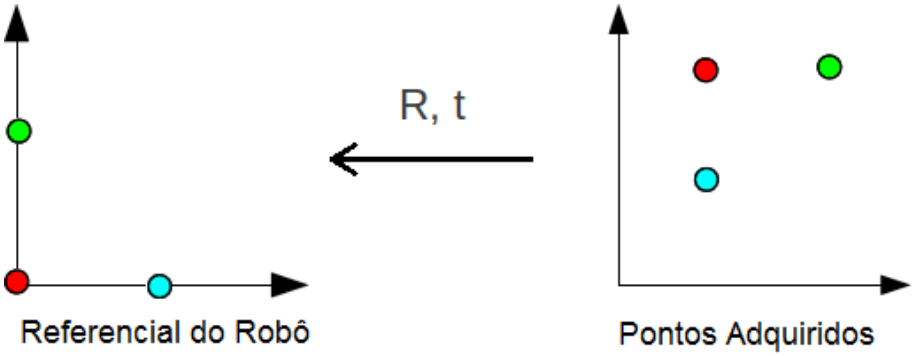


Figura 4.16: Rotação e translação entre o referencial do robô e o pontos adquiridos (Adaptado de [7])

Este problema é modelado pela seguinte equação:

$$Ref_{Robo} = R \times Ref_{Pontos} + t \tag{4.13}$$

Inicialmente são identificados os centroides de rotação de cada conjunto de pontos. No Ref_Robo é o ponto $(0,\ 0)$ e no Ref_pontos é o ponto de origem definido na tabela 4.3.

Para cálculo da rotação ótima entre os dois referencias é utilizada a decomposição em valores singulares. Mas, para se poder utilizar este método, é necessário remover a componente de translação entre os conjuntos de pontos.

$$H = \sum_{i=1}^{N} (P^i_{Pontos} - centroide_{Pontos})(P^i_{Robo} - centroide_{Robo})^T \tag{4.14}$$

$$[U,\ S,\ V] = SVD(H) \tag{4.15}$$

$$R = VU^T \tag{4.16}$$

A translação é obtida pela seguinte equação:

$$t = -R \times centroide_{Pontos} + centroide_{Robo} \tag{4.17}$$

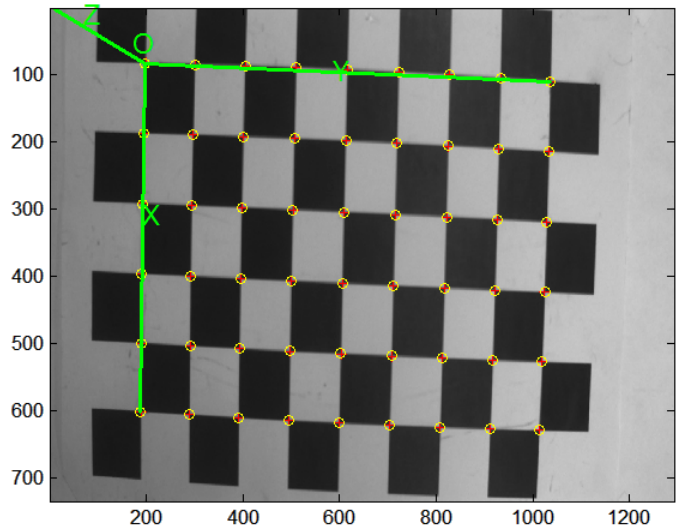


Figura 4.17: Calibração externa da câmara

Obtendo-se a matriz homogénea:

$$H_{Base\_do\_Robo}^{Padrão\_Xadrez} = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.0406 & -0.9992 & -0.0054 & 0.7424 \\ 0.9992 & 0.0406 & 0.0013 & 0.1527 \\ -0.0011 & -0.0054 & 1.0000 & -0.1956 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{4.18}$$

4.5.3.2 Transformação entre a câmara e o padrão de calibração

A transformação homogénea entre a câmara e o padrão de calibração equivale à calibração das caraterísticas extrínsecas da mesma, ou seja, a sua posição em relação ao mundo.

Este processo foi realizado com recurso à *toolbox* de calibração de câmaras para Matlab [41]. Inicialmente são determinados os parâmetros intrínsecos da câmara, como apresentado em 4.5.1. Em seguida, a câmara é orientada para o padrão de calibração, calculada a transformação 4.19 e guardada a posição e orientação do manipulador robótico 4.20.

$$H_{Padrão\_Xadrez}^{Câmara} = \begin{bmatrix} -0.033678 & 0.999400 & 0.008081 & -0.16503801 \\ 0.998053 & 0.033205 & 0.052798 & -0.99002173 \\ 0.052498 & 0.009844 & -0.998572 & 0.52561520 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.19}$$

$$H_{End\_Effector}^{Base\_do\_Robo} = \begin{bmatrix} 0.0637 & -0.9980 & -0.0028 & -0.0229 \\ -0.9814 & -0.0632 & 0.1812 & 0.4613 \\ -0.1810 & -0.0088 & -0.9835 & 0.7860 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{4.20}$$

4.5.3.3 Transformação entre o *end-effector* e a câmara

A transformação em falta, $H_{Câmara}^{End\_Effector}$, é obtida através da resolução da seguinte equação:

$$H_{Base\_do\_Robo}^{Padrão\_Xadrez} \times H_{End\_Effector}^{Base\_do\_Robo} \times H_{Câmara}^{End\_Effector} \times H_{Padrão\_Xadrez}^{Câmara} = I \tag{4.21}$$

$$H_{End\_Effector}^{Padrão\_Xadrez} \times H_{Câmara}^{End\_Effector} \times H_{Padrão\_Xadrez}^{Câmara} = I \tag{4.22}$$

$$H_{Câmara}^{End\_Effector} = H_{End\_Effector}^{Padrão\_Xadrez\ -1} \times H_{Padrão\_Xadrez}^{Câmara\ \ -1} \tag{4.23}$$

Obtendo-se a seguinte transformação homogénea:

$$H_{Câmara}^{End\_Effector} = \begin{bmatrix} 0.0218 & 0.9996 & 0.0176 & -0.0834 \\ -0.9776 & 0.0177 & 0.2095 & -0.1357 \\ 0.2091 & -0.0217 & 0.9776 & 0.0409 \\ 0.0000 & 0.0000 & 0.0000 & 1.0000 \end{bmatrix} \tag{4.24}$$

4.6 Resultados

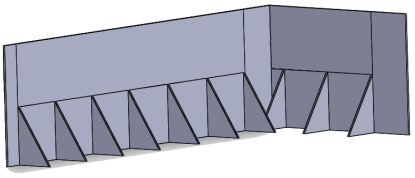
Por forma a validar o conceito proposto foram realizados testes práticos. Numa fase inicial foram realizados testes apenas com o projetor e calibração manual no robô do projeto CARLoS e sem o sistema de interação humano-robô por intermédio de um ponteiro laser. Em seguida, foi implementado o método de calibração com o braço robótico e o *setup* projetor-câmara e testado com o mesmo, incluindo a interface homem-robô.

4.6.1 Testes com manipulador robô do projeto CARLoS (GUARDIAN)

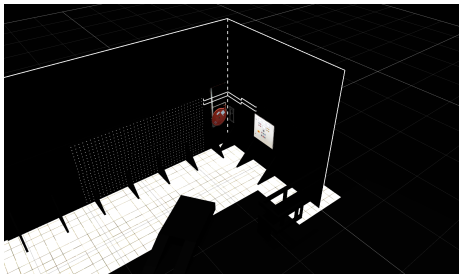
Numa fase inicial do projeto, o sistema de *projection mapping* sem HRI foi testado no robô do projeto CARLoS. Para tal, foram determinados os parâmetros intrínsecos do projetor e medida manualmente a transformação entre o projetor e o centro do robô (ponto conhecido no mundo).

A representação virtual 3D do mundo, apresentada na figura 4.18b, foi criada com recurso ao *software* SolidWorks, e importada para o simulador gazebo bem como todos os objetos para projeção (canalizações, *studs*, extintores, quadro elétrico, ...), obtendo-se o resultado apresentado na figura 4.18c. O mundo virtual construído é todo preto, à exceção dos objetos que se pretendem visualizar, dado que em RGB significa ausência de luz, ou seja, a câmara virtual apenas vê os objetos com as oclusões criadas pelo meio.

(a) Mockup de infraestrutura do navio

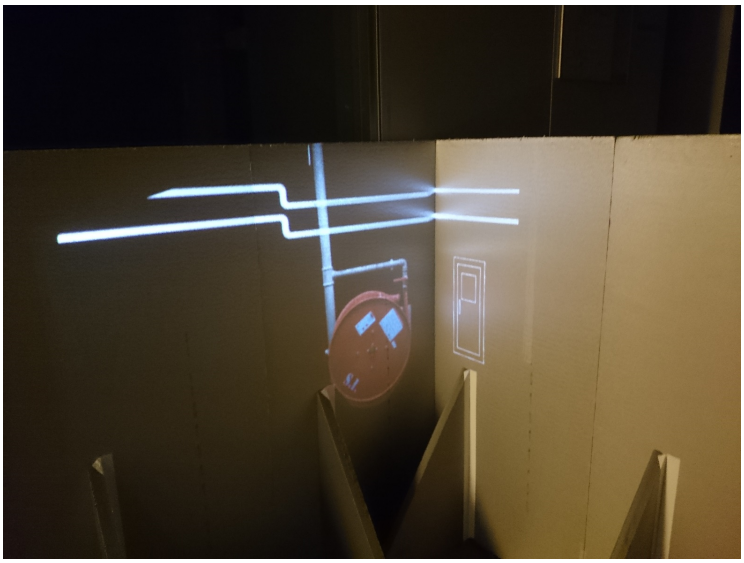


(b) Modelo tridimensional do ambiente

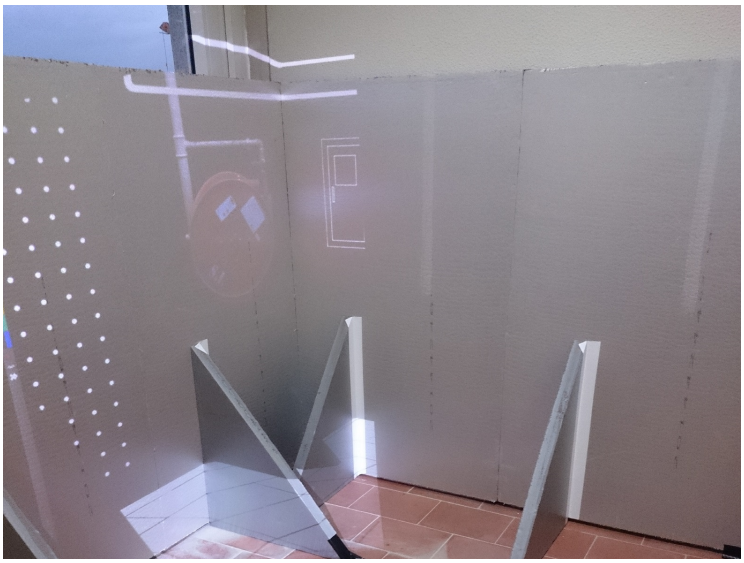


(c) Ambiente simulado

Figura 4.18: Criação de uma representação virtual tridimensional do mundo real para *Projection mapping* com o robô do projeto CARLoS



(a)



(b)

Figura 4.19: *Projection mapping* de informação sobre paredes utilizando o robô móvel do projeto CARLoS

Como se pode observar na figura 4.19, o erro associado à projeção de informação é elevado, principalmente segundo o eixo vertical. Este, segundo o eixo vertical, pode variar entre 10cm a 25cm, com o deslocamento do robô pelo ambiente. Por outro lado, o erro horizontal estava muito dependente do ângulo de projeção variando desde 2cm, quando o projetor se encontrava direcionado para o canto a uma distância de 2m (figura 4.19b), até 10cm quando projetava segundo outras direções.

Estes erros advêm do facto de a localização do robô não ser exata, mas principalmente, de as calibrações das transformadas terem sido realizados manualmente e ângulo de visão ainda não estar definido com a precisão requerida.

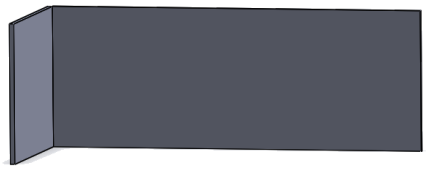
4.6.2 Testes com manipulador robótico ABB IRB140_T

Numa segunda fase, os procedimentos de calibração apresentados foram implementados bem como a interação humano-robô por intermédio de um apontador laser e uma interface gráfica no computador.

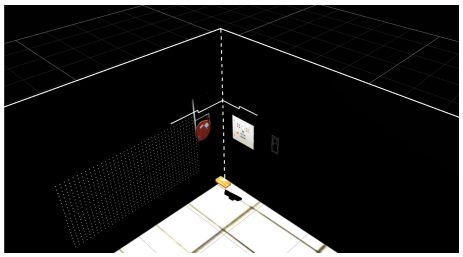
Nesta etapa, o sistema físico é constituído por um manipulador robótico com o conjunto câmara-projetor colocado no *end-effector*. Por forma a que o sistema desenvolvido soubesse a posição do projetor em relação ao mundo, foi desenvolvido um programa em RAPID que lê as posições das juntas do braço robótico e por cinemática direta calcula a posição e orientação do *end-effector* do mesmo, as quais são concatenadas numa string e enviadas pela porta série. Por sua vez, um nó de ROS recebe esta informação e converte-a para uma TF a qual é subscrita pelo sistema.

Foi criada uma representação virtual do mundo, apresentada na figura 4.20b, e importada para o simulador, bem como todos os objetos que o utilizador pretende visualizar, obtendo-se o resultado apresentado na figura 4.20c.

(a) Ambiente real

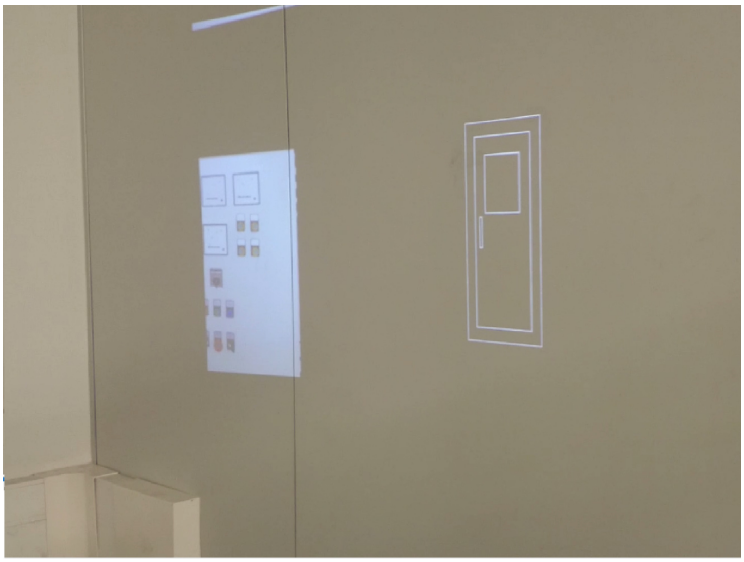


(b) Modelo tridimensional do ambiente

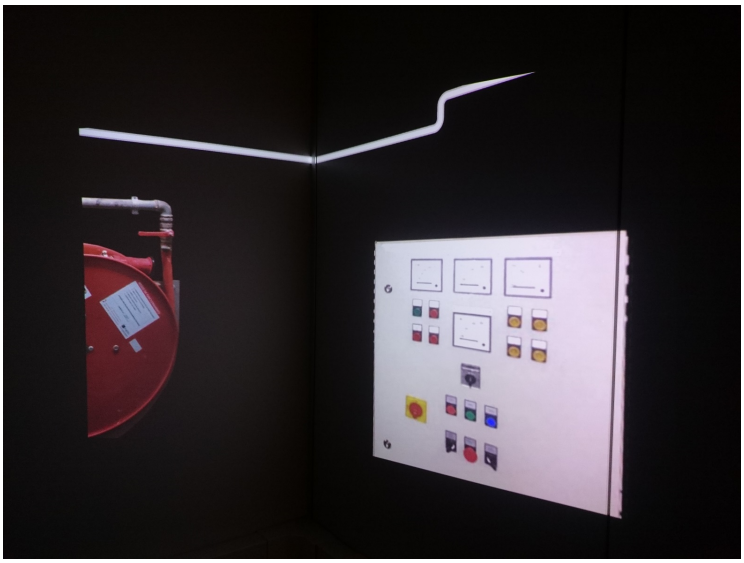


(c) Ambiente simulado

Figura 4.20: Criação de uma representação virtual tridimensional do mundo real para *Projection mapping* com um manipulador robótico



(a) Projeção sobre o ambiente

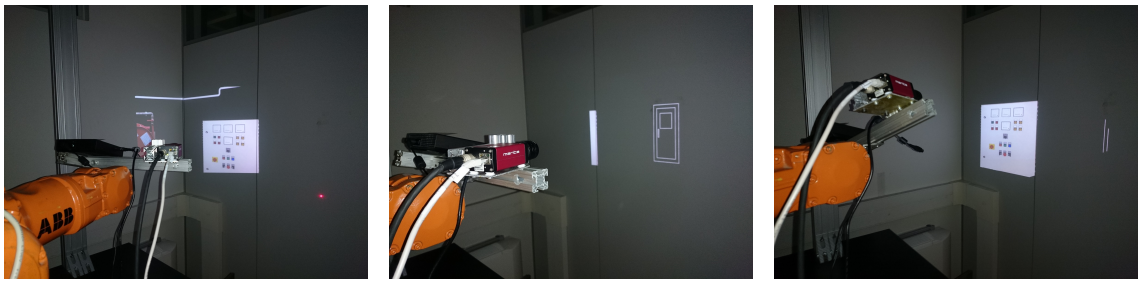


(b) Projeção sobre o ambiente

Figura 4.21: *Projection mapping* de informação sobre paredes utilizando um manipulador robótico

Os erros de projeção obtidos são inferiores a 3cm, independentemente da posição e orientação do projetor.

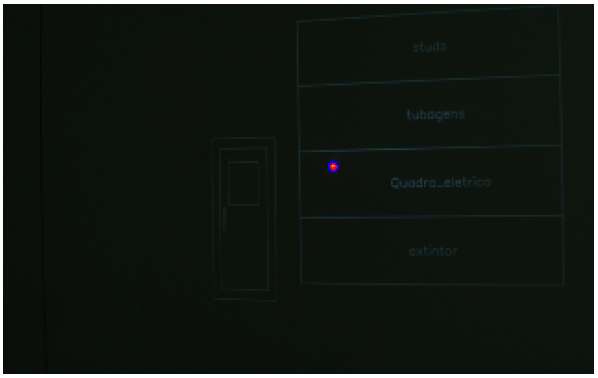
O projetor foi inclinado 15º por forma a que o centro da imagem projetada esteja centrado com a posição do projetor e da câmara e o *keystone* corrigido para que a imagem projetada seja retangular. Esta configuração foi adotada para que a área de projeção coincida com a área de visualização da câmara, possibilitando a implementação de HRI com um apontador laser. No entanto, esta configuração não permite que a razão 16:10 (1.60) seja mantida, obtendo-se apenas 1.56. Daqui constata-se que os píxeis não são quadrados como era esperado impondo erro de projeção segundo o eixo vertical do projetor que é mais notório quando não está horizontal como se pode ver na figura 4.22c.



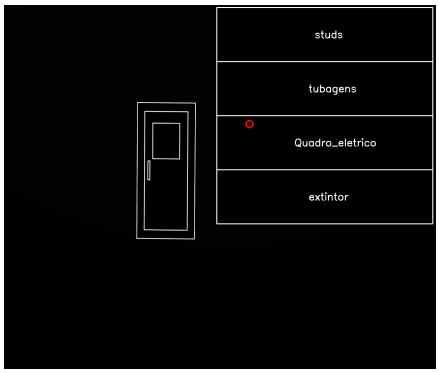
(a) (b) (c)

Figura 4.22: *Projection mapping* de informação sobre paredes com o projetor em diferentes orientações

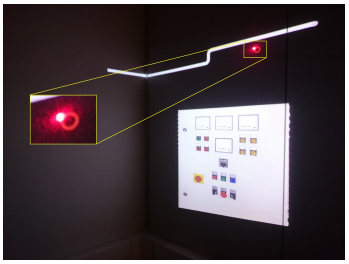
A interação entre o utilizador e o robô foi implementada com recurso ao conjunto projetor-câmara e um apontador laser de duas frequências. O utilizador aponta para a área de projeção, clica no botão direito e aparece um conjunto de caixas de dialogo na área de projeção correspondentes aos ficheiros importados na interface gráfica do computador.



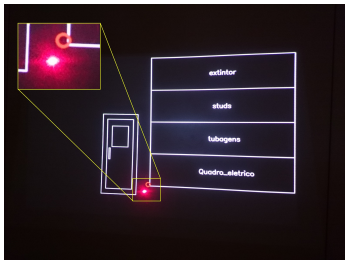
(a) Imagem adquirida pela câmara com identificação da posição do ponteiro laser



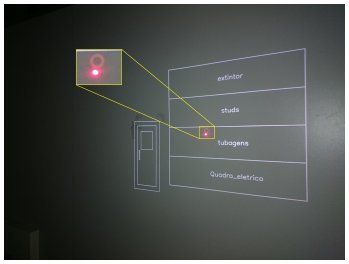
(b) Imagem proveniente do gazebo com adição de caixas de diálogo e identificação da posição do ponteiro laser no referencial do projetor



(c) Identificação da posição do apontador laser na imagem projetada - exemplo 1



(d) Identificação da posição do apontador laser na imagem projetada - exemplo 2



(e) Identificação da posição do apontador laser na imagem projetada - exemplo 3

Figura 4.23: Interação humano-robô através de um apontador laser

O apontador laser é corretamente localizado e identificada a frequência a que pisca (clique direito e clique esquerdo), com recurso a uma câmara.

Após identificação de clique direito, as caixas de diálogo são sobrepostas na imagem proveniente do simulador 4.23b e calculada a posição do apontado laser no referencial do projetor. Como se pode constatar pelas figuras 4.23c, 4.23d e 4.23e, o erro associado à conversão é reduzido e aceitável perante as dimensões das caixas de diálogo. Devido ao facto de o formato da imagem projetada não ser exatamente 16:10 o erro de localização é de 2cm na parte superior da imagem aumentando progressivamente para 4cm na parte inferior.

Por forma a que o sistema seja fluido, todos estes processos devem ser realizados periodicamente a uma frequência de 20Hz (50ms). Para tal,o algoritmo de identificação do apontador laser foi otimizado para que o prazo temporal seja cumprido.

O tempo de execução, no computador ASUS n56vz (Intel Core i7-3610QM, 6GB RAM, NVIDIA GT650M, 240GB SSD), foi medido 492 vezes obtendo-se os tempos apresentados na tabela 4.4. Como se pode verificar o tempo médio de execução é 33.28ms com desvio padrão de 5ms e tempo máximo de 49.65ms, sendo inferiores aos 50ms estipulados.

Tabela 4.4: Tempos de execução do sistema

	Média	**Máximo**	**Mínimo**	**Desvio Padrão**
Tempo(ms)	33.2812	49.6471	22.6462	4.9961

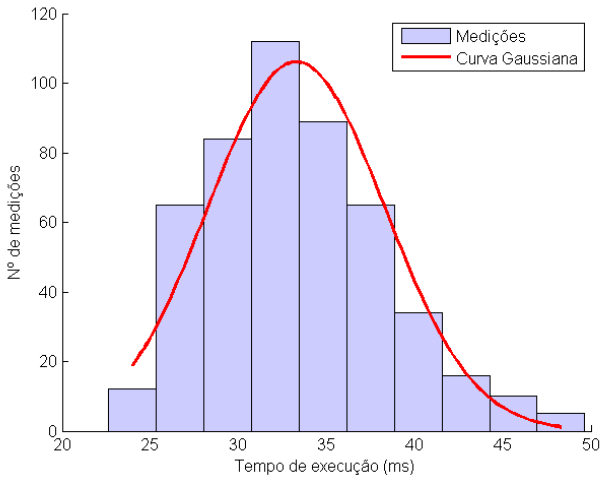


Figura 4.24: Tempos de execução do sistema

4.6.3 Testes com robô móvel jarvis

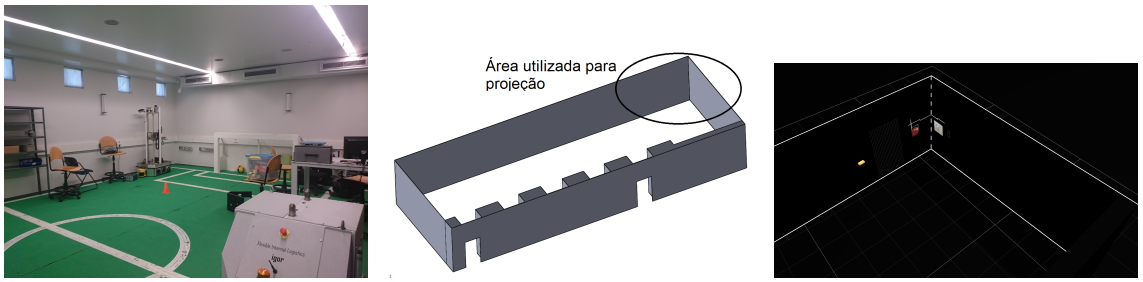
O sistema foi também testado no robô móvel jarvis 4.25 (AGV). Este, foi desenvolvido na FEUP com a *framework* ROS e tem a capacidade de se localizar em mundos conhecidos através de um sistema de localização *Sick NAV350-3232*. No entanto, ao contrário dos outros robôs em que a localização é 6DoF, no jarvis é apenas é 3DoF obtendo-se a posição em xyθ. Isto implica

que se o robô subir uma rampa nem ele nem o sistema de *projection mapping* têm conhecimento de tal levando a um mau posicionamento da câmara virtual e consequente erro de mapeamento das imagens no ambiente real.

Tal como nos casos anteriores, o espaço físico 4.26a é modelado no SolidWorks 4.26b e importado para o gazebo criando o mundo virtual 4.26c, adicionando-se em seguida todos os objetos necessários. Este mundo virtual foi posicionado de forma à origem da localização do robô real coincidir com a simulada.

A *framework* ROS permite a implementação de uma arquitetura distribuída. Assim sendo, o sistema pode correr noutro computador da mesma rede subscrevendo normalmente o(s) tópico(s) necessários. Neste caso em particular, apenas foi necessário alterar o nome das TF para coincidirem com as do robô e definir a TF entre um ponto conhecido do robô (base do sistema de localização laser) e o suporte câmara-projetor.

Figura 4.25: Robô móvel jarvis (AGV)

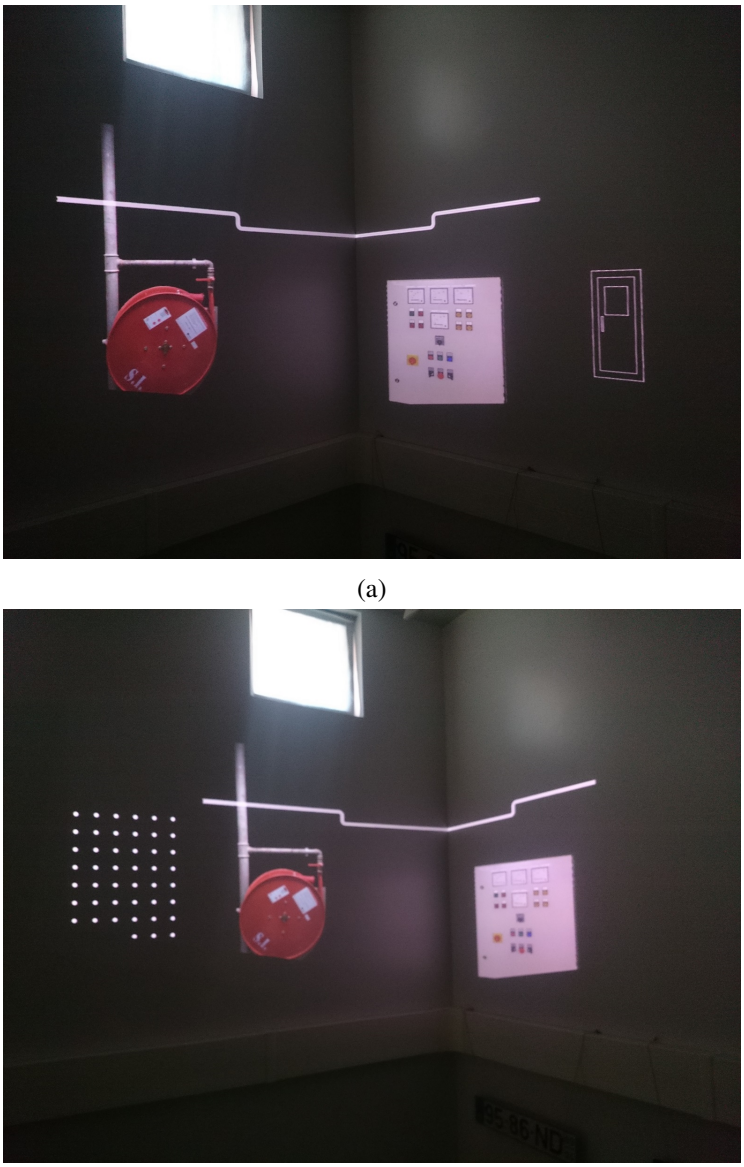


(a) Ambiente real

(b) Modelo tridimensional do ambiente

(c) Ambiente simulado

Figura 4.26: Criação de uma representação virtual tridimensional do mundo real para *Projection mapping* com robô móvel jarvis



(a)

(b)

Figura 4.27: *Projection mapping* de informação sobre paredes utilizando o robô móvel jarvis

Os erros de projeção obtidos assemelham-se aos dos testes realizados com o braço robótico ABB, inferiores a 3cm (distância de projeção 3m). Isto deve-se ao facto de o sistema de localização ter uma precisão de $\pm 4mm$. O mesmo foi verificado para a identificação do apontador laser, com um erro inferior a 5cm.

Nesta fase do projeto foi adicionado ao sistema funcionalidade de *drag and drop* de objetos virtuais sobre o mundo real com o apontador laser. Desta forma, o utilizador, mesmo sem qualquer tipo de experiência, tem a capacidade de reposicionar objetos num ambiente tridimensional dado que as alterações são efetuadas sobre o mesmo e não numa interface 2D, ecrã de um computador. Para tal, apenas é necessário clicar (clique esquerdo) sobre o objeto que pretende mover e clicar de novo (clique esquerdo) na posição desejada para que este mude de localização.

Numa fase mais avançada, estas alterações efetuadas por utilizadores poderiam ser afetadas diretamente os ficheiros CAD de construção, reduzindo a necessidade de um profissional para os alterar constantemente, mantendo-os sempre atualizados.

4.6.4 Fontes de erro

O sistema apresentado contêm erros de projeção que em algumas situações não podem ser desprezados. Estes advêm de três fontes de erro, do projetor, da calibração das transformações do sistema real e virtual, da localização do robô e da recriação do mundo virtual.

Ao nível do projetor, destaca-se o facto de o píxel não ser quadrado e não ser possível modelar este comportamento virtualmente. Isto, deve-se ao facto de o projetor esta inclinado 15^{o} por forma a que a área de projeção seja coincidente com a de visão abrangida pela câmara e consequente compensação do *keystone* para que a imagem seja retangular. No entanto, este ajuste faz com que a relação 1.6 (1280×800) seja alterada para 1.56, implicando um erro vertical máximo de 1.3cm a 2m de distância da área de projeção.

A calibração do sistema, que consiste em encontrar as caraterísticas intrínsecas e extrínsecas da câmara e projetor bem como todas as transformações entre referenciais, incluindo com o mundo, é um processo feito por etapas cada uma delas acarretando erros de precisão. No caso da câmara e projetor, a limitação é dada pela sua resolução e distorção de lente, dado que os resultados obtidos são tão precisos quanto menor for a distância correspondente a 1 píxel. No que toca à identificação da transformada entre o padrão de calibração e a base do robô, o erro é consequente da calibração da ferramenta e colocação da mesma nos três pontos. A maior fonte de erro, no caso do braço robótico é a sua localização em relação mundo a qual foi medida manualmente. No entanto tal não se verifica no AGV e no robô do projeto CARLoS dado que estes são capazes de se localizar. Nesta situação, o erro está dependente da precisão de localização.

O mundo virtual foi modelado com o *software SolidWorks* assumindo que as paredes dos locais de projeção são planas e verticais, que poderá não corresponder exatamente à realidade.

Por forma a ter noção de como os erros de localização e orientação afetam o sistema global foi calculado o erro de projeção em função do erro de deslocação e orientação.

$$I_0 + (I_1 - I_0)t = 0 \tag{4.25}$$

onde $I_0 = (x_0, y_0, z_0)$ e $I_1 = (x_1, y_1, z_1)$, representam a reta de projeção com origem em I_0 (localização do projetor) e orientação $I_1 - I_0$.

$$I_{0x} + (I_{1x} - I_{0x})t + I_{0y} + (I_{1y} - I_{0y})t + I_{0z} + (I_{1z} - I_{0z})t = 0 \tag{4.26}$$

$$a(x - xp_0) + b(y - yp_0) + c(z - zp_0) = 0 \tag{4.27}$$

representa o plano de projeção definido pelo ponto (xp_0, yp_0, zp_0) e pela normal (a, b, c).

Visto que o que se pretende calcular é o erro de projeção (localização atual do píxel - localização correta do píxel), por forma a simplificar as operações, o plano é centrado em $(0,0,0)$ e a reta de projeção tem origem em $I_0 = (deslocamento_x, deslocamento_y, deslocamento_z)$. Desta forma, o erro de projeção é dado pela distância euclidiana entre a origem e a interseção da reta no plano.

$$a \times (I_{0x} + (I_{1x} - I_{0x})t) + b \times (I_{0y} + (I_{1y} - I_{0y})t) + c \times (I_{0z} + (I_{1z} - I_{0z})t) = 0 \tag{4.28}$$

Resolvendo em ordem a t e substituindo na equação 4.25 obtêm-se o ponto de interseção.

Considerando um plano rodado 45º segundo o eixo z $(a = 1, b = 1, c = 0)$, o ponto de projeção com origem em $(2,0,0) + 0.005$ e sentido $(-1.0, 0.01, 0.01)$ obtém-se um erro de projeção de 0.0438m.

4.7 Conclusões

Neste capítulo foi apresentada a arquitetura e o sistema implementado até ao momento, o qual se baseia na recriação em ambiente virtual do mundo físico para gerar as transformações na imagem, um ponteiro laser para interação com o sistema e ROS nas comunicações. A transformação das imagens é conseguida com o correto posicionamento da câmara virtual, que simula o projetor real, no mundo virtual. Desta forma, a imagem adquirida por esta câmara vai ficar corretamente mapeada no mundo real. A interação humano-robô foi conseguida com a implementação de um apontador laser de duas frequências (clique direito, clique esquerdo). O utilizador aponta para a área de visualização do robô e com o clique direito é lhe apresentada uma lista de objetos da qual pode selecionar o que pretender que seja projetado com um clique esquerdo. Tornou-se então necessário converter a posição do apontador laser do referencial da câmara para o do projetor, o qual foi conseguido por intermédio de um *kinect* virtual. O apontador é identificado através da câmara, é calculada a sua localização 3D com recurso aos dados do *kinect* e convertida para o referencial do projetor conhecendo as suas caraterísticas intrínsecas e extrínsecas. Todo este sistema foi implementado com recurso a ROS proporcionando abstração de hardware e modularidade.

Para o sistema funcionar corretamente e visto que este se baseia na posição do projetor no mundo para gerar as transformações necessárias foi preciso calibrar corretamente todo o sistema. As caraterísticas intrínsecas da câmara e do projetor foram determinadas minuciosamente para uma correta modelação dos mesmos no ambiente virtual e as transformações homogéneas entre todos os referencias calculadas com recurso a um manipulador robótico para se obter a precisão desejada.

O sistema implementa HRI através do conjunto câmara-projetor-apontador laser e o utilizador interage com o robô através de um ponteiro laser que pisca a duas frequências. Este oferece a possibilidade ao utilizador de selecionar os objetos que pretende visualizar, bem como alterar a sua posição mundo.

Por fim, foram apresentados resultados obtidos do sistema numa fase inicial acoplado ao robô do projeto CARLoS, ainda sem HRI implementado. Em seguida com o sistema na sua fase final fixado a um manipulador robótico e a um AGV. Os resultados obtidos são muito superiores no segundo caso, visto todas as calibrações terem sido efetuadas metódica e minuciosa observando-se um erro de projeção inferior a 3cm e de conversão da posição do laser de 4cm.

Capítulo 5

Conclusões e trabalho futuro

5.1 Conclusões

Os robôs são cada vez mais parte integrante da sociedade atual, em industrias, em serviços, aplicações médicas e entretenimento. Isto levou a que os sistemas robóticos que antes estavam confinados a áreas predeterminadas, tenham de partilhar espaço com os utilizadores e interagir com estes em ambientes cada vez menos estruturados.

É nesta área de interação entre humanos e robôs, designada de HRI, que a presente dissertação está inserida. Apresenta uma aplicação de HRI com base num apontador laser de duas frequências para interação entre o utilizador e o robô e *projection mapping* para o sentido oposto, fechando a malha de comunicação. A aplicação desenvolvida será depois incorporada no projeto CARLoS 3.1, que tem o propósito desenvolver em robô móvel para operações *fit-out* autonomamente em cooperação com os operadores em ambientes industriais.

O sistema utiliza um *setup* câmara-projetor como forma de interação com o meio ambiente, um simulador gráfico 3D (gazebo) para implementação de técnicas de *projection mapping*, uma interface gráfica para importação de objetos e um laser para interação.

A base do sistema assenta numa recriação virtual do mundo real num simulador com o ambiente (paredes, chão, tetos) representado a preto e os objetos (sistemas elétricos, infraestruturas, locais de soldadura, ...) a cor colocados sobre este. São também colocado no mundo virtual uma câmara, que simula o projetor real, e um *kinect* que simula a câmara real e estão colocados nas posições correspondentes à sua localização física. Desta forma, a área de visão da câmara virtual corresponde à do projetor real, aplicando todas as transformações necessárias à imagem para que esta seja mapeada corretamente no mundo real. No que toca à interação homem máquina, esta é realizada por intermédio de um apontador laser com duas frequências que permite ao utilizador adicionar, remover e mover objetos na cena. Todo este processo é realizado sobre o ambiente tridimensional permitindo que utilizadores com pouca ou nenhuma experiência possam interagir diretamente com a cena em vez da necessidade de aceder a um computador com uma interface 2D (ecrã) a qual implica familiarização e prática para um controlo fluído. Os objetos são adicionados clicando no botão direito seguido de clique esquerdo sobre a caixa de diálogo correspondente

ao objeto desejado e a sua movimentação (*drag and drop*) com dois cliques esquerdos, um para seleção e outro para posicionamento.

Dado que, o sistema utiliza a localização do projetor para geração das transformações na imagem, foi implementada uma metodologia de calibração de parâmetros intrínsecos e extrínsecos da câmara e projetor e todas as transformações entre referenciais por forma minimizar erros. Para tal, foi utilizado um braço robótico disponível no laboratório dada a sua elevada precisão.

Por forma a poder ser integrado no projeto CARLoS, a aplicação foi desenvolvida com recurso a ROS tornando o sistema modular e possibilitando abstração de hardware.

Os resultados obtidos apresentam-se promissores, o sistema implementado é capaz de realizar projeções sobre superfícies com erros inferiores 3cm e localização do apontador lazer inferiores a 4cm. Estes erros advêm maioritariamente do facto de a razão da imagem projetada não ser exatamente 16:10, constatando-se que os píxeis projetados não são precisamente quadrados.

5.2 Trabalho futuro

Relativamente a trabalho futuro, uma primeira etapa passaria por tornar o sistema mais robusto, principalmente ao nível da localização pois é nesta que a metodologia de transformação de imagens está assente. Uma possível solução seria substituir a câmara utilizada por um sensor RGBD, tal como um *kinect* ou *Asus xtion*, permitindo obter uma representação 3D da área de visão do sistema a qual serviria numa primeira fase para validar a localização atual e numa segunda fase para aumentar a sua precisão. Ao mesmo tempo, a conversão da posição do apontador laser entre referenciais (câmara-projetor) deixaria de necessitar de um *kinect*, aumentado a precisão de identificação do mesmo.

A utilização de um apontador laser como forma de interação pode apresentar-se ineficiente em ambientes de muita luminosidade e com superfícies espelhadas. Para tal, poderia implementar-se um sistema multimodal de interação, ou seja, a par do apontador laser existiria outra forma de transmissão de informação tais como comandos de voz, identificação de gestos, luvas instrumentadas entre outros.

Bibliografia

[1] B. Poling, “A Tutorial On Camera Models.”

[2] B. Frank, C. Stachniss, G. Grisetti, K. Arras, and W. Burgard, “Robotics 2 Camera Calibration.”

[3] Oculus-rift, “L’histoire de la réalité virtuelle.” [Online]. Disponivel em: http://www.oculus-rift.fr/lhistoire-realite-virtuelle/ [Acedido: 2015-02-08]

[4] Projection Mapping Central, “The Illustrated History of Projection Mapping.” [Online]. Disponivel em: http://projection-mapping.org/the-history-of-projection-mapping/ [Acedido: 2015-02-07]

[5] O. Bimber and R. Raskar, *Spatial Augmented Reality Merging Real and Virtual Worlds*, 2005, vol. 6.

[6] THE CARLoS PROJECT, “The CARLoS Project.” [Online]. Disponivel em: http://carlosproject.eu/ [Acedido: 2015-01-28]

[7] “Finding optimal rotation and translation between corresponding 3D points | Nghia Ho.” [Online]. Disponivel em: http://nghiaho.com/?page_id=671 [Acedido: 2015-06-27]

[8] M. a. Goodrich and A. C. Schultz, “Human-Robot Interaction: A Survey,” *Foundations and Trends® in Human-Computer Interaction*, vol. 1, no. 3, pp. 203–275, 2007.

[9] ROS, “ROS Wiki.” [Online]. Disponivel em: http://wiki.ros.org/ [Acedido: 2015-02-15]

[10] “SARKKIS@ AUTOMATICA Trade Fair 2014 - SARKKIS.” [Online]. Disponivel em: http://www.sarkkis.com/mechatronics/?p=2212 [Acedido: 2015-06-28]

[11] “DLP_texas_instruments.” [Online]. Disponivel em: http://www.ti.com/lsds/ti/dlp-technology/about-dlp-technology/how-dlp-technology-works.page [Acedido: 2015-06-15]

[12] Projection Mapping News, “Projection mapping definition. Video mapping, 3D mapping.” [Online]. Disponivel em: http://www.mapping-projection.com/projection-mapping-definition/ [Acedido: 2015-02-07]

[13] Projection Mapping Central, "Projection Mapping Central." [Online]. Disponivel em: http://projection-mapping.org/ [Acedido: 2015-02-07]

[14] A. Osorio, J.-A. Galan, J. Nauroy, S. Dahdouh, J.-J. Lobato, P. Donars, and I. Navarro, "Real time planning, guidance and validation of surgical acts using 3D segmentations, augmented reality projections and surgical tools video tracking," *Medical Imaging 2010: Visualization, Image-Guided Procedures, and Modeling*, vol. 7625, pp. –\n1180, 2010. [Online]. Disponivel em: <GotoISI>://000285047500079

[15] M. Billinghurst, "Augmented Reality in Education," *International Journal of Gaming and Computer-Mediated Simulations*, vol. 3, no. figure 1, pp. 91–93.

[16] "BMW Augmented Reality : Introduction." [Online]. Disponivel em: http://www.bmw.com.pt/com/en/owners/service/augmented_reality_introduction_1.html [Acedido: 2015-06-08]

[17] S. Tate, "Virtual Reality: A Historical Perspective." [Online]. Disponivel em: http://ei.cs.vt.edu/~history/Tate.VR.html [Acedido: 2015-02-08]

[18] P. lint, "The history of augmented reality." [Online]. Disponivel em: http://www.pocket-lint.com/news/108888-the-history-of-augmented-reality [Acedido: 2015-02-08]

[19] H. E. Lowood, "Virtual reality (VR)." [Online]. Disponivel em: http://www.britannica.com/EBchecked/topic/630181/virtual-reality-VR/253103/Early-work#ref884304 [Acedido: 2015-02-08]

[20] M. Naimark, "Michael Naimark - Biography." [Online]. Disponivel em: http://www.naimark.net/bio.html [Acedido: 2015-02-08]

[21] ——, "Displacements." [Online]. Disponivel em: http://www.naimark.net/projects/displacements.html [Acedido: 2015-02-08]

[22] R. Raskar, G. Welch, M. Cutts, A. Lake, L. Stesin, and H. Fuchs, "The Office of the Future: A Unified Approach to Image-Based Modeling and Spatially Immersive Displays," *SIGGRAPH '98 Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pp. 1–10, 1998.

[23] H. Fuchs and K. Gaskill, "Office of the Future." [Online]. Disponivel em: http://www.cs.unc.edu/Research/stc/index.html [Acedido: 2015-02-08]

[24] R. Raskar, P. Beardsley, J. van Baar, Y. Wand, P. Dietz, J. Lee, D. Leigh, and T. Willwacher, "RFIG Lamps: Interacting with a Self-Describing World via Photosensing Wireless Tags and Projectors," *Transactions on Graphics (TOG)*, vol. 23, pp. 406–415, 2004. [Online]. Disponivel em: http://dx.doi.org/10.1145/1015706.1015738

[25] Phys.org, "Smart projectors do not require artificial canvases." [Online]. Disponivel em: http://phys.org/news2790.html [Acedido: 2015-02-08]

[26] I. Markelic, "Tutorial on Homographies," *Markelic.De*, pp. 1–11. [Online]. Disponivel em: http://www.markelic.de/professional/Documents/PG.pdf

[27] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*, 2nd ed. Cambridge University Press, 2002, vol. 37.

[28] H. Gilje, "VPT 7." [Online]. Disponivel em: https://hcgilje.wordpress.com/vpt/ [Acedido: 2015-02-16]

[29] Derivative, "Derivative TouchDesigner." [Online]. Disponivel em: https://www.derivative.ca/ [Acedido: 2015-02-16]

[30] HeavyM, "HeavyM." [Online]. Disponivel em: http://heavym.net/en/ [Acedido: 2015-02-16]

[31] Painting With Light, "Painting With Light." [Online]. Disponivel em: http://pwl.bigfug.com/ [Acedido: 2015-02-16]

[32] "Human-Robot Interaction: A historical perspective and current research trends | Human-Robot Interaction." [Online]. Disponivel em: http://humanrobotinteraction.org/human-robot-interaction-a-historical-perspective-and-current-research-trends/ [Acedido: 2015-06-28]

[33] D. Feil-Seifer and M. Mataric, "Human-robot interaction," *Invited contribution to Encyclopedia of Complexity and Systems Science*, pp. 4643–4659, 2009. [Online]. Disponivel em: http://robotics.usc.edu/publications/585/

[34] I. Asimov, *I, Robot*. Doubleday, 1950.

[35] T. B. Sheridan and W. L. Verplank, "Human and Computer Control of Undersea Teleoperators," *ManMachine Systems Lab Department of Mechanical Engineering MIT Grant N0001477C0256*, p. 343, 1978.

[36] "Leap Motion | Mac & PC Motion Controller for Games, Design, & More." [Online]. Disponivel em: https://www.leapmotion.com/ [Acedido: 2015-06-28]

[37] "devDept Software S.a.s." [Online]. Disponivel em: https://www.devdept.com/ [Acedido: 2015-06-28]

[38] "Qt | Cross-platform application & UI development framework." [Online]. Disponivel em: https://www.qt.io/ [Acedido: 2015-06-19]

[39] "tf - ROS Wiki." [Online]. Disponivel em: http://wiki.ros.org/tf [Acedido: 2015-06-20]

[40] "Arduino - ArduinoBoardNano." [Online]. Disponivel em: http://www.arduino.cc/en/Main/ArduinoBoardNano [Acedido: 2015-06-21]

[41] "Camera Calibration Toolbox for Matlab." [Online]. Disponivel em: http://www.vision.caltech.edu/bouguetj/calib_doc/ [Acedido: 2015-06-24]

[42] “Projector-Camera Calibration / 3D Scanning Software.” [Online]. Disponivel em: http://mesh.brown.edu/calibration/ [Acedido: 2015-06-25]

[43] D. Moreno and G. Taubin, “Simple, accurate, and robust projector-camera calibration,” *Proceedings - 2nd Joint 3DIM/3DPVT Conference: 3D Imaging, Modeling, Processing, Visualization and Transmission, 3DIMPVT 2012*, pp. 464–471, 2012.