



**Jorge Miguel de
Almeida Pereira**

**Aquisição e tratamento de dados 3D para modelação
acústica de salas**



**Jorge Miguel de
Almeida Pereira**

**Aquisição e tratamento de dados 3D para modelação
acústica de salas**

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Eletrónica e Telecomunicações, realizada sob a orientação científica do Doutor Paulo Miguel de Jesus Dias e do Doutor António Guilherme Rocha Campos, ambos Professores Auxiliares do Departamento de Eletrónica, Telecomunicações e Informática (DETI) da Universidade de Aveiro.

Dedico este trabalho à minha família pelo incansável apoio.

o júri

presidente

Professor Doutor Joaquim João Estrela Ribeiro Silvestre Madeira
Professor Auxiliar da Universidade de Aveiro

arguente principal

Professor Doutor Rui Pedro Amaral Rodrigues
Professor Auxiliar no Departamento de Engenharia Informática da Faculdade de Engenharia da
Universidade do Porto

orientador

Professor Doutor Paulo Miguel de Jesus Dias
Professor Auxiliar da Universidade de Aveiro

agradecimentos

Gostaria de expressar o meu reconhecimento a todas as pessoas que, ao longo do meu percurso académico, estiveram presentes e ajudaram nas várias fases do processo que levaram à realização desta dissertação.

No entanto, não posso deixar de manifestar o meu sincero agradecimento às pessoas que mais ativamente deram o seu contributo, nomeadamente:

Ao meu orientador Professor Doutor Paulo Dias pela inteira disponibilidade, paciência e apoio motivacional que demonstrou na resolução e orientação de todos os problemas que surgiram ao longo desta dissertação, assim como toda a orientação necessária para planeamento das várias fases do trabalho desenvolvido.

Ao meu coorientador Professor Doutor Guilherme Campos pela motivação transmitida, que contribuiu para a escolha desta dissertação, assim como pela total disponibilidade ao longo de todo o seu desenvolvimento.

Ao IEETA (Instituto de Engenharia Electrónica e Telemática de Aveiro) pela disponibilização de um local para trabalhar, assim como o acesso a algum equipamento utilizado no decorrer deste trabalho.

Ao André Oliveira e Nuno Silva pela disponibilidade em ajudar na criação de uma demonstração que permitiu testar os resultados desta dissertação.

Por fim, gostaria de agradecer a todos os meus amigos que me acompanharam ao longo do meu percurso académico e que, apesar de não possuírem conhecimentos nesta área específica, sempre estiveram presentes para fornecer apoio e motivação essenciais na realização desta dissertação, em especial ao Cristóvão Leal, Pedro Aleixo, Beatriz Martins, Maria Matos, Pedro Narra, Maria , Filipa Rodrigues , Maria Kaoutzani e João Lopes.

palavras-chave

Reconstrução 3D, Auralização, Modelos Poligonais, Voxelização, Kinect

resumo

Esta dissertação tem como objectivo desenvolver ferramentas de apoio à componente de áudio em ambientes de Realidade Virtual. Trata-se de possibilitar a aquisição e configuração de dados 3D para alimentar modelos acústicos de salas. Estes podem ser geométricos (como o método das imagens virtuais, baseado nas reflexões das ondas sonoras nas superfícies que delimitam a sala) ou físicos (como o método das malhas de guias de onda digitais, baseado na discretização da equação de onda numa grelha tridimensional abrangendo o volume da sala).

Para a aquisição de modelos poligonais representativos das superfícies delimitadoras de ambientes reais, foi utilizado um sensor *Microsoft Kinect* associado à aplicação *Kinect Fusion*. Em seguida, foram desenvolvidas ferramentas de identificação dos materiais constituintes das superfícies presentes, recorrendo ao *Visualisation Toolkit (VTK)*. Este passo permite associar características acústicas, nomeadamente coeficientes de absorção/reflexão, a cada polígono da malha. É assim possível alimentar modelos acústicos geométricos, utilizados em algoritmos de auralização em tempo real.

Para alimentar modelos físicos, foi desenvolvido um algoritmo para, a partir dos modelos poligonais das superfícies das salas, gerar uma malha tridimensional de nós, de topologia retangular, abrangendo todo o domínio de propagação por elas delimitado. Cada nó é configurado em função da sua posição no domínio (*nó de ar* ou *nó-fronteira*) e, no caso de ser nó-fronteira, do respectivo material.

Foi utilizada, para efeitos de teste, uma sala de reuniões situada no IEETA (Instituto de Engenharia Electrónica e Telemática de Aveiro). Para esta sala, construiu-se um modelo poligonal com os materiais corretamente identificados em cada face, para alimentar o modelo geométrico. Aplicaram-se critérios de limitação do número de polígonos. Construiu-se igualmente uma grelha tridimensional de nós para modelação física. Conjugando as informações proporcionadas pelos dois modelos (absorção das superfícies, por um lado, e volume, por outro), foi possível gerar uma estimativa automática do tempo de reverberação (RT_{60}) de acordo com a fórmula de Sabine (modificada).

keywords

3D Reconstruction, Auralization, Polygonal Models, Voxelization, Kinect

abstract

The goal of this dissertation is the development of tools for audio in Virtual Reality environments, namely the acquisition and configuration of 3D data to feed room acoustic models. These models can be either *geometric* (such as the *mirror-image source* method, based on the reflection of sound waves on the surfaces delimiting the room) or *physical* (such as the *digital waveguide mesh* method, based on the discretisation of the wave equation on a three-dimensional grid covering the whole volume of the room).

A *Microsoft Kinect* sensor and the *Kinect Fusion* application were used for acquiring boundary surface polygonal models in real rooms. Tools to identify the surface materials were then developed using the *Visualisation Toolkit (VTK)*. This allowed the assignment of acoustic properties, namely absorption/reflection coefficients, to each mesh polygon, making it possible to configure geometric acoustic models, used in real-time auralization applications.

In the physical model configuration front, an algorithm was developed for generating three-dimensional grids of nodes based on the polygonal models of room boundary surfaces. These 3D grids, with rectangular topology, cover the whole volume of the room. Each node is configured according to its position (*air node* or *boundary node*) and, for boundary nodes, according to the respective boundary material.

A meeting room in IEETA was used for testing purposes. A polygonal model, with surface materials carefully identified, was built to create a geometric model of this room. Polygon number limitation criteria were applied. A 3D grid for physical modelling was also built. Combining the information from both models (surface absorption from one and volume information from the other), it was possible to automatically estimate reverberation time (RT_{60}) according to the (modified) Sabine's formula.

ÍNDICE

ÍNDICE.....	I
LISTA DE FIGURAS.....	III
1 INTRODUÇÃO.....	1
1.1 Auralização.....	1
1.1.1 Modelos geométricos.....	2
1.1.2 Modelos Físicos.....	3
1.2 Motivação.....	4
1.3 Objetivos.....	4
1.4 Estrutura da tese.....	6
2 FERRAMENTAS DE DESENVOLVIMENTO.....	9
2.1 Aplicações.....	9
2.1.1 Visualization Toolkit (VTK).....	9
2.1.2 Point Cloud Library (PCL).....	10
2.1.3 Meshlab.....	10
2.2 Sensor Microsoft Kinect.....	12
3 AQUISIÇÃO 3D DE UM AMBIENTE REAL.....	15
3.1 Nuvem de pontos e malhas poligonais.....	15
3.2 Aquisição de nuvens de pontos.....	16
3.2.1 Kinect Fusion.....	17
3.2.2 PCL Kinfu.....	19
3.2.3 PCL Kinfu large scale.....	21
3.2.4 Kscan3D, Skanect e ReconstructME.....	22
3.2.5 Análise e resultados das aplicações testadas.....	26
3.3 Tratamento das nuvens de pontos obtidas.....	28
3.3.1 Alinhamento manual inicial e aplicação do algoritmo ICP.....	29
3.3.2 Obtenção dos planos principais através do algoritmo RANSAC.....	29
3.3.3 Projeção de pontos.....	31
3.3.4 Convex Hull.....	33
3.3.5 Algoritmo de reconstrução de Poisson.....	35
4 APLICAÇÃO PARA CONFIGURAÇÃO DO MODELO PARA AURALIZAÇÃO.....	41
4.1 Ficheiro para armazenar características dos materiais.....	41
4.2 Seleção manual dos materiais.....	42
4.3 Seleção semiautomática de materiais recorrendo à textura.....	44
4.3.1 Segmentação de uma imagem.....	44
4.3.2 Métodos de representação de cores.....	45
4.3.3 Implementação de um método de seleção semiautomático e resultados.....	47

5	FERRAMENTA PARA CRIAÇÃO E CONFIGURAÇÃO DE MODELOS FÍSICOS	53
5.1	Voxelização	53
5.2	Determinação do espaço aberto na malha a partir da voxelização inicial.....	58
5.3	Cálculo do RT_{60}	60
5.3.1	Fórmula de Millington-Sette.....	60
5.3.2	Cálculo da Área	61
5.3.3	Cálculo do Volume	61
5.4	Exportação do modelo voxelizado	62
6	RESULTADOS OBTIDOS NA SALA DE REUNIÕES DO IEETA.....	65
6.1	Conversão para nuvem de pontos	65
6.1.1	Aquisição	65
6.1.2	Tratamento da nuvem de pontos para obter a malha poligonal do modelo	66
6.2	Seleção de materiais	70
6.3	Voxelização e preenchimento.....	71
6.4	Demonstração.....	72
7	CONCLUSÃO	77
7.1	Discussão.....	77
7.2	Trabalho Futuro	78
8	REFERÊNCIAS.....	81

LISTA DE FIGURAS

Ilustração 1 - Ilustração de um modelo geométrico.	2
Ilustração 2 - Visualização da propagação das ondas sonoras num modelo físico de um espaço fechado [7].	3
Ilustração 3 - Diagrama da informação necessária para alimentar o processo de auralização.	6
Ilustração 4 - Visualização em Meshlab da malha poligonal da Anta pintada de Antelas.	11
Ilustração 5 - Hardware do sensor Kinect [19].	13
Ilustração 6 - Nuvem de pontos de um bule de chá (lado esquerdo) e respetiva malha triangular (lado direito).	16
Ilustração 7 - Fases de processamento no funcionamento da Kinect Fusion [22].	18
Ilustração 8 - Comparação entre duas malhas poligonais sem textura (superior e inferior) obtidas com a Kinect Fusion (lado esquerdo) e as respetivas fotografias do local (lado direito).	19
Ilustração 9 - Malha poligonal, com textura associada, obtida numa única aquisição utilizando a aplicação Kinfu.	20
Ilustração 10 - Exemplo de funcionamento do software PCL Kinfu Large Scale [35].	21
Ilustração 11 - Malha poligonal com textura associada obtida, numa única aquisição, pela Kinfu Large Scale, após a “colagem” de três malhas resultantes.	22
Ilustração 12 - Fotografia de uma zona do IEETA usada como teste de aplicações comerciais de aquisição.	23
Ilustração 13 - Visualização a utilização da ferramenta KScan3D.	24
Ilustração 14 - Aquisição utilizando o software Skanect de uma nuvem com 1 217 617 pontos (imagem superior) e uma malha poligonal, obtida após reconstrução dessa nuvem, com 124 482 triângulos (imagem inferior).	25
Ilustração 15 - Utilização da ferramenta ReconstructMe [26].	26
Ilustração 16 - Segmentação duma nuvem de pontos com uma cor diferente associada a cada plano [39].	31
Ilustração 17 - Representação de 3 planos principais obtidos.	31
Ilustração 18 - Representação da sala de reuniões do IEETA após os pontos serem projetados.	33
Ilustração 19 - Representação de um conjunto de pontos que define um polígono inicial (lado esquerdo) o processamento do algoritmo que aplica o Convex Hull (centro) e o resultado obtido (lado direito).	34
Ilustração 20 - Resultado após aplicar o algoritmo que aplica o Convex Hull a cada um dos planos encontrados.	35
Ilustração 21 - Imagem de uma nuvem de pontos (lado esquerdo) e aplicação da Reconstrução de Poisson (lado direito) [40].	36
Ilustração 22 - Visualização dos passos da aplicação da reconstrução através do método de Poisson [37].	37
Ilustração 23 - Reconstrução de Poisson em que foi dado como entrada a nuvem de pontos representativa da sala (lado esquerdo) e o resultado desse algoritmo (lado direito) aplicado e visualizado no Meshlab.	38

Ilustração 24 - Resultado da decimação do número de polígonos para 150, a partir do modelo obtido após a Reconstrução de Poisson.	39
Ilustração 25 - Formatação utilizada para armazenar os coeficientes associados a cada material.	42
Ilustração 26 - Seleção de um retângulo da Anta, associando um material a esse conjunto de células.....	43
Ilustração 27 – Sala com todos os polígonos que definem o chão associados a um material.	43
Ilustração 28 - Representação do sistema de coordenadas RGB.....	46
Ilustração 29 - Representação do sistema de coordenadas HSV.....	46
Ilustração 30 - Nuvem de pontos singular adquirida pela Kinect com textura associada.	47
Ilustração 31 - Nuvem de pontos com textura associada adquirida pela Kinect (lado esquerdo) e aplicação no Meshlab da Reconstrução de Poisson e associação de textura à malha (lado direito).....	48
Ilustração 32 - Abertura no VTK do ficheiro .ply exportado do Meshlab com a malha triangular com textura associada (lado esquerdo) e segmentação de materiais (lado direito).....	49
Ilustração 33 - Falhas na transferência de cor entre a nuvem de pontos (lado esquerdo) e a malha obtida pela Reconstrução de Poisson (lado direito).	50
Ilustração 34 - Segmentação da malha poligonal colorida (lado esquerdo) através do HSV e seleção de 3 materiais diferentes para parede, chão e cadeira (lado direito).	50
Ilustração 35 - Comparação entre o algoritmo de segmentação RGB (lado esquerdo) e o algoritmo que utiliza HSV (lado direito).	51
Ilustração 36 - Resultados de toda a informação necessário para a produção de um modelo geométrico.	52
Ilustração 37 - Voxelização de um bule de chá [52].....	54
Ilustração 38 - Malha poligonal de um bule de chá (lado esquerdo) e respectiva voxelização a 22 050Hz (lado direito).....	56
Ilustração 39 - Exemplo de voxelização da Anta a 11 025 Hz (lado esquerdo) e 22 050Hz (lado direito).....	56
Ilustração 40 - Seleção de dois materiais diferentes (lado esquerdo) e respetivo resultado da voxelização (lado direito).	57
Ilustração 41 - Imagem onde é visível o modelo da sala de reuniões do IEETA (azul) e o cubo na origem da voxelização (vermelho).	59
Ilustração 42 - Segmento do algoritmo usado para preenchimento do volume exterior.	59
Ilustração 43 - Propagação do som numa sala e representação do valor do RT ₆₀	60
Ilustração 44 - Processo de voxelização com o cálculo do valor de RT ₆₀	62
Ilustração 45 - Cabeça colocada no centro de uma sala rectangular 5x4x3 m e 3 nós representando a colocação de fontes.....	63
Ilustração 46 - Resultado do alinhamento das várias nuvens de pontos no interior da sala de reuniões do IEETA.	66
Ilustração 47 - Nuvem de pontos da sala de reuniões em que cada plano obtido é apresentado por uma cor diferente.	67
Ilustração 48 - Pontos que pertencem ao mesmo plano após aplicação do algoritmo RANSAC (lado esquerdo) processa o algoritmo que aplica o Convex Hull a esses pontos (lado direito).	67

Ilustração 49 - Resultado após implementar o algoritmo que aplica o Convex Hull a cada um dos planos encontrados.	68
Ilustração 50 - Resultado da implementação do algoritmo que aplica o Convex Hull a toda a nuvem de pontos após a projeção dos mesmos.	68
Ilustração 51 - Imagem interior da sala após a implementação do algoritmo que aplica o Convex Hull a toda a nuvem de pontos após a projeção dos mesmos.	69
Ilustração 52 - Várias perspetivas do resultado da aplicação do algoritmo da Reconstrução de Poisson.	69
Ilustração 53 - Comparação entre a malha poligonal obtida (cinzento) e a sala original modelada recorrendo ao SketchUp (amarelo).	70
Ilustração 54 - Seleção do chão como madeira e o resto da sala como gesso e obtenção do RT_{60} durante a voxelização.	71
Ilustração 55 - Resultado da voxelização a 11 025Hz da sala de reuniões do IEETA.	72
Ilustração 56 - Malha poligonal original após a aplicação da Reconstrução de Poisson (lado esquerdo) e resultado após simplificação através de decimação (lado direito).	73
Ilustração 57 - Fotografia tirada na sala de reuniões durante a simulação.	73
Ilustração 58 - Imagem visualizada pelo utilizador durante a simulação.	74
Ilustração 59 - Diagrama com toda a informação necessária para fornecer um modelo físico à aplicação de auralização.	75

1 Introdução

Com a expansão e desenvolvimento da tecnologia, houve um aumento progressivo da motivação em estudar mais profundamente o ambiente que nos rodeia de modo a compreender o seu comportamento e possibilitar a sua simulação virtual.

Neste sentido, surge uma disciplina de estudo, denominada Realidade Virtual, cujo objetivo é um ambiente artificial o mais idêntico possível ao mundo real em termos sensoriais. Isto implica, não só reproduzir a perceção visual, mas também as sensações táteis, auditivas e olfativas, de modo a criar uma sensação de presença física em locais reais ou mundos criados pela imaginação [1].

As primeiras aplicações que surgiram nesta área foram desenvolvidas a nível militar (nomeadamente para simulação de pilotagem de aviões ou situações de combate). No entanto, nos últimos anos a sua importância tem aumentado e tem-se assistido à investigação e desenvolvimento de produtos a nível comercial (desde o uso educacional até à inclusão em jogos de computador, visualização de paisagens naturais de difícil acesso ou visitas virtuais a museus).

1.1 Auralização

Dada a importância do termo auralização para esta dissertação, começaremos por explicar o seu significado. Esta palavra surge como analogia ao termo visualização, sendo a visualização definida como o processo de tornar algo visível em animação e computação gráfica. Deste modo, entende-se como auralização a criação de cenários acústicos através de dados gerados computacionalmente [2].

De um modo mais completo, podemos definir a auralização como o processo de representação de áudio, através de modelos matemáticos ou físicos, que simule a experiência auditiva binaural ([3, 4]) de um ouvinte colocado em determinada posição do espaço que foi modelado. Os dados sobre o material, fontes sonoras (música, discurso) e posição do ouvinte são tomados em conta para a filtragem computacional dos sinais sonoros originais.

Como resultado, esses sinais ao chegar ao ouvinte, permitem simular uma sensação auditiva que tenha em conta os fenômenos inerentes à propagação do som (reflexão, difração, refração, interferência) [5].

A auralização recorre essencialmente a modelos acústicos de dois tipos: modelos geométricos e modelos físicos.

1.1.1 Modelos geométricos

Os modelos geométricos são alimentados por uma malha poligonal fechada que representa as características geométricas do local a auralizar. Adicionalmente é necessário fornecer a posição espacial da fonte sonora e ouvinte, assim como as propriedades acústicas (coeficiente de reflexão/absorção) que caracterizam cada uma das células que compõem essa malha, através da identificação dos materiais que lhes estão associados.

Usando esses dados, foi desenvolvido no âmbito do projeto AcousticAVE, *software* que, através de cálculos geométricos e tendo em conta as posições das células e os coeficientes de reflexão/absorção dos materiais nelas presentes, calcula as reflexões dos sons emitidos pelas fontes. Após este processamento, através da introdução de atrasos no sinal sonoro injetado, é simulada a propagação do som e são reproduzidos os sons que chegam à posição do ouvinte [6]. É possível observar esta situação na Ilustração 1, em que se representa a posição de uma fonte (F) e um ouvinte (O) e se observam algumas reflexões possíveis e as várias direções com que chegam ao recetor.

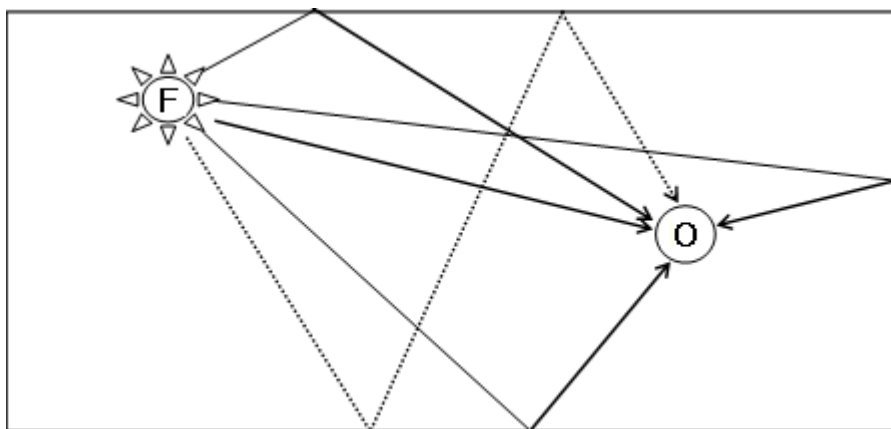


Ilustração 1 - Ilustração de um modelo geométrico.

1.1.2 Modelos Físicos

Os métodos que recorrem a modelos físicos fazem a simulação acústica tendo em conta as propriedades de uma onda sonora (frequência, comprimento de onda, amplitude, direção, pressão e intensidade sonora). Estes modelos são alimentados com uma grelha tridimensional que cobre todo o volume que constitui o meio de propagação, sendo cada nó da grelha identificado relativamente ao material que lhe corresponde, ou seja, cada nó pode estar no interior do modelo (nós de ar), exterior (nós inativos pois não influenciam a propagação sonora) e fronteira (nós que identificam o material associado à posição do modelo onde se encontram). A esta informação junta-se a localização de um ouvinte e a injeção de uma fonte sonora contendo um som “puro” gravado numa câmara anecoica, ou seja, um sinal sonoro que não contenha as reflexões geradas pela sala durante a gravação.

Com estes dados é feita uma simulação da propagação sonora (iniciada no nó fonte que possui o sinal sonoro injetado) ao longo de cada nó de ar dentro da sala para os seus vizinhos, até atingir as fronteiras e ser refletido mediante as propriedades dos materiais presentes nos nós fronteira. Para melhor compreensão deste método, é possível observar a Ilustração 2, que apresenta uma visualização da propagação das ondas sonoras num modelo físico de uma sala retangular com três compartimentos, evidenciando fenómenos (nomeadamente difração e interferência), que os modelos geométricos são incapazes de modelar.

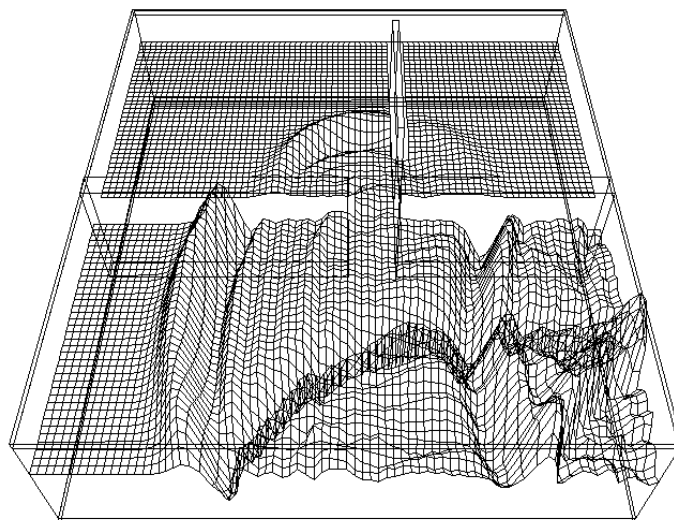


Ilustração 2 - Visualização da propagação das ondas sonoras num modelo físico de um espaço fechado [7].

1.2 Motivação

Esta dissertação foi proposta na sequência do projeto de investigação AcousticAVE, em curso no Instituto de Engenharia Eletrónica e Telemática de Aveiro (IEETA) e financiado pela Fundação para a Ciência e a Tecnologia (FCT), que visa o desenvolvimento de modelos e aplicações de auralização em ambientes de Realidade Virtual. O objetivo principal do projeto é desenvolver um pacote completo de auralização baseado em auscultadores que permita a espacialização do som tendo em conta as propriedades acústicas da sala e a trajetória da fonte sonora, assim como a orientação da cabeça e posição de um ouvinte, ambos seguidos em tempo real. Isto ajudará a compreender melhor o mundo que nos rodeia, na medida em que tornará possível por um lado estudar o passado, ao recriar virtualmente a acústica de antigos monumentos já destruídos pelo tempo (por exemplo uma Anta), por outro prever o futuro ao planear e otimizar a acústica de um projeto de um local (por exemplo uma sala de aulas).

Os principais desafios são a sincronização audiovisual e a construção de modelos acústicos de salas reais de forma computacionalmente eficiente.

Esta dissertação procura dar uma contribuição através do desenvolvimento de *software* que permita a construção de modelos acústicos de salas a partir de dados sobre a sua configuração física. Estes dados podem ser obtidos através de medição *in situ* com instrumentos (por exemplo *laser scanners* ou sensores como o Microsoft Kinect estudado nesta dissertação na secção 2.2) ou por desenho com ferramentas CAD de modelação (SketchUp [8], AutoCAD [9], Blender [10]). Com os dados adquiridos pretende-se construir modelos adequados para prever o comportamento das salas em termos de propagação do som (fornecendo-se também as características das fontes sonoras e ouvintes presentes na sala, nomeadamente as suas posições).

1.3 Objetivos

Como já referido, tanto o modelo geométrico como o modelo físico utilizados para auralização de uma sala, necessitam de ser alimentados com dados relativos à geometria, materiais e posição da fonte e ouvintes no espaço. Deste modo, nesta dissertação vão ser

desenvolvidos métodos de aquisição e tratamento de dados que servem para alimentar esses modelos para posteriormente possibilitar a auralização de uma sala.

O primeiro passo consiste em obter um modelo 3D/malha poligonal que represente a geometria da sala em estudo. Para criar esses modelos já existem bastantes pacotes *software* de modelação atualmente no mercado (SketchUp [8], AutoCAD [9], Blender [10]). No entanto, quanto mais complexa é a geometria do modelo, mais tempo é necessário para fazer a sua modelação precisa. Como forma de facilitar esta aquisição, têm sido utilizados sistemas baseados em *laser* profissionais para proceder ao levantamento da geometria (FARO Photon80 ou Creaform VIUScan utilizados para adquirir nuvens de pontos [11]). Apesar de apresentarem bons resultados, o seu preço de mercado é elevado. No âmbito desta dissertação foi estudada a possibilidade de obter nuvens de pontos para modelação de espaços usando um sensor de custo mais reduzido: uma Microsoft Kinect. De notar que o sensor Kinect tem ainda a possibilidade de aquisição de textura o que permite uma visualização mais realista (com recurso a texturas reais) e pode ainda ser um auxílio na identificação dos materiais de um modelo, através das suas cores.

Para permitir a auralização, a geometria não é suficiente, é ainda necessário atribuir materiais às várias células da malha poligonal obtida. Para este efeito pretende-se criar uma aplicação de identificação dos materiais presentes em cada célula, quer manualmente através da seleção de células e associação direta ao respetivo material, quer através da utilização das cores obtidas na textura para associar cada cor a um material correspondente (por exemplo associar a cor castanha do chão de uma sala a madeira). Esta ferramenta também deve permitir definir a posição e orientação do ouvinte e das fontes.

Por fim, um objetivo desta dissertação é permitir gerar uma grelha de nós tridimensionais a partir de um modelo poligonal (designando-se este processo de voxelização que é explicado na secção 5.1). O resultado deste processo de voxelização vai permitir alimentar o modelo físico de auralização (secção 1.1.2) com uma grelha, em que todos os nós estão devidamente identificados como cabeça do ouvinte, fonte, ar, fronteira/material e/ou inativos.

Durante este processo irá obter-se também o RT_{60} , sendo este um valor utilizado na auralização e que se encontra diretamente relacionado com o volume do espaço físico e a área dos materiais constituintes.

No final, para efeitos de teste é feita a modelação de uma sala de reuniões situada no IEETA e são apresentados os resultados obtidos para este espaço físico em cada uma das fases do processo.

O objetivo é a criação de um modelo poligonal de uma sala real que, após a aquisição e tratamento desses dados com vista a configurar um modelo acústico da mesma, possibilite uma simulação virtual que inclui sensações visuais e auditivas simultaneamente.

É possível observar na Ilustração 3 o diagrama que resume toda a informação que é necessário obter para a produção de um modelo geométrico e físico que alimentam o processo de auralização.

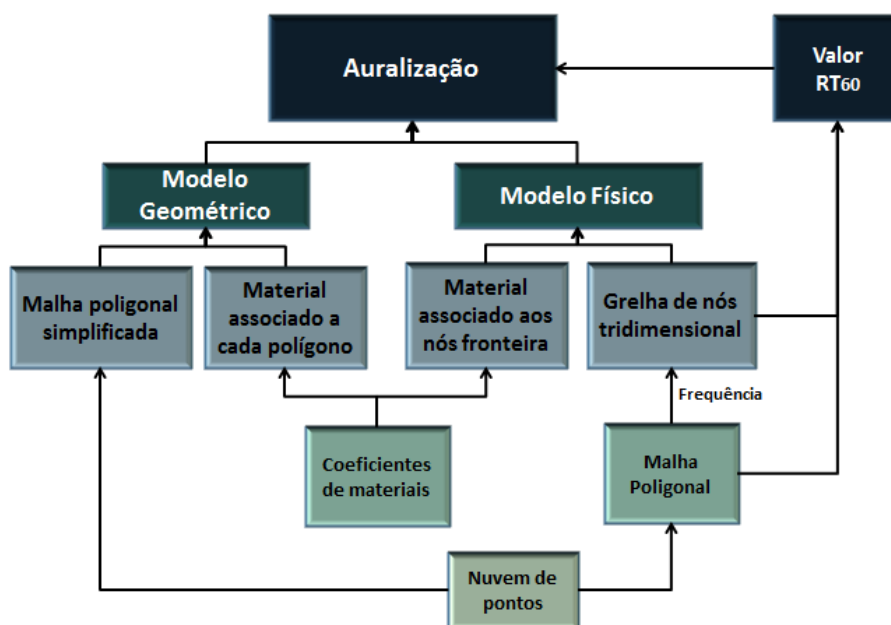


Ilustração 3 - Diagrama da informação necessária para alimentar o processo de auralização.

1.4 Estrutura da tese

A organização desta dissertação apresenta a seguinte estrutura:

O **Capítulo 2** (Ferramentas) começa por descrever os recursos utilizados, apresentando os pacotes de *software* e as características que suportaram a escolha dos mesmos e o sensor Microsoft Kinect que é utilizado para aquisição.

No **Capítulo 3** (Aquisição 3D de um ambiente real) é apresentado o estudo de várias possibilidades de aplicações existentes para aquisição de malhas poligonais e todo o tratamento que permite obter uma malha poligonal para alimentar o modelo geométrico para auralização.

O **Capítulo 4** (Aplicação para configuração do modelo para auralização) descreve as técnicas de configuração utilizadas para permitir fazer a seleção e respetiva associação de materiais a cada célula da malha poligonal. Inicialmente é estudado o processo de seleção manual dos materiais que permite uma associação direta e, na busca de um método semiautomático de seleção, é estudada a possibilidade da utilização da textura para seleccionar materiais através da cor.

O **Capítulo 5** (Processamento do modelo poligonal para alimentar modelos físicos de auralização) define o conceito de voxelização que nos permite a geração de uma grelha de nós tridimensional. De seguida são apresentados todos os passos que permitem, através da alimentação com uma malha poligonal, gerar uma grelha tridimensional de nós com a identificação de nós de ar, fronteira/material e inativos. É também apresentada a definição e importância do valor de RT_{60} na projecção acústica de salas e os procedimentos que permitem a sua obtenção após o processo de voxelização.

No **Capítulo 6** (Resultados obtidos na sala de reuniões do IEETA) é apresentada a aplicação dos processos descritos nos capítulos anteriores a uma sala de reuniões do IEETA e são expostos os resultados obtidos em cada uma das fases de processamento.

O **Capítulo 7** (Conclusões e trabalho futuro) resume os resultados obtidos com a realização desta dissertação e identifica as contribuições tanto para o projeto AcousticAVE, como para a área de modelação acústica de salas. Por fim são referidos os vários caminhos que podem ser tomados para futuros desenvolvimentos desta dissertação.

2 Ferramentas de desenvolvimento

Neste capítulo irão ser apresentados os recursos utilizados (bibliotecas e sensor Kinect) para a elaboração deste trabalho.

2.1 Aplicações

De seguida são apresentadas as aplicações e bibliotecas utilizadas, assim como as suas principais características e as razões que levaram à escolha das mesmas.

2.1.1 Visualization Toolkit (VTK)

O VTK (abreviatura de Visualization Toolkit) é uma biblioteca *open-source* gratuita que teve origem em Dezembro de 1993, tendo sido criada como um *software* de suporte para o livro "Visualization Toolkit: An Object Oriented Approach to 3D Graphics" [12]. O VTK tem a sua maior utilização no auxílio a aplicações médicas mas também é muito utilizado para interação em ambientes 3D.

Esta biblioteca é compatível com a maior parte dos sistemas operativos Windows, Mac e plataformas UNIX.

Optou-se pela utilização do VTK para o trabalho proposto porque é uma aplicação orientada a objetos de alto nível, o que permite uma maior abstração e um uso simples e intuitivo das suas funções, quando comparado com outras bibliotecas como por exemplo a OpenGL [13]. Possui centenas de classes e funções que permitem uma grande diversidade nos formatos de importação de malhas poligonais (.obj e .ply, por exemplo). Trata-se ainda de uma biblioteca, que incorpora classes necessárias a esta dissertação, para visualização 3D das malhas poligonais ou nuvens de pontos, assim como o seu tratamento (triangulação, voxelização). Já para exportação, vai permitir igualmente alguma variedade na escolha em relação ao tipo de ficheiro, possibilitando alguma liberdade de escolha para fornecer as malhas poligonais (explicadas na secção 3.1) dos modelos geométricos que alimentam a auralização.

Em contrapartida, como desvantagem principal, surge a sua fraca documentação,

que por vezes torna complicada a compreensão correta das suas funções e aumenta o tempo de desenvolvimento.

2.1.2 Point Cloud Library (PCL)

O Point Cloud Library (PCL) é um projeto *open-source* em larga escala (o código fonte é disponibilizado de acordo com a licença BSD, Berkeley Software Distribution, sendo gratuito o seu uso para investigação) e autónomo, desenvolvido para imagens 2D/3D e processamento de nuvens de pontos [14]. Esta biblioteca, pela característica *open-source* que possui, com a ajuda de financiadores importantes (como a Google, Nvidia ou a Toyota) e utilizadores de todo o mundo, apresenta funções e algoritmos constantemente otimizados e adaptados às características computacionais dos computadores que são apresentados no mercado.

A biblioteca está dividida numa coleção de pequenos módulos em linguagem C++. No caso particular para esta dissertação, um dos módulos designado IO (Input-Output), irá conter classes que possibilitam implementar, através de uma interface OpenNI (Open Natural Interaction), uma conexão com a Microsoft Kinect, que irá ser utilizada para obtenção de nuvens de pontos. Permite também a visualização (recorrendo ao módulo Visualization da biblioteca PCL) e processamento das nuvens adquiridas, de modo a criar uma malha poligonal que é possível ser visualizada através da biblioteca VTK que está integrada no PCL. Já para visualização de nuvens de pontos, recorre à função PCD Viewer, que permite abrir o formato PCD (Point Cloud Data), específicos desta biblioteca e com que são adquiridos os pontos obtidos através da Microsoft Kinect.

2.1.3 Meshlab

O Meshlab é um programa disponibilizado a partir de 2005 como um projeto elaborado por um conjunto de estudantes da Universidade de Pisa [15]. Disponibilizado em versão *open-source* para Windows ou Linux, permite manipular malhas triangulares em 3D. Como principal característica do uso desta ferramenta encontra-se a variedade de aplicações disponibilizadas (visualização, edição, reconstrução e tratamento de malhas), aliado à simplicidade de uso das mesmas. Permite também a exportação para vários tipos

de ficheiros, entre eles *.obj* e *.ply* que irão ser os mais utilizados neste trabalho. Como principais vantagens no decorrer desta dissertação, pode-se referir a facilidade e simplicidade, quando comparado com o VTK, nas seguintes funções: correr o algoritmo Iterative Closest Point (ICP) após um alinhamento inicial definido pelo utilizador (secção 3.3.1), a limpeza de pontos que possuam erros de aquisição nas malhas adquiridas (secção 3.3.1) e correr algoritmos de simplificação das malhas poligonais (secção 3.3.5). Este programa possui ainda uma implementação da reconstrução de Poisson, que irá ser utilizada nesta tese (secção 3.3.5), assim como um processo já implementado de projeção dos valores de textura da nuvem de pontos original para a malha poligonal (secção 4.3.3).

Todos estes algoritmos integrados no Meshlab têm a vantagem, quando comparado com o VTK ou OpenGL, de não necessitar de qualquer fase de programação para a sua utilização, sendo apenas necessário interagir com o programa e escolher a função que pretendemos, o que permite trabalhar eficazmente sobre a nuvem de pontos obtida na Kinect (ao abstrair da programação), permitindo uma rápida análise e comparação de resultados das várias funções e parâmetros testados.

Como exemplo de visualização no Meshlab, podemos observar na Ilustração 4, a importação de uma malha poligonal, previamente obtida recorrendo a um laser SICK LMS 200 modificado [16], da Anta pintada de Antelas situada em Oliveira de Frades.

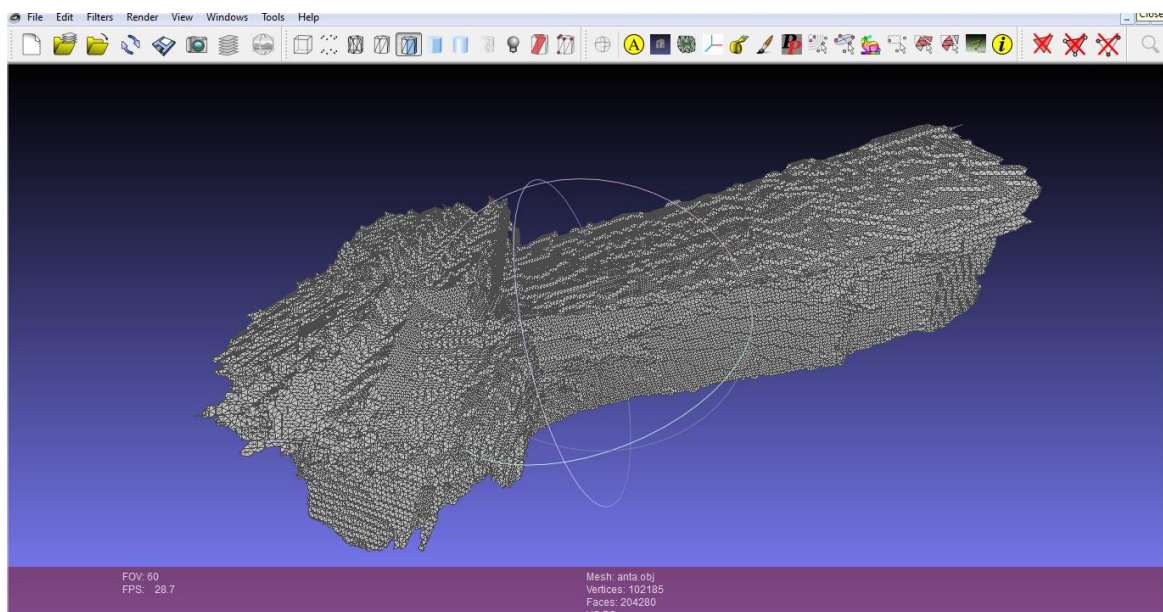


Ilustração 4 - Visualização em Meshlab da malha poligonal da Anta pintada de Antelas.

2.2 Sensor Microsoft Kinect

A Kinect é um sensor de movimento que possui uma tecnologia de *hardware* desenvolvida pela empresa Primesense e *software* desenvolvido pela empresa Rare, que é propriedade da Microsoft. O *software* original permite a utilização da Kinect na consola Xbox 360 mas, dadas as potencialidades da utilização desta tecnologia como *laser scanner* económico, a Kinect foi adaptada pela Microsoft para o sistema operativo Windows, convertendo o seu cabo de saída para tipo USB (Universal Serial Bus). Surgiu assim uma tecnologia de aquisição de nuvens pontos a um valor bastante acessível (a rondar os 150 euros) a qualquer programador para uso pessoal.

Na Ilustração 5 podemos ver a constituição da Kinect a nível de *hardware*: uma câmara RGB que permite guardar imagens com uma resolução de 1280x960 permitindo assim armazenar informação ao nível das cores e um emissor de infravermelho e um sensor de profundidade. O emissor funciona ao emitir vários raios que são reflectidos nas superfícies e adquiridos pelo sensor e convertidos em informação de profundidade através da distorção do padrão infravermelho [17, 18]. Esta tecnologia possibilita, através da informação de cor e profundidade, fornecer uma nuvem de pontos com textura em tempo real.

A Kinect também permite a gravação de sons através dos quatro microfones incorporados mas, no contexto desta dissertação, esta característica não irá ser utilizada. Possui ainda um acelerómetro de 3 coordenadas, que permite determinar a orientação da Kinect em cada instante. Pelas especificações fornecidas pela Microsoft em [19], podemos ver que a aquisição de nuvens de pontos pode ser feita entre um mínimo de distância de cerca de 40 centímetros e, um máximo para um valor próximo dos 3 metros. A aquisição de pontos é feita a uma velocidade de 30 FPS (*frames* por segundo) no sistema de coordenadas (x,y,z).

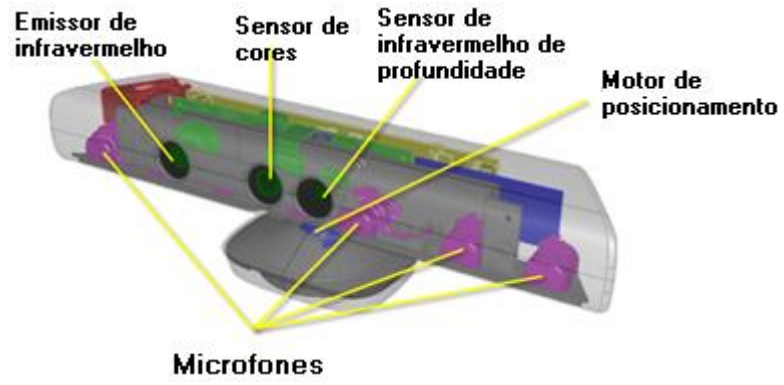


Ilustração 5 - Hardware do sensor Kinect [19].

3 Aquisição 3D de um ambiente real

Neste capítulo é apresentado todo o processo que permite adquirir a informação geométrica de uma sala real, resultando numa malha poligonal. São estudados e comparados vários pacotes de *software* existentes para aquisição de nuvens de pontos. De seguida, são apresentados os métodos encontrados para obter modelos poligonais a partir dessas nuvens. Além disso, e na medida em que a complexidade computacional dos métodos geométricos depende do número de polígonos considerado, procuraram-se métodos para simplificar o modelo e assim limitar o número de polígonos, ponto crucial para permitir uma auralização com modelos geométricos em tempo real.

3.1 Nuvem de pontos e malhas poligonais

Por nuvem de pontos entende-se um conjunto desorganizado de pontos no espaço 3D, definidos pelas coordenadas (x,y,z) que usualmente representam a superfície exterior de um objeto.

Para gerar nuvens de pontos a partir de objetos/superfícies reais, usualmente recorre-se a *scanners* 3D, que permitem uma grande cadência de aquisição (na ordem das dezenas ou centenas de milhar de pontos por segundo). É possível ver um exemplo da representação de uma nuvem de pontos de um bule de chá e uma possível malha triangular reconstruída a partir dessa nuvem na Ilustração 6.

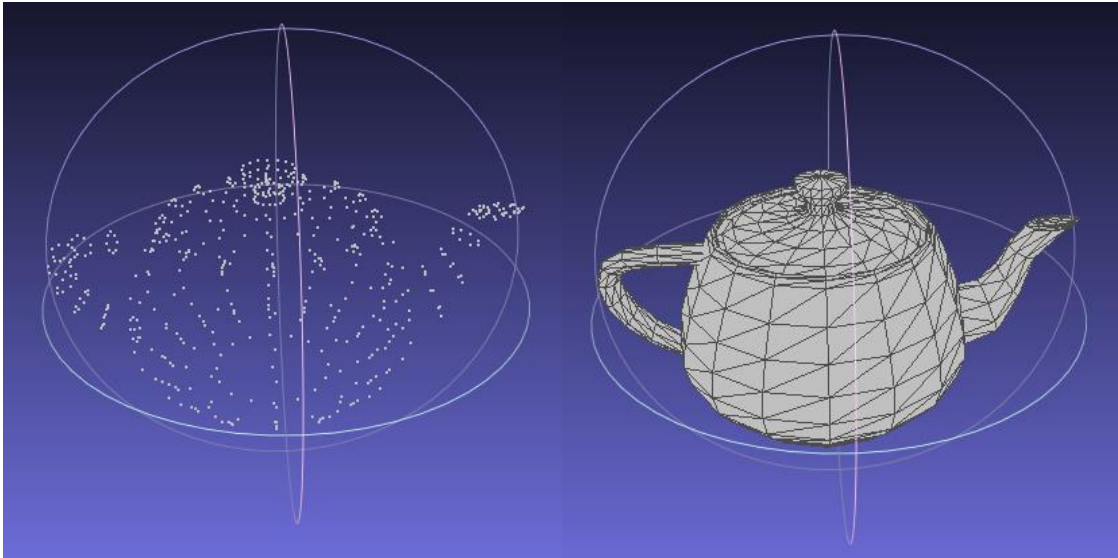


Ilustração 6 - Nuvem de pontos de um bule de chá (lado esquerdo) e respectiva malha triangular (lado direito).

A partir destas nuvens é possível gerar uma malha poligonal que é representada como um conjunto de polígonos adjacentes (no caso desta dissertação serão triângulos, fornecendo assim malhas triangulares), definindo superfícies 3D. As malhas poligonais podem ser fechadas, sendo cada polígono definido pelo conjunto dos seus vértices. O processo de conversão de uma nuvem de pontos numa malha triangular é designado de triangulação e foi inicialmente estudado por E. Keppel [20] em 1975.

Com o passar dos anos e o aumento de recursos computacionais disponíveis, desenvolveram-se algoritmos com melhor desempenho, entre eles a triangulação de Delaunay [21], que permite a triangulação de um conjunto de pontos com um bom desempenho ao nível da velocidade de processamento.

3.2 Aquisição de nuvens de pontos

Para obter uma representação geométrica de uma sala utilizando a Kinect é possível recorrer a algumas aplicações já existentes, nomeadamente: Kinect Fusion [22], KinFu e KinFu Large Scale incluídos na biblioteca PCL [23], KScan3D desenvolvido pela empresa d3D3 Solutions [24], Skanect produzido pela empresa ManCTL [25] e o programa ReconstructMe da autoria da PROFACTOR GmbH [26].

Todas estas ferramentas apresentam como principal característica a facilidade no seu uso para fazer digitalização de objetos através da utilização de um *scanner* comercial,

como por exemplo a Kinect. Estas aplicações foram sobretudo desenvolvidas para capturar pequenos objetos, contudo algumas delas (KinFu Large Scale, ReconstructMe e Scanect) publicitam também o seu uso para adquirir espaços fechados mais amplos (salas ou quartos por exemplo). Estas ferramentas baseiam-se na aquisição sequencial de nuvens de pontos sucessivas que são depois alinhadas entre si.

O uso destas aplicações requer uma elevada capacidade de processamento para apresentar em tempo real os resultados do alinhamento das várias nuvens de pontos, resultados esses que permitem constatar se está a funcionar corretamente ou é necessário recomeçar ou alterar a velocidade de rotação da Kinect.

3.2.1 Kinect Fusion

A Microsoft criou um projeto [27] que tem como objetivo a captura e reprodução virtual de objetos reais. Este objetivo é alcançado através do mapeamento do espaço que nos rodeia (desde espaços grandes como uma sala, a objetos que se encontrem no interior dessa mesma sala) para possibilitar uma posterior interação em tempo real. Deste projeto resultou uma aplicação designada de Kinect Fusion.

Apesar de a aplicação não ser *open-source*, os passos e algoritmos usados para reconstrução encontram-se descritos em vários artigos [28, 29].

O conceito fundamental para o processamento em tempo real (para permitir a interação com o utilizador) está na utilização das capacidades computacionais da Unidade de Processamento Gráfico (GPU), em detrimento do habitual processador utilizado pelo computador (CPU). Isto porque, o processador possui menos de uma dezena de núcleos, enquanto uma boa GPU chega a possuir valores da ordem das centenas, o que permite obter velocidades de processamento superiores através de algoritmos que possuam operações paralelizáveis.

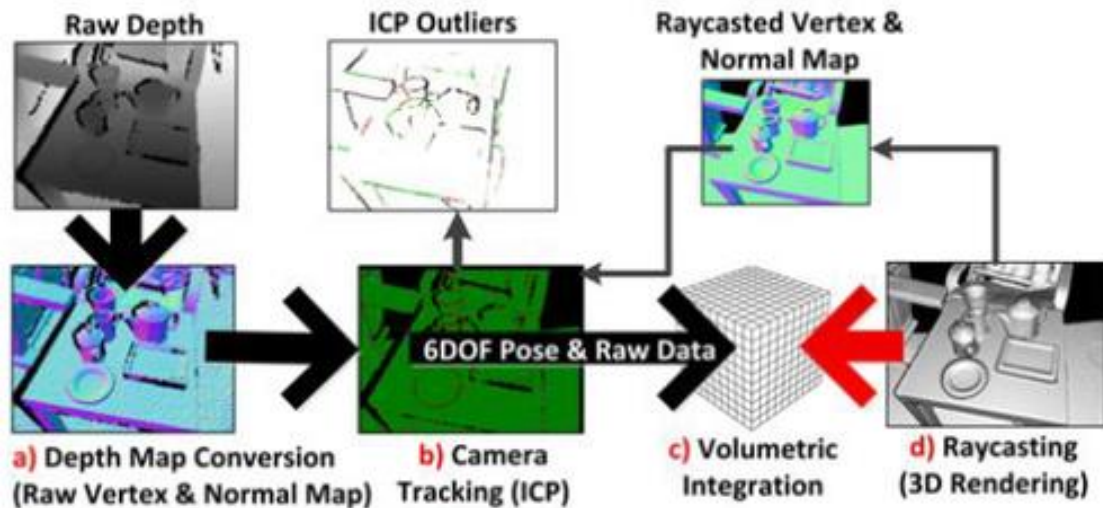


Ilustração 7 - Fases de processamento no funcionamento da Kinect Fusion [22].

A Ilustração 7, disponibilizada pela Microsoft em [22], apresenta os principais passos utilizados pela Kinect Fusion: inicialmente é adquirido pela própria Kinect um mapa de profundidade que é convertido numa nuvem de pontos 3D (imagem a) da Ilustração 7), contendo informação sobre os valores dos vértices e normais associadas [30].

No início da aquisição, a primeira nuvem é adquirida e colocada num cubo volumétrico (de dimensões 3x3x3 metros apresentado na imagem c) da Ilustração 7), sendo apresentado ao utilizador os resultados, em tempo real, através da técnica de Ray Casting [31] (visível na imagem d) da Ilustração 7).

Cada aquisição seguinte possui um passo adicional ao ser comparada com a última que tiver sido guardada no modelo e é processado com um algoritmo baseado no ICP [32] (imagem b) da Ilustração 7), para registo de duas nuvens de pontos.

Este algoritmo necessita que as duas nuvens de pontos a comparar tenham um alinhamento inicial razoável. No caso da Kinect Fusion, como referido anteriormente, esta necessidade é garantida pela elevada taxa de aquisição da Kinect (30 *frames* por segundo) e o processamento de imagens ser feito aproximadamente em tempo real (devido às capacidades já referidas da GPU). Deste modo, irá existir uma diferença bastante pequena entre duas aquisições consecutivas garantindo um alinhamento inicial aceitável e reduzindo o tempo de processamento.

Após o alinhamento, a *frame* adquirida, sabendo a sua posição relativamente à armazenada anteriormente, irá ser adicionada ao cubo volumétrico.

A dimensão do cubo volumétrico de armazenamento (3x3x3 metros) é uma das limitações deste *software*, apenas permitindo adquirir informação de um modelo dentro destas dimensões (suspendendo a obtenção de novas *frames* caso estas se situem fora do cubo volumétrico). A Ilustração 8 apresenta duas aquisições utilizando a Kinect Fusion. Quando comparados os resultados deste *software* (lado esquerdo da Ilustração 8), com a imagem original retirada utilizando uma máquina fotográfica (lado direito da Ilustração 8), podemos observar que a malha poligonal adquiriu corretamente a geometria da sala e objetos presentes.

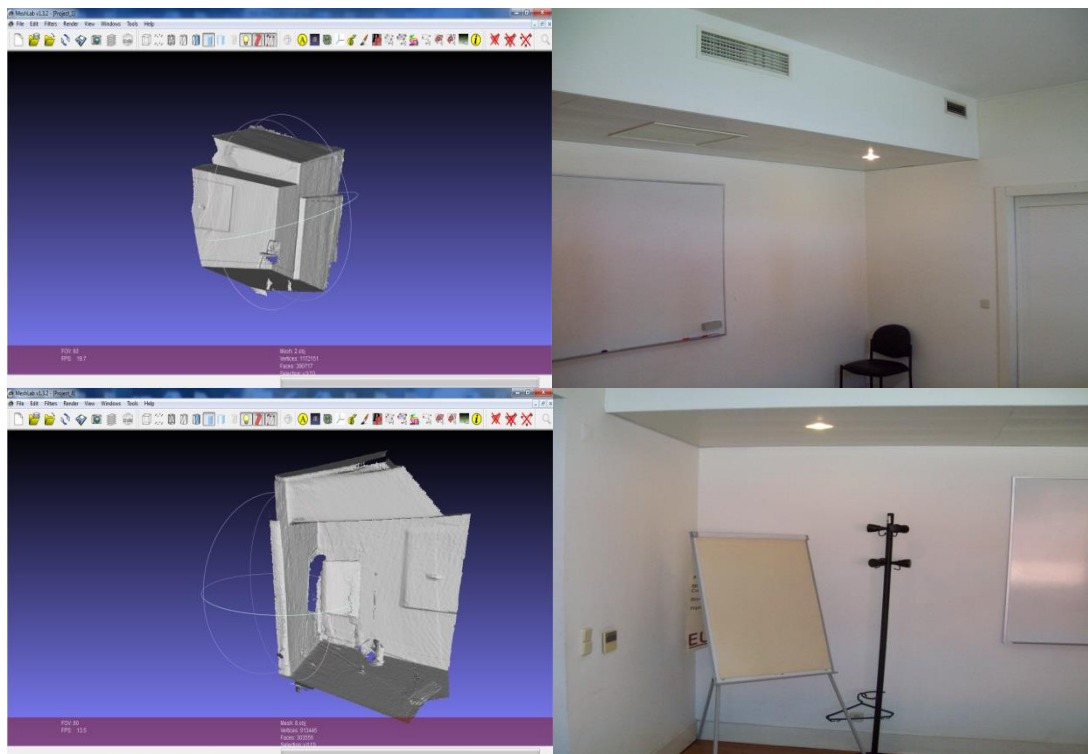


Ilustração 8 - Comparação entre duas malhas poligonais sem textura (superior e inferior) obtidas com a Kinect Fusion (lado esquerdo) e as respectivas fotografias do local (lado direito).

3.2.2 PCL Kinfu

Sendo a Kinect Fusion o resultado de um projeto da Microsoft, com base nos artigos publicados [28, 29] (em que é explicado o seu funcionamento), foi desenvolvido na biblioteca PCL uma implementação *open-source* da aplicação Kinect Fusion, a que se deu o nome de Kinfu [33].

No entanto, comparando a KinFu com o algoritmo da Microsoft [34], verificam-se as seguintes diferenças:

- A versão da Kinect Fusion é feita para Windows, já a versão PCL está focada para correr em Linux;
- Enquanto a Microsoft disponibilizou um *software* que não adquire textura, a KinFu incorpora a integração de informação relativa à cor, resultando numa reconstrução de uma superfície já com textura associada;
- A aplicação Fusion da Microsoft só é compatível com a Kinect, já a KinFu pode ser utilizada com qualquer câmara de aquisição de profundidade, compatível com o *software* OpenNI;
- Na aplicação KinFu foi desenvolvida uma versão, com o nome Large Scale (secção 3.2.3), que não possui a limitação espacial da Fusion.

Na Ilustração 9 é possível observar o resultado da aquisição de uma malha com textura associada utilizando a KinFu.

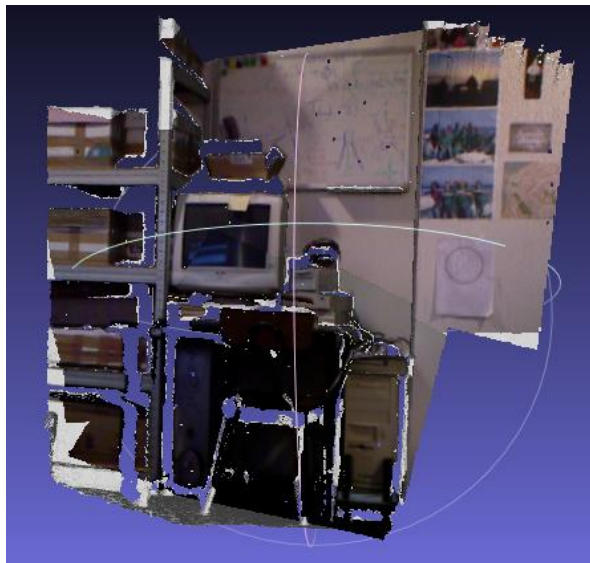


Ilustração 9 - Malha poligonal, com textura associada, obtida numa única aquisição utilizando a aplicação KinFu.

3.2.3 PCL Kinfu large scale

A Kinect Fusion, assim como a versão *open-source* Kinfu, possuem ambas a limitação da aquisição de uma nuvem de pontos num volume definido (aproximadamente de 3x3x3 metros), sendo mais apropriado para a obtenção de objetos pequenos, ao invés de grandes superfícies.

Para colmatar esta dificuldade, os programadores que contribuem para o desenvolvimento da biblioteca PCL, criaram uma versão da Kinfu, a que deram o nome de Kinfu Large Scale, que permite a obtenção de espaços de maior dimensão (como por exemplo uma sala) e menos apropriada para aquisição de pequenos objetos.

O seu funcionamento baseia-se na Kinfu apresentada anteriormente que permite adquirir nuvens de pontos num espaço 3x3x3 metros. No entanto, ao invés de limitar a aquisição de pontos ao atingir o limite do cubo de aquisição, o *software* armazena a nuvem de pontos inicial e começa a adquirir uma nova nuvem, mantendo a informação da posição e orientação desta nova aquisição em relação à armazenada anterior.

Este processo permite a acumulação final de todas as nuvens de pontos alinhadas através de uma só aquisição. A Ilustração 10 demonstra este processo, em que cada cubo representa a limitação da Kinfu e da Kinect Fusion, estando também representada a posição e ângulo de visão da Kinect (pirâmide rosa) que originou a criação do cubo seguinte, ao serem atingidos os limites de aquisição para o cubo inicial.

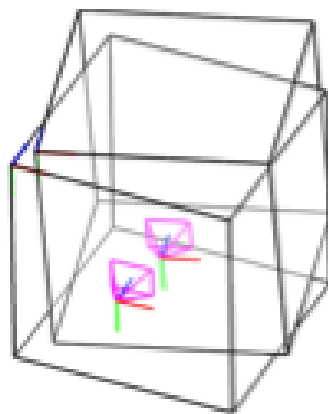


Ilustração 10 - Exemplo de funcionamento do software PCL Kinfu Large Scale [35].

É de notar, que a dimensão e detalhe do modelo final, irá depender do número de imagens adquiridas podendo resultar em modelos de uma dimensão muito significativa (facilmente se atingem modelos com centenas de Mb, ou mesmo na ordem de Gb).

De referir que, o resultado final desta ferramenta, não é apenas uma malha triangular mas sim cada uma das malhas armazenadas, de dimensões 3x3x3 metros que limitam a Kinfu. Isto significa que irá ser necessário recorrer a uma ferramenta (como por exemplo o Meshlab) para fazer o registo de todas essas aquisições para obter apenas uma malha com o resultado final. Na Ilustração 11 observa-se o resultado da aquisição, num único varrimento com a Kinfu Large Scale, de três malhas (com textura associada). Pode-se observar o local que foi possível adquirir na Kinfu (Ilustração 9) para compreender que foram ultrapassadas as limitações da Kinfu.

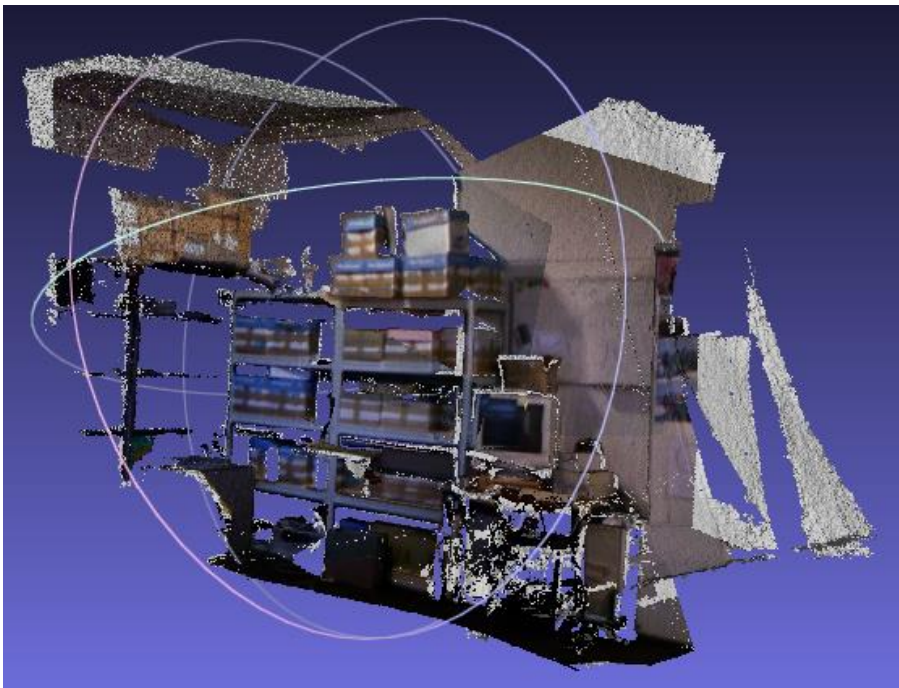


Ilustração 11 - Malha poligonal com textura associada obtida, numa única aquisição, pela Kinfu Large Scale, após a "colagem" de três malhas resultantes.

3.2.4 Kscan3D, Skanect e ReconstructME

Optou-se por apresentar estes três pacotes de *software* numa só secção dado que são programas comerciais, não disponibilizados gratuitamente e que implicam a aquisição de licenças para a sua utilização sem limitações. Nesta dissertação apenas foram testadas as

versões livres e são apresentadas para fornecer alguma informação sobre opções comerciais disponíveis em alternativa às ferramentas da Microsoft (Kinect Fusion) e da biblioteca PCL (KinFu).

Devido à impossibilidade de acesso ao código e documentação sobre o seu funcionamento apenas são apresentadas algumas imagens da sua utilização, assim como uma breve análise das mesmas e aquisição de um pequeno espaço do IEETA representado na Ilustração 12.



Ilustração 12 - Fotografia de uma zona do IEETA usada como teste de aplicações comerciais de aquisição.

A ferramenta KScan3D funciona através da aquisição de várias nuvens de pontos individuais, obrigando, de seguida, a um alinhamento manual e está projetada para adquirir objetos pequenos. Ao utilizar esta ferramenta para espaços de grande dimensão podemos observar, pela observação da Ilustração 13, que em cada aquisição é obtida muito pouca informação e apenas sobre objetos que se situem próximos da Kinect.

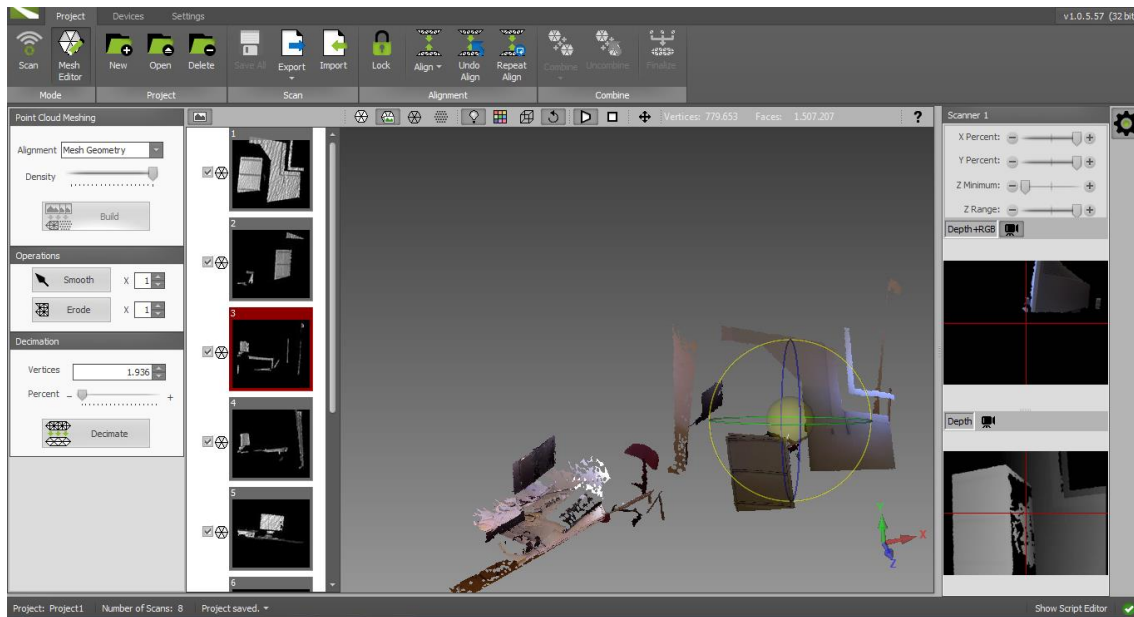


Ilustração 13 - Visualização a utilização da ferramenta KScan3D.

Após alguns testes com o *software* Skanect verificou-se que em relação ao KScan3D, tem a vantagem de possibilitar a aquisição de mais informação sobre objetos mais afastados da Kinect. É possível observar na Ilustração 14 a nuvem de pontos (imagem superior da Ilustração 14) e a respetiva triangulação e projeção da textura (imagem inferior da Ilustração 14). Este programa possui a desvantagem de bloquear após a câmara começar a ser movimentada o que, como é possível ver na Ilustração 14, apenas possibilitou obter o que era visível inicialmente pela câmara e descontinuou a aquisição ao mover a Kinect para o lado direito.

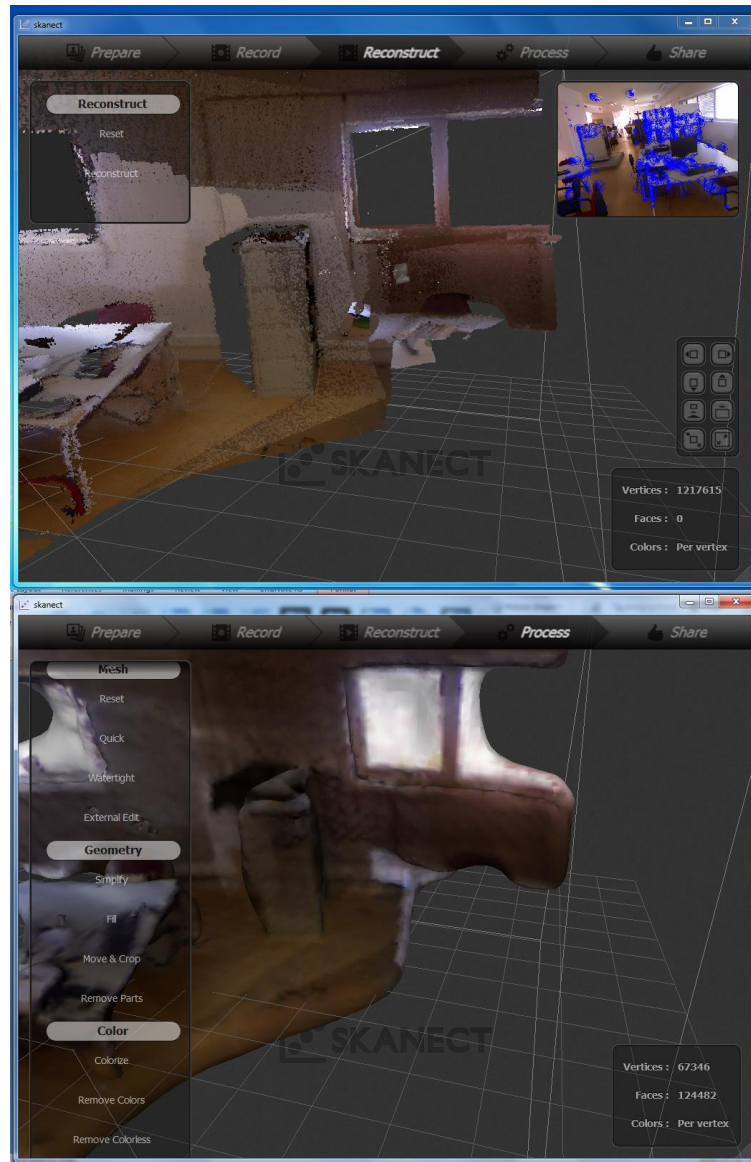


Ilustração 14 - Aquisição utilizando o software Skanect de uma nuvem com 1 217 617 pontos (imagem superior) e uma malha poligonal, obtida após reconstrução dessa nuvem, com 124 482 triângulos (imagem inferior).

Relativamente ao ReconstructMe, não foi possível correr este programa no computador testado. No entanto, visitando a *web page* da aplicação [26], foi possível obter uma imagem do seu funcionamento (Ilustração 15).

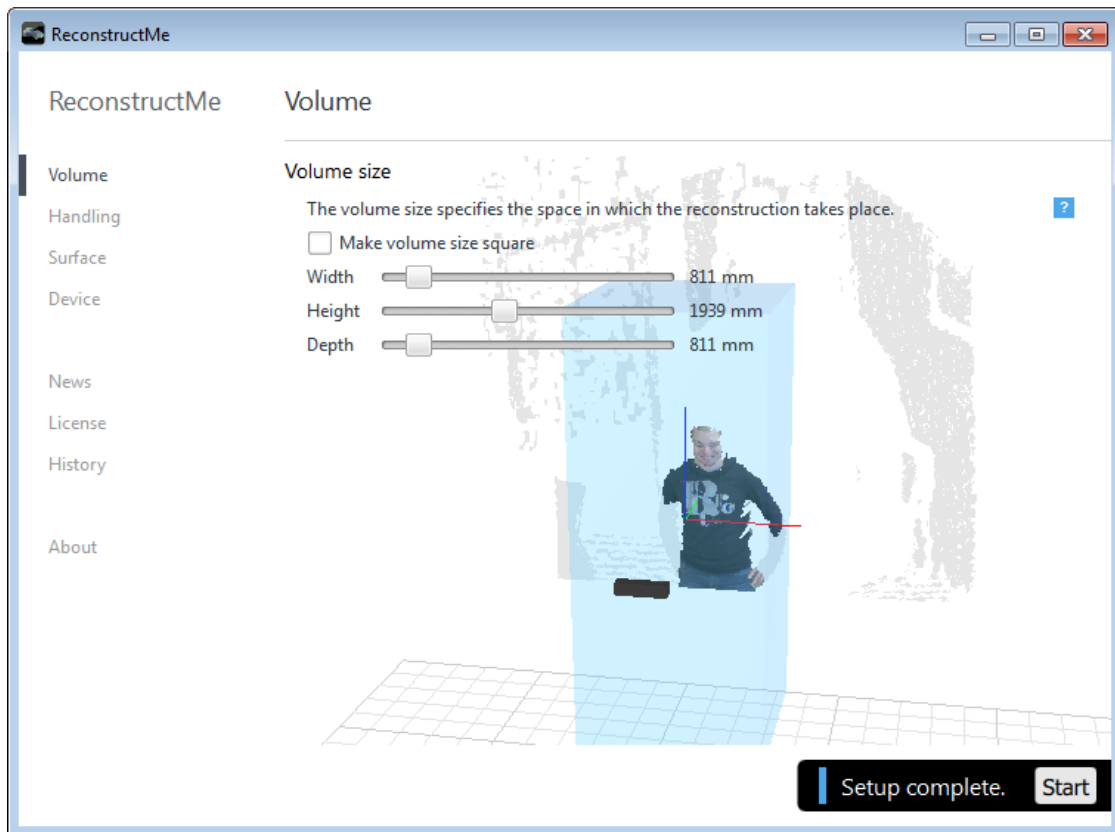


Ilustração 15 - Utilização da ferramenta ReconstructMe [26].

3.2.5 Análise e resultados das aplicações testadas

A Tabela 1 apresenta as principais ferramentas estudadas assim como as suas configurações mínimas para possibilitar a sua utilização.

Tabela 1 - Características das várias aplicações estudadas para aquisição de nuvens de pontos utilizando a Kinect.

	Kinect Fusion	Kinfu/ Kinfu Large Scale	Kscan3D	Skaneect	Reconstruct Me
Sistema Operativo	Windows	Linux	Windows	Windows	Windows
Versão	Versão gratuita	Versão gratuita	Versão limitada	Versão limitada	Versão limitada
Open-Source	Não	Sim	Não	Não	Sim
CPU	3GHz	2 GHz	2GHz	2 GHz	2 GHz
Memória	4 GB	4 GB	2 GB	4 GB	4 GB
Placa gráfica	1GB DirectX 11 CUDA 2.0	1GB DirectX 11 CUDA 2.0	256MB DirectX 9.0	1GB DirectX 9.0 CUDA 2.0	512MB DirectX 9.0
Versão beta?	Sim	Sim	Não	Não	Não
Funções	Capturar	Capturar	Capturar Alinhar	Capturar Editar	Capturar
Exportação	Malha poligonal	Malha poligonal	Nuvem de pontos	Malha poligonal	Malha poligonal
Formatos	.obj	.obj	.obj .ply	.obj .ply	.obj .ply
Textura?	Não	Sim	Não	Sim	Não

Como é possível observar na Tabela 1, estas ferramentas necessitam de máquinas com boas capacidades de processamento. O *software* mais exigente ao nível do processamento (CPU) e placa gráfica (GPU) é a Kinect Fusion. A Kinect Fusion, Kinfu e Skaneect necessitam da tecnologia CUDA que apenas está disponível nas placas gráficas da NVIDIA, sendo esta uma característica obrigatória no computador a ser utilizado. A placa gráfica necessita ainda de ser compatível com o DirectX 11 no caso da Kinfu e Kinect Fusion.

Após alguns testes do KScan3D, Skanect e ReconstructMe num computador com uma placa gráfica de 512MB, verificou-se que houve problemas de aquisição no caso do KScan3D e ReconstructMe que apenas detetam na aquisição a geometria de objetos e ignoram as paredes planas. Já no caso do Skanect esta aplicação bloqueava e deixava de adquirir ao movimentar a Kinect.

Com o acesso a um computador compatível com o DirectX 11 e com o *software* CUDA instalado (que permite aumentar a velocidade de processamento através da paralelização das operações através da GPU), testou-se a ferramenta Kinect Fusion. Esta, apesar de ser uma ferramenta ainda em fase de desenvolvimento, já fornece bons resultados (Ilustração 8), apresentando na saída uma malha poligonal em ficheiro *.obj* (como era necessário) e apresenta malhas poligonais de boa qualidade.

Devido a problemas de logística apenas se obteve acesso a um computador com sistema operativo Windows e com a Kinect Fusion já instalada. Por esta razão optou-se pelo uso desta aplicação.

Como limitações da sua utilização apenas permite aquisições com dimensões de 3x3x3 metros e não possui textura pelo que, havendo *hardware* disponível, seria de avaliar melhor a opção KinFu Large Scale.

3.3 Tratamento das nuvens de pontos obtidas

A utilização da Kinect Fusion para reconstrução em ambientes reais apresenta os seguintes desafios:

- Necessidade de alinhamento das várias aquisições;
- Geração de uma malha com elevado número de polígonos que necessita de ser decimada (processo de redução do número de polígonos) para obter uma malha simples, que permita uma auralização através de modelos geométricos (utilizando o algoritmo desenvolvido no projeto AcousticAVE [6]).

Relativamente à redução do número de polígonos, uma estratégia que se experimentou consistiu em identificar os planos principais que definem um espaço físico fechado. Para esta identificação, após o alinhamento inicial das várias malhas poligonais obtidas pela Kinect Fusion, foi feita uma segmentação para obter os planos principais. De seguida foi testada a projeção dos pontos para planos mais próximos (para efeitos de

simplificação geométrica) e por fim a utilização do algoritmo que aplica o Convex Hull [36] para obter um modelo fechado dessa nuvem de pontos.

Como alternativa a este processo de simplificação através da identificação de planos é estudado um algoritmo, designado Reconstrução de Poisson [37], que possibilita a construção de uma malha triangular da sala, logo após a fase de alinhamento inicial.

3.3.1 Alinhamento manual inicial e aplicação do algoritmo ICP

Para proceder ao alinhamento, recorreu-se ao algoritmo de alinhamento ICP disponibilizado no Meshlab [15], alinhando assim as várias malhas triangulares obtidas com a Kinect Fusion.

Para alinhar todas as nuvens de pontos foi utilizado o seguinte processo:

1. Carregamento de duas nuvens de pontos;
2. Seleção e limpeza manual dos pontos que visualmente se situassem em posições com um erro bastante elevado;
3. Alinhamento manual das duas nuvens no Meshlab através da seleção de pontos equivalentes nas duas nuvens;
4. Aplicação do algoritmo ICP disponível no Meshlab;
5. Registo das duas nuvens obtendo-se uma nuvem final;
6. Abertura da nuvem seguinte e repetir o processo a partir de 2. utilizando esta nuvem e a final (resultante em 5.) até estarem todas as nuvens adquiridas alinhadas.

Após este processo obtém-se uma nuvem de pontos completa de uma sala composta pelas várias aquisições individuais da Kinect Fusion.

3.3.2 Obtenção dos planos principais através do algoritmo RANSAC

Numa sala real existem vários objetos, nomeadamente: quadro, mesa, cadeiras, etc. Contudo, dado a necessidade de salas simples para os algoritmos de auralização, o nosso objetivo neste trabalho é conseguir um modelo da geometria global da sala. Para tal é necessário obter os planos principais que definem a sala (plano de cada uma das 4 paredes,

o plano do chão e do teto, e ainda pequenos planos que definem pequenos detalhes geométricos da sala). Assim para a aquisição destes planos recorreu-se a um algoritmo, que faz uma segmentação de planos principais, denominado de RANSAC (Random Sample Consensus) [38].

Os planos principais do modelo foram detetados recorrendo à biblioteca PCL, nomeadamente a função *pcl::SampleConsensusModelPlane* para definir, que queremos analisar especificamente planos e a *pcl::RandomSampleConsensus*, que irá ter como entrada a nuvem de pontos da nossa sala após ter sido alinhada corretamente e irá fornecer como saída o plano definido pelo maior número de pontos.

Para obter todos os principais planos, a metodologia usada foi a seguinte:

1. Utilizar o algoritmo RANSAC para obter o plano com maior suporte;
2. Eliminar os pontos pertencentes ao plano encontrado;
3. Repetir o processo desde 1. até existirem apenas 20% dos pontos iniciais da sala;

Este processo finalizava ao existirem 20% dos pontos iniciais da sala porque, através de confirmação visual (neste caso específico), não resulta em nenhum plano importante.

O resultado final apresenta os pontos pertencentes a cada plano e os respetivos coeficientes. No caso dos coeficientes, a função *pcl::RandomSampleConsensus* fornece os seus valores na forma normal hessiana, sendo a equação característica definida por (1) para cada um dos planos.

$$Ax + By + Cz + D = 0 \quad (1)$$

Neste caso os coeficientes são dados pelos valores de A, B, C e D e a normal dada pelos valores (A,B,C).

Na Ilustração 16 é possível observar uma nuvem de pontos segmentada em vários planos, sendo cada segmentação apresentada com uma cor diferente.

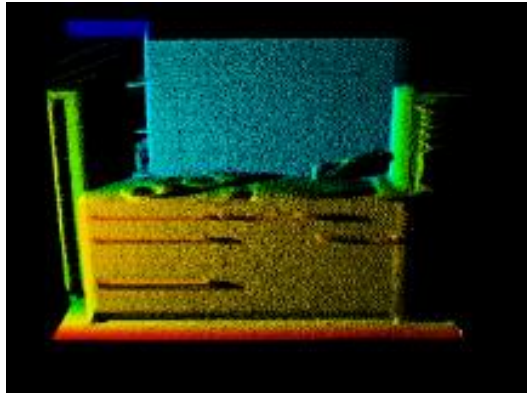


Ilustração 16 - Segmentação duma nuvem de pontos com uma cor diferente associada a cada plano [39].

3.3.3 Projeção de pontos

Possuindo agora os pontos todos pertencentes aos vários planos principais, verificou-se que existe uma carência de pontos ao longo de todo plano. Estes erros surgem devido a erros na detecção de planos, que devido aos erros de alinhamento das várias nuvens de pontos, não possuem um preenchimento total de pontos como pode ser visível na Ilustração 17 que apresenta 3 planos principais e a seta aponta a ausência de pontos nesse plano.

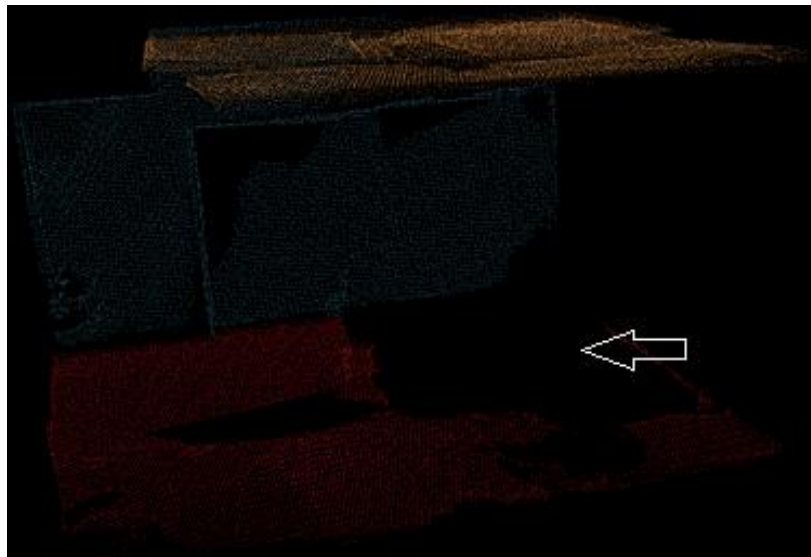


Ilustração 17 - Representação de 3 planos principais obtidos.

Para solucionar este problema da carência de pontos em certas zonas do plano foi desenvolvido um algoritmo que, através da projeção de pontos, permite que os pontos interiores de cada plano sejam projetados para o plano vizinho mais próximo de modo a definir apenas pontos que delimitem um polígono associado a cada plano principal.

Para se conhecer qual o plano mais próximo de um dado ponto recorre-se à função *vtkPlane::DistanceToPlane* em que é fornecido como entrada: a normal e um ponto pertencente ao plano.

As normais de cada um dos planos, como referido na secção 3.3.2, são dadas por (A,B,C) da equação do plano resultante do algoritmo RANSAC. Já para obter um ponto do plano, podemos recorrer ao resultado da expressão (2) em que, através da normal (A,B,C), se obtém o ponto D pertencente ao plano.

$$\frac{(A, B, C) * D}{(A^2 + B^2 + C^2)} \quad (2)$$

Detendo agora a representação dos vários planos principais, o algoritmo desenvolvido funciona do seguinte modo:

1. Análise de um dos planos;
2. É percorrido cada ponto existente nesse plano e calculada a distância aos outros planos;
3. Identificação do plano que se situe mais próximo de cada ponto;
4. Projeção desse ponto para o plano vizinho mais próximo através da função *vtkPlane::ProjectPoint* e registo do ponto projetado;
5. Após análise de todos os pontos desse plano, analisa o plano seguinte e repete desde 1. até serem percorridos todos os planos existentes;
6. Criação de uma nuvem de pontos com todos os pontos projetados.

Na Ilustração 18 pode ser vista uma imagem da sala de reuniões do IEETA, com os pontos projetados através do método anteriormente descrito em que se observa que os pontos estão situados em arestas de um sólido que delimite a sala.

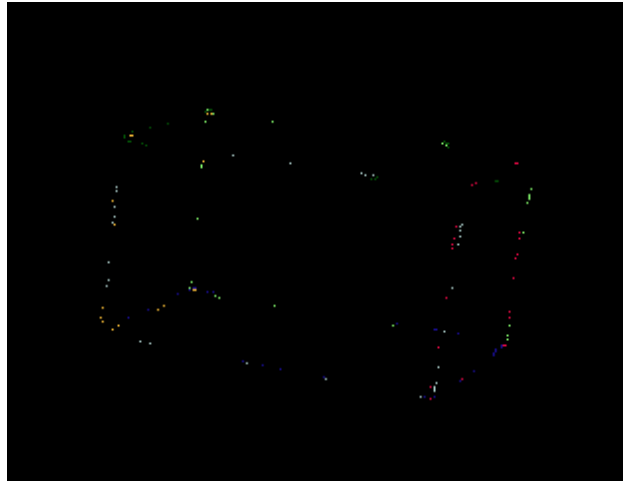


Ilustração 18 - Representação da sala de reuniões do IEETA após os pontos serem projetados.

3.3.4 Convex Hull

Como o objetivo deste processo é criar uma malha simples, depois de obter os pontos projetados, torna-se necessária a formação dos vários polígonos presentes na sala.

Os valores fundamentais para definir o nosso plano irão ser, deste modo, os pontos projetados que definem a fronteira e o limite de cada um dos planos. Para obter os polígonos que representam cada um dos planos, recorre-se ao termo Convex Hull [36] que forma um polígono envolvente a uma nuvem de pontos que seja fornecida.

O algoritmo que implementa este conceito irá receber como entrada uma nuvem de pontos (lado esquerdo da Ilustração 19) e, aplicando o algoritmo que implementa o Convex Hull, vai criar um polígono envolvente à nuvem fornecida (representado a azul no centro da Ilustração 19), resulta num polígono presente no lado direito da Ilustração 19.

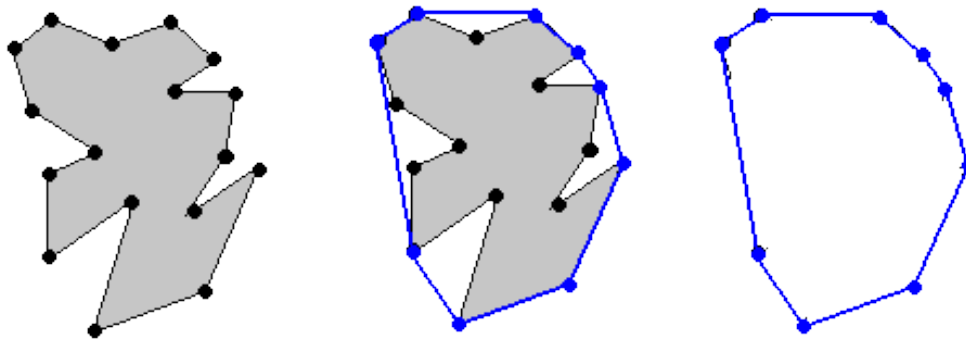


Ilustração 19 - Representação de um conjunto de pontos que define um polígono inicial (lado esquerdo) o processamento do algoritmo que aplica o Convex Hull (centro) e o resultado obtido (lado direito).

Para efeitos de simulação acústica é necessário um modelo fechado e, como tal, é necessário que os casos de janelas e/ou portas existentes numa sala sejam fechadas. Como é fácil de perceber, neste caso deixa de haver esse problema, observando-se que o algoritmo desenvolvido, por si só, irá definir um polígono que contém os pontos à volta do plano e, por conseguinte, a sua representação já é um plano fechado. No entanto, apesar de cada um dos planos ser representado por um polígono fechado, a aplicação do algoritmo RANSAC apresentava falhas no preenchimento como foi observado na Ilustração 17. Apesar da tentativa de corrigir este defeito com a projeção de pontos, é possível observar na Ilustração 20 que o polígono representativo do plano surge com um tamanho menor que o esperado. Deste modo há falhas de interseção entre alguns dos polígonos que provocam buracos no modelo.

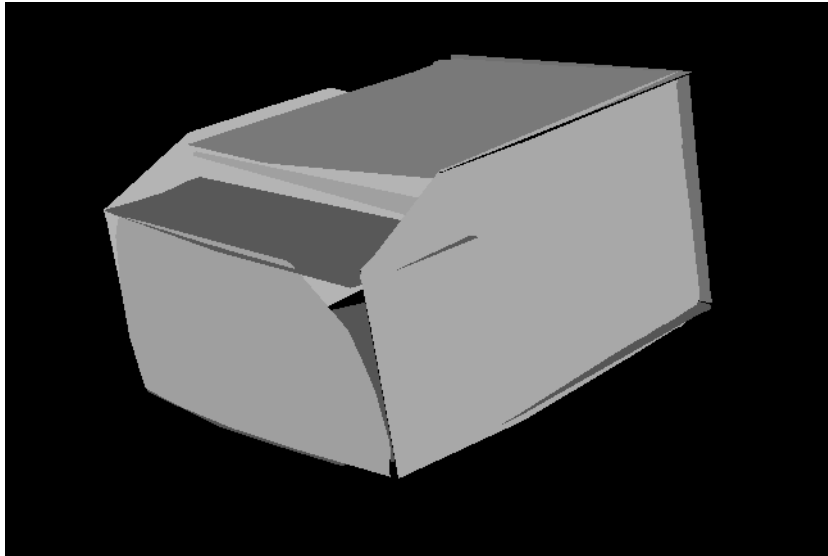


Ilustração 20 - Resultado após aplicar o algoritmo que aplica o Convex Hull a cada um dos planos encontrados.

Estes buracos chegam a atingir dimensões razoáveis como é visível na Ilustração 20, que tornam complicada a aplicação de métodos de cobrir falhas em malhas para se obter o modelo fechado pretendido. Esta complicação surge porque a maior parte dos algoritmos existentes (*vtkFillHolesFilter* do VTK, função *Close Holes* do Meshlab, assim como de outros programas de reparação de malhas) estão preparados para a existência de bastantes falhas na malha poligonal, mas pressupõe que todas elas são de pequenas dimensões.

Como os modelos para simulação acústica necessitam de ser completamente fechados e este método ainda não satisfazia essa necessidade essencial, foi estudado um processo alternativo.

3.3.5 Algoritmo de reconstrução de Poisson

Uma outra possibilidade para a reconstrução de uma malha triangular a partir de uma nuvem de pontos recorre à Reconstrução de Poisson [37]. Este algoritmo tem como valores de entrada uma nuvem de pontos orientada, ou seja, que possua a definição de cada uma das suas normais associadas à localização espacial dos pontos e irá apresentar na saída uma malha triangular sólida e completamente fechada. Um exemplo, obtido na publicação [40], é apresentado na Ilustração 21, em que é possível visualizar uma nuvem de pontos

(lado esquerdo) e o resultado da aplicação do algoritmo de Reconstrução do Poisson (lado direito).

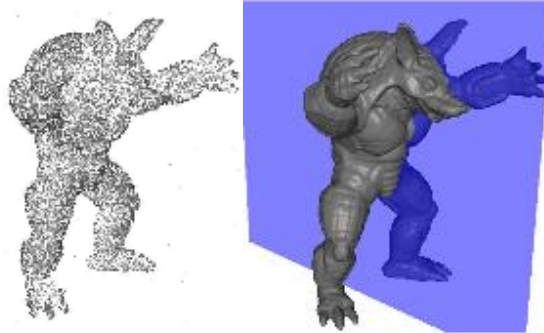


Ilustração 21 - Imagem de uma nuvem de pontos (lado esquerdo) e aplicação da Reconstrução de Poisson (lado direito) [40].

A função específica utilizada nesta dissertação é a função Poisson Reconstruction desenvolvida no programa Meshlab [37], que surge como um aperfeiçoamento do algoritmo desenvolvido por Kazhdan [40].

Como explicado em [40], este algoritmo usa o Teorema Fundamental do Cálculo, para relacionar os pontos (com a orientação/normal devidamente definida) e um modelo sólido. Mais especificamente, recorre a uma versão deste teorema designada por Teorema da Convergência ou Teorema de Gauss. Em primeiro lugar é necessário obter a função característica do sólido que possui uma forma/contorno volumétrica M , que é obtida recorrendo à nuvem de pontos adquirida através da Kinect Fusion. Esta função característica X_M é definida na equação (3).

$$X_M(x, y, z) = \begin{cases} 1 & \text{se } (x, y, z) \in M \\ 0 & \text{se } (x, y, z) \notin M \end{cases} \quad (3)$$

Um dos elementos chave deste algoritmo está na igualdade (4) entre o campo vetorial das normais (n) e o gradiente da função característica (∇X_M).

$$\nabla X_M = n \quad (4)$$

De seguida, é possível resolver a equação de Poisson apresentada em (5) para obter a função característica X_M .

$$\nabla^2 X_M = \nabla \cdot n \quad (5)$$

Como se pode observar na Ilustração 22, possuindo um conjunto de pontos com a normal associada é obtido o campo vetorial (primeira imagem da Ilustração 22). Calcula-se a função característica através do cálculo da divergência (segunda imagem da Ilustração 22) e resolução da equação de Poisson, obtendo-se a função característica visível na terceira imagem da Ilustração 22. Por fim é extraída a superfície, resultando na quarta imagem da Ilustração 22.

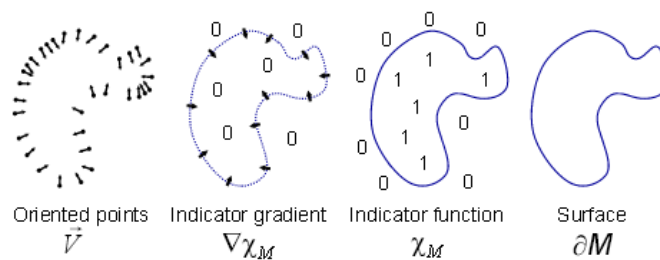


Ilustração 22 - Visualização dos passos da aplicação da reconstrução através do método de Poisson [37].

A função (6) possibilita a obtenção da função característica.

$$X_M(x, y, z) = \sum_{l,m,n} \hat{X}_M(l, m, n) e^{i(lx+my+nz)} \quad (6)$$

Esta equação (6) analisa todos os pontos da nuvem ao mesmo tempo, sem recorrer à divisão por secções, pelo que vai estar bastante suscetível aos pontos com erros de aquisição e, pela mesma razão, a complexidade deste algoritmo está diretamente relacionada com a quantidade de pontos existentes na nuvem de pontos.

De seguida, dado possuir valores de 1 e 0 que diferenciam o interior do exterior do nosso sólido, recorre-se a uma função de Iso-surfacing que irá extrair a fronteira do nosso

objeto (quarta imagem da Ilustração 22), através da utilização do algoritmo Marching Cubes [41].

Como já referido, o algoritmo da Reconstrução de Poisson é bastante influenciado pelos erros, sendo possível, no caso de existir uma pequena quantidade de pontos próximos um dos outros e afastados do modelo, que se forme durante o processo uma malha isolada.

Ao invés de uma limpeza posterior pode-se também ter especial atenção e fazer um pré-tratamento manual das nuvens de pontos obtidas na Kinect depois de alinhadas, fazendo uma limpeza manual dos pontos que se encontrem bastante afastados e não façam parte da sala de reuniões.

Após testes com a sala de reuniões obtida com a Kinect, optou-se por uma pré-limpeza da nuvem de pontos (lado esquerdo da Ilustração 23 em que é possível observar a identificação dos problemas de alinhamento, falhas onde não existem pontos e pontos afastados do modelo).

Como resultado da aplicação do algoritmo Reconstrução de Poisson da aplicação Meshlab obteve-se um modelo triangulado e completamente fechado, razoável da sala (lado direito da Ilustração 23 em que é possível observar que os buracos foram fechados com sucesso).

No final, optou-se por este algoritmo como resolução para o problema de triangulação da nuvem de pontos que era fornecida pela Kinect Fusion, dado apresentar vantagens, em relação ao processamento anteriormente desenvolvido, de criar um modelo fechado e preservar a geometria da sala, incluindo os pequenos detalhes como por exemplo o degrau existente no teto e a zona de entrada.

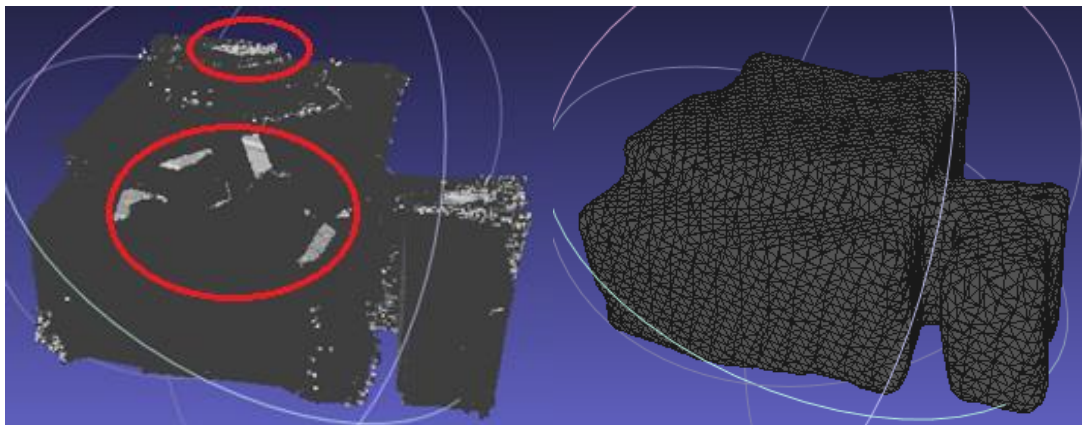


Ilustração 23 - Reconstrução de Poisson em que foi dado como entrada a nuvem de pontos representativa da sala (lado esquerdo) e o resultado desse algoritmo (lado direito) aplicado e visualizado no Meshlab.

O resultado da aplicação da Reconstrução de Poisson pode ser então decimado de modo a reduzir o número de polígonos do resultado obtido na Ilustração 23 (que possuía 15 786 triângulos). Após o processo de redução do número de polígonos através da função Quadratic Edge Collapse Decimation disponível em Meshlab, que utiliza o método Quadric Based Mesh Decimation [42], contraindo repetidamente pares de vértices para simplificar o modelo, mantendo-o o mais aproximado possível da sua forma original.

O resultado final é um modelo com a geometria semelhante ao original (comparação feita na Ilustração 53 na secção 6.1.2), mas com apenas 150 triângulos (como é possível observar na Ilustração 24), dado que para uma redução a partir desse número de triângulos apresentava resultados visuais errados.

Assim, através deste método, obteve-se um modelo fechado que representa a geometria do espaço real e que pode ser utilizado pela biblioteca desenvolvida no projeto AcousticAVE [6], como originalmente idealizado.

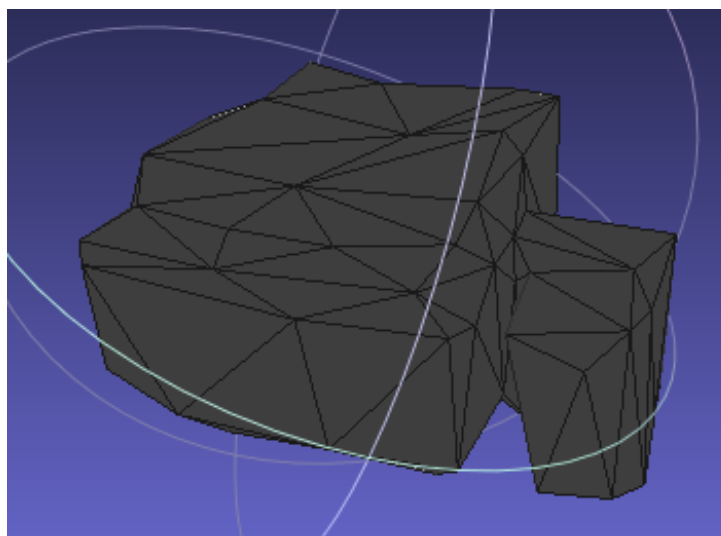


Ilustração 24 - Resultado da decimação do número de polígonos para 150, a partir do modelo obtido após a Reconstrução de Poisson.

4 Aplicação para configuração do modelo para auralização

Neste capítulo são apresentadas as ferramentas que permitem associar materiais a cada um dos triângulos da malha triangular conseguida na secção 3. Esta associação é essencial para alimentar as duas aplicações de auralização: quer seja para fornecer o material existente em cada face (para a ferramenta que usa o modelo geométrico), quer facultando o material presente em cada nó fronteira (para a ferramenta que recorre ao modelo físico para auralizar através de uma grelha de nós tridimensional).

4.1 Ficheiro para armazenar características dos materiais

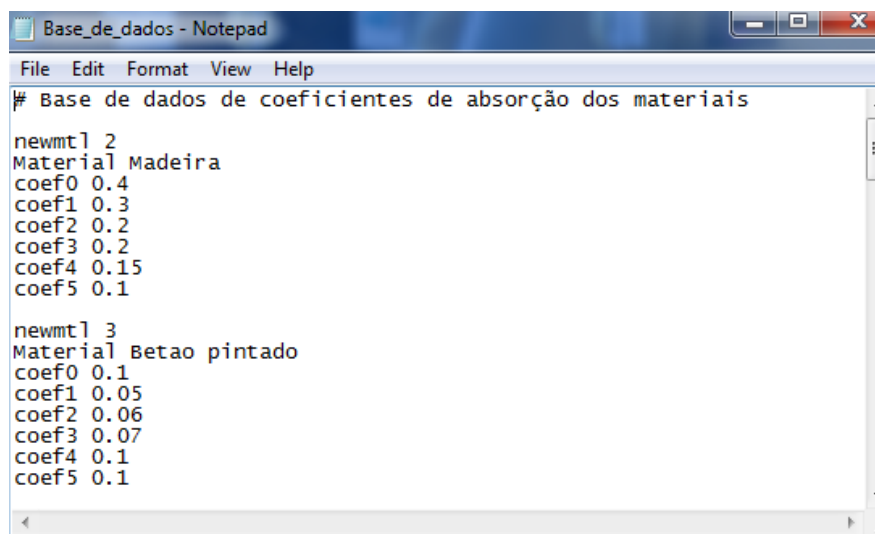
Cada material apresenta um valor acústico característico, denominado coeficiente de absorção, que se define como um valor entre 0 e 1 indicador da relação entre o som que é absorvido pela superfície do material e o som que é refletido. Ao fazer a seleção de cada material é necessário aceder a uma base de dados para obter esses coeficientes que são usados no cálculo do valor RT_{60} (secção 5.3). No entanto, os coeficientes dos materiais apresentam valores dependentes da frequência e, por esta razão, são guardados os valores dos coeficientes em bandas de oitava (de 125 Hz a 4KHz).

Esses coeficientes são calculados para diversos materiais em [43] e alguns desses valores são apresentados na Tabela 2.

Tabela 2 - Alguns exemplos dos coeficientes de materiais a diversas frequências retirados de [43].

Materiais	Coeficientes de absorção sonora à frequência (Hz)					
	125	250	500	1000	2000	3000
Betão pintado	0.1	0.05	0.06	0.07	0.1	0.1
Tijolo	0.03	0.03	0.03	0.04	0.05	0.07
Vidro frequentemente usado em janelas	0.3	0.2	0.2	0.1	0.07	0.04
Chão de madeira	0.4	0.3	0.2	0.2	0.15	0.1

Para a seleção dos materiais foi produzida uma base de dados num ficheiro .txt, que associa um número a cada material. Cada número está associado a um dos coeficientes pertencentes ao material respetivo, para as várias frequências: coef0 (125KHz), coef1 (250KHz), coef2 (500KHz), coef3 (1000KHz), coef4 (2000 KHz) e coef5 (4000KHz). Na Ilustração 25, apresenta-se como exemplo o material Madeira, que tem associado o número 2.



```
# Base de dados de coeficientes de absorção dos materiais
newmtl 2
Material Madeira
coef0 0.4
coef1 0.3
coef2 0.2
coef3 0.2
coef4 0.15
coef5 0.1

newmtl 3
Material Betao pintado
coef0 0.1
coef1 0.05
coef2 0.06
coef3 0.07
coef4 0.1
coef5 0.1
```

Ilustração 25 - Formatação utilizada para armazenar os coeficientes associados a cada material.

4.2 Seleção manual dos materiais

Para a configuração de materiais foi utilizada a função de interação *VtkInteractorRubberBandPick*, que permite ao utilizador selecionar um retângulo (Ilustração 26).

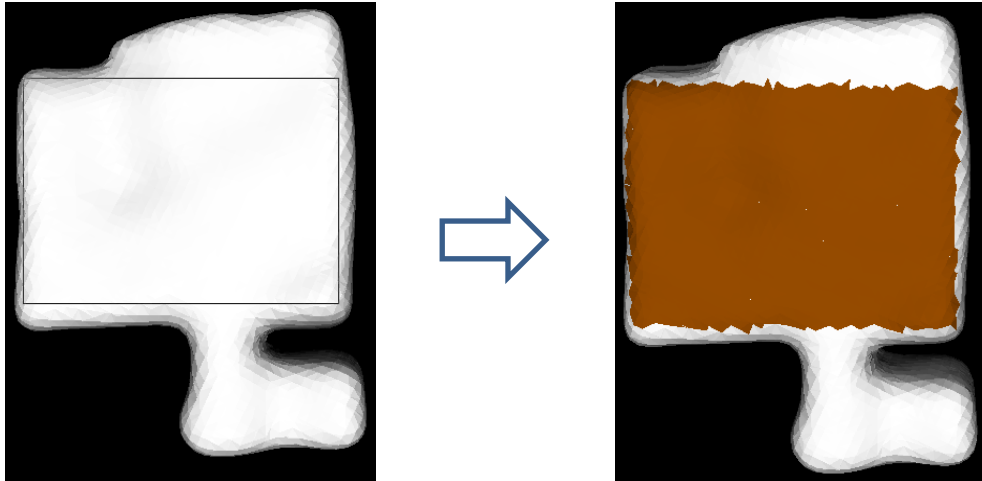


Ilustração 26 - Seleção de um retângulo da Anta, associando um material a esse conjunto de células.

A aplicação identifica todos os polígonos que se situam parcialmente ou totalmente na zona seleccionada, permitindo ao utilizador seleccionar o material a associar a esses polígonos.

Como exemplo, a Ilustração 27 representa o resultado após a seleção de todos os polígonos que definem o chão da sala e associa a uma único material.

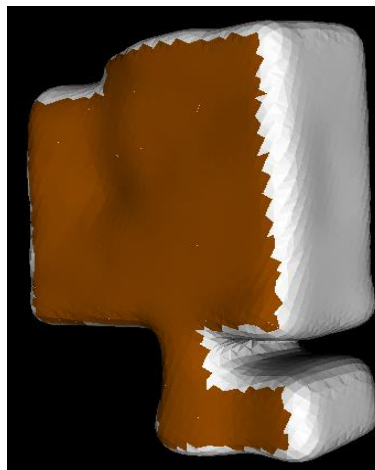


Ilustração 27 – Sala com todos os polígonos que definem o chão associados a um material.

4.3 Seleção semiautomática de materiais recorrendo à textura

Para automatizar o processo de associação de materiais aos modelos foi ainda investigada, no âmbito desta dissertação, a possibilidade de usar a textura quando a mesma estiver disponível. A ideia consiste em segmentar o modelo por cor, podendo depois associar materiais às cores da textura.

4.3.1 Segmentação de uma imagem

Durante as últimas décadas o processo de segmentação de uma imagem foi utilizado em várias aplicações: reconhecimento de objetos, controlo de tráfego automático ou localização de estradas em imagens de satélite. Também na medicina estas técnicas são bastante populares, sendo usadas em imagens de ressonância magnética ou tomografia e, em bioinformática, são utilizadas para localização de proteínas.

Assim, dada a vasta variedade de utilização, surgiram diferentes técnicas para segmentar imagens: técnicas baseadas no uso de *threshold* [44], histograma [45], deteção de contornos (*edge detection*) [46], enchimento de regiões (*region growing*) [47] e métodos de agrupamento (*clustering*) [48].

As técnicas que recorrem ao *threshold* são mais utilizadas em imagens que contêm os valores em tons de cinzento. É uma das técnicas mais populares, dado que a sua implementação é bastante simples. Em 2D a imagem é constituída pelos vários tons de cinzento e assume-se estar dividida em duas partes: o fundo e os objetos de interesse. O valor de *threshold* T é definido analisando todos os valores de intensidade em cada um dos *pixéis* da imagem. Em cada *pixel* (x,y) , em que o seu valor seja superior a T , o ponto irá pertencer ao conjunto de objetos, caso contrário é considerado fundo.

As técnicas baseadas em histogramas são bastantes fáceis de utilizar quando comparadas com as demais. Geralmente são utilizadas imagens em tons de cinzento, sendo inicialmente feito um histograma do número de *pixéis* existentes para cada valor de intensidade e, mediante as curvaturas máximas e mínimas dos níveis de intensidade, são agrupadas em conjuntos diferentes. Para o caso da utilização das cores, em que cada pixel é constituído por 3 valores RGB, é possível construir um histograma que pode ser

combinado com a técnica de *threshold* para segmentação através do agrupamento em valores de RGB semelhantes.

A detecção de contornos é a técnica mais usada para detetar discontinuidades em imagens reproduzidas em valores de cinzento. Para a sua implementação recorre-se à primeira e segunda derivada para detecção de fronteiras entre *pixéis* de valores de intensidade bastante afastados.

Nas técnicas baseadas em crescimento de regiões, é inicialmente fornecido um ponto (*seed*), que irá servir como base a uma determinada região, de seguida essa zona irá expandir-se para os *pixéis* vizinhos, caso estes possuam propriedades semelhantes ao ponto de *seed* previamente escolhido.

Por fim, a segmentação baseada em *clusterings* permite dividir a informação em subgrupos que possuam características semelhantes. Uma das desvantagens deste método está na necessidade da introdução prévia do número de regiões em que queremos subdividir a imagem.

Nesta dissertação optou-se pelo método de *region growing*. A escolha desta técnica deveu-se à possibilidade de utilização direta das cores fornecidas em RGB, sem ser necessário a passagem para uma escala de cinzentos e ao fato de não existir conhecimento prévio sobre o número de regiões que queremos segmentar.

4.3.2 Métodos de representação de cores

Para apresentar as cores numa imagem existem várias codificações possíveis (RGB, HSV, HSL, CIELab, entre outros). Cada uma delas favorece determinadas características em detrimento de outras (por exemplo: cor, saturação, intensidade e/ou luminância).

Neste trabalho vai ser dado especial destaque às codificações RGB e HSV, dado serem representações que permitem uma análise mais simples da “proximidade/afastamento” entre as cores presentes em dois pontos diferentes.

O RGB é um sistema de cores constituído por vermelho (*Red*), Verde (*Green*) e Azul (*Blue*) representado de uma maneira semelhante ao sistema de coordenadas x,y,z (Ilustração 28), o que nos permite trabalhar sobre este sistema para detetar a diferença entre cores, como a distância entre dois pontos 3D.

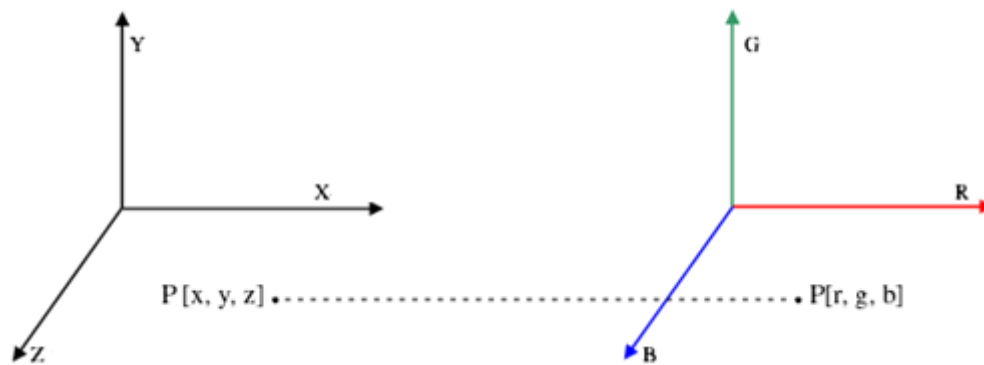


Ilustração 28 - Representação do sistema de coordenadas RGB.

Devido a estas características pode-se definir diretamente como condição de crescimento, as diferenças entre cada um dos valores R,G e B separadamente.

Um outro modelo possível para representar as cores de uma imagem é designado HSV. Esta sigla representa as componentes matiz (*Hue* que representa o valor da cor), saturação (*Saturation* que está relacionado com os tons de cinza) e valor (*Value* que define o brilho que a cor apresenta). Na Ilustração 29, é possível observar o diagrama de cores representativo do HSV que permite identificar visualmente a influência da variação de cada um dos seus valores.

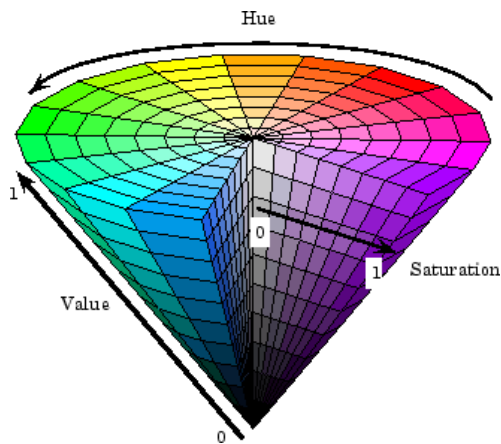


Ilustração 29 - Representação do sistema de coordenadas HSV.

Neste caso específico, o mais importante é definir uma boa condição de crescimento para o valor de H, que tem vantagem em relação ao RGB de ter especial relevância apenas num valor de medida das cores. Como principal desvantagem, esta

representação tem mais dificuldades na identificação da cor branca ou cores bastantes claras. Esta limitação é facilmente identificada em paredes em tons de bege ou que apresentem um branco mais escuro devido à diferente luminosidade. Este problema pode ser facilmente identificado na Ilustração 29, em que é possível observar, no centro do cone, a presença de cores bastante próximas do branco. Esta situação implica que ao existir uma proximidade entre duas cores claras a nível visual, o mesmo pode significar uma diferença bastante significativa no valor de H (*Hue*).

4.3.3 Implementação de um método de seleção semiautomático e resultados

Dado que não estava disponível a textura na Kinect Fusion, para esta dissertação foram feitos testes em aquisições individuais da Kinect. Para essas aquisições recorreu-se à biblioteca PCL que fornece um método (disponível em [49]), que permite guardar a imagem que a Kinect está a captar numa nuvem de pontos com textura, que pode ser observada na Ilustração 30.

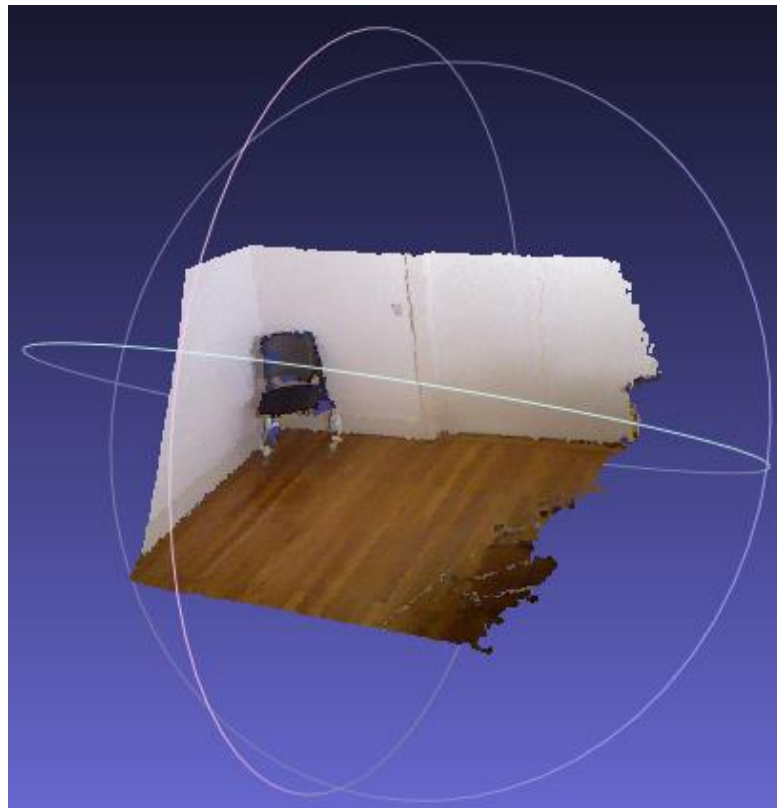


Ilustração 30 - Nuvem de pontos singular adquirida pela Kinect com textura associada.

Na Ilustração 30 cada ponto possui um valor de RGB associado. Para simular a malha poligonal com textura associada que se devia obter na secção 3.3 é processada a Reconstrução de Poisson no Meshlab. Para isso nas opções deste programa seleciona-se Filters->Sampling->Vertex Attribute Transfer, que irá permitir fazer a transferência da cor da nuvem de pontos original, diretamente para a malha resultante da Reconstrução de Poisson, como observável na transformação da imagem esquerda para a direita, na Ilustração 31.

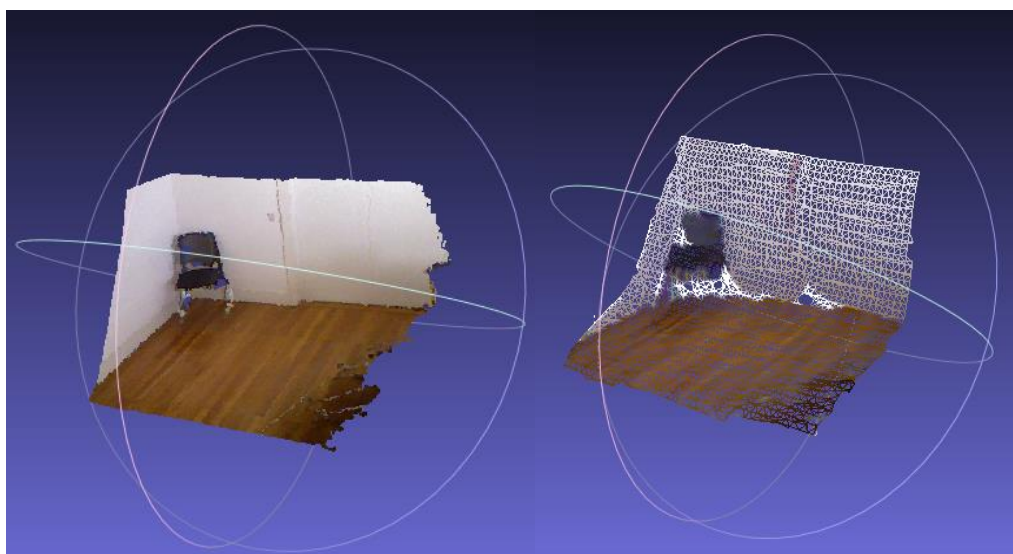


Ilustração 31 - Nuvem de pontos com textura associada adquirida pela Kinect (lado esquerdo) e aplicação no Meshlab da Reconstrução de Poisson e associação de textura à malha (lado direito).

Para esta transferência o algoritmo do Meshlab assume que as malhas são semelhantes e alinhadas (o que é verdadeiro neste caso), de seguida para cada vértice da malha triangular final irá identificar o ponto mais próximo (e não o vértice) na nuvem de pontos original. A transferência de textura da nuvem para a malha é obtida copiando o valor RGB do ponto mais próximo encontrado para o vértice correspondente.

De seguida, foi desenvolvida uma aplicação que utiliza o método de enchimento de regiões elegido anteriormente (fim da secção 4.3.1).

A aplicação começa por fazer uma média de cada um dos valores R, G e B presentes nos 3 pontos que compõem cada triângulo da malha, ficando esta representada apenas por um valor para cada uma das componentes RGB. Após este processo, é selecionado um triângulo pelo utilizador ao qual irá associar um material. O algoritmo analisa então os triângulos vizinhos, expandindo o material aos polígonos vizinhos de uma

forma iterativa até que não haja mais nenhuma célula vizinha que reúna as condições de crescimento.

Nesta fase, ainda exploratória, foi empiricamente escolhido um valor de *threshold* para a condição de crescimento de 80 em cada componente RGB.

Após alguns testes, verificou-se que trabalhando com estes valores obtiveram-se já alguns resultados na separação entre o chão e a parede, visíveis do lado direito da Ilustração 32, em que se verificou que foi feita a seleção de materiais diferentes para a parede (verde), cadeira (azul), chão (amarelo) e o vermelho a cor definida inicialmente em todas as células.

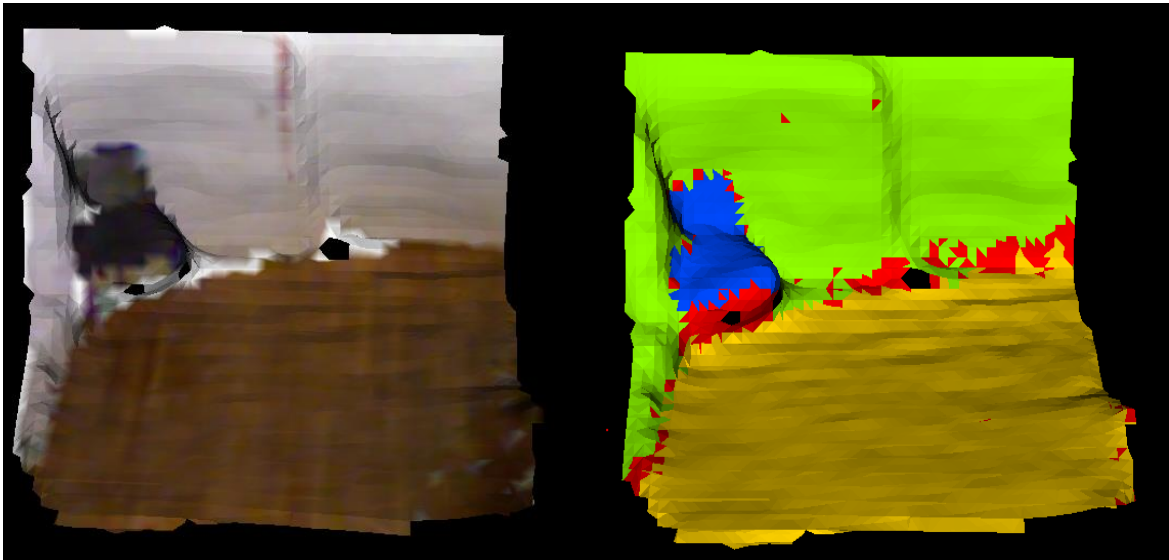


Ilustração 32 - Abertura no VTK do arquivo .ply exportado do Meshlab com a malha triangular com textura associada (lado esquerdo) e segmentação de materiais (lado direito).

É possível observar que vários triângulos não foram englobados em nenhuma região após as expansões das 3 regiões (a vermelho na Ilustração 32). Esta indefinição deve-se em parte ao processamento da Reconstrução de Poisson em que, não existindo uma malha completa resultante da aquisição do *scanner*, ao fazer a transferência das cores (dado inicialmente não existir nenhum valor naquela região), irá transferir a cor totalmente branca, que tem um valor de RGB [255,255,255]. Esta situação pode ser observada na Ilustração 33 em que não houve aquisição de pontos nem textura em baixo da cadeira e mesmo em algumas zonas do próprio objeto (lado esquerdo da Ilustração 33) e, ao ser feita

a Reconstrução de Poisson e respetiva transferência de cor, essas zonas foram colocadas como textura branca (lado direito da Ilustração 33).

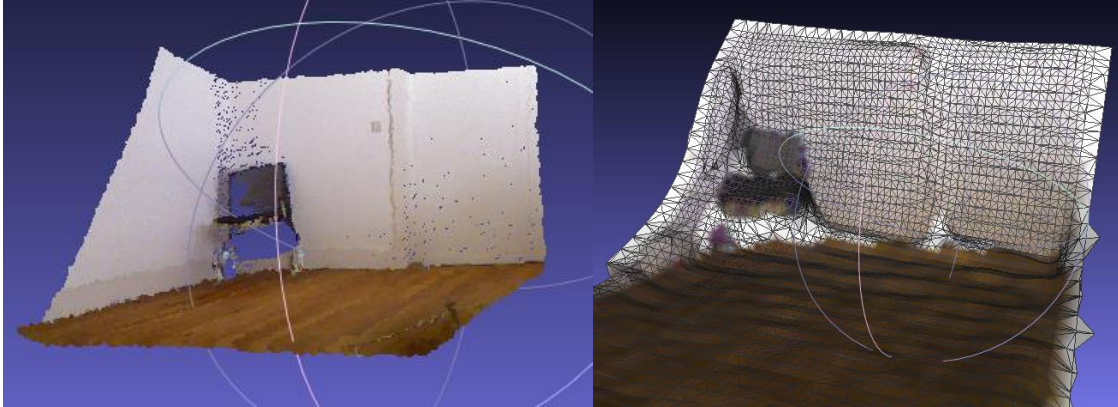


Ilustração 33 - Falhas na transferência de cor entre a nuvem de pontos (lado esquerdo) e a malha obtida pela Reconstrução de Poisson (lado direito).

Foram também efetuados testes com os valores de HSV em que pela lógica, foi dado um valor de *threshold* mais baixo na variação de H (0.1) e um valor mais elevado em S e V (0.4) para possibilitar uma maior variância a nível de saturação e mais controlo a nível das diferentes cores. Pelos resultados obtidos na Ilustração 34 verificamos que, para os valores do chão em tons de castanho, a segmentação apresentou resultados razoáveis. No entanto, ao seleccionar a parede, dado possuir o problema já discutido da existência de diferentes tons de branco e bege, apenas permitiu um crescimento bastante reduzido dessa secção.



Ilustração 34 - Segmentação da malha poligonal colorida (lado esquerdo) através do HSV e seleção de 3 materiais diferentes para parede, chão e cadeira (lado direito).

Comparando os dois algoritmos (RGB e HSV) na detecção do chão em madeira (ver Ilustração 35), nota-se que mais uma vez a segmentação utilizando o RGB permite uma segmentação melhor.

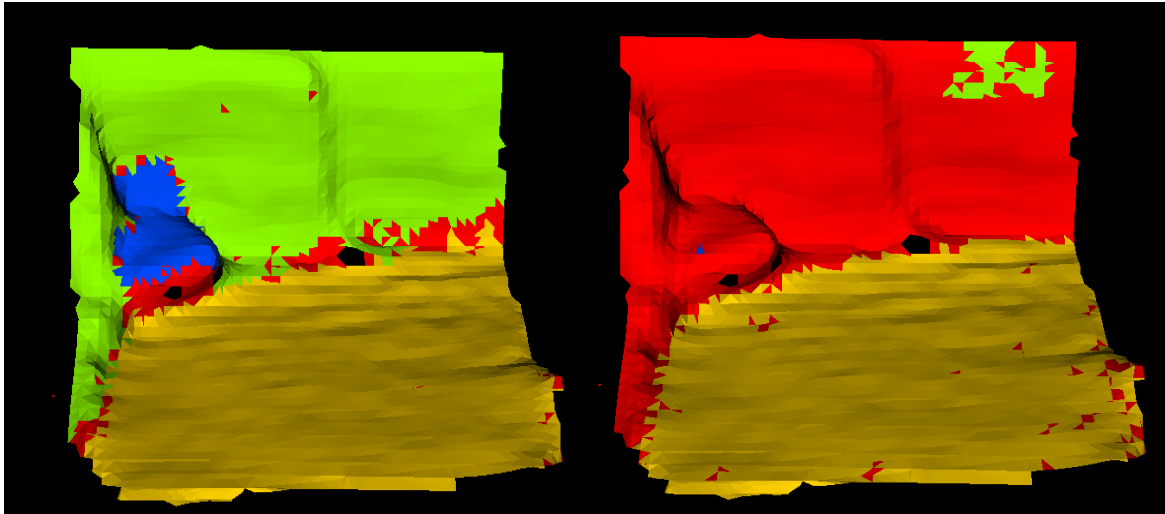


Ilustração 35 - Comparação entre o algoritmo de segmentação RGB (lado esquerdo) e o algoritmo que utiliza HSV (lado direito).

Apesar dos desenvolvimentos nesta aplicação, assim como os testes efetuados, terem sido baseados em aquisições individuais da Kinect, foi possível obter resultados encorajadores utilizando os valores RGB para identificação de materiais recorrendo à textura. No entanto, a evolução desta aplicação, está dependente do desenvolvimento do processo de aquisição de uma malha poligonal de uma sala com textura associada a todos os polígonos presentes na malha.

Assim é possível observar na Ilustração 36 que foi possível obter toda a informação necessária para fornecer um modelo geométrico que possa ser utilizado na aplicação de auralização.

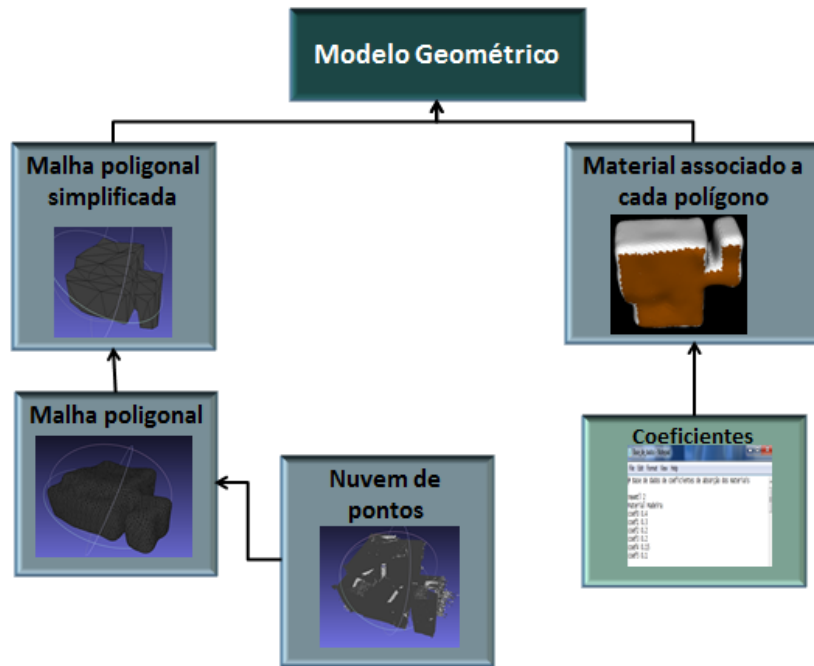


Ilustração 36 - Resultados de toda a informação necessário para a produção de um modelo geométrico.

5 Ferramenta para criação e configuração de modelos físicos

Para a implementação da auralização através do uso de modelos físicos, como já foi referido, recorre-se à utilização de uma malha volumétrica (grelha de nós 3D) que identifique cada nó como sendo exterior, interior ou fronteira relativamente à malha poligonal. Neste capítulo é explicado como é feita a geração dessa malha volumétrica a partir de uma malha poligonal fechada. Para tal é usado um processo designado de voxelização. A partir do modelo voxelizado é apresentada a fórmula que permite calcular o RT_{60} . Este valor, importante para a auralização, indica o tempo que um sinal sonoro demora a desvanecer numa sala e depende do volume da sala e da área ocupada por cada material.

5.1 Voxelização

A palavra voxel provém da abreviatura de *pixel* volumétrico (*volume pixel*) e pode ser definido como o equivalente a um *pixel* numa imagem tridimensional [50].

Cada voxel é um pequeno elemento volumétrico que permite armazenar informação, que pode ser uma única característica (cor em formato RGB ou RGBA), ou várias (normal, transparência, entre outras).

Deste modo, através da associação de informação podemos, além da auralização, definir as partes visíveis ou invisíveis para uma correta visualização de objetos ou a nível científico (Raio-X, Tomografia computada ou Ressonância Magnética).

Nesta dissertação, os vóxeis gerados na malha tridimensional vão ser usados especificamente para armazenar informação sobre o tipo de superfície (ar no caso de nós exteriores ao modelo, inativo para nós interiores ou identificação de um material específico para nós fronteira) sendo essa informação necessária para correr uma simulação acústica.

A voxelização [51] é o procedimento que converte a representação de uma superfície ou malha poligonal, numa representação discreta composta por vóxeis. Neste

processo, procede-se à análise dos vóxeis para determinar: se estão no interior ou exterior do objeto ou, se por outro lado, se encontram na fronteira.

Na Ilustração 37, que foi obtida em [52], pode ser observado um exemplo do resultado esperado de uma voxelização de um bule de chá, em que são visíveis apenas os vóxeis de fronteira.



Ilustração 37 - Voxelização de um bule de chá [52].

Apesar de o conceito de voxel já existir há vários anos, os primeiros processos estudados de voxelização de linhas e círculos 3D, assim como uma diversidade de malhas poligonais começaram a surgir a partir de 1886, através de Kaufman, A. [50, 53, 54]. Um desses algoritmos foi desenvolvido recorrendo a um algoritmo de preenchimento Scan-line [55]. Com o passar dos anos outros algoritmos mais aperfeiçoados e com maior rapidez de processamento foram surgindo [56, 57].

No caso específico para a ferramenta VTK surgiu um artigo [58] que estuda a função *vtkVoxelModeller* para voxelização sem discutir a associação de valores a cada nó. Esta associação é um dos objetivos principais da aplicação deste processo nesta dissertação.

Existem ainda ferramentas online como o Acropora [59], que permite trabalhar sobre as malhas poligonais que desejarmos e inclui ainda a possibilidade de voxelização das mesmas. No entanto, esta ferramenta apenas permite observar os resultados visuais da voxelização e não permite guardar o resultado como uma grelha de nós com as características dos mesmos (interior, fronteira ou exterior). Por esta razão e por uma

questão de usar uma ferramenta própria para as necessidades desta dissertação foi desenvolvido um algoritmo de voxelização específico para este trabalho.

Para desenvolver esse processo começou por ser estudada a função *VtkVoxelModeller* que recebe o modelo poligonal e o número de nós que queremos obter em cada. Como resultado fornece uma grelha tridimensional regular associando aos nós o valor 1 (se o cubo centrado nesse nó possuir algum polígono) ou caso contrário o valor 0.

A dimensão da grelha é calculada a partir das dimensões da malha poligonal fornecida a dividir pela distância entre cada nó.

No caso da modelação física, a distância entre dois nós consecutivos é fornecida pelo comprimento de onda, que está inversamente relacionado com a frequência, através da fórmula (7).

$$\text{Comprimento de onda} = \sqrt{3} * \left| \frac{\text{velocidade do som}}{\text{frequência}} \right| \quad (7)$$

O valor da frequência é fornecido como parâmetro de entrada e para a velocidade do som é usado o valor 340 m/s.

Na Tabela 3 são apresentados o número de nós para uma sala retangular com dimensões 5x4x3m e frequências (taxas de amostragem) mais comuns em áudio.

Tabela 3 - Exemplo de valores, em função da frequência, relacionados com a voxelização de uma sala rectangular.

Frequência (Hz)	Distância entre dois nós consecutivos (cm)	Número total de nós para uma sala cúbica 5x4x3 metros
44 100	1.34	24 982 048
22 050	2.67	3 141 600
11 025	5.34	394 800

Um dos problemas desta abordagem é o elevado tempo de processamento (podendo chegar a horas) para malhas de maior dimensão. De notar que uma malha de teste que usamos, o modelo da Anta Pintada de Antelas tem 204 280 triângulos.

Na Ilustração 38 é possível observar os resultados do algoritmo desenvolvido aplicado ao bule de chá, assim como a voxelização da Anta de Antelas pintada para duas frequências, uma de 11 025 Hz (lado esquerdo da Ilustração 39) e outra de 22 050 Hz (lado

direito da Ilustração 39). É fácil observar que, quanto maior a frequência, menor irá ser o tamanho de cada cubo e maior o número total de cubos, possibilitando a visualização voxelizada dos objetos com um maior nível de detalhe.

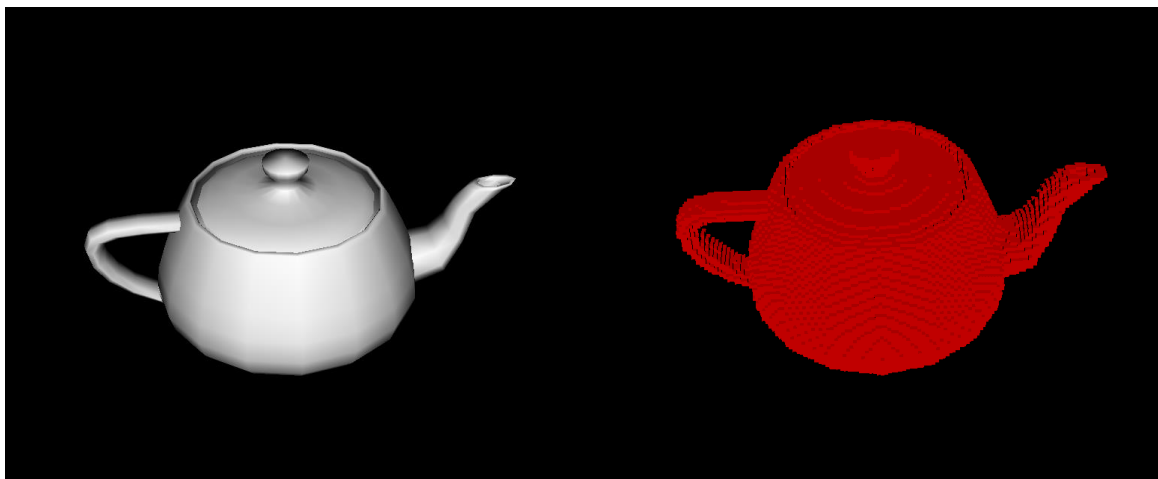


Ilustração 38 - Malha poligonal de um bule de chá (lado esquerdo) e respectiva voxelização a 22 050Hz (lado direito).

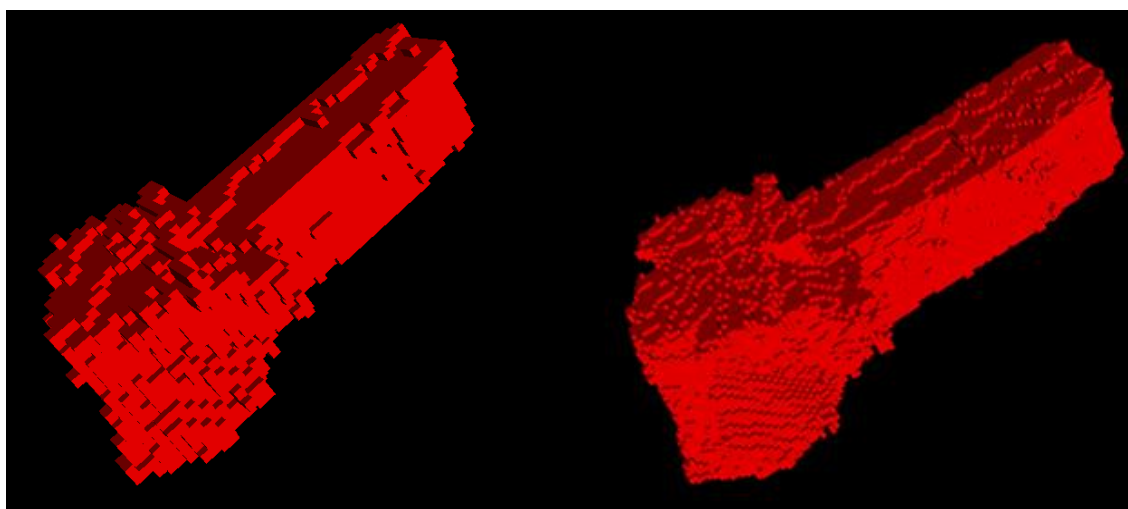


Ilustração 39 - Exemplo de voxelização da Anta a 11 025 Hz (lado esquerdo) e 22 050Hz (lado direito).

Devido à utilização do *vtkVoxelModeller* ser muito onerosa computacionalmente e a saída desta função produzir um aumento exagerado da *bounding box* (que resulta na identificação de bastantes nós exteriores dispensáveis), foi estudada a possibilidade de otimizar o algoritmo através da substituição desta função.

A nova abordagem consistia em criar no VTK uma grelha tridimensional com o número de nós em cada eixo desejados. De seguida, é percorrido cada nó dessa grelha e é

calculada a distância a que se situa o polígono mais próximo do modelo fornecido (função *vtkCellLocator->FindClosestPointWithinRadius*, fornecendo como raio de pesquisa metade da diagonal do cubo que representa cada voxel). Em caso da distância calculada ser menor que um voxel identificamos esse nó como uma fronteira e identifica-se simultaneamente o material associado ao polígono mais próximo. Os restantes casos são colocados a 0.

Deste modo, obteve-se um processo mais eficiente, que não recorre à função *vtkVoxelModeller* e tem a vantagem de identificar simultaneamente os materiais associados a cada nó.

Na Ilustração 40 surge um exemplo de voxelização usando o novo processo desenvolvido. Neste caso o bule foi configurado com dois materiais diferentes (lado esquerdo da Ilustração 40) e para visualização apenas são visíveis os vóxeis correspondentes a um material (lado direito da Ilustração 40).

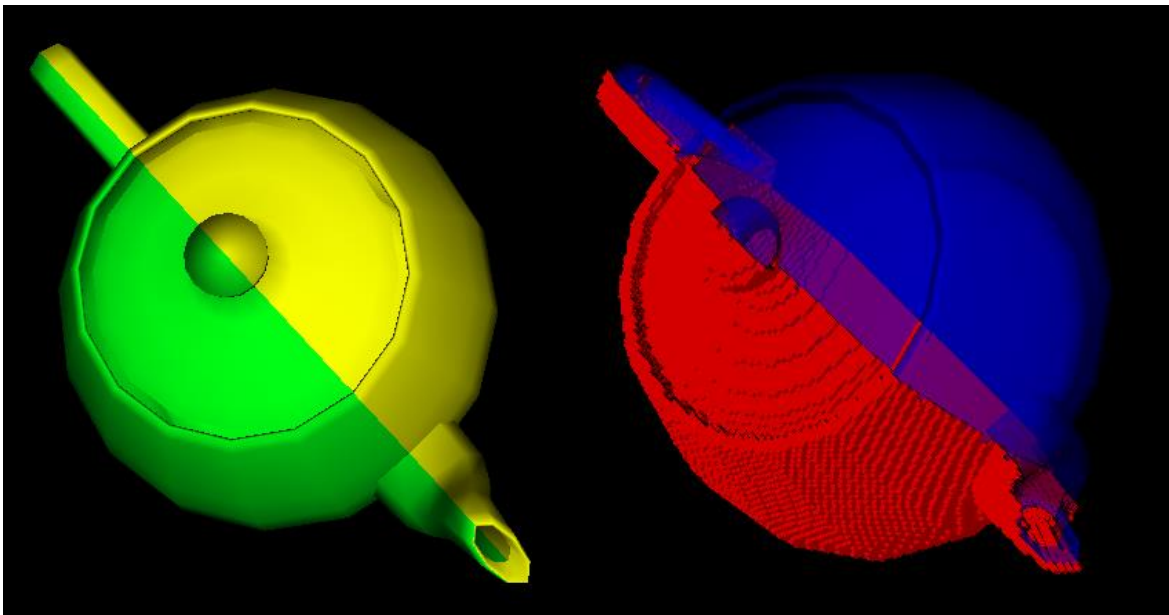


Ilustração 40 - Seleção de dois materiais diferentes (lado esquerdo) e respetivo resultado da voxelização (lado direito).

5.2 Determinação do espaço aberto na malha a partir da voxelização inicial

A voxelização inicial só distingue os nós que pertencem à fronteira da malha poligonal fornecida. Para efeitos de auralização, é essencial existir uma diferenciação que identifique quais os nós exteriores (nós inativos) e interiores (nós de ar) do modelo. Para determinar os nós ativos da malha foram analisados algoritmos de preenchimentos.

Inicialmente, foram feitos alguns testes de preenchimento com o algoritmo scan-line que percorria cada linha do modelo e, dependendo do número de interseções com o modelo (par ou ímpar), definia a posição de cada nós como pertencendo ao exterior ou interior do modelo. Este algoritmo é tipicamente usado em imagens 2D e foi adaptado para correr nos eixos x-y e percorrer ao longo de z (funcionando como *slices*). Após alguns testes com o modelo os resultados foram poucos encorajadores com identificação incorreta dos nós e foi estudado outro algoritmo de preenchimento.

O algoritmo Flood-fill funciona através do crescimento de uma região, isto significa que fornecendo um nó inicial semente (*seed*), o algoritmo propaga-se abrangendo os nós vizinhos que possuam as mesmas características.

Para a utilização do algoritmo Flood-fill tem de se garantir que o primeiro nó que vamos analisar se encontra no exterior da malha poligonal. Por esta razão, foi calculado o tamanho de cada voxel e aumentando a *bounding box* de voxelização em 2 vezes essa distância, ou seja, em 2 vóxeis. Este procedimento foi feito de modo a garantir que o nó semente era sempre exterior em relação a qualquer modelo. Na Ilustração 41 é possível observar esta operação no modelo da sala de reuniões do IEETA, em que é usado como semente o voxel (0,0,0) distanciado de 2 vóxeis de distância do modelo.

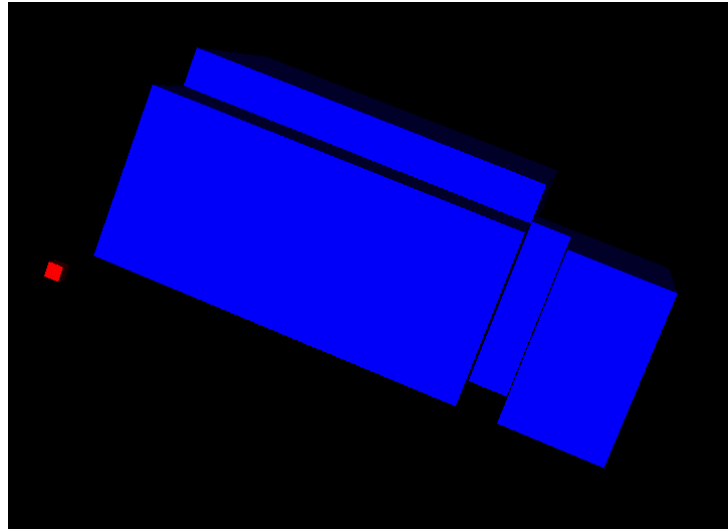


Ilustração 41 - Imagem onde é visível o modelo da sala de reuniões do IEETA (azul) e o cubo na origem da voxelização (vermelho).

Um segmento do algoritmo, aplicado a partir da semente, é apresentado na Ilustração 42.

```
Colocar valor no primeiro ponto como inativo 'I' e inserir na fila;
while(tamanho da fila > 0)
{
    retirar um ponto e analisar os seus pontos vizinhos;

    //Detetar limite de dimensões do cubo e se o ponto é exterior à malha poligonal
    If(coordenadas no ponto à esquerda > 0 && valor no vizinho da esquerda == 0)
    {
        Coloca o valor no vizinho da esquerda a inativo 'I' e coloca esse vizinho na fila
        para posteriormente serem analisados os seus vizinhos
    }
}
```

Ilustração 42 - Segmento do algoritmo usado para preenchimento do volume exterior.

Este algoritmo (Ilustração 42) é replicado para todos os vizinhos do nó em análise: esquerda ($x-1, y, z$), direita ($x+1, y, z$), superior ($x, y+1, z$), inferior ($x, y-1, z$), à frente ($x, y, z+1$) e, por fim, o vizinho que está atrás ($x, y, z-1$). Após esta análise teremos os materiais associados a cada nó da fronteira.

No final, os valores possíveis na malha 3D obtida são: material, inativos com valor 'I' e nós interiores de ar (que não foram alterados pelo algoritmo) com o valor 0.

5.3 Cálculo do RT_{60}

A auralização depende de vários parâmetros acústicos importantes que caracterizam uma sala com uma dada geometria. Um parâmetro importante é o tempo de Reverberação (RT_{60}), que se define como o tempo necessário para um som decair em 60dB numa sala, após a fonte ter parado de emitir qualquer som (ver Ilustração 43).

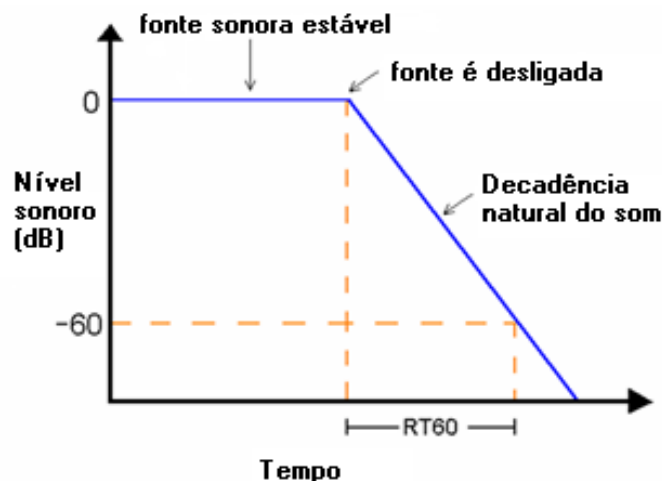


Ilustração 43 - Propagação do som numa sala e representação do valor do RT_{60} .

5.3.1 Fórmula de Millington-Sette

A primeira fórmula a surgir para o cálculo do RT_{60} foi estudada por W.C. Sabine [60] para cálculos que assume uma propagação uniforme do som que não se verifica para salas pequenas. Com a crescente necessidade de modelar o som em espaços físicos diversificados surgiram outras fórmulas: a de Eyring [61] e por fim uma modificação da fórmula de Sabine designada de Millington-Sette [62].

Como no nosso modelo existem diferentes materiais, que estão associados a superfícies com áreas diferentes e, dado que toda a informação de superfícies e áreas está disponível, é relativamente fácil usar fórmulas mais complexas que podem ser computacionalmente processadas de modo eficaz, sendo assim foi utilizada a fórmula (8) designada de Millington-Sette.

$$RT60 = \frac{0.161 * V}{-\sum Si * \ln(1 - ai(f))} \quad (8)$$

Sendo o valor V o volume do modelo físico em m^3 , Si a área do i -ésimo material e $ai(f)$ o valor do coeficiente associado a esse material.

5.3.2 Cálculo da Área

O cálculo da área da superfície vai depender do tipo de malha em causa: triangulares ou baseadas em outros polígonos. No caso de malhas triangulares, são usadas funções do VTK para determinar a área (classe *vtkTriangle*). No caso de a malha possuir qualquer polígono quadrilátero, recorre-se à fórmula de Brahmagupta (9) para o cálculo da sua área.

$$K = \sqrt{(s - a)(s - b)(s - c)(s - d)} \quad (9)$$

Sendo a,b,c,d os comprimentos dos lados do quadrilátero e s o semi-perímetro que é dado pela equação (10).

$$s = \frac{a + b + c + d}{2} \quad (10)$$

O algoritmo para o cálculo da área vai percorrer cada polígono, calcular a sua área, verificar o material que lhe está associado e a área será adicionada à área do material correspondente somando a área de todas as superfícies de um dado material. O resultado final é a área associada a cada material que é utilizado na fórmula de Millington-Sette anterior (secção 5.3.1).

5.3.3 Cálculo do Volume

Para o cálculo do volume do modelo físico foram utilizados os volumes dos vóxeis que surgem após a voxelização, bastando somar o volume de todos os vóxeis que representam o interior (nós ar) e a fronteira do modelo (nós fronteira). Este método permite

facilmente obter uma razoável aproximação do volume do modelo independentemente da sua complexidade.

Para testar todo o processo de voxelização e cálculo de RT_{60} , após a atribuição de um material (betão pintado) a todos os polígonos presentes no modelo da Anta (lado esquerdo da Ilustração 44), é feito o processamento da voxelização do modelo (lado direito da Ilustração 44), na qual foi usada uma frequência de 44 100Hz (valor necessário para alimentar a aplicação de auralização que recorre a modelos físicos) e obter os valores relacionados com o número de nós, número de cubos interiores, volume de cada cubo, área de cada material, volume total e os valores de RT_{60} em função da frequência.

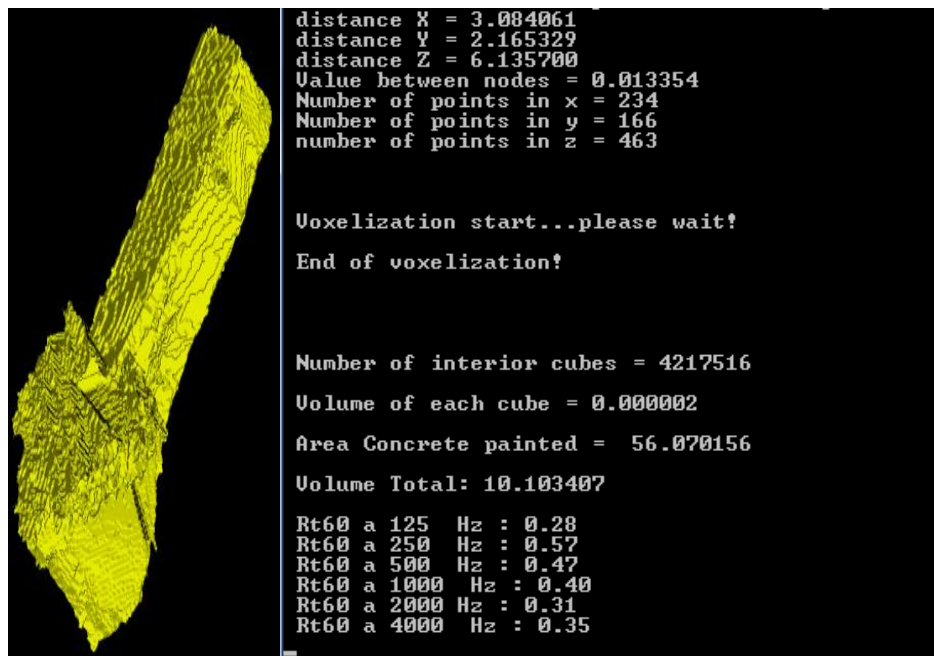


Ilustração 44 - Processo de voxelização com o cálculo do valor de RT_{60} .

5.4 Exportação do modelo voxelizado

Um passo importante do trabalho era a possibilidade de exportar a malha final num formato conhecido e que pudesse ser utilizado por um programa de auralização. A opção tomada foi a de criar um ficheiro de texto (.txt), que contivesse toda a informação relevante: frequência a que tinha sido amostrado o espaço físico, número de nós existente em cada um dos eixos (x, y e z) e uma lista de todos os nós (x,y,z), identificando os mesmos como inativos ' ' (espaço), fronteira (em que é fornecido o valor do material que

lhe corresponde), cabeça do ouvinte 'C', fonte 'X' e por fim dois nós que definem os ouvidos, sendo o direito identificado pela letra 'R' e o esquerdo por 'L'.

Na Ilustração 45 é possível observar a representação da cabeça (incluindo os ouvidos) no centro de uma sala retangular (5x4x3 metros) em que foi utilizado o processo de voxelização para facilitar a respetiva visualização.

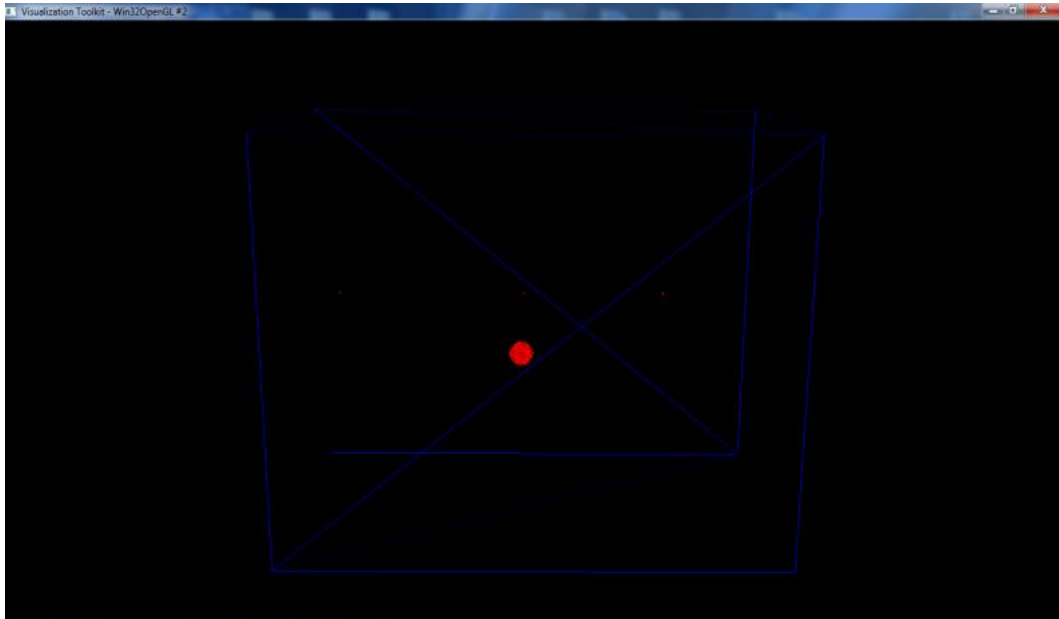


Ilustração 45 - Cabeça colocada no centro de uma sala rectangular 5x4x3 m e 3 nós representando a colocação de fontes.

Deste modo foi possível obter toda a informação necessária para fornecer um modelo físico à aplicação de auralização que está em desenvolvimento no âmbito do projeto AcousticAVE.

6 Resultados obtidos na sala de reuniões do IEETA

Neste capítulo, é aplicado o processo descrito anteriormente para modelar e criar os dois modelos necessários (geométrico e físico) para auralização de uma sala de reuniões do IEETA.

6.1 Conversão para nuvem de pontos

6.1.1 Aquisição

A aquisição da nuvem de pontos foi feita através da Kinect Fusion, apesar das limitações já referidas (ver secção 3.2.1).

De modo a ser possível trabalhar nestas condições, foi necessário adquirir várias nuvens de pontos (limitando a sua aquisição ao atingir o limite dos 3 metros) em detrimento de um só varrimento. Foram obtidas 7 nuvens de pontos diferentes de modo a manter-se alguma geometria comum entre duas nuvens consecutivas para uma maior facilidade de alinhamento. Após o alinhamento manual de todas as nuvens com recurso ao Meshlab (secção 3.3.1), obteve-se uma nuvem de pontos da sala de reuniões, visível na Ilustração 46.

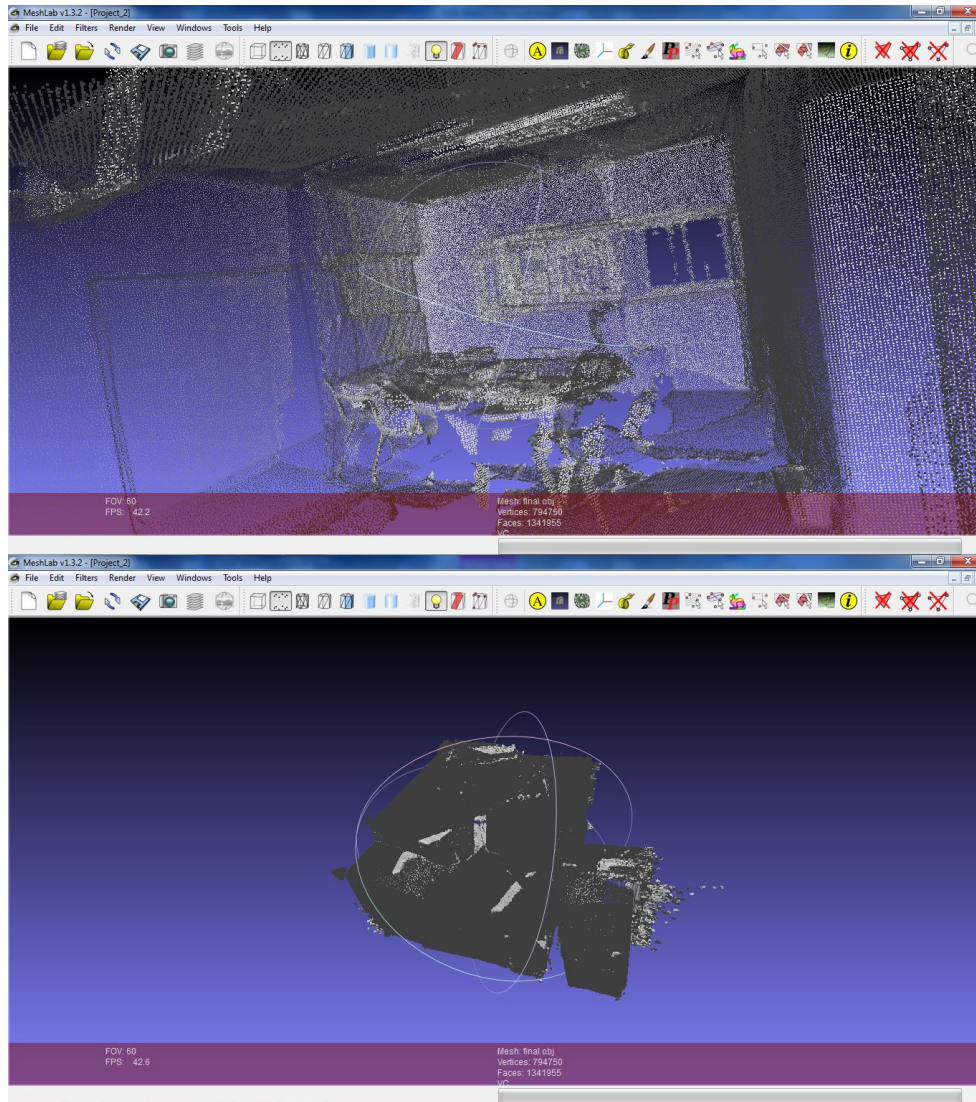


Ilustração 46 - Resultado do alinhamento das várias nuvens de pontos no interior da sala de reuniões do IEETA.

6.1.2 Tratamento da nuvem de pontos para obter a malha poligonal do modelo

Após a aquisição e alinhamento das nuvens de pontos procedeu-se ao tratamento através da utilização do algoritmo RANSAC (secção 3.3.2), para obtenção dos planos principais da sala. Na Ilustração 47, podemos observar os planos principais resultantes, estando representado cada um com uma cor diferente para uma perceção mais facilitada da imagem. Devido aos erros associados ao processo de alinhamento manual, planos de dimensões mais reduzidas podem facilmente não ser detetados.

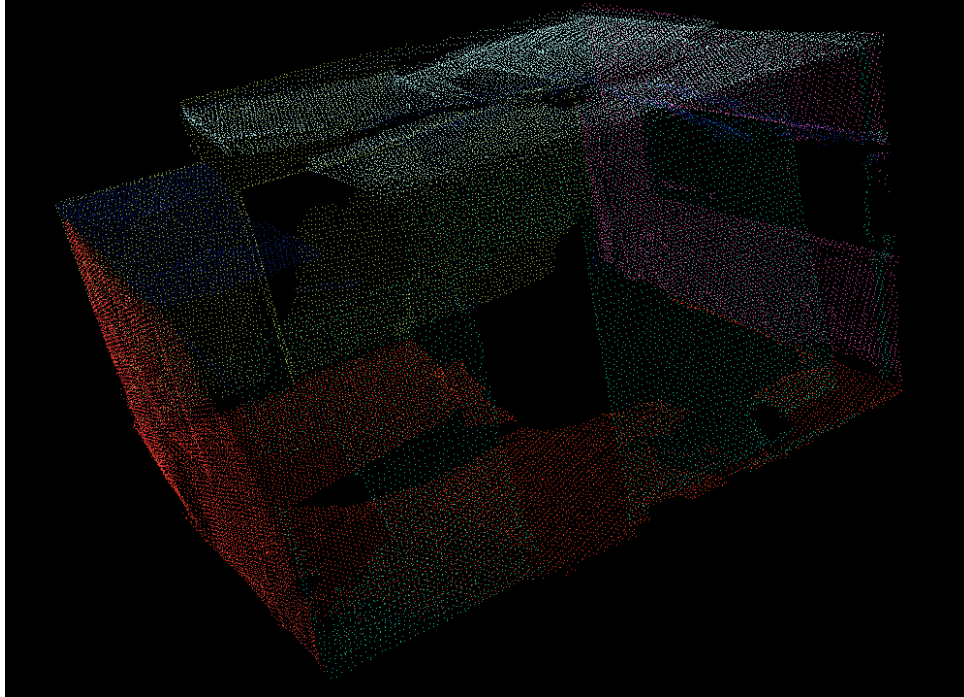


Ilustração 47 - Nuvem de pontos da sala de reuniões em que cada plano obtido é apresentado por uma cor diferente.

Na secção esquerda da Ilustração 48, podemos observar um dos planos com mais pormenor, sendo este plano a separação entre a sala e a porta de saída.



Ilustração 48 - Pontos que pertencem ao mesmo plano após aplicação do algoritmo RANSAC (lado esquerdo) processa o algoritmo que aplica o Convex Hull a esses pontos (lado direito).

De seguida, é aplicado o algoritmo que permite aplicar o Convex Hull a esse plano (lado direito da Ilustração 48), obtendo-se um polígono que envolve os pontos que formam o plano (como explicado na secção 3.3.4). O resultado depois de aplicar o processo a toda a sala é apresentado na Ilustração 49.

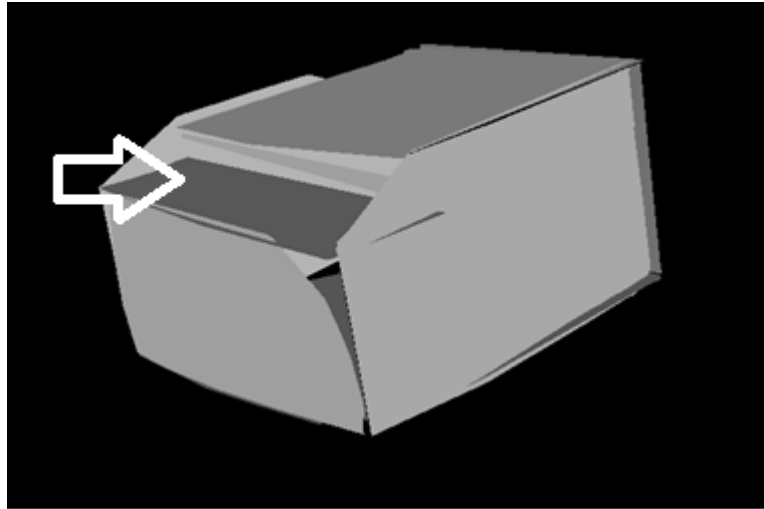


Ilustração 49 - Resultado após implementar o algoritmo que aplica o Convex Hull a cada um dos planos encontrados.

Este resultado apresenta grandes falhas e, apesar de se terem feito tentativas para o aumento das dimensões de cada um dos planos, para se intersectarem entre eles ou a técnica da projeção de pontos, tornou-se impossível obter um modelo fechado da sala.

Foi ainda feita uma tentativa de utilizar a técnica de projeção de pontos (secção 3.3.3), e implementação do algoritmo que aplica o Convex Hull a toda a nuvem de pontos em vez de aplicar a planos individuais, obtendo-se os resultados visíveis na Ilustração 50.

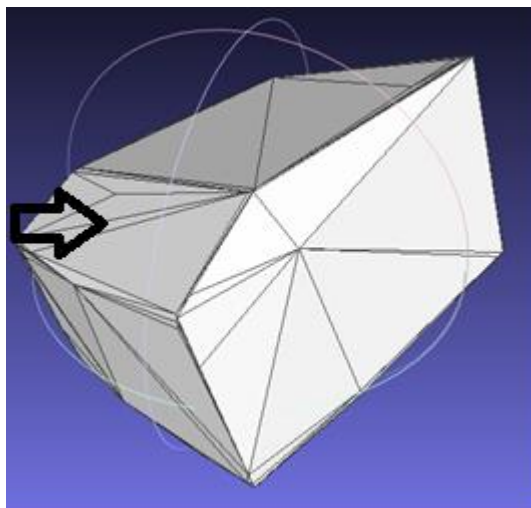


Ilustração 50 - Resultado da implementação do algoritmo que aplica o Convex Hull a toda a nuvem de pontos após a projeção dos mesmos.

No entanto, este método provoca a perda de muitos detalhes geométricos da sala, como é possível ver na alteração do degrau apontado pela seta da Ilustração 49 para a Ilustração 50 em que se perde o degrau do interior da sala. Do mesmo modo é possível ver esta alteração na imagem do interior da sala apresentada na Ilustração 51.

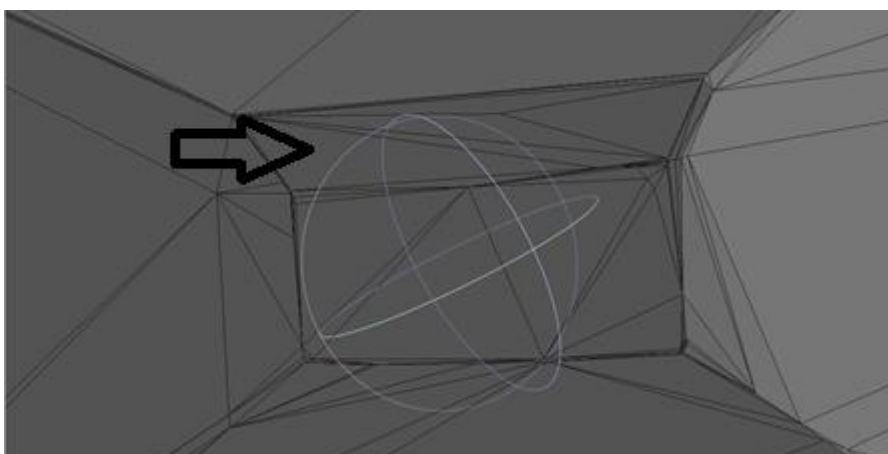


Ilustração 51 - Imagem interior da sala após a implementação do algoritmo que aplica o Convex Hull a toda a nuvem de pontos após a projeção dos mesmos.

Dadas estas dificuldades, optou-se pelo uso da Reconstrução de Poisson (secção 3.3.5), que resulta nas malhas apresentadas pela Ilustração 52. Com esta técnica foi possível obter uma malha poligonal da sala de reuniões com a geometria correta e sem a perda da zona de saída da sala.

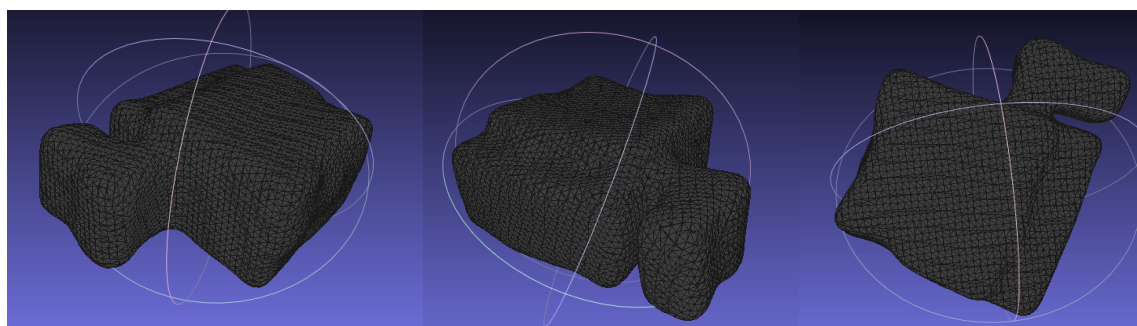


Ilustração 52 - Várias perspetivas do resultado da aplicação do algoritmo da Reconstrução de Poisson.

Quando comparado com a sala original (obtido através de uma modelação no *software* SketchUp (representado a amarelo na Ilustração 53), pode-se observar que a

malha final com reconstrução de Poisson apresenta resultados ao nível das dimensões e geometria, bastante próximos do modelo original.

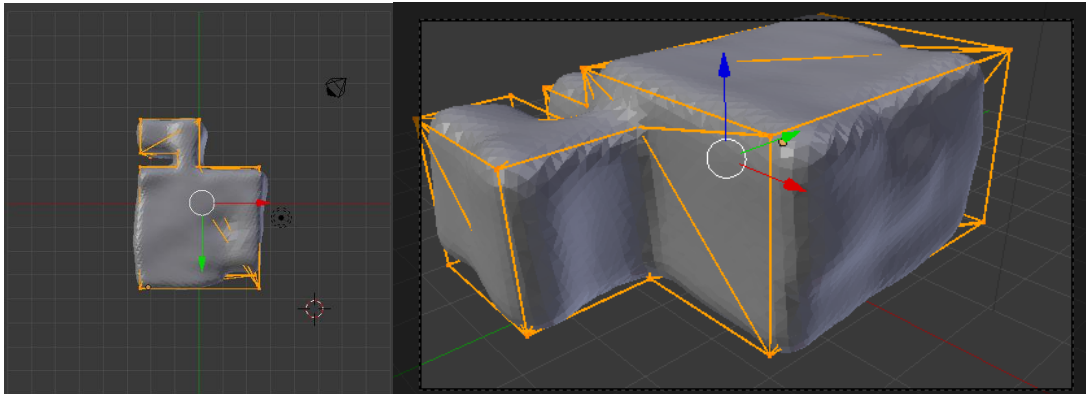


Ilustração 53 - Comparação entre a malha poligonal obtida (cinzento) e a sala original modelada recorrendo ao SketchUp (amarelo).

6.2 Seleção de materiais

Para a seleção de materiais foi efetuado um caso simples em que foram associados dois materiais diferentes, um para o chão e outro para as paredes. Deste modo, escolheu-se para o chão o material madeira (representado com a cor amarela no lado esquerdo da Ilustração 54) e placa de gesso para o resto da sala (representado com a cor verde na Ilustração 54).

É possível visualizar no lado direito da Ilustração 54 informação sobre a voxelização que inclui: a área de cada material e o volume total da sala permitindo o cálculo do valor de RT_{60} (secção 5.3) para cada uma das frequências.

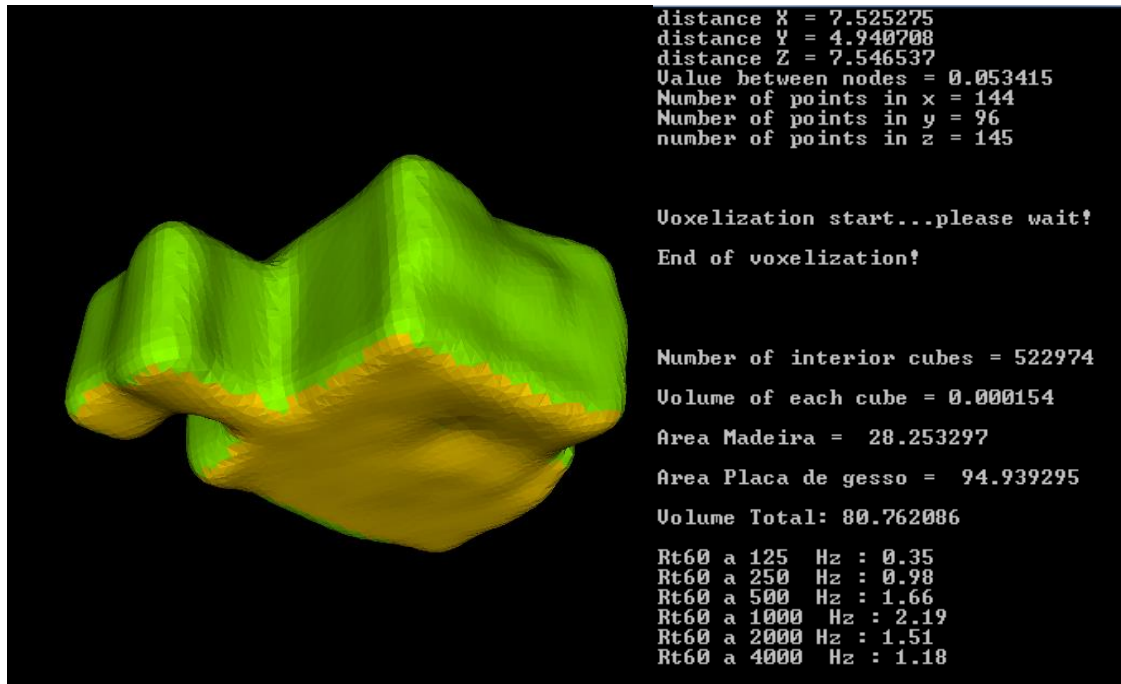


Ilustração 54 - Seleção do chão como madeira e o resto da sala como gesso e obtenção do RT_{60} durante a voxelização.

6.3 Voxelização e preenchimento

Após associação de materiais foi efetuada a voxelização e exportação dos resultados. Para efeitos visuais de voxelização podemos observar a Ilustração 55, onde são representados cubos centrados nos vários nós fronteira da malha poligonal, representativa da sala de reuniões do IEETA.

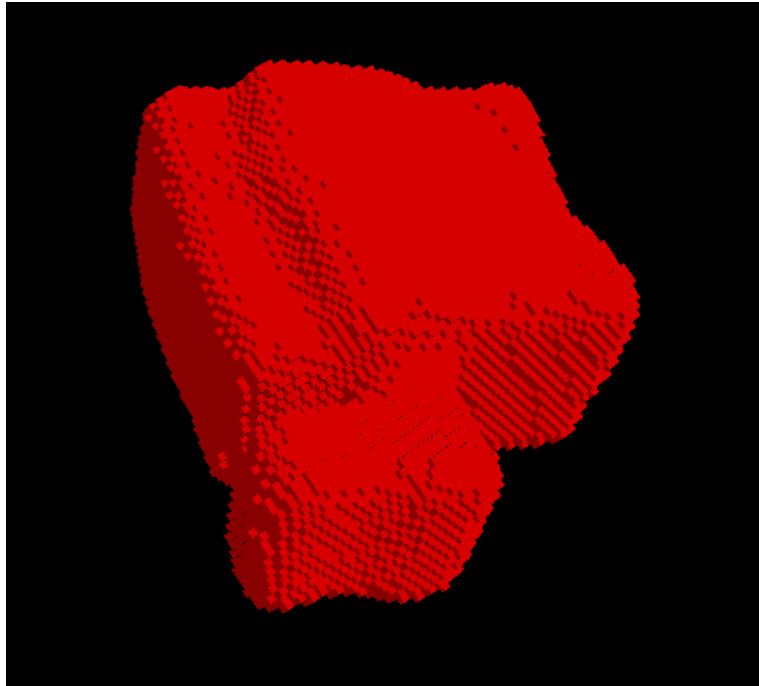


Ilustração 55 - Resultado da voxelização a 11 025Hz da sala de reuniões do IEETA.

6.4 Demonstração

Após todo o processamento dos dados obtidos pela Kinect Fusion, foram exportados os resultados para fazer uma demonstração na aplicação desenvolvida no projeto AcousticAVE [6] que é alimentado pelo modelo geométrico (informações apresentadas na Ilustração 36). O modelo obtido resultante da aplicação do algoritmo da Reconstrução de Poisson (ver secção 3.3.5) é apresentado no lado esquerdo da Ilustração 56 e o modelo simplificado através de decimação é apresentado do lado direito da Ilustração 56. Como o custo computacional da aplicação utilizada está diretamente relacionado com a complexidade do modelo geométrico fornecido foi utilizado o modelo simplificado, implicando isso uma visualização menos atrativa do interior da sala.

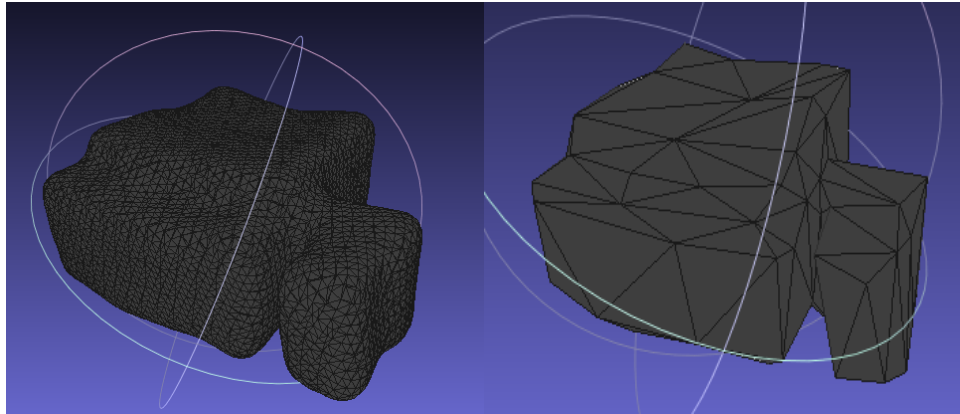


Ilustração 56 - Malha poligonal original após a aplicação da Reconstrução de Poisson (lado esquerdo) e resultado após simplificação através de decimação (lado direito).

Para esta simulação foram utilizadas os seguintes componentes: um dispositivo Head Mounted Display (HMD I-Glasses SVGA da empresa i-O Display Systems [63]) que permite a simulação visual da deslocação no interior da sala, uns auscultadores (Sony MDR – CD470) alimentados pelo resultado da auralização para simular a audição e um dispositivo de *tracking* (IntertiaCube BT desenvolvido pela empresa Intersense [64]) que permitia detetar a rotação da cabeça. Todos estes dispositivos podem ser observados na Ilustração 57.



Ilustração 57 - Fotografia tirada na sala de reuniões durante a simulação.

Para a simulação, dado que os nossos resultados não possuíam textura, esta foi associada manualmente aos polígonos que representam o chão, parede, zona de ventilação e teto da sala. Na Ilustração 58 pode ser observada uma imagem que estava a ser visualizada durante a simulação do mesmo ângulo de visão em que foi tirada a fotografia da Ilustração 57. A esfera azul representa a fonte sonora colocada na sala.

Apesar do modelo fornecido ainda não ser visualmente apelativo como era esperado, foi possível fornecer o modelo geométrico para simulação acústica como era proposto.



Ilustração 58 - Imagem visualizada pelo utilizador durante a simulação.

Também foi possível exportar toda a informação necessária do modelo para simulação acústica alimentada por um modelo físico como é possível observar no diagrama da Ilustração 59. No entanto, devido a esta aplicação de auralização estar ainda em fase de desenvolvimento no decorrer do projeto AcousticAVE, não foi possível testar a mesma com o modelo gerado.

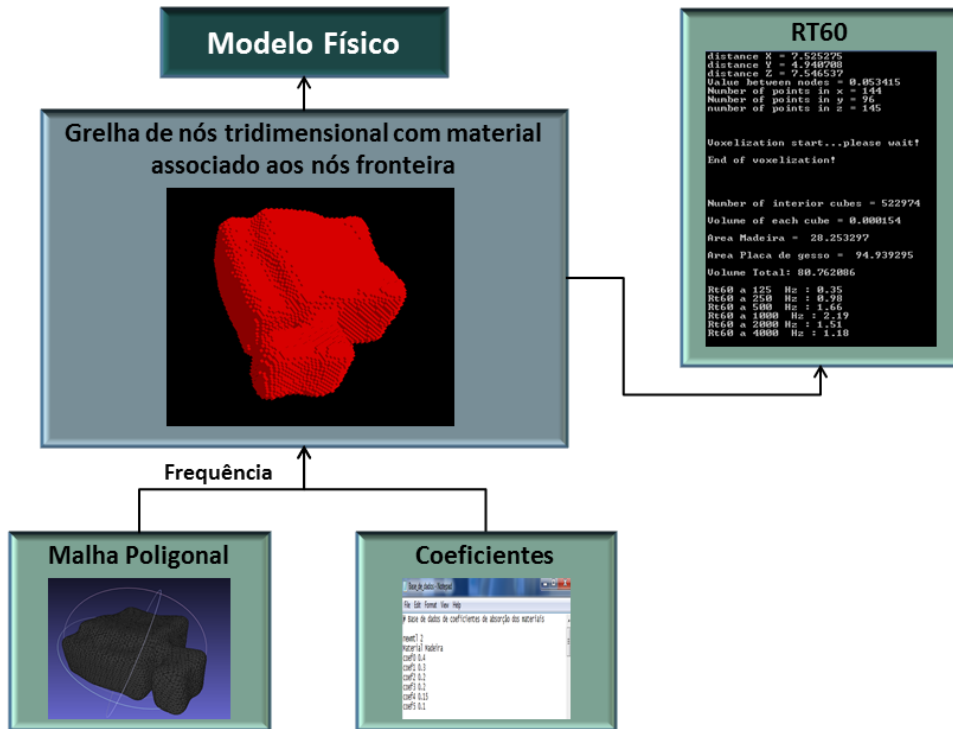


Ilustração 59 - Diagrama com toda a informação necessária para fornecer um modelo físico à aplicação de auralização.

7 Conclusão

7.1 Discussão

Este trabalho permitiu estudar e desenvolver ferramentas necessárias para aquisição de um modelo poligonal de uma sala real para posterior auralização do mesmo. As principais contribuições foram:

- Apresentação de um processo de aquisição que permite a aquisição de nuvens de pontos recorrendo à ferramenta Microsoft Kinect e aplicação Kinect Fusion;
- Desenvolvimento de processos de tratamento que, a partir dessa nuvem de pontos, permitem obter uma malha poligonal de uma sala real (alinhamento, detecção de planos principais, projeção de pontos e aplicação dos algoritmos, um que aplica o Convex Hull e outro que utiliza a Reconstrução de Poisson);
- Obtenção de um modelo geométrico com um número de polígonos limitado que possibilita a alimentação do processo de auralização;
- Desenvolvimento de uma aplicação automatizada para configurar os modelos para auralização que permite associar materiais (e o respetivo coeficiente de absorção) a cada polígono da malha recorrendo à textura;
- Criação de uma aplicação que, recorrendo ao processo de voxelização, permite colocar um ouvinte e fontes sonoras no interior de uma sala e, como resultado, exportar uma grelha tridimensional de nós para alimentar um modelo físico a ser utilizado no processo de auralização;
- Cálculo do valor RT_{60} durante o processo de voxelização desenvolvido.

É assim possível obter um modelo poligonal simplificado, com os materiais devidamente associados a cada superfície, que vai permitir alimentar um modelo geométrico ao processo de *auralização* desenvolvido pelo projeto AcousticAVE [6].

Por outro lado possibilita também fornecer uma grelha tridimensional de nós que vai identificar cada nó como pertencendo ao interior, exterior ou fronteira do modelo poligonal. Esta grelha irá permitir alimentar um modelo físico para outro dos processos de

auralização existente, processo esse que insere informações em cada nó fornecido, com o objetivo de simular a propagação do som efetuada ao longo dos vários nós.

O trabalho aqui desenvolvido também mostrou que existem entraves e limitações nas abordagens utilizadas passíveis de serem melhoradas através da utilização de outros métodos ou outras tecnologias. As principais limitações do trabalho aqui apresentado são:

- Impossibilidade de aquisição de uma malha completa numa única aquisição e com textura associada através do uso da Kinect Fusion;
- Necessidade de um computador com elevadas características computacionais para correr aplicações de aquisição de nuvens de pontos de um modo fluido.
- Não se conseguiu, através dos algoritmos de alinhamento, obtenção de planos, projeção de pontos e Convex Hull, obter uma malha completamente fechada, essencial para efeitos de simulação acústica;
- A limitação do número de polígonos de uma malha ainda não apresenta os resultados ideais de um polígono por cada plano principal da sala real adquirida;
- Aplicação que associa materiais aos diversos polígonos da malha apenas foi testada numa única nuvem de pontos adquirida na Kinect.

7.2 Trabalho Futuro

Como objeto de estudo para trabalho futuro, dadas as limitações encontradas anteriormente, pode ser considerado o desenvolvimento/utilização de ferramentas que permitem associar textura à nuvem de pontos, durante a aquisição e alinhamento. No caso da ferramenta Kinect Fusion ser desenvolvida com sucesso nesse sentido pode ser estudada a utilização desta para a resolução do problema da textura. Após a existência de uma nuvem de pontos (ou malha poligonal), de uma sala completa, que já contenha informações de textura, pode ser estudado melhor a questão da segmentação das cores como ferramenta de auxílio para identificação dos materiais. Por fim, pode ser estudado um processo para uma maior simplificação do número de polígonos presentes na malha poligonal adquirida, idealmente pretende-se a representação de cada plano principal existente na sala com dois triângulos (ou através de um único polígono quadrilátero). Para este processo de simplificação uma nova abordagem pode ser feita, utilizando os nós fronteira obtidos como

resultado do processo de voxelização, aplicando-lhes um algoritmo que faça detecção de planos em que possui a vantagem de ser alimentado com uma nuvem de pontos apenas com pontos que representam a fronteira do modelo em vez de ser fornecido uma nuvem de pontos que possui pontos com erros associados provenientes de erros de aquisição através da Kinect Fusion e erros durante o processo de alinhamento.

8 Referências

- [1] Q. Zhao, "A survey on virtual reality," *Science in China Series F: Information Sciences*, vol. 52, pp. 348-400, 2009/03/01 2009.
- [2] M. Vorländer, "*Auralization: Fundamentals of acoustics, modelling, simulation, algorithms and acoustic virtual reality*": Springer, 2007.
- [3] W. a. F. Ahnert, Rainer, "EARS Auralization software," presented at the Audio Engineering Society Convention 93, 1992.
- [4] K. H. Kuttruff, "Auralization of Impulse Responses Modeled on the Basis of Ray-Tracing Results," *J. Audio Eng. Soc.*, vol. 41, pp. 876--880, 1993.
- [5] M. Kleiner, B.-I. Dalenbäck, and P. Svensson, "Auralization-An Overview," *J. Audio Eng. Soc.*, vol. 41, pp. 861--875, 1993.
- [6] A. Oliveira, G. Campos, P. Dias, D. Murphy, J. Vieira, C. Mendonça, and J. Santos, "Real-time dynamic image-source implementation for auralization," presented at the Proc. of the 16th Int. Conference on Digital Audio Effects (DAFx-13), Maynooth, Ireland, 2013.
- [7] L. Savioja, M. Karjalainen, and T. Takala, "DSP Formulation of a Finite Difference Method for Room Acoustics Simulation," presented at the Proceedings of the IEEE Nordic Signal Processing Symposium NORSIG'96, Espoo, Finland, 1996.
- [8] A. L. Software. (2000). *SketchUp* [3D modeling software]. Available: <http://www.sketchup.com/>
- [9] Autodesk®. (1982). *AutoCAD®*. Available: <http://www.autodesk.com/autocad>
- [10] B. Foundation. (1998). *Blender (2.68 ed.)* [Open-Source]. Available: www.blender.org
- [11] Y. Wun-Bin, C. Min-Bin, Y. Ya-ning, and C. Hung-Ming, "An integrated 3D laser scanning technique for the digitization of historic buildings," in *Virtual Systems and Multimedia (VSM), 2010 16th International Conference on*, 2010, pp. 332-335.
- [12] W. Schroeder, K. M. Martin, and W. E. Lorensen, "*The visualization toolkit: an object-oriented approach to 3D graphics*": Prentice-Hall, Inc., 1998.
- [13] G. Sellers, R. S. Wright, and N. Haemel Jr., "*OpenGL SuperBible: Comprehensive Tutorial and Reference*", 6 edition ed.: Addison-Wesley Professional, 2013.
- [14] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on* Shanghai 2011.
- [15] P. Cignon, M. Corsini, and G. Ranzuglia. (2008). *MeshLab: an Open-Source 3D Mesh Processing System*. Available: <http://meshlab.sourceforge.net>
- [16] P. Dias, G. Campos, V. Santos, R. Casaleiro, R. Seco, and B. Sousa Santos, "3D reconstruction and spatial auralization of the Painted Dolmen of Antelas," presented at the Proceedings of the Electronic Imaging, San Jose, California, USA, 2008.
- [17] M. R. Andersen, T. Jensen, P. Lisouski, A. K. Mortensen, M. K. Hansen, T. Gregersen, and P. Ahrendt, "Kinect Depth Sensor Evaluation for Computer Vision Applications " Electrical and Computer Engineering Department of Engineering, Aarhus University, 2012.
- [18] Z. Zalevsky, A. Shpunt, A. Maizels, and J. Garcia, "Method and system for object reconstruction," ed: Google Patents, 2010.
- [19] Microsoft. (2013). *Kinect for Windows sensor components and specifications*. Available: <http://msdn.microsoft.com/en-us/library/jj131033.aspx>
- [20] E. Keppel, "Approximating complex surfaces by triangulation of contour lines," *IBM J. Res. Dev.*, vol. 19, pp. 2-11, 1975.

- [21] S. Mohan and J. F. Lee, "Three-dimensional finite element mesh generation using Delaunay triangulation," *Antennas and Propagation Society International Symposium, AP-S. Digest*, pp. 312-315 vol.1, June 28 1993-July 2 1993 1993.
- [22] Microsoft. (2011). *Kinect Fusion*. Available: <http://msdn.microsoft.com/en-us/library/dn188670.aspx>
- [23] PCL. (2012). *Kinfu*. Available: http://pointclouds.org/documentation/tutorials/using_kinfu_large_scale.php
- [24] D. Solutions. (2012). *KScan3D*. Available: <http://www.kscan3d.com/>
- [25] ManCTL. (2013). *Skanect*. Available: <http://skanect.com/>
- [26] P. GmbH. (2012). *ReconstructMe*. Available: <http://reconstructme.net/>
- [27] M. Research. (2011). *KinectFusion Project Page*. Available: <http://research.microsoft.com/en-us/projects/surfacerecon/>
- [28] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon, "KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera," presented at the 28th Chaos Communication Congress (28C3) by the Chaos Berlin Congress Center [bcc]; Alexanderstr. 11; 10178 Berlin; Germany, 2011.
- [29] R. A. Newcombe, S. Izadi, H. Otmar, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, "KinectFusion: Real-time dense surface mapping and tracking," presented at the Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality, 2011.
- [30] J. Kramer, N. Burrus, F. Ehtler, C. D. Herrera, and M. Parker, "*Hacking the Kinect*": Apress, 2012.
- [31] S. Woop and P. Slusallek, "RPU: a programmable ray processing unit for realtime ray tracing," *ACM Trans. Graph.*, vol. 24, pp. 434-444, 2005.
- [32] P. J. Besl and N. D. McKay, "A method for registration of 3-D shapes," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 14, pp. 239-256, 1992.
- [33] R. B. Rusu. (2011). *An open source implementation of KinectFusion*. Available: <http://librarywww.pointclouds.org/news/2011/12/08/kinectfusion-open-source/>
- [34] M. Pirovano, "Kinfu - an open source implementation of Kinect Fusion + case study: implementing a 3Dscanner with PCL," 2012.
- [35] M. Vona. (2013). *Geometric and Physical Computing (GPC)*. Available: <http://www.ccs.neu.edu/research/gpc/mvkinfu/index.html>
- [36] P. J. Schneider and D. Eberly, "*Geometric Tools for Computer Graphics*": Elsevier Science Inc., 2002.
- [37] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson surface reconstruction," presented at the Proceedings of the fourth Eurographics symposium on Geometry processing, Cagliari, Sardinia, Italy, 2006.
- [38] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, pp. 381-395, 1981.
- [39] N. Blodow. (2011). *PointCloudLibrary: PCL :: Segmentation*. Available: http://www.pointclouds.org/assets/rss2011/04_segmentation.pdf
- [40] M. Kazhdan, "Reconstruction of solid models from oriented point sets," presented at the Proceedings of the third Eurographics symposium on Geometry processing, Vienna, Austria, 2005.
- [41] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," *SIGGRAPH Comput. Graph.*, vol. 21, pp. 163-169, 1987.

- [42] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," presented at the Proceedings of the 24th annual conference on Computer graphics and interactive techniques, 1997.
- [43] L. L. Doelle, *Environmental acoustics*: McGraw-Hill, 1972.
- [44] K. V. Mardia and T. J. Hainsworth, "A spatial thresholding method for image segmentation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 10, pp. 919-927, 1988.
- [45] F. Kurugollu, B. Sankur, and A. E. Harmanci, "Color image segmentation using histogram multithresholding and fusion," *Image and Vision Computing*, vol. 19, pp. 915-928, 2001.
- [46] J. Canny, "A Computational Approach to Edge Detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. PAMI-8, pp. 679-698, 1986.
- [47] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*: Addison-Wesley Longman Publishing Co., 2001.
- [48] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*: Prentice-Hall, Inc., 1988.
- [49] PCL. (2011). *The OpenNI Grabber Framework in PCL*. Available: http://pointclouds.org/documentation/tutorials/opensni_grabber.php
- [50] A. Kaufman and E. Shimony, "3D scan-conversion algorithms for voxel-based graphics," presented at the Proceedings of the 1986 workshop on Interactive 3D graphics, Chapel Hill, North Carolina, USA, 1987.
- [51] D. Haumont and N. Warzee, "Complete polygonal scene voxelization," *ACM Journal of Graphics Tools*, vol. 7, pp. 27-41, 2002.
- [52] O. Mattausch. (2011). *GPU-based Scene Voxelization*. Available: <http://christl.cg.tuwien.ac.at/courses/projekte/ve/voxelization.html>
- [53] F. Dachille and A. Kaufman, "Incremental triangle voxelization," *In Proceedings of Graphics Interface*, pp. 205-212, 2000.
- [54] M. Sramek and A. Kaufman, "Alias-Free Voxelization of Geometric Objects," *IEEE Transactions on Visualization and Computer Graphics*, vol. 5, pp. 251-267, 1999.
- [55] A. Kaufman, "Efficient algorithms for 3D scan-conversion of parametric curves, surfaces, and volumes," *SIGGRAPH Comput. Graph.*, vol. 21, pp. 171-179, 1987.
- [56] J. Huang, R. Yagel, V. Filippov, and Y. Kurzion, "An accurate method for voxelizing polygon meshes" presented at the Proceedings of the 1998 IEEE symposium on Volume visualization, Research Triangle Park, North Carolina, USA, 1998.
- [57] N. Stolte and A. Kaufman, "Novel techniques for robust voxelization and visualization of implicit surfaces," *Graph. Models*, vol. 63, pp. 387-412, 2001.
- [58] C. Quammen and R. Taylor II, "Grid voxelization with partial volume effects in VTK", *The VTK Journal*, 2011.
- [59] Voxelogic. (2013). *Acropora*. Available: <http://www.voxellogic.com>
- [60] W. C. Sabine, *Collected papers on acoustics*: Cambridge : Harvard University Press, 1922.
- [61] C. F. Eyring, "Reverberation Time in "Dead" Rooms ", *The Journal of the Acoustical Society of America*, vol. 1, p. 1, 1930.
- [62] G. Millington, "A modified formula for reverberation.", *The Journal of the Acoustical Society of America*, vol. 4, pp. 69-69, 1932.
- [63] I.-O. D. Systems. (2012). *HMD i-Glasses i3pc*. Available: <http://www.i-glassesstore.com/i-glasses-i3pc.html>
- [64] Intersense. (2013). *InertiaCube BT™*. Available: <http://www.intersense.com/pages/18/60/>

