

**Luciana Brasil Sondermann Alves**

**Um estudo sobre a captura de objetos em movimento com  
múltiplas câmeras RGB-D para efeitos especiais**

**Dissertação de Mestrado**

Dissertação apresentada como requisito parcial para  
obtenção do título de Mestre pelo Programa de Pós-  
Graduação em Informática da PUC-Rio.

Orientador: Prof. Bruno Feijó

Rio de Janeiro,  
julho de 2015



**Luciana Brasil Sondermann Alves**  
**Um estudo sobre a captura de objetos em**  
**movimento com múltiplas câmeras RGB-D**  
**para efeitos especiais**

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre pelo Programa de Pós-Graduação em Informática do Departamento de Informática do Centro Técnico e Científico da PUC-Rio. Aprovada pela Comissão Examinadora abaixo assinada.

**Prof. Bruno Feijó**

Orientador

Departamento de Informática - PUC-Rio

**Prof. Hélio Côrtes Vieira Lopes**

Departamento de Informática - PUC-Rio

**Prof. Luiz Eduardo Azambuja Sauerbronn**

UFRJ

**José Eugenio Leal**

Coordenador(a) Setorial do Centro Técnico Científico - PUC-Rio

Rio de Janeiro, 17 de julho de 2015

Todos os direitos reservados. É proibida a reprodução total ou parcial do trabalho sem autorização da universidade, da autora e do orientador.

**Luciana Brasil Sondermann Alves**

Licenciada em Matemática pela Universidade do Estado do Rio de Janeiro e formada em Análise de Sistemas pela FAETERJ-Rio. Atualmente, participa do ICAD-VisionLab, Puc-Rio e trabalha no desenvolvimento de games para a área

Ficha Catalográfica

Alves, Luciana Brasil Sondermann

Um estudo sobre a captura de objetos em movimento com múltiplas câmeras RGB-D para efeitos especiais / Luciana Brasil Sondermann Alves ; orientador: Bruno Feijó. – 2015.

59 f. : il. (color.) ; 30 cm

Dissertação (mestrado)–Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, 2015.

Inclui bibliografia

1. Informática – Teses. 2. Nuvem de pontos. 3. Kinect. 4. VFX. 5. Efeitos especiais. 6. Krakatoa. I. Feijó, Bruno. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

CDD: 004

A Deus, apenas devolvendo o que me foi dado.

## Agradecimentos

Agradeço à minha família pelo apoio recebido.

Agradeço aos professores da FAETERJ-Rio que me deram a base necessária para que eu pudesse ingressar no Mestrado.

Agradeço aos professores do Departamento de Botânica da UFRJ que entenderam minha missão e permitiram que eu fosse liberada de minhas funções.

Agradeço ao meu orientador Bruno Feijó que se mostrou paciente desde o início.

Agradeço aos colegas de curso da PUC-Rio que me ajudaram em momentos críticos.

E, por fim, agradeço à PUC-Rio que me agraciou com uma bolsa de isenção, fazendo com que este meu sonho se tornasse real.

## Resumo

Alves, Luciana Brasil Sondermann; Feijó, Bruno. **Um estudo sobre a captura de objetos em movimento com múltiplas câmeras RGB-D para efeitos especiais.** Rio de Janeiro, 2015. 59p. Dissertação de Mestrado - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Esta dissertação faz um estudo sobre a geração de efeitos visuais (VFX) a partir da captura de objetos em movimento como nuvens muito densas de pontos utilizando múltiplas câmeras RGB-D de baixo custo. Para este estudo, são utilizados um software comercial para renderização de partículas e alguns módulos já desenvolvidos pelo Departamento de P&D em VFX da TV Globo em parceria com o laboratório ICAD/VisionLab para a captura simultânea de múltiplas câmeras MS Kinect. No pipeline de produção proposto, um servidor sincroniza os disparos das várias câmeras, unifica as nuvens e gera um arquivo em formato padrão (PRP ou PLY). Este arquivo é, então, utilizado para a renderização de partículas volumétricas com adição de efeitos visuais utilizando o software Krakatoa MX para o 3DS Max. O objetivo é filmar cenas de maneira que o diretor possa posteriormente definir as cenas finais com qualquer trajetória de câmera e adicionar efeitos visuais.

## Palavras-chave

Nuvem de pontos; Kinect; VFX; Efeitos visuais; Krakatoa.

## Abstract

Alves, Luciana Brasil Sondermann; Feijó, Bruno (Advisor). **A study about capture of objects in moving with multiple RGB-D cameras for special effects.** Rio de Janeiro, 2015. 59p. MSc. Dissertation - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

This dissertation is an investigation on the generation of visual effects (VFX) from the capture of moving objects as very dense point clouds using multiple low-cost RGB-D cameras. For this investigation, we use a commercial software for particle rendering and some modules already developed by the Department of VFX R&D of TV Globo in partnership with ICAD/VisionLab laboratory for the simultaneous capture of multiple MS Kinect cameras. In the proposed production pipeline, a server synchronizes the shooting of multiple cameras, unifies the clouds and generates a file in standard format (PRP or PLY). This file is then used for volumetric particle rendering with added visual effects using the Krakatoa MX software for 3DS Max. The goal is to shoot scenes in such a way that the film director can later define the final scenes with any camera path and adding visual effects.

## Keywords

Point cloud; Kinect; VFX; Visual effects; Krakatoa.

# Sumário

1. Introdução	13
2. Nuvem de Pontos	16
2.1. Definição e Processos	16
2.1.1. Geração de Malhas	18
2.1.2. Segmentação	19
2.2. Point Cloud Library (PCL)	20
2.3. Captura	21
2.3.1. Scanners a laser 3D	21
2.3.2. Câmeras RGB-D	23
2.3.2.1. Cálculo de Profundidade	25
2.3.3. Dificuldades na Captura	27
2.4. Softwares para VFX	27
2.4.1. Krakatoa	29
2.5. Dificuldades	30
3. Sistemas de Captura com múltiplas câmeras	31
3.1. Arquitetura	32
3.1.1. Modelagem	32
3.1.1.1. KinectCapture:	32
3.1.1.2. KinectConvert:	33
3.1.1.3. KinectCapture Sync:	33
3.1.1.4. KinectMerger:	34
3.1.2. Diagramas	34
3.2. Requisitos do Software	37
3.2.1. Funcionais	37
3.2.2. Não-Funcionais	38
4. Tutorial e Resultados	40
4.1. Utilização do Krakatoa	40
4.2. Resultados obtidos	45
4.2.1. Resultado dos Módulos desenvolvidos	45



4.2.2. Resultados Finais	46
4.2.2.1. Exemplo 1	47
4.2.2.2. Exemplo 2	48
4.2.2.3. Exemplo 3	49
4.2.2.4. Exemplo 4	51
4.2.2.5. Exemplo 5	52
5. Considerações Finais	54
5.1. Conclusões	54
5.2. Trabalhos Futuros	55
6. Referências Bibliográficas	57

## Lista de figuras

Figura 1 – Efeito especial com partículas no filme G.I. JOE: The Rise of Cobra usando Krakatoa. Fonte: DMN NEWSWIRE. Imagem com direito de autor copiada sob a política de <i>fair use</i> .	17
Figura 2 – Desintegração do robô do filme Elysium. Para obter este efeito, pequenos conjuntos de pontos são associados a pequenos pedaços de objetos 3D e depois os pontos. são animados conforme os impactos que sofrem (levando juntos os objetos a que estão associados) (a). Fonte: FilmRave. Imagem com direito de autor copiada sob a política de <i>fair use</i> .	17
Figura 3 – Laser Scanner 3D da Faro. Modelo Foco 3D X 330. Fonte: <a href="http://www.faro.com">www.faro.com</a>	22
Figura 4 – Kinect for Windows sensor lançado pela Microsoft em 2010. Fonte: ROBOTIQ	24
Figura 5 – Imagem RGB à esquerda e informação de profundidade captada por uma câmera RGB-D à direita. Fonte: Henry et al. (2014)	25
Figura 6 - Motion blur causado por movimento rápido do jogador no game Crysis 3. Fonte: TWEAKGUIDES.	25
Figura 7 – Sistema Time-of-flight. Fonte: CASTANEDA and NAVAB (2011)	26
Figura 8 – Nuvem de pontos melhorada pelo software SCENE da Faro. Fonte: <a href="http://www.faro.com">www.faro.com</a>	28
Figura 9 – Imagem do filme Lucy (2014). Fonte: Site do Krakatoa. Imagem com direito de autor copiada sob a política de <i>fair use</i> .	30
Figura 10 – Diagrama de classe do módulo KinectCapture	35
Figura 11 – Diagrama de classe do módulo KinectCapture Sync	35
Figura 12 - Diagrama de classe do módulo KinectMerger	36
Figura 13 – Diagrama de caso de uso para 1 kinect	36
Figura 14 – Diagrama de caso de uso para mais de 1 kinect	37
Figura 15 – Seleção do Krakatoa no 3DS (a) e botão para carregar nuvem de pontos (b)	40
Figura 16 – Botão de configuração da timeline do 3DS	41
Figura 17 – Porcentagem de renderização das partículas	42
Figura 18 – Seleção do Krakatoa como renderer (Parte 1)	43

Figura 19 – Seleção do Kraktoa como renderer (Parte 2)	43
Figura 20 – Seleção de frames a serem renderizados e opção de saída dos arquivos	44
Figura 21 – Escolha de nome, tipo de arquivo e local para salvamento.	45
Figura 22 – Imagem original na escala RGB (a). Imagem obtida pela câmera de infravermelho (b). Nuvem de pontos gerada (c). Fonte: A autora.	46
Figura 23 – Imagem extraída do vídeo realizado para exibição no canal Futura. Cenário é o do laboratório ICAD/VisionLab.	47
Figura 24 – Outras imagens do mesmo vídeo para o canal Futura que comprovam a animação de câmera. Cenário é o do laboratório ICAD/VisionLab.	48
Figura 25 – Imagem do vídeo no início da animação	49
Figura 26 – Imagens intermediária e final ( $z=0$ ) da animação respectivamente	49
Figura 27 – Nuvem de pontos renderizada extraída de vídeo. Imagem com direito de autor copiada sob a política de <i>fair use</i> .	50
Figura 28 – Nuvem de pontos renderizada extraída de vídeo. Imagem com direito de autor copiada sob a política de <i>fair use</i> .	51
Figura 29 – Efeito bend aplicado a uma nuvem de pontos	52

## Lista de tabelas

Tabela 1 – Requisitos funcionais do software de captura	38
Tabela 2 – Requisitos não-funcionais do software de captura	39

## 1. Introdução

Efeitos visuais em cinema e TV, abreviadamente VFX (*Visual Effects*), são processos através dos quais imagens são criadas ou manipuladas após a filmagem de cenas ao vivo. Normalmente VFX completa cenas gravadas em tela verde/azul (usando *chroma key* e composição com Nuke ou Fusion), com elementos criados por softwares de modelagem e renderização 3D (usualmente 3DS Max, Maya e Renderman). Atualmente, a indústria de cinema e TV tem utilizado *scanners* de alta precisão para gerar nuvens de pontos 3D como fonte de dados para VFX. Estas nuvens de pontos podem ser usadas em vários processos, tais como: construir ambientes virtuais para cenas em tela verde/azul; construir extensões virtuais de estúdios; pré-visualização; referência para medições feitas pelas equipes de arte; auxílio para definição de movimentos de câmera. Entretanto, estes *scanners* de precisão não são utilizados para objetos que se movem. Se nuvens de pontos fossem geradas para objetos em movimento (incluindo atores), o diretor poderia testar vários movimentos de câmera e efeitos nos objetos após a cena ter sido filmada. Infelizmente, na indústria de entretenimento, não há ainda uma solução adequada para capturar objetos em movimento e transformá-los em nuvens de pontos de alta qualidade. Esta dissertação propõe capturar objetos em movimento como nuvens muito densas de pontos utilizando múltiplas câmeras RGB-D de baixo custo. Esta não seria uma solução definitiva, pois estas câmeras baratas não são compatíveis com as altas resoluções utilizadas em TV e cinema (4K e 8K). Mas, por outro lado, alguns efeitos com partículas podem ser realizados com alta qualidade. Um personagem pode, por exemplo, se desintegrar ou participar de uma cena de *flashback* ou sonho (onde a textura é fragmentada ou nebulosa). Integrar várias câmeras RGB-D não é uma tarefa trivial, o que fica claro no presente trabalho.

Há vários sistemas propostos na literatura para a reconstrução de modelos tridimensionais em tempo real a partir da captura de uma única câmera de profundidade (RGB-D). Esta reconstrução de cenas 3D pode acelerar a produção de efeitos especiais, ajudar na produção de games e dar suporte para sistemas de visão computacional. Por exemplo, Newcombe et al. (2011) foram um dos primeiros a proporem um método em tempo real para mapear e rastrear

superfícies usando uma câmera de profundidade de baixo custo. Henry et al. (2014) utilizaram uma técnica de otimização global para produzir mapas 3D para navegação de robôs em ambientes internos. Zhou e Koltun (2013) usaram pontos de interesse para reconstruir cenas de grande complexidade, também utilizando câmeras de baixo custo.

Entretanto, os sistemas mencionados acima não têm o objetivo mais simples de capturar pontos para trabalhar efeitos diretamente com estes pontos, num esquema de registro *frame-to-frame*. Nestes sistemas, o registro é feito *frame-to-model*, o que de fato é mais robusto quando o objetivo é a reconstrução da superfície 3D.

É importante deixar registrado que *frame-to-frame* e *frame-to-model* estão relacionados com fluxo óptico que é o campo de movimento calculado sob a hipótese de que as intensidades em redor de pontos em correspondência de duas imagens se mantêm constante. Ele nos permite dado um conjunto de pontos numa determinada imagem, encontrar esses mesmos pontos numa outra imagem (BRITO, p. 21). Em Brito (2009) é proposto um algoritmo para cálculo de fluxo óptico.

Na presente dissertação, queremos experimentar a captura em massa de pontos *frame-to-frame* com a maior resolução possível considerando os movimentos tanto da câmera como dos objetos. Para tanto, utilizaremos múltiplas câmeras de profundidade.

Nesta dissertação, vamos implementar um *pipeline* simples de produção para gerar efeitos especiais diretamente em cima de uma nuvem de pontos obtidos *frame-a-frame*. Para tanto, utilizamos um software comercial para renderização de partículas e usamos alguns módulos já desenvolvidos pelo Departamento de P&D em VFX da TV Globo em parceria com o laboratório ICAD/VisionLab que fazem a captura simultânea de múltiplas câmeras MS Kinect. No pipeline de produção proposto, um servidor sincroniza os disparos das várias câmeras, unifica as nuvens e gera um arquivo em formato padrão (PRP ou PLY). Utilizamos então o software Krakatoa MX para carregar e renderizar os arquivos no 3DS Max, aproveitando este para gerar os efeitos visuais.

Esta dissertação não pretende esgotar o assunto, mas apenas experimentar o uso da captura de objetos em movimento com múltiplas câmeras RGB-D para a produção de efeitos visuais. Apesar de usarmos módulos já desenvolvidos em projetos anteriores, estamos contribuindo com a montagem

destes módulos de modo a formar uma arquitetura adequada para o *pipeline* proposto.

Esta dissertação está estruturada da seguinte forma. No capítulo 2, definimos nuvem de pontos e exploramos algumas formas de a mesma ser capturada. Citamos ainda alguns softwares utilizados para se trabalhar com nuvens de pontos no mercado de VFX, inclusive o Krakatoa MX que foi o escolhido para ser utilizado neste trabalho. No capítulo 3, descrevemos os quatro módulos do software utilizado para captura de uma nuvem de pontos a partir de vários kinects, citamos seus requisitos (funcionais e não-funcionais) e as funcionalidades das classes principais de cada um deles. Mostramos ainda um exemplo de nuvem de pontos capturada pelo software. O capítulo 4 contém um mini-tutorial sobre como carregar nuvens de pontos no 3DS Max com o uso do Krakatoa MX. Exibimos ainda alguns exemplos de combinação de nuvem de pontos com animações realizadas no 3DS Max. No capítulo 5, apresentamos as conclusões desta dissertação, algumas dificuldades sobre a implementação do projeto em televisão e, ainda, algumas sugestões de trabalhos futuros.

## 2. Nuvem de Pontos

### 2.1. Definição e Processos

Uma nuvem de pontos é uma coleção de posições no espaço tridimensional ( $x$ ,  $y$ ,  $z$ ) com informações adicionais, como, por exemplo, cor e normais e pode ser considerada como uma amostra de uma superfície contínua (Katz, 2007). Cada ponto pertencente a uma nuvem possui os valores correspondentes de suas posições relativas no espaço de acordo com a posição do objeto utilizado para realizar a captura.

Como são elementos unidimensionais, os pontos não contêm quaisquer outros tipos de geometria, logo, não possuem faces ou arestas. Além disto, uma nuvem de pontos não pode ser convertida para uma geometria, contudo, é possível criar uma geometria que esteja em conformidade com a forma da nuvem de pontos utilizando técnicas disponíveis em softwares específicos, como, por exemplo, o *snap para pontos* no 3DS Max.

Uma nuvem de pontos pode ser utilizada para rastrear, representar e manipular objetos e ambientes tridimensionais referentes ao mundo real a partir de importações de dados de captura e, devido ao aumento do poder computacional das placas gráficas, tornam-se cada vez mais incluídas nas aplicações de RV e RA, principalmente quando se trata de animação e visualização. Uma outra vantagem da nuvem de pontos é que podemos incluir objetos realistas em uma cena 3D, sem a necessidade de modelá-los explicitamente.

Atualmente, a indústria do cinema tem realizado efeitos especiais visuais (VFX) de alta qualidade a partir de nuvens muito densas de partículas (Fig. 1). Em outras cenas, nuvens não tão densas de pontos são utilizadas para criar outros efeitos, tal como na desintegração do robô do filme *Elysium* (Fig. 2). Em cenas como estas, seria bastante produtivo se a nuvem de pontos viesse da captura em tempo real de objetos em movimento.





Figura 1 – Efeito especial com partículas no filme G.I. JOE: The Rise of Cobra usando Krakatoa. Fonte: DMN NEWSWIRE<sup>1</sup>. Imagem com direito de autor copiada sob a política de *fair use*.

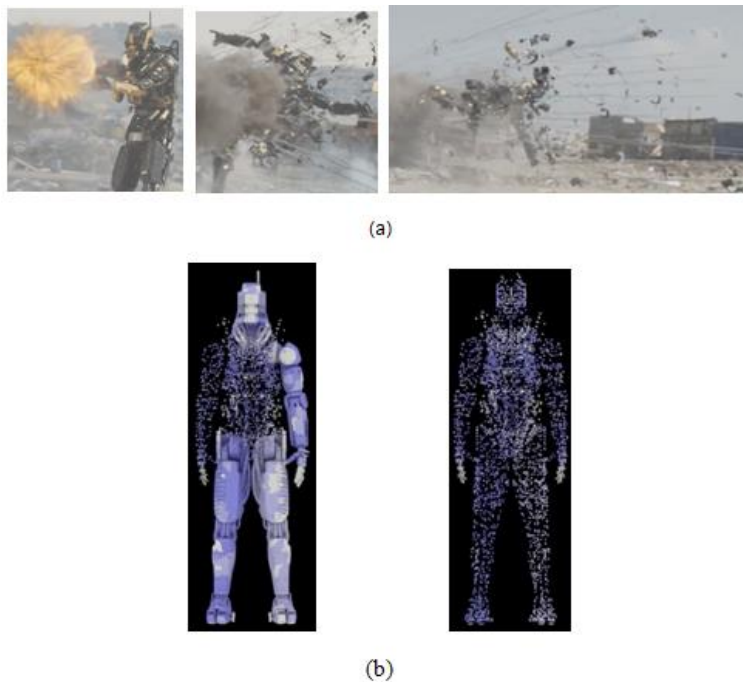


Figura 2 – Desintegração do robô do filme Elysium. Para obter este efeito, pequenos conjuntos de pontos são associados a pequenos pedaços de objetos 3D e depois os pontos. são animados conforme os impactos que sofrem (levando juntos os objetos a que

<sup>1</sup> Disponível em <http://dmnnewswire.digitalmedianet.com/article/Prime-Focus-VFX-Helps-Elevate--GI-JOE:-The-Rise-of-Cobra-826540>

estão associados) (a). Fonte: FilmRave<sup>2</sup>. Imagem com direito de autor copiada sob a política de *fair use*.

### 2.1.1. Geração de Malhas

Assim como um objeto pode ser decomposto em uma nuvem de pontos, este processo pode ser invertido. Temos, assim, que uma nuvem pode ser utilizada para geração de malhas desde que algumas exigências sejam satisfeitas.

Os métodos existentes de reconstrução de malhas buscam satisfazer dois pontos:

- A superfície  $S$  gerada deve possuir a mesma topologia que a superfície  $S'$  que contém os pontos da nuvem;
- Um ponto contido em  $S'$  deve ter uma distância suficientemente pequena em relação à  $S$ .

Gois (2004) esclarece que: “A correta reconstrução de uma superfície é possível apenas se o conjunto  $P^3$  for adequado” (p. 26). Ainda segundo Gois (op.cit.), o algoritmo utilizado é que determinará se o conjunto de pontos é adequado ou não. Geralmente, uma das exigências é que haja uma maior densidade de pontos nas regiões em que as curvaturas sofrem grandes variações ou ainda que a nuvem seja densa e uniforme. Ou seja, precisamos somente de pontos para gerar uma malha.

Ainda segundo Gois, em geral, os métodos de reconstrução de superfícies a partir de nuvem de pontos podem ser classificados de quatro formas distintas. São elas: Métodos baseados em esculpimentos ou decomposição espacial, métodos baseados em funções implícitas, métodos incrementais e métodos baseados em modelos deformáveis. Como a geração de malhas não é o foco deste trabalho, descreveremos abaixo, apenas de forma resumida, cada um dos métodos citados:

---

<sup>2</sup> Prints retirados do filme disponível em <http://www.filmrave.co.uk/2013/09/elysium-movie-review-allegory-and-action.html>

<sup>3</sup> Pontos pertencentes à nuvem de pontos

- Métodos baseados em esculpimentos ou decomposição espacial: consiste em obter a triangulação de Delaunay do conjunto de pontos para em seguida utilizar heurísticas denominadas esculpimentos para remover um conjunto de simplexes<sup>4</sup> e, assim, obter uma malha que represente o objeto original de modo mais fidedigno possível;
- Métodos baseados em funções implícitas: Consiste em utilizar a amostragem de pontos para definir uma função distância  $f$  (com sinal) e calcular o conjunto zero de  $f$ , denotado por  $Z(f)$ . Também é chamado de métodos baseados em conjunto zero de função;
- Métodos incrementais: Este método consiste em construir uma superfície de modo iterativo a partir de uma aresta inicial (ou de um conjunto de arestas iniciais) construída conectando dois pontos da nuvem. Após serem definidas as arestas iniciais, obtém-se as demais arestas a partir desta;
- Métodos baseados em modelos deformáveis: Neste método, aplicam-se deformações em uma superfície inicial gerando uma aproximação para a nuvem de pontos;

### 2.1.2. Segmentação

O processo de segmentação tem como objetivo “agrupar regiões ou objetos similares, como por exemplo: paredes, esquadrias, cobertura.” (GROETELAARS and AMORIM, 2011).

A segmentação é o primeiro passo para se interpretar uma nuvem de pontos. Ela permite o particionamento dos dados do objeto em nuvem a fim de criar subconjuntos significativos, coerentes e conectados, denominados segmentos. Cada segmento inclui pontos com atributos semelhantes e representa uma superfície, um objeto. O resultado da segmentação é o conjunto de segmentos extraídos (RYCHARD, 2010). A segmentação é possível a partir

---

<sup>4</sup> Generalização do conceito do triângulo em outras dimensões. No caso do espaço 3D, se refere a tetraedro

de critérios de agregação, como a definição de limites devido às alterações de cor, plano, inclinação ou de material. Ela pode ser realizada através de processos interativos de seleção de trechos das nuvens em programas simples ou utilizando processos automatizados em programas mais complexos.

O método de Contornos Ativos Snake também é uma boa solução para a segmentação de imagens digitais, pois possui a característica de movimentar uma curva de maneira dinâmica sobre a imagem até que se alinhe às bordas do objeto de interesse (MELO, 2005).

Como exemplo de programa que realiza a classificação a partir de nuvem de pontos, podemos citar o EdgeWISE da ClearEdge3D que pode ser utilizado junto com o Revit (Seção 2.3). Ele é capaz de identificar automaticamente e extrair paredes, janelas, portas e outras características de nuvens de pontos e exportá-las como objetos da família do Revit.

## 2.2. Point Cloud Library (PCL)

Apesar de ainda não haver nenhuma empresa ou universidade localizados na América Latina que participem do projeto PCL, é muito importante destacar a sua existência.

O Point Cloud Library (ou PCL) é um projeto aberto em larga escala para processamento de imagens 2D/3D e de nuvens de pontos. O framework PCL contém inúmeros algoritmos que realizam filtragem, estimativa de recurso, reconstrução da superfície, registro, ajuste de modelo e segmentação. Esses algoritmos podem ser usados, por exemplo, para filtrar valores atípicos de dados com ruídos, pontos sobrepostos em nuvens de pontos e segmentos de partes relevantes da cena, além de computar descritores para reconhecer objetos do mundo com base em sua aparência geométrica e criar superfícies a partir de nuvens de pontos e visualizá-las.

O PCL é *open source* e totalmente livre para o uso comercial e acadêmico. Ele é dividido em bibliotecas de código menores, que podem ser compilados separadamente.

Este projeto está sendo desenvolvido por engenheiros e cientistas de diferentes organizações distribuídas ao redor do mundo. Entre elas podemos citar as Universidades de Toronto, Nova York, Carlos III de Madri, Houston, além das empresas Leica Geosystems e Toyota. Dentre as várias organizações que financiam este projeto estão Google, Nvidia, Toyota, Intel e Leica.

## **2.3. Captura**

Para a captura de nuvens de pontos, os métodos mais utilizados são os *scanners* a laser 3D e câmeras de profundidade ou RGB-D.

### **2.3.1. Scanners a laser 3D**

A técnica de escaneamento utilizando scanner a laser 3D (Fig. 3) é uma tecnologia que permite obter uma nuvem de pontos realizando a varredura das superfícies dos objetos através de feixes de raios laser, utilizando aparatos exclusivos. Desta maneira, torna-se possível a captura automática de grande quantidade de dados em um curto intervalo de tempo (GROETELAARS and AMORIM, 2011). De acordo com a qualidade do *scanner* e da distância entre ele e o elemento a ser escaneado, é possível realizar a varredura de objetos de tamanhos variados, sejam eles prédios ou peças milimétricas.

Ainda de acordo com GROETELAARS and AMORIM (2011, p. 490), para se utilizar esta tecnologia, pelo menos cinco etapas se fazem necessárias. São elas: Planejamento, aquisição dos dados, pré-processamento, processamento e análise da precisão do modelo gerado.



**Figura 3 – Laser Scanner 3D da Faro. Modelo Foco 3D X 330. Fonte: [www.faro.com](http://www.faro.com)**

A finalidade do levantamento, os objetos a serem escaneados e o nível de detalhamento necessário são definidos na fase de planejamento. Na etapa de aquisição de dados, realiza-se a varredura a laser do objeto a partir de vários pontos, a fim de que diversas cenas sejam obtidas, evitando, assim, que nenhuma parte do objeto seja ignorada e reduzam-se ao máximo as zonas de oclusões. Ainda nesta fase, realiza-se o levantamento de pontos de controle<sup>5</sup> para que se possa fazer a associação da nuvem de pontos para um sistema de coordenadas e para o controle de resultados.

Na fase de pré-processamento, realiza-se o alinhamento das cenas. Este, basicamente, consiste em posicionar as várias cenas com suas coordenadas locais em um único arquivo com o mesmo referencial, podendo ser usado para isto o posicionamento do scanner sobre um ponto conhecido de coordenadas. Enquanto que na fase de processamento realizam-se normalmente as operações de segmentação, modelagem, otimização e edição do modelo, além da exportação dos produtos em diversos formatos.

Após o processamento, temos, finalmente, a fase de análise da precisão do modelo gerado, onde é possível sobrepor a nuvem de pontos a um modelo geométrico para que se possa verificar automaticamente sua precisão a partir do uso de diferentes cores que indicam os desvios da modelagem.

---

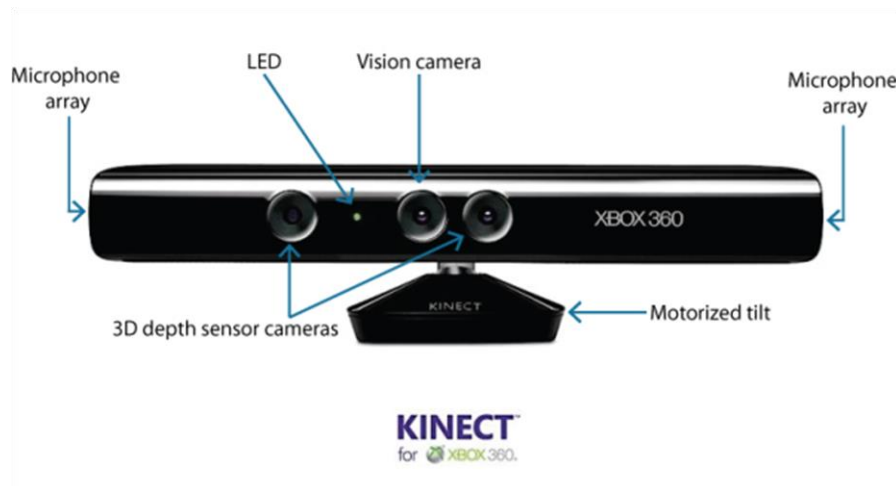
<sup>5</sup> Podem ser feições naturais do objeto ou alvos aplicados sobre a superfície do mesmo

### 2.3.2. Câmeras RGB-D

Câmeras de profundidade (RGB-D) são sistemas de sensores que capturam imagens RGB juntamente com as informações de profundidade de cada pixel (Henry, 2014), geralmente são baseadas em *time-to-flight* (ToF) ou detecção de luz estruturada (*structured light sensing*). O uso em massa destas câmeras pelo usuário comum surgiu com o advento do Kinect da Microsoft em 2010 que usa detecção de luz estruturada.

As informações obtidas por uma câmera RGB-D eram a base do que precisávamos para criar uma nuvem de pontos, pois ela nos informa a posição (x,y,z) de cada um dos pontos no espaço (tomando como referência a posição da câmera), bem como as componentes R, G e B destes pontos.

Com o passar do tempo, o kinect sofreu algumas melhorias e, em 2014, a Microsoft lançou o Kinect for Windows V2 sensor. Este possui resolução de 1080p, na captura RGB e 512x424 na captura de profundidade, com capacidade de captura de até 30 frames por segundo. Contudo, neste trabalho, não tivemos a oportunidade de utilizar o novo Kinect. Logo, a câmera RGB-D utilizada foi o sensor Kinect lançado em 2010 para utilização com o XBOX 360 (Fig. 4) que possui resolução de 640x480 a (câmera RGB) e 320x240 pixels (câmera de profundidade), ambas com capacidade de até 30 frames por segundo. Para utilizá-lo conectado ao computador, usamos um SDK (Software Development Kit) disponibilizado pela Microsoft para tal fim.



**Figura 4 – Kinect for Windows sensor lançado pela Microsoft em 2010. Fonte: ROBOTIQ**

Apesar das melhorias sofridas ao longo do tempo, ainda encontramos algumas dificuldades ao utilizar o Kinect para construir imagens de profundidade. Um dos problemas mais comuns é o aparecimento de buracos. Estes podem ocorrer pelos seguintes motivos: oclusão, limites de alcance de profundidade do sensor (0,8m – 4m), limite do ângulo do campo de visão (58,5° na horizontal e 45,6° na vertical), o fato de o material não refletir a luz infravermelha ou devido à incidência de luz em superfícies muito finas (Fig. 5). No caso de movimentação da câmera, pode haver perda de informação causada pelo motion blur (Fig. 6) produzido por um movimento rápido da mesma. No caso de se realizar trajetórias longas e complexas com a câmera ao redor de um objeto, podemos ter acúmulo de erros de registro.





Figura 5 – Imagem RGB à esquerda e informação de profundidade captada por uma câmera RGB-D à direita. Fonte: Henry et al. (2014)

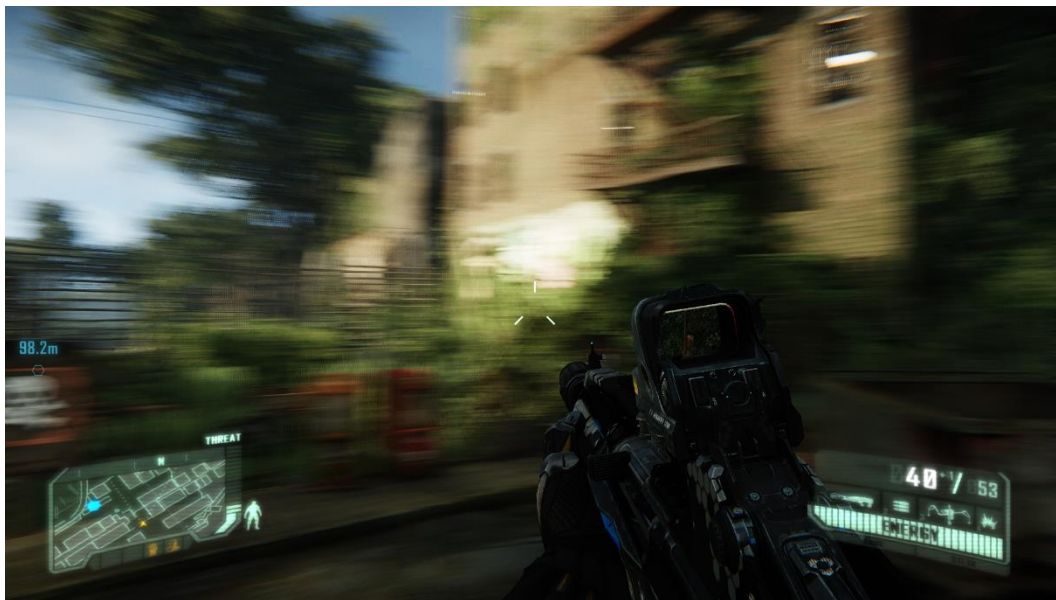


Figura 6 - Motion blur causado por movimento rápido do jogador no game Crysis 3. Fonte: TWEAKGUIDES.

#### 2.3.2.1. Cálculo de Profundidade

Apesar de câmeras a cores realizarem de forma eficiente a captura da informação visual, é extremamente difícil obtermos os dados de profundidade

a partir somente dos dados da câmera; principalmente quando se trata de ambientes internos muito escuros ou em áreas com pouca textura.

Já as Câmeras RGB-D possuem um sistema de som ativo ou sensores *time-of-flight*<sup>6</sup> para calcularem a profundidade estimada da maioria dos pixels. No caso do Kinect do Xbox 360, a câmera de *time-of-flight* emite sinais de luz pulsada e, em seguida, mede quanto tempo os sinais levam para voltar (Fig. 7). Com estas medidas, a câmera é capaz de diferenciar a luz que reflete dos objetos ao ambiente que o envolve. Isso fornece uma estimativa de profundidade mais precisa e permite que as formas dos objetos sejam calculadas.

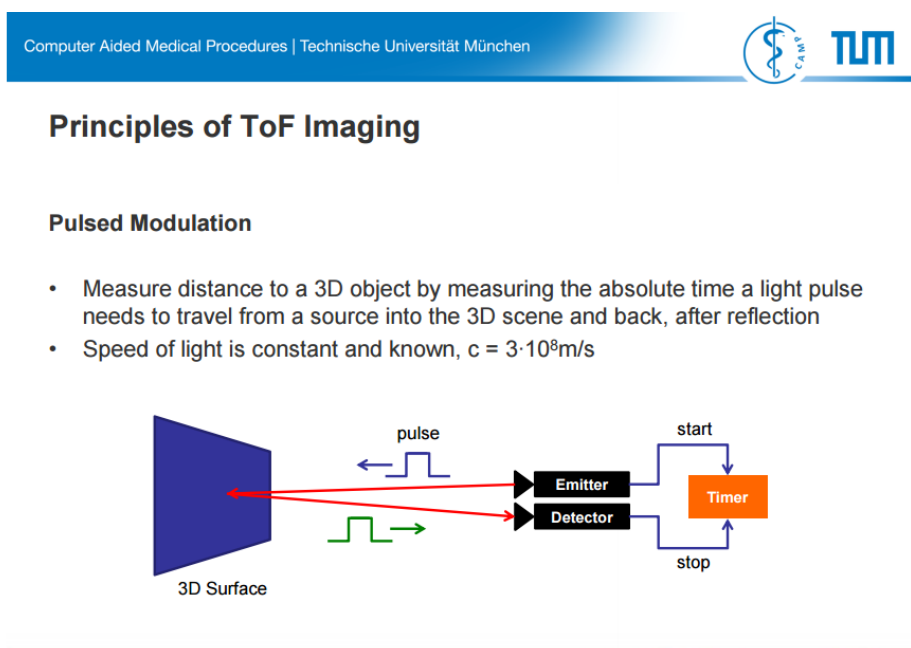


Figura 7 – Sistema Time-of-flight. Fonte: CASTANEDA and NAVAB (2011)

Apesar de este cálculo poder ser feito, as profundidades são apenas aproximadas. Isto ocorre devido ao fato de a câmera RGB-D possuir algumas restrições em relação ao mapeamento 3D: Elas somente fornecem profundidade até um certo limite de distância dos objetos que é de aproximadamente até 4m. Lembremos ainda que a profundidade estimada possui muito ruído e que o seu campo de visão é de aproximadamente 45,6° apenas.

<sup>6</sup> Time-of-flight (TOF) descreve uma variedade de métodos que medem o tempo que leva para um objeto ou partícula ou onda eletromagnética ou outra onda para percorrer uma distância através de um meio.

### 2.3.3. Dificuldades na Captura

Não é aconselhável realizar a mapeamento 3D de objetos sob sol forte e as informações dos dados gerados também não são muito confiáveis no caso de superfícies translúcidas ou espelhadas.

Ao utilizarmos sensores simples para rastrear cenas complexas, podemos encontrar, pelo menos, duas dificuldades. Uma delas é a falta de fidelidade de dados devido ao campo estreito de visão e ao erro de operação na faixa de poucos centímetros. Outra dificuldade tem a ver com a complexidade da trajetória da câmera pelo ambiente devido à captura dos detalhes.

Para que haja o mínimo de oclusões e o máximo de detalhamento, é necessário que muitos pontos sejam capturados, diminuindo ao máximo a média de erros entre os frames.

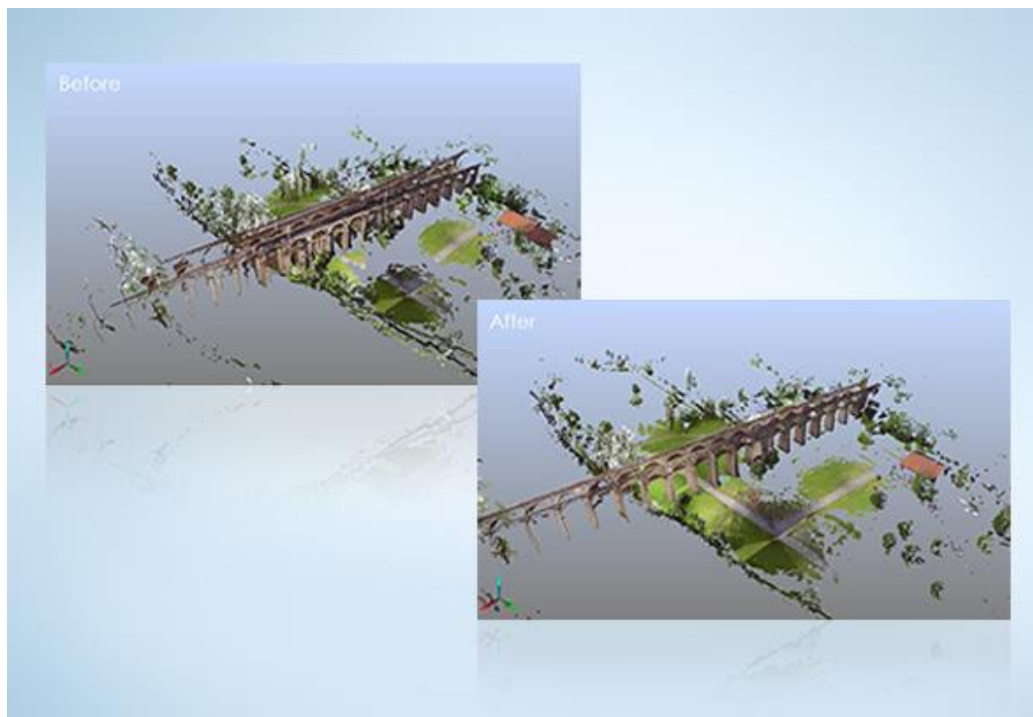
## 2.4. Softwares para VFX

Há vários softwares no mercado capazes de trabalhar com nuvem de pontos, seja para uso na indústria VFX ou não. Podemos citar, por exemplo, o Revit<sup>7</sup> da Autodesk, utilizado para realizar projetos de edificações e construção. O Revit utiliza nuvem de pontos para fornecer a condição de uma construção ou terreno. Por ser um software de CAD, ele possui grandes vantagens ao utilizar nuvem de pontos, pois esta pode ser exibida em várias vistas de modelagem, como uma vista 3D, plantas e cortes, sendo possível isolar seções da mesma, além do benefício de ela geralmente se comportar como um objeto de modelo dentro do Revit.

---

<sup>7</sup> Versão gratuita de avaliação disponível em <http://www.autodesk.com.br/products/revit-family/free-trial>

Outro software bastante útil para quem deseja trabalhar com nuvem de pontos é o Leica Cyclone da Leica Geosystems, pois facilita o gerenciamento eficiente dos dados da nuvem na base de dados. Assim, os usuários podem trabalhar simultaneamente na base de dados, reduzindo a necessidade de copiar e/ou transmitir grandes arquivos de projeto de nuvem de pontos. O Leica Cyclone é composto por módulos individuais que auxiliam na personalização e um deles, o Cyclone-IMPORTER possui o diferencial de permitir a importação direta de formatos de nuvem de pontos de diferentes fabricantes; bem diferente do software SCENE<sup>8</sup> da Faro que, apesar de gerenciar os dados digitalizados de forma fácil e eficiente usando o reconhecimento de objetos (Fig. 8), foi projetado para funcionar somente com o FARO Focus3D.



**Figura 8 – Nuvem de pontos melhorada pelo software SCENE da Faro. Fonte: [www.faro.com](http://www.faro.com)**

Um software que vem sendo amplamente utilizado pela indústria de VFX para criação de nuvem de pontos é o NUKEX da The Foundry<sup>9</sup>. Nele é

<sup>8</sup> <http://www.faro.com/en-us/products/faro-software/scene/overview>

<sup>9</sup> <https://www.thefoundry.co.uk/>

possível gerar uma nuvem extremamente densa através de dados de imagem utilizando o Nuke Point Cloud Generator. Um tutorial sobre como gerar esta nuvem de pontos através do NUKEX pode ser encontrado no site <http://lesterbanks.com/2013/03/using-3d-point-cloud-projections-and-reconcile-3d-nodes-for-compositing-in-nuke/>.

Podemos citar ainda o software HOUDINI da Side Effects Software<sup>10</sup> que também trabalha com nuvem de pontos. Ele utiliza a GPU para realizar o render, permitindo que o mesmo seja efetuado mais rapidamente.

#### **2.4.1. Krakatoa**

Neste trabalho, para se trabalhar com nuvem de pontos, foi utilizado o software Krakatoa MX desenvolvido pela Thinkbox<sup>11</sup>. Ele é um plugin que pode ser instalado no 3DS Max e possui capacidade de armazenar, transformar, sombrear e renderizar grandes quantidades de partículas volumétricas a uma velocidade suficiente para que se possa representar fenômenos naturais, tais como: poeira, fumaça, espuma da superfície do oceano, plasma e até mesmo objetos sólidos.

Os recursos de gerenciamento de partículas incluem a capacidade de gerar milhões de partículas, usando um método de particionamento único que supera as limitações de memória e sistema do 3ds Max, por isto, faz-se necessário o uso do Krakatoa devido à sua capacidade de armazenar um grande volume de partículas. Ele suporta entrada e saída de dados no formato PRT, ou seja, é totalmente compatível para ser utilizado com os módulos desenvolvidos.

O Krakatoa tem sido amplamente utilizado em grandes produções cinematográficas, como por exemplo, Lucy (2014) (Fig. 9) e Homem de Aço (2013).

---

<sup>10</sup> <http://www.sidefx.com/>

<sup>11</sup> <http://www.thinkboxsoftware.com/Krakatoa/>



Figura 9 – Imagem do filme Lucy (2014). Fonte: Site do Krakatoa<sup>12</sup>. Imagem com direito de autor copiada sob a política de *fair use*.

## 2.5. Dificuldades

É altamente desejável se trabalhar com nuvens densas de pontos, pois uma vez que temos bastante informação, mais fácil é para se realizar segmentação e combinação. Contudo, possuir um grande volume de dados também pode ser um problema que nem sempre é simples de ser resolvido.

Uma das dificuldades encontradas ao se trabalhar com nuvem de pontos é que por ser um ponto a menor medida geométrica, um simples objeto decomposto em nuvem de pontos possui uma quantidade imensa de informações. Logo, é necessário que o computador utilizado possua bastante memória RAM, placa de vídeo com memória dedicada e um bom processador. Caso contrário, uma tarefa que poderia ser realizada em poucos minutos, levaria horas para ser finalizada.

<sup>12</sup> Quadro do trailer disponível em <http://www.thinkboxsoftware.com/Krakatoa-in-production/>

### 3. Sistemas de Captura com múltiplas câmeras

Para a obtenção dos pontos, foi desenvolvido um software pelo Departamento de P&D em VFX da TV Globo em parceria com o laboratório ICAD/VisionLab, do qual faço parte, capaz de capturar as imagens através do sensor Kinect para, posteriormente, transformá-las em uma nuvem de pontos.

Como o sensor kinect possui duas câmeras (RGB e profundidade), o software é capaz de realizar uma sincronização entre estas câmeras a fim de que os pixels capturados por cada uma delas coincidam. Caso isto não ocorresse, a nuvem de pontos possuiria falhas, no sentido de que um pixel poderia ter sua cor incorreta apesar de a sua profundidade estar correta.

A fim de que uma cena pudesse ser visualizada por distintos ângulos, os pontos gerados através de um único kinect não seriam suficientes para tal feito. Portanto, o software realiza a captura a partir de vários kinects ao mesmo tempo e faz a interpolação dos dados capturados no momento em que a nuvem de pontos é gerada, ou seja, no caso de um mesmo pixel ser capturado por mais de um kinect, esta duplicidade é descartada. A nuvem de pontos é gerada pelo software em três formatos de arquivos diferentes: PRT, CSV e PLY. O formato CSV (Comma-separated values) é importante, pois permite que as informações dos pontos possam ser lidas e utilizadas mais facilmente por outro software, enquanto que o formato PLY (Polygon File Format) permite a visualização da nuvem em um software que lê arquivos de dados tridimensionais, como o MeshLab<sup>13</sup> por exemplo.

Para que fosse possível utilizar o Krakatoa MX para renderizar as partículas volumétricas e adicionar efeitos visuais, foi utilizado o arquivo gerado no formato PRT por ser suportado pelo software.

---

<sup>13</sup> <http://meshlab.sourceforge.net/>

### 3.1. Arquitetura

A Linguagem utilizada para a implementação dos módulos desenvolvidos que geram uma nuvem de pontos a partir de capturas realizadas por câmeras RGB-D foi C# com .NET versão 4.5.

Os módulos estão brevemente descritos nos subitens abaixo

#### 3.1.1. Modelagem

O software foi desenvolvido em 4 módulos diferentes (Conforme Requisito [RNF003] citado no próximo tópico). São eles:

##### 3.1.1.1. KinectCapture:

Realiza a captura de 1 kinect com áudio e canais de zdepth e imagem em cor. Adicionalmente ele grava os dados de rotação, timecode e parâmetros intrínsecos da câmera do mesmo para reprojeção.

Classes principais:

- Effect - Aplica efeitos nas imagens renderizadas
- RenderTarget - Renderiza as imagens processadas utilizando

GLSL

- ResourceManager - Classe Singleton do Gerente de Memória
- ResourceManagerCreator - Classe de Criação do Singleton do

ResourceManager para multithreads

- Texture - Realiza a representação de uma textura 2D RGBA
- VertexBuffer - Classe de representação de um Vertex Buffer

Object (VBO)



- Program - Classe com a função Main -> entrada de pontos para a aplicação
- mainform - Classe que constrói o formulário para a aplicação principal
- VertexTypes - Container de Estruturas de Representação de Vértices
- TransmitData - Classe aninhada que contém variáveis úteis para se realizar a transmissão de dados
- SaveData - Classe aninhada que contém variáveis para armazenamento de dados recebidos
- TransmitterMessage - Classe aninhada que contém propriedades para transmissão de mensagens

#### **3.1.1.2. KinectConvert:**

Grava parâmetros para conversões.

Classes principais:

- Program - Classe que possui a função Main -> "chama" a classe BaseConverter se houver arquivos a serem convertidos
- BaseConverter - Classe que gera os arquivos de decodificação para que seja possível a conversão de .png para .ply e .craw no Módulo Merge

#### **3.1.1.3. KinectCapture Sync:**

Sincroniza os disparos dos demais kinects.

Classes principais:

- mainForm - Classe que constrói o formulário para a aplicação de sincronização para que se possa utilizar mais de 1 kinect
- ClientServer - Classe aninhada que contém método e as variáveis necessárias para conexão do cliente (kinect) com o servidor

- Program - Classe com a função Main -> entrada de pontos para a aplicação

#### **3.1.1.4. KinectMerger:**

Realiza o trabalho de unificar as nuvens e converter os dados para formatos comerciais.

Classes principais:

- Program - Classe com a função Main -> entrada de pontos para a aplicação
- MainForm - Classe que constrói o formulário para a aplicação de conversão dos arquivos
- CRawConverter - Classe que carrega os arquivos decodificados para que a conversão dos arquivos possa ser feita corretamente

#### **3.1.2. Diagramas**

Logo abaixo, temos os diagramas de classe de cada um dos módulos citados, bem como os diagramas de casos de uso quanto utilizamos 1 ou mais sensores kinect.



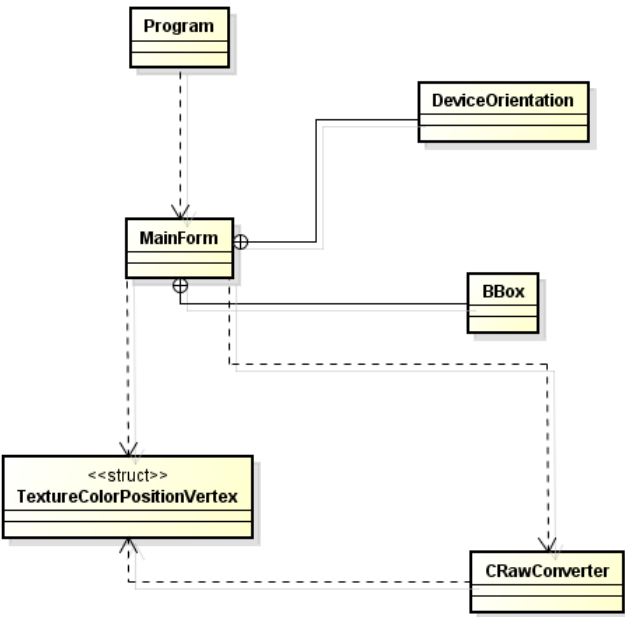


Figura 12 - Diagrama de classe do módulo KinectMerger

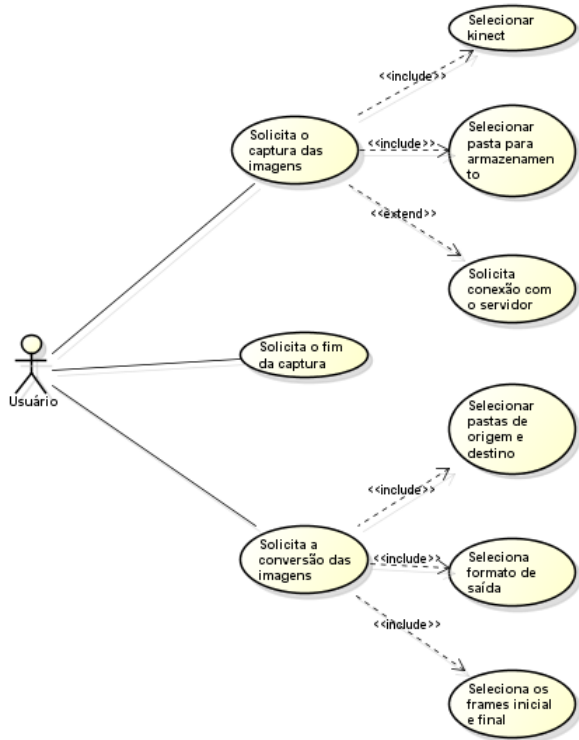


Figura 13 – Diagrama de caso de uso para 1 kinect

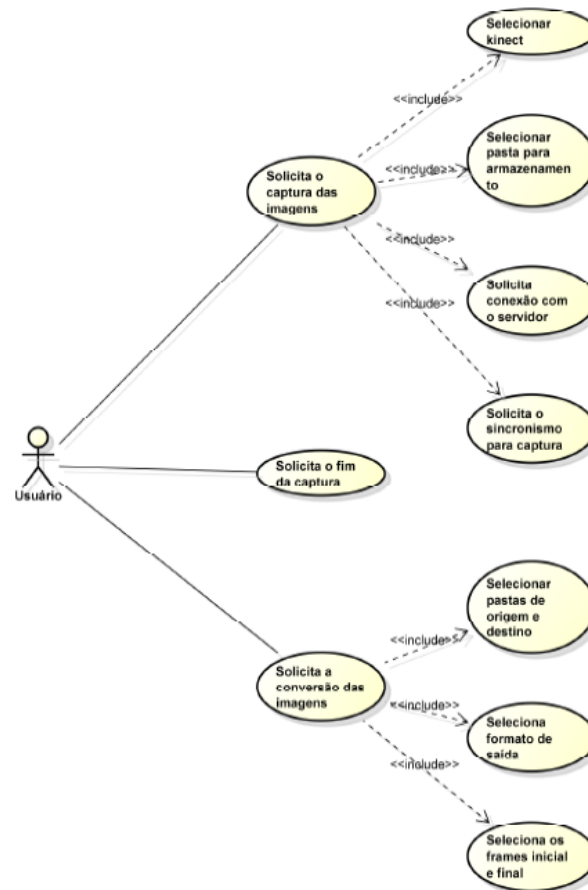


Figura 14 – Diagrama de caso de uso para mais de 1 kinect

## 3.2. Requisitos do Software

### 3.2.1. Funcionais

“Requisitos Funcionais são assuntos de importância fundamental ou essencial ao produto. Eles descrevem o que o produto tem de fazer ou que ações processuais deve tomar” ROBERTSON and ROBERTSON (2009).

[RF001]	O sistema deve utilizar kinect para captar imagens (entrada)
[RF002]	O Sistema deve armazenar no pc as imagens captadas pelo kinect
[RF003]	O sistema deve oferecer opção de local de destino para o cliente armazenar as imagens
[RF004]	O Sistema deve ser capaz de utilizar mais de um Kinect ao mesmo tempo
[RF005]	O Sistema deve ser capaz de iniciar e terminar a captura de vários Kinects ao mesmo tempo
[RF006]	O Sistema dever gerar nuvens de pontos a partir das imagens captadas
[RF007]	O Sistema dever converter as imagens em formatos comerciais (saída)
[RF008]	O sistema deve informar quando a conversão estiver finalizada
[RF009]	O Sistema dever fazer a conversão em qualquer momento, sem a necessidade de os Kinects estarem conectados
[RF010]	O sistema deve oferecer opção de local de destino para o cliente armazenar a nuvem de pontos

Tabela 1 – Requisitos funcionais do software de captura

### 3.2.2. Não-Funcionais

“Requisitos Não Funcionais são as propriedades que as funções devem ter, tais como desempenho e usabilidade” ROBERTSON and ROBERTSON (2009)..

[R NF001]	O sistema deve ser implementado em C#
[R NF002]	O sistema deve armazenar imagens no formato .png
[R NF003]	O sistema deve ser desenvolvido em diferentes módulos, para que a conversão possa ser feita em um momento diferente ao da captura
[R NF004]	O sistema deve converter as imagens para o formato .prt
[R NF005]	O sistema não deverá exigir mais do que cinco minutos para treinamento
[R NF006]	O tamanho do pacote de instalação do sistema deve ser inferior a 25MB
[R NF007]	O tamanho de cada arquivo de pontos deve ser inferior a 6MB, caso se utilize 2 kinects e inferior a 3MB, no caso de se utilizar somente 1 (formato .prt).
[R NF008]	A conversão dos arquivos deve ser inferior a 1h para 200 frames gerados.
[R NF009]	As imagens captadas devem possuir tamanho inferior a 1MB
[R NF010]	O sistema deve ser capaz de gravar 300 imagens em tempo inferior a 5min

Tabela 2 – Requisitos não-funcionais do software de captura

## 4. Tutorial e Resultados

### 4.1. Utilização do Krakatoa

Para se utilizar o software Krakatoa, primeiro é necessário entrar em contato com a empresa Thinkbox através do e-mail [sales@thinkboxsoftware.com](mailto:sales@thinkboxsoftware.com), e informar alguns dados. Após isto, eles enviam o link de download do software por e-mail juntamente com um arquivo de licença de 15 dias de validade, após este prazo, ainda será possível continuar utilizando o software desde que a resolução das imagens não ultrapasse de 480x320 pixels, caso contrário, haverá anúncio nas mesmas.

Neste estágio, utilizaremos o arquivo da nuvem de pontos gerada pelo software do capítulo anterior em formato .prt.

Após a instalação do Krakatoa no 3DS Max, cria-se uma nova cena, seleciona-se Krakatoa na lista suspensa em Create – Geometry e clicando em PRT Loader carregamos a nuvem de pontos (Fig. 16).

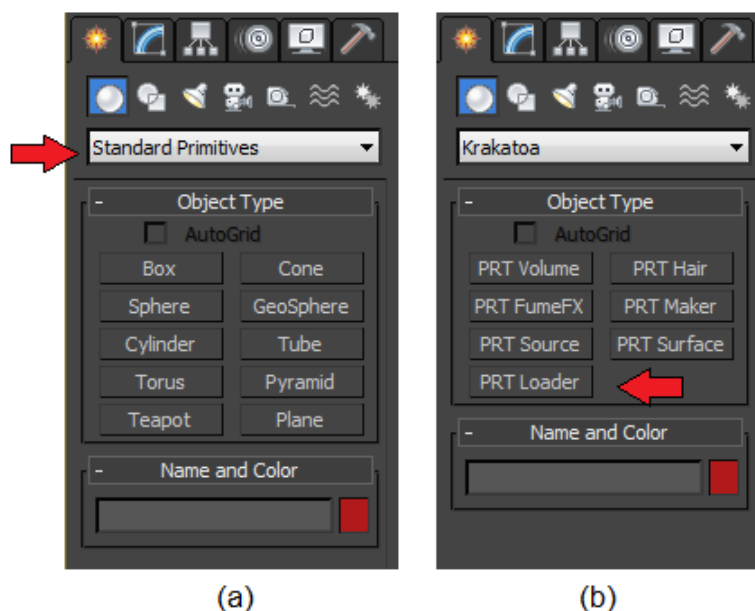
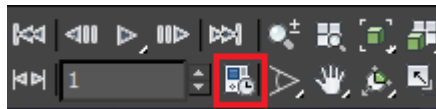


Figura 15 – Seleção do Krakatoa no 3DS (a) e botão para carregar nuvem de pontos (b)



É recomendável carregar a nuvem de pontos no centro do modelo. Ao se selecionar a nuvem, basta eleger um único arquivo da sequência e toda ela será carregada. Um segundo passo é ajustar a Time configuration do 3DS Max de acordo com os frames da sequência (Fig. 17).



**Figura 16 – Botão de configuração da timeline do 3DS**

É possível também ajustar a porcentagem dos pontos que se deseja visualizar renderizada na tela, esta pode ser escolhida de acordo com a capacidade da placa de vídeo do computador. Contudo, escolher 20% já é suficiente (Fig. 18).

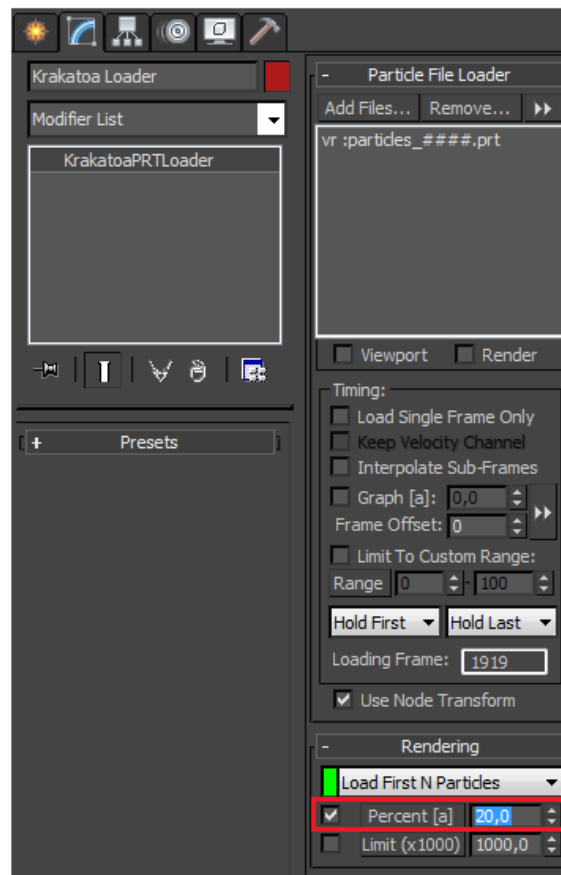


Figura 17 – Porcentagem de renderização das partículas

É importante adicionar uma câmera na cena e posicioná-la de modo que se esteja em seu campo visual somente o que se almeja renderizar. Lembramos que estamos trabalhando com o 3DS Max, um software onde se pode adicionar animação e podemos utilizar isto também na câmera, ou seja, podemos movê-la no decorrer dos frames e, assim, evitamos que renderizemos o que não é conveniente em um determinado frame, evitando gastos desnecessários. Para acionar a câmera e saber o que está em seu campo de visão, basta apertar a tecla **c**, caso queira voltar à perspectiva, basta teclar **p**.

A fim de que a nuvem de pontos renderizada possa ser visualizada, é imprescindível que se adicione luzes à cena. Normalmente, uma Target Spot já é suficiente.

Para que o Krakatoa seja utilizado para realizar o render, é necessário ajustar o renderizador (apertar F10) e rolar o rollout até o fim, abrir a aba Assign Renderer e selecionar Krakatoa no Production (Figs. 19 e 20).

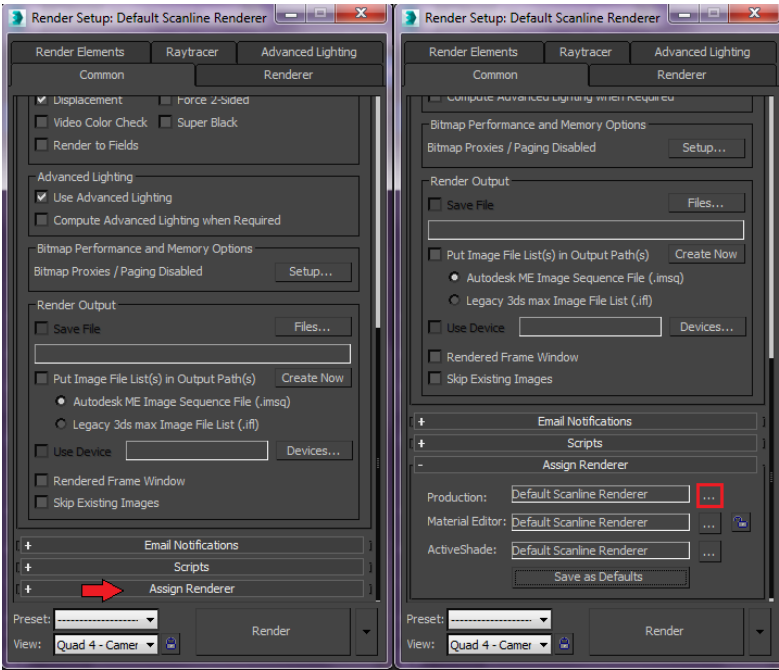


Figura 18 – Seleção do Krakatoa como renderer (Parte 1)

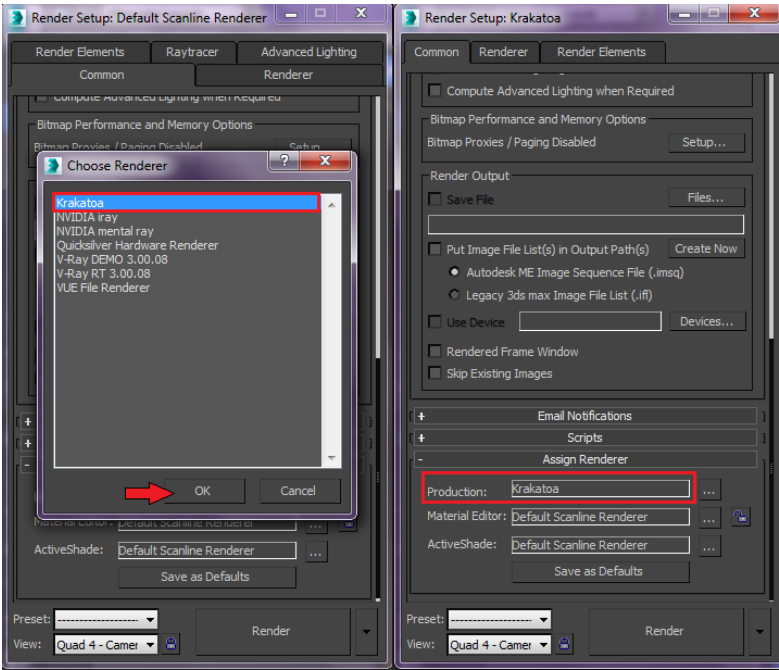


Figura 19 – Seleção do Krakatoa como renderer (Parte 2)

Após estes passos, basta indicar a faixa de frames que será renderizada, escolher o nome, o formato do arquivo de saída: .avi, .mov, .bmp, .jpg, .tiff, .exr, ... e o local onde o mesmo será salvo (Figs. 21 e 22).

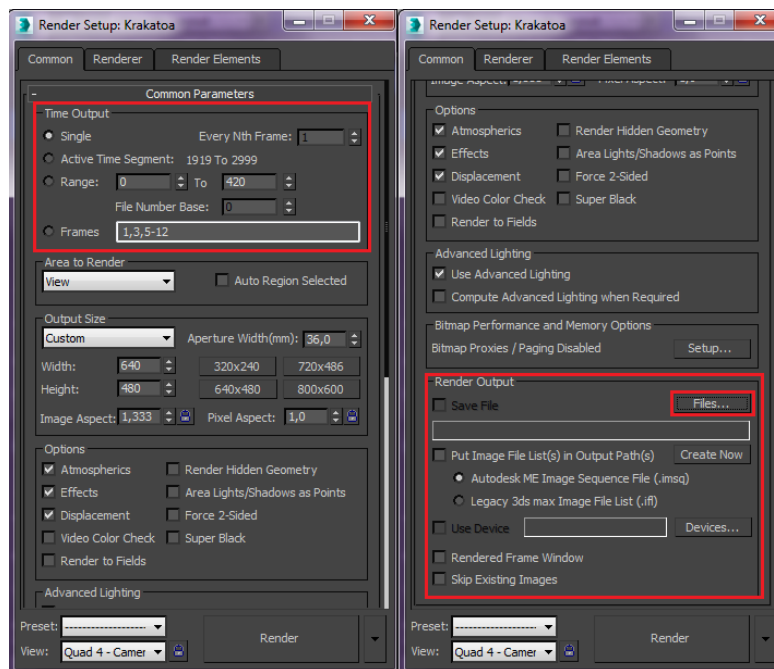


Figura 20 – Seleção de frames a serem renderizados e opção de saída dos arquivos

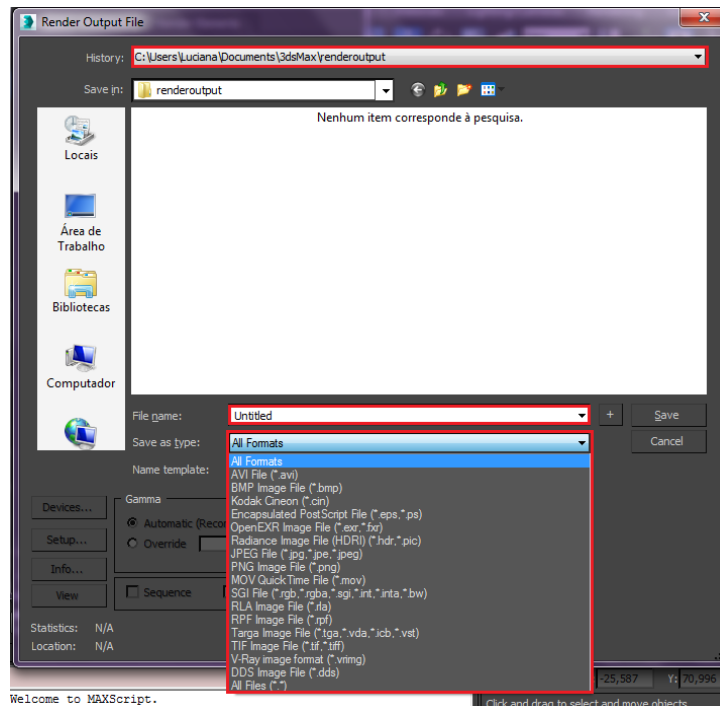


Figura 21 – Escolha de nome, tipo de arquivo e local para salvamento.

## 4.2. Resultados obtidos

Para os resultados, foi utilizada uma máquina com os seguintes requisitos:

- Sistema Operacional: Windows 7 Home Premium.
- Hardware: 6GB de RAM, Processador Intel Core i7-3612QM, placa AMD Radeon HD 7650M Series\Intel(R) HD Graphics 4000.

### 4.2.1. Resultado dos Módulos desenvolvidos

A figura. 15(a) foi capturada pela câmera RGB-D e a figura 15(b) foi obtida pela câmera de profundidade. Nesta imagem, é possível observar os pontinhos brancos que representam os raios infravermelhos emitidos pela câmera. A Figura 15(c) representa a nuvem de pontos gerada através dos módulos desenvolvidos. A mesma foi gerada no formato .ply, possui 15,1 MB e levou 3,9s para ser gerada.

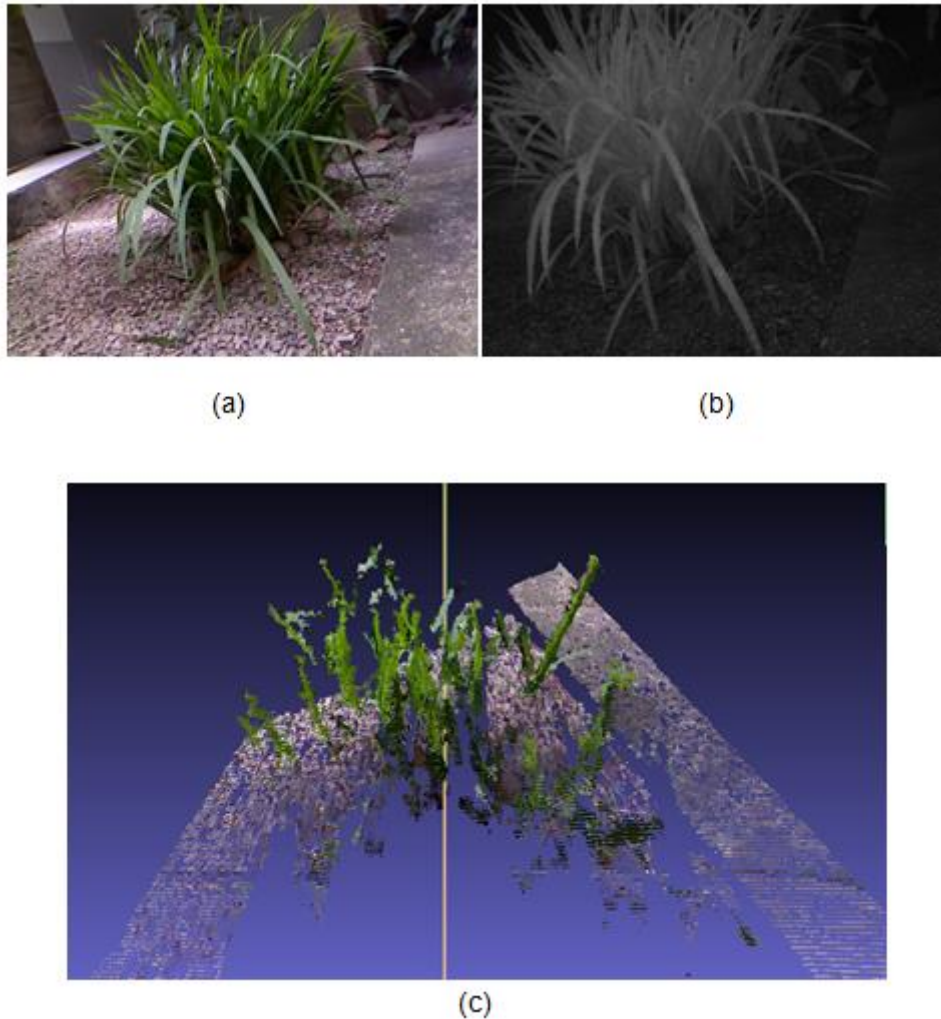


Figura 22 – Imagem original na escala RGB (a). Imagem obtida pela câmera de infravermelho (b). Nuvem de pontos gerada (c). Fonte: A autora.

#### 4.2.2. Resultados Finais

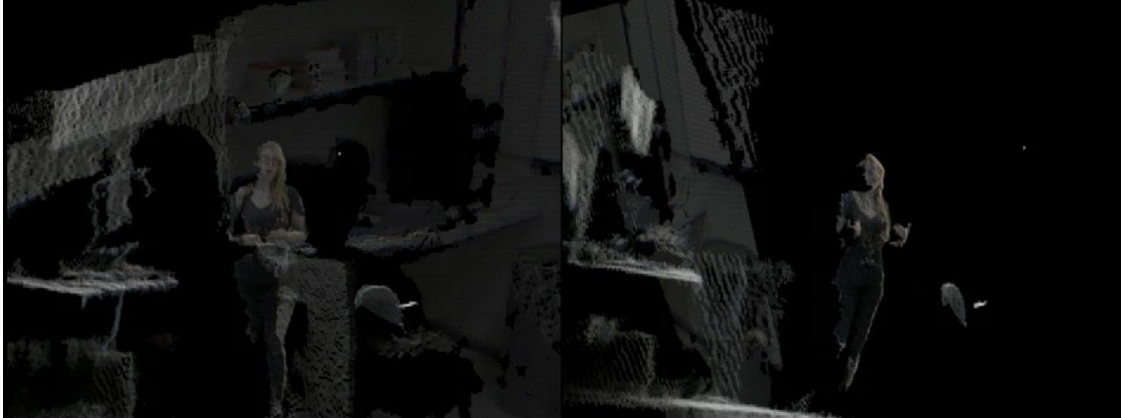
#### 4.2.2.1. Exemplo 1

Abaixo, seguem algumas imagens extraídas do vídeo realizado para exibição no canal Futura. O arquivo final foi salvo no formato .avi. Além do render da nuvem, foi incluída uma animação de giro de câmera (Figs. 23 e 24).

O vídeo está no formato .avi e foi gerado pelo 3DS Max, utilizando o Krakatoa como render a uma taxa de 30fps. Possui 44,4 MB, 720 x 480 p, 388 frames e levou 20min para ser gerado.



**Figura 23 – Imagem extraída do vídeo realizado para exibição no canal Futura. Cenário é o do laboratorio ICAD/VisionLab.**



**Figura 24 – Outras imagens do mesmo vídeo para o canal Futura que comprovam a animação de câmera. Cenário é o do laboratório ICAD/VisionLab.**

#### **4.2.2.2. Exemplo 2**

As imagens abaixo foram retiradas de outro vídeo de uma nuvem de pontos renderizada. O arquivo inicial foi gerado a partir de uma planta filmada. Desta vez, foi incluída uma animação de câmera acrescida de uma animação simples onde o eixo z tem seu valor diminuído até chegar a zero (Figs. 25 e 26). O vídeo está no formato .avi e foi gerado pelo 3DS Max, utilizando o Krakatoa como render a uma taxa de 30fps. Possui 12,8 MB, 720 x 480 p, 111 frames e levou 7min para ser gerado.



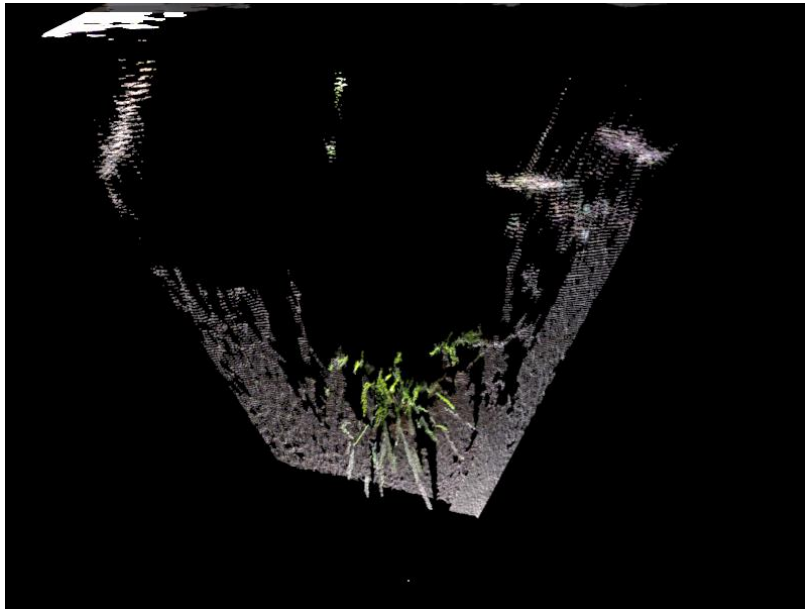


Figura 25 – Imagem do vídeo no início da animação

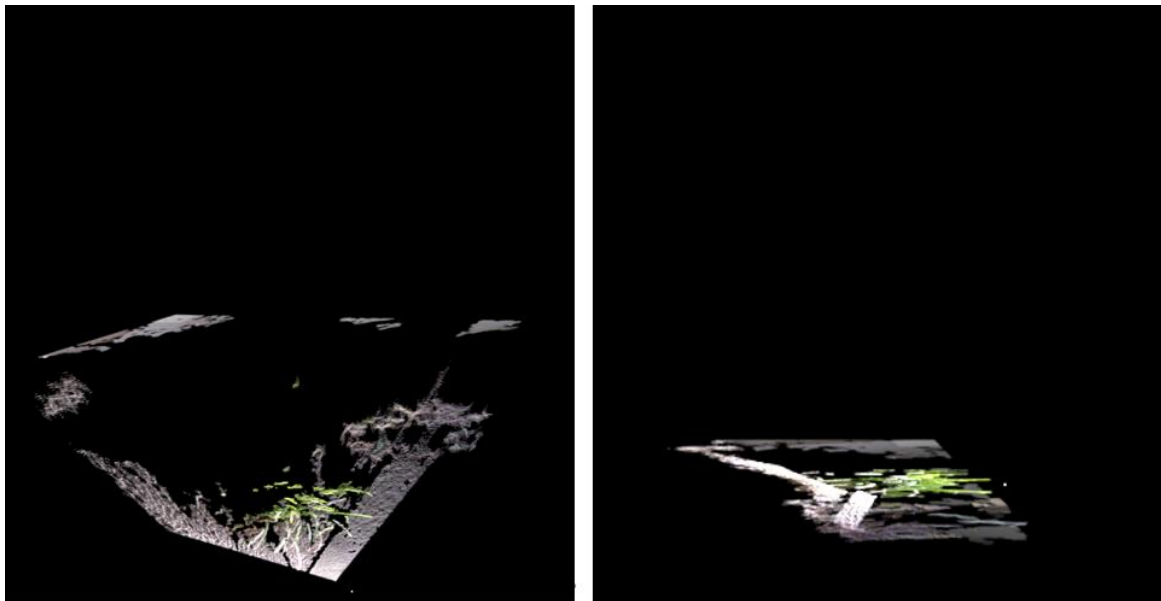


Figura 26 – Imagens intermediária e final ( $z=0$ ) da animação respectivamente

#### 4.2.2.3. Exemplo 3

Este exemplo contém imagens cedidas pela rede Globo de televisão. São imagens de uma atriz em uma cena que foi capturada por dois kinects, transformada em nuvem de pontos e acrescida de animação de câmera (Figs. 27 e 28).

O vídeo está no formato .avi e foi gerado pelo 3DS Max, utilizando o Krakatoa como render a uma taxa de 30fps. Possui 123 MB, 720 x 480 p, 1081 frames e levou 30min para ser gerado.



Figura 27 – Nuvem de pontos renderizada extraída de vídeo. Imagem com direito de autor copiada sob a política de *fair use*.



Figura 28 – Nuvem de pontos renderizada extraída de vídeo. Imagem com direito de autor copiada sob a política de *fair use*.

#### 4.2.2.4. Exemplo 4

Este exemplo contém tiras de imagens extraídas de um vídeo onde é aplicado o efeito bend em uma nuvem de pontos renderizada. O arquivo inicial também foi gerado a partir de uma planta filmada (Fig. 29).

O vídeo está no formato .avi e foi gerado pelo 3DS Max, utilizando o Kraktoa como render a uma taxa de 30fps. Possui 11,5 MB, 720 x 480 p, 98 frames e levou 7min para ser gerado.

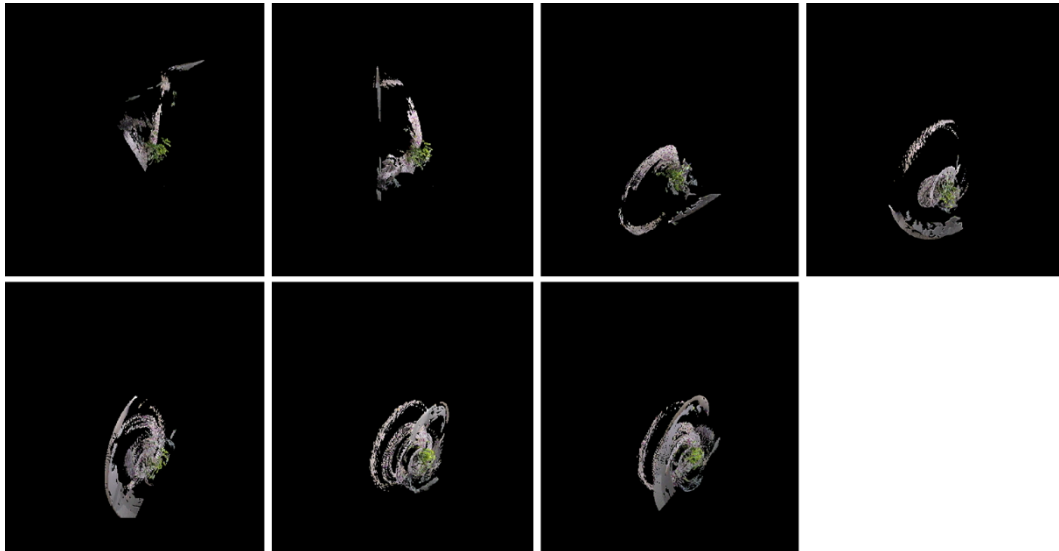


Figura 29 – Efeito bend aplicado a uma nuvem de pontos

#### 4.2.2.5. Exemplo 5

Este exemplo contém tiras de imagens extraídas de um vídeo onde são aplicados os efeitos bend + twister em uma nuvem de pontos renderizada. O arquivo inicial também foi gerado a partir da planta filmada (Fig. 30).

Estas animações também podem ser aplicadas para simularem um efeito de redemoinho do Saci em um programa infantil de TV, por exemplo.

O vídeo está no formato .avi e foi gerado pelo 3DS Max, utilizando o Krakatoa como render a uma taxa de 30fps. Possui 11,5 MB, 720 x 480 p, 98 frames e levou 7min para ser gerado.

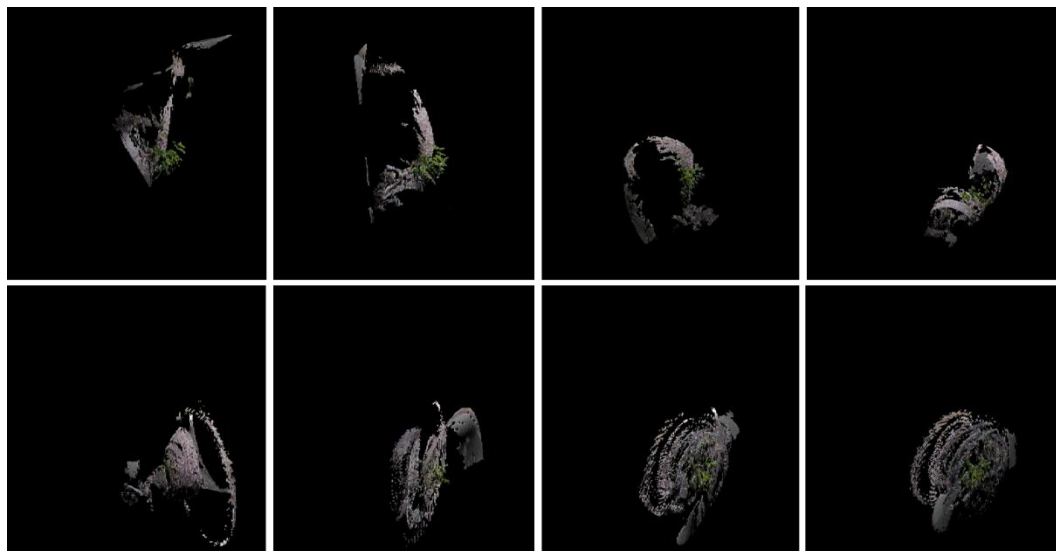


Figura 30 - Efeito bend + twist aplicado a uma nuvem de pontos

## 5. Considerações Finais

### 5.1. Conclusões

Nesta dissertação, fizemos um estudo sobre a geração de efeitos especiais visuais (VFX) a partir da captura de objetos em movimento como nuvens muito densas de pontos utilizando múltiplas câmeras RGB-D de baixo custo. Implementamos um protótipo e testamos com até duas câmeras. Os resultados mostraram a viabilidade de uso de câmeras RGB-D para gerar cenas que um diretor possa testar várias trajetórias de câmera e aplicar efeitos nos objetos capturados.

Apesar de os efeitos gerados apresentados neste trabalho terem sido poucos devido ao tempo disponível para tal e à deficiência de material encontrado, o que foi aqui apresentado pode ser utilizado como base para novos experimentos, conforme está escrito na seção de Trabalhos Futuros.

Ao utilizarmos o 3DS Max, podemos incluir várias animações combinadas utilizando como base a nuvem de pontos gerada pelo software do capítulo 3. Além disso, ao utilizarmos vários Kinects para captura, teremos uma nuvem de pontos mais densa que, apesar de exigir maior configuração de máquina, aumenta a qualidade da nuvem. À medida que utilizamos mais câmeras, obtemos melhores efeitos visuais.

Infelizmente, ainda enfrentamos um problema de resolução das câmeras RGB-D. Estas câmeras, apesar de terem evoluído ao longo dos últimos anos, geram imagens que não apresentam resolução desejável para exibição em TV. Contudo, se tivermos capturas a partir de ângulos diferentes (Kinects colocados em pontos estratégicos realizando a captura ao mesmo tempo), teremos uma nuvem bem densa e, somando-se a isso as possibilidades do 3 DS Max, teremos um arquivo final com boa qualidade.

Contudo, é importante deixar registrado que quanto menos Kinects forem utilizados, menos ruídos encontraremos. Mas no caso de ser necessário

utilizar dois Kinects ou mais, um algoritmo de interpolação pode melhorar os ruídos. Poderá ser como valores ajustados por regressão (linear regression).

Outro problema sério que enfrentamos é em relação ao volume e tamanho dos dados. Apesar de termos obtido êxito em conseguir gerar o arquivo de nuvem em formato .prt que é, em média, sete vezes menor que o .csv, ainda temos arquivos com 2,2 MB para cada ponto de uma nuvem, o que é muito se considerarmos a quantidade de pontos necessária para uma nuvem com qualidade.

Uma opção que temos para diminuirmos este problema é que podemos utilizar diversos computadores para realizarem a captura (um Kinect em cada computador diferente), concentrando somente em uma máquina os arquivos nos formatos aceitáveis para entrada no Krakatoa

Ainda em relação aos dados, outro ponto a ser considerado é o desempenho da máquina necessário para se realizar as capturas e renderizações. Quanto mais núcleos a máquina tiver, mais rapidamente as tarefas serão concluídas.

## 5.2. Trabalhos Futuros

Como trabalhamos com grande volume de dados que precisam trafegar pela rede local, seria importante realizar como trabalho futuro uma pesquisa sobre como aumentar a velocidade da rede local para este grande número de informações, sendo necessário talvez avaliar a necessidade de se realizar alteração no protocolo.

Ainda sobre o grande volume de dados, sugerimos realizar uma pesquisa acerca de como armazená-lo sem a necessidade de se realizar compressões.

Uma outra sugestão de trabalho futuro seria utilizar o novo Kinect e um número maior de câmeras, pois nos testes que fizemos foram empregadas no máximo duas. E, conforme foi citado na seção **2.3.2.**, o novo Kinect possui um maior campo de visão, além de alcances mínimo e máximo superiores quando comparado ao Kinect que foi utilizado.

Outra sugestão seria testar efeitos com partículas, conforme foi feito no personagem do filme Elysium (Figura 2), ou mesmo transformações em outros tipos de objetos.

Outra sugestão seria pesquisar para utilizar as informações da Normal, pois quando a mesma está perpendicular há maior perda de informação.

Por fim, um outro trabalho seria combinar a captura de objetos em movimento com o escaneamento em alta resolução do cenário com scanners a laser.



## 6. Referências Bibliográficas

BRITO, Paulo Miguel Franco Correia de. 2009. Acessibilidade - Emulação do rato através de movimentos faciais e voz -Mobility Camera. 67f. Dissertação (Mestrado em Informática) – Escola de Engenharia, UMINHO.

CASTANEDA, Victor and NAVAB, Nassir. 2011. Time-of-Flight and Kinect Imaging. Computer Aided Medical Procedures.

FARO. 2013. Laser Scanner Focus 3D – Faro. [Online]. Disponível em [www2.faro.com/site/resources/share/944](http://www2.faro.com/site/resources/share/944) [Acessado em 13 de outubro de 2014].

GOIS, João Paulo. 2004. Reconstrução de superfícies a partir de nuvens de pontos. 131f. Dissertação (Mestrado em Ciências da computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, USP.

GROETELAARS, Natalie Johanna and DE AMORIM, Arivaldo Leão. 2011. Tecnologia 3D Laser Scanning: características, processos e ferramentas para manipulação de nuvens de pontos. SIGraDi 2011. 490-494.

HENRY, P. et al. 2014. RGB-D Mapping: Using Depth Cameras for Dense 3D Modeling of Indoor Environments. In: HENRY, P., KRAININ, M., HERBST, E., REN, X., FOX, D. (eds). Experimental Robotics. Springer Tracts in Advanced Robotics v. 79. University of Washington, Seattle: Springer Berlin Heidelberg.

KATZ, S. et al. 2007. Direct Visibility of Point Sets. Transactions on Graphics (TOG) - SIGGRAPH 2007 Conference Proceedings - 26(3).

LEICA GEOSYSTEMS. 2015. Leica Cyclone - 3D Point Cloud Processing Software. [Online]. Disponível em [http://hds.leica-geosystems.com/en/Leica-Cyclone\\_6515.htm](http://hds.leica-geosystems.com/en/Leica-Cyclone_6515.htm) [Acessado em 12 de abril de 2015].

MELO, Fagner Daniel. 2005. Contornos Ativos Snakes para a Segmentação de Imagens Digitais. 84p. Monografia de Graduação. Faculdade de Informática Presidente Prudente: UNOESTE.

MICROSOFT. 2014. Kinect for Windows features. [Online]. Disponível em: <http://www.microsoft.com/en-us/kinectforwindows/meetkinect/features.aspx> [Acessado em: 25 de agosto de 2014].

NEWCOMBE, R. A. et al. 2011. KinectFusion: Real-Time Dense Surface Mapping and Tracking. Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality, p. 127-136. IEEE Computer Society Washington, DC, USA.

PCL. 2015. What is PCL? [Online]. Disponível em <http://pointclouds.org/about/> [Acessado em 19 de junho de 2015].

ROBERTSON, James & ROBERTSON, Suzanne. 2009. Modelo para Especificações de Requisitos. Volere – Requirements Resources.

ROBOTIQ. Using the Kinect for Robotic Manipulation. 2014. [Online]. Disponível em <http://blog.robotiq.com/bid/40428/Using-The-Kinect-For-Robotic-Manipulation> [Acessado em 12 de junho de 2015].

RYCHARD, Małgorzata Jarząbek. 2010. Implementation of 3d Point Clouds Segmentation Based on Plane Growing Method. JUNIORSTAV 2010.

THE OFFICIAL MICROSOFT BLOG. 2013. Collaboration, expertise produce enhanced sensing in Xbox One. [Online]. Disponível em <http://blogs.microsoft.com/blog/2013/10/02/collaboration-expertise-produce-enhanced-sensing-in-xbox-one/> [Acessado em 20 de junho de 2015].

THE URBAN CANUK. 2014. Comparing Kinect v1 and v2 Depth Data. [Online]. Disponível em <http://www.bryancook.net/2014/02/comparing-kinect-v1-and-v2-depth-data.html> [Acessado em: 12 de outubro de 2014].

THINKBOX SOFTWARE. 2014. Krakatoa MX Overview. [Online]. Disponível em: <http://www.thinkboxsoftware.com/kmx-overview/> [Acessado em: 30 de agosto de 2014].

ZHOU, Qian-Yi and KOLTUN, Vladlen. 2013. Dense Scene Reconstruction with Points of Interest. ACM Transactions on Graphics (TOG) - SIGGRAPH 2013 Conference Proceedings - 32(4).