Captura Completa de Avatares com uma Única Câmera RGBD

Marcos Vinícius Lenz Balatka* Matheus Ramos Fernandes da Silva Gabriel Caixeta Silva
Marcelo da Silva Hounsell André Tavares da Silva

Universidade do Estado de Santa Catarina (UDESC), Departamento de Ciência da Computação (DCC), Brasil

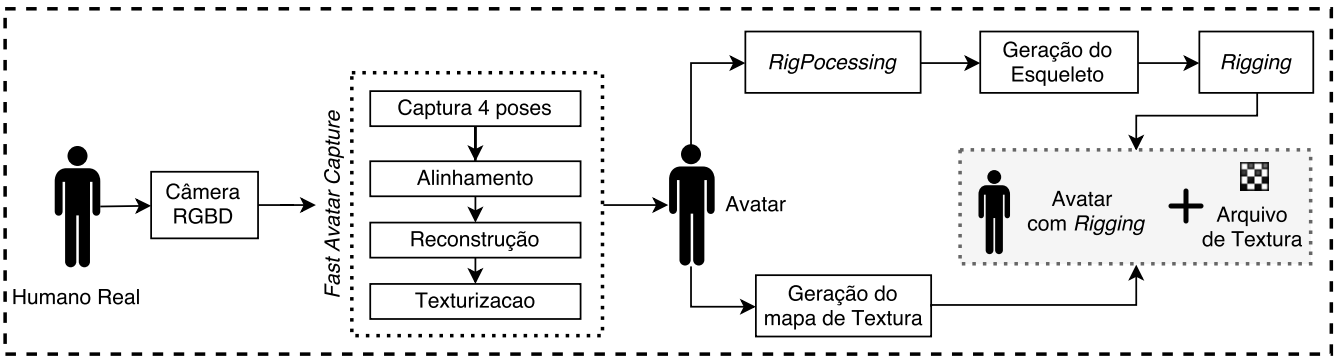

Figura 1: Processo do sKanner3D: captura, texturização e *rigging* automático.

RESUMO

Humanos virtuais capturados utilizando câmeras RGBD de baixo custo podem ser utilizados em diversas aplicações como animação, simulação e jogos, dentre outras. Existem *softwares* que realizam a captura de modelos tridimensionais, textura e *rigging* (associação de um esqueleto à malha geométrica), porém normalmente não contemplam todas estas tarefas, são de difícil integração (devido aos formatos de arquivos) ou requerem equipamentos complexos e caros. Este artigo detalha estes processos e compara as ferramentas encontradas que se propõe a realizá-los. Ao final, este artigo apresenta um método, chamado sKanner3D, que é um processo automático de captura que gera uma malha com mapa de textura e *rigging*, de modo rápido, fácil e integrado, utilizando apenas uma única câmera Microsoft Kinect v1.0. O sKanner3D se mostra robusto quando a pessoa assume uma determinada posição, está devidamente trajada e em ambiente bem iluminado. Os modelos capturados podem ser imediatamente manipulados e animados, sendo adequados para prototipação e análises, antes de se investir numa solução de maior precisão e detalhamento.

Keywords: captura 3D, reconstrução 3D, *rigging*, modelagem geométrica.

1 INTRODUÇÃO

Modelos geométricos 3D capturados (escaneados) de objetos reais têm inúmeras aplicações na indústria cinematográfica, de jogos [17], moda [16] e saúde [32], principalmente quando o modelo é uma pessoa. Modelos 3D de pessoas podem ser desenvolvidos por ferramentas de modelagem interativa, ou reconstrução baseadas em fotos 2D, vídeos ou tecnologias a laser [18]. Estes modelos podem ser usados como substitutos de "pessoas reais" em testes ergométricos baseados em computador [20].

*e-mail: m.balatka5@hotmail.com

Modelos de humanos virtuais vem sendo largamente utilizados em aplicações de computação gráfica, como realidade virtual, simulações e jogos [28], possuindo uma série de usos como:

- Populações virtuais para aprendizado e treinamento baseado em simulações, desenvolvimento de habilidades, coordenação de times e tomadas de decisão [18];
- Usuários virtuais para análise ergonômica em ambientes virtuais de trabalho e veículos [18];
- Pacientes virtuais para os diversos tipos de cirurgias [18];
- Manequins virtuais para a indústria têxtil [18];
- Atores virtuais para filmes, seja personagens principais ou figurantes, e apresentadores virtuais para a TV e web [18];
- Análise de diferenças entre humanos que relaciona um conjunto de características que são as variantes do indivíduo [20];
- Simulação biomédica em que o ser humano é uma estrutura complexa tanto em termos físicos quanto funcionais [20];
- Análise de forma e movimento: o entendimento do que se vê e sente no mundo real deve ser transportado até o modelo do mundo virtual. Com formas geométricas e deformações de objetos mais próximas da realidade [20].

Por fim, capturar humanos é uma tarefa de capturar pontos no espaço, alinhá-los e construir a malha do modelo 3D, fechando espaços não capturados por oclusão, de forma que fique realmente parecido com a "pessoa real" capturada. Mas também deve gerar um esqueleto para o modelo 3D a fim de produzir avatares deformáveis para que sejam utilizados na animação. Um humano virtual deve promover a junção de um esqueleto à malha geométrica e texturização. Estas etapas podem levar uma grande quantidade de tempo para um artista, situação que piora quando é preciso criar

vários e distintos humanos virtuais [4]. Geralmente, devido à complexidade dos corpos humanos, formas realistas são construídas por um exaustivo e longo processo de design gráfico.

Além, de requererem a presença de grupos ou multidões de personagens, uma diversidade de formas e tipos de animações são necessárias para popular realisticamente um ambiente virtual [28]. Como consequência, os artistas envolvidos devem criar manualmente cada um destes personagens, aumentando a complexidade de sua tarefa e exigindo muito tempo e talento artístico [30]. Desta forma, o objetivo desta pesquisa é mostrar um processo de captura e reconstrução de um humano virtual com *rigging* e textura, a partir da captura de uma pessoa real por meio de uma única câmera RGBD.

Este artigo está estruturado da seguinte forma: na seção 2 são apresentados conceitos sobre captura e reconstrução de modelos tridimensionais, esqueleto e textura; na seção 3 são apresentados alguns trabalhos correlatos; na seção 4 são apresentados algumas ferramentas correlatas para captura e *rigging*; na seção 5 é apresentado o processo do sKanner3D como um processo automatizado; na seção 6 são mostrados os resultados e realizada uma discussão sobre o processo; e, por fim, na seção 7 é apresentada a conclusão.

2 Conceitos básicos

O presente artigo enfoca três temáticas: captura 3D, textura e *rigging*. A seguir são aprofundados esses conceitos com o enfoque na geração de personagens utilizando escâneres.

2.1 Captura

Escaneamento de corpos humanos 3D está se tornando uma tecnologia madura que gera modelos estáticos e foto-realistas de humanos reais. Porém, os dados coletados permitem apenas a reconstrução da superfície externa do corpo, uma vez que os modelos escaneados não possuem nenhuma estrutura interna ou propriedades físicas a respeito de esqueleto, pele ou camada de gordura do humano escaneado [21].

Um escâner 3D é um dispositivo que analisa um objeto do mundo real e coleta dados de suas formas e possivelmente cor. Estes dispositivos são usados pela indústria de entretenimento na produção de personagens virtuais para filmes e jogos. Esse dispositivo concebe formas individualizadas com um realismo visual através de texturas de alta qualidade [28].

Escâneres 3D convencionais são equipamentos caros, como por exemplo, os modelos vendidos pela Artec3D[1] onde os preços variam de US$ 9.800,00 a US$ 25.800. Estes realizam a captura de pontos do modelo, depois passam por um processo de alinhamento e reconstrução a fim de transformar a nuvem de pontos obtidas pelo escâner em um modelo geométrico. Neste processo, alguns espaços vazios criados pela oclusão de algumas partes do corpo, como topo da cabeça e sola do pé, terão que ser preenchidos [27].

Com o barateamento e a popularização das câmeras de profundidade, também conhecidas como câmeras RGBD (*Depth Sensor*), como é o caso dos dispositivos *Microsoft Kinect*[2], *RealSense*[3], *StructureSensor*[4], e *Asus Xtion*[5], existe a possibilidade de escanear objetos do mundo real e possui como vantagens: ser mais baratos que os escâneres 3D convencionais, como por exemplo o *Microsoft Kinect* por ser encontrado por R$200,00; tem mobilidade no transporte; e devido ao seu baixo custo muitos pesquisadores e praticantes de ciência da computação e outras áreas tem desenvolvido diversas maneiras de interagir com este tipo de dispositivo [33]. E como desvantagem, tem uma menor resolução de captura comparado aos escâneres tradicionais.

[1] https://www.artec3d.com/

[2] https://developer.microsoft.com/pt-br/windows/kinect/

[3] https://software.intel.com/pt-br/realsense/home

[4] https://structure.io/

[5] https://www.asus.com/3D-Sensor/

2.2 Textura

Textura pode ser definido como algo genérico para diferenciar objetos ou, em computação gráfica, como uma imagem projetada para um espaço multidimensional, sendo a textura aplicada a um objeto por um processo de mapeamento [11]. Este mapeamento é tradicionalmente utilizado para adicionar realismo em imagens de computação gráfica [10], sendo uma técnica popular para adicionar detalhes visuais contidos em uma textura para renderizar modelos geométricos 3D [15].

A texturização de um modelo pode ser realizada por duas abordagens [12]: a primeira é a abordagem da colorização de vértice (*vertex coloring*) que vincula uma cor para cada vértice do modelo. Quando renderizado, a cor de um ponto da face de um polígono é resultante da interpolação da cor dos vértices que estão ao entorno e que geralmente conduz a uma aparência visual borrada; A segunda é por mapeamento de textura (*texture-mapping*), em que os vértices são atribuídos a uma única coordenada de textura que se refere a um local específico de uma imagem colorida. Quando renderizado, as coordenadas da textura para os pontos da malha geométrica entre as localizações dos vértices são interpolados um a um, e as cores são lidas diretamente destes locais do mapa de textura associado. A vantagem do mapeamento de textura sobre a coloração de vértice é a possibilidade de aumentar a resolução da cor independentemente da geometria do modelo e consequentemente um menor custo computacional. A Figura 2 mostra à esquerda um malha geométrica sem textura, no centro a texturização por coloração de vértice, à direita por mapeamento de textura.

Figura 2: Comparação entre entre coloração de vértice e mapeamento de textura [12]

O mapeamento de textura é uma maneira relativamente eficiente para criar características mais complexas na aparência do modelo, sem a necessidade de modelar e renderizar cada detalhe 3D de uma superfície, devido a estas últimas serem tarefas tediosas e demoradas [11].

2.3 *Rigging*

Um esqueleto pode ser definido como um sistema hierárquico de objetos representado por ossos e suas juntas. Onde um esqueleto hierárquico de um humano virtual pode ser definido por uma estrutura na qual o início é a pélvis, os nodos filhos são as partes adjacentes e as partes extremas são os pés, mãos e cabeça. Neste tipo de esqueleto o movimento de um osso irá afetar também seus nodos filhos, por exemplo, o movimento do osso da coxa vai alterar a posição da panturrilha e do pé [27].

O esqueleto é extraído como uma estrutura em árvore com segmentos unidos nos extremos. Sua segmentação é a forma de representar a estrutura do corpo, onde cada segmento do esqueleto corresponde a uma parte da malha original [14]. A Figura 3 mostra alguns exemplos de esqueletos: o da esquerda um esqueleto capturado pelo dispositivo *Microsoft Kinect*; à direita um esqueleto gerado pela ferramenta Blender[6].

Para animação de humanos virtuais é utilizado um esqueleto, que possui ligações entre as juntas, e este é encoberto por superfícies

[6] https://www.blender.org/

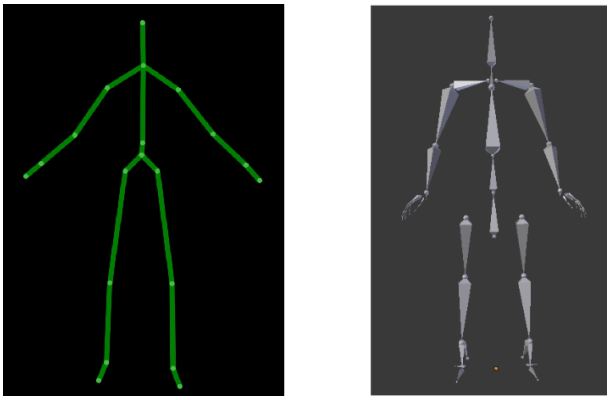

Figura 3: Exemplos de esqueletos

deformáveis para a modelagem da pele. Esta estrutura chamada *rigging* permite realizar transformações e deformações no modelo [20]. Assim, o *rigging* pode ser definido como um processo de vincular cada polígono da malha de um humano virtual para corresponder aos ossos do esqueleto. A vinculação é controlada por um sistema de pesos, em que cada polígono é vinculado a um ou mais ossos com um dado peso que determina a influência de cada osso sobre um determinado polígono. Dessa forma, cada polígono "conhece" quais ossos o influência e o quanto [3]. Também um determinado vértice da malha pode depender de mais de uma junta do esqueleto [24].

O *rigging* é uma técnica para criar um controle que permite um artista manipular um modelo 3D para gerar movimentos em um personagem 3D, realizando uma ligação entre o modelo e o processo de animação [23].

3 Trabalhos Relacionados

Através de uma pesquisa bibliográfica da literatura, foi possível identificar alguns trabalhos que utilizam câmeras RGBD para o processo de captura de modelos de pessoas e outros com foco no *rigging*.

3.1 Múltiplos dispositivos

Sistema para capturar modelos humanos 3D de corpo inteiro usando múltiplos *Kinects* [29] evitam ruídos ao utilizar dois *Kinects* para capturar as partes superior e inferior do corpo e um terceiro *Kinect* é usado para a capturar a parte central na direção oposta dos demais. Inicialmente, um gabarito de malha poligonal é usado para deformar a geometria através do registro dos vértices em quadros sucessivos. Posteriormente, um alinhamento global é realizado a fim de resolver eficientemente o problema de oclusão. Os resultados experimentais mostram a aplicabilidade e eficiência do sistema proposto utilizando multiplos dispositivos RGBD, obtendo resultados "impressionantes", segundo os autores, em poucos minutos através de dispositivos de baixo custo.

Vários *Kinects* são utilizados para capturar objetos em movimento [1], visando a utilização dos modelos 3D reconstruídos em aplicações de tempo real. Além da captura de modelos existem também trabalhos focados na reconstrução de uma malha a partir de nuvens de pontos, capturadas por escâneres [6].

3.2 Único dispositivo

Outro método é a reconstrução de formas humanas baseadas em nuvem de pontos e utilizando a silhueta obtida pelo sensor RGBD [31]. Neste método a estimativa precisa da forma 3D é obtida pela combinação de múltiplas vistas de uma mesma pessoa se movendo em frente do sensor. Também é possível utilizar um método para minimizar a distância entre o contorno da projeção 3D do corpo e a silhueta da imagem através de uma função objetiva. Baseada em formas humanas capturadas e dados obtidos por escâneres 3D, os modelos também necessitam estar sem roupa.

Utilizando um *hardware* de baixo custo é possível capturar modelos geométricos tridimensionais com apenas um dispositivo [26]. A aquisição do modelo, neste caso, requer quatro poses estáticas em um ângulo de noventa graus em relação às outras. A malha gerada pode ser utilizada em ambientes de simulação.

Também utilizando apenas uma câmera RGBD [19] apresenta-se um método de captura que utiliza dezoito quadros de seis poses estáticas e então as alinha localmente e em seguida globalmente.

3.3 Sistema de *rigging*

Já para gerar o *rigging* de um modelo capturado é possível utilizar uma base de modelos 3D com esqueleto através de um método de transferência de atributos [7]. Neste processo é realizado a identificação de um modelo da base de dados que mais se aproxima do modelo escaneado. Depois é realizado a transferência do esqueleto para o modelo escaneado realizando ajustes para que fique bem encaixado.

Um método chamado *Pinocchio* [2] tentou reduzir o esforço manual de fazer o processo de *rigging*, este método possui resultados razoáveis e requer uma malha conectada e completamente fechada para trabalhar.

Outra técnica para automaticamente criar e animar modelos obtidos de dados escaneados de todo corpo humano, desenvolve uma camada no modelo sob o esqueleto, simplificando a superfície e mapeando-a para a estrutura do esqueleto gerado sem intervenção manual [22]. O algorítimo utiliza um conjunto de marcadores na superfície, que normalmente é provido por programa de escaneamento, para gerar o esqueleto. Este esqueleto, entretanto, é sempre criado em um *template* fixo, e não permite criar um esqueleto customizado, que deveria preencher melhor as necessidades do usuário. O processo de mapear os ossos para a pele, é realizado dividindo a camada da malha da superfície e aplicando uma técnica de crescimento de superfície, utilizando uma função de distância para gerar os pesos na superfície.

3.4 Discussão da Sessão

A Figura 4 mostra uma classificação para os sistemas de captura apresentadas neste artigo que podem conter os componentes de texturização e *rigging*. Esta classificação permite uma visão mais abrangente da área mostrando suas contribuições e lacunas. Os trabalhos identificados mostram que a captura pode ser feita utilizando escâneres 3D ou câmeras RGBD e estes podem utilizar utilizam vários dispositivos *kinects* [29][1] ou apenas um [31][26][19] tratando apenas o processo de captura de um modelo 3D, que por sua vez podem ser texturizados ou não, porém nenhum trata da geração de um mapa de textura. Também identificou-se que os trabalhos realizavam apenas o *rigging* [7] e não eram englobados no contexto da captura, mas como processo isolado. Assim, este trabalho visa gerar uma solução completa envolvendo captura, esqueleto, *rigging* e textura mapeada, com uma única câmera RGBD.

4 Análise de Ferramentas

O impacto do *Kinect* foi além da indústria de jogos. Com o barateamento e sua popularização, muitos pesquisadores e praticantes de ciência da computação, engenharia elétrica e robótica estão desenvolvendo novos meios criativos para interagir com máquinas usando o *Kinect*. A *Microsoft* chama isso de "Efeito *Kinect*" [33].

A fim de analisar as ferramentas existentes na questão da captura e *rigging* automatizado será mostrado um levantamento de softwares que utilizam o *Kinect* para escaneamento, geração de malha, texturização e, se possível, *rigging* automático. Ao todo foram encontrados sete softwares para a captura e dois para o *rigging*, e to-

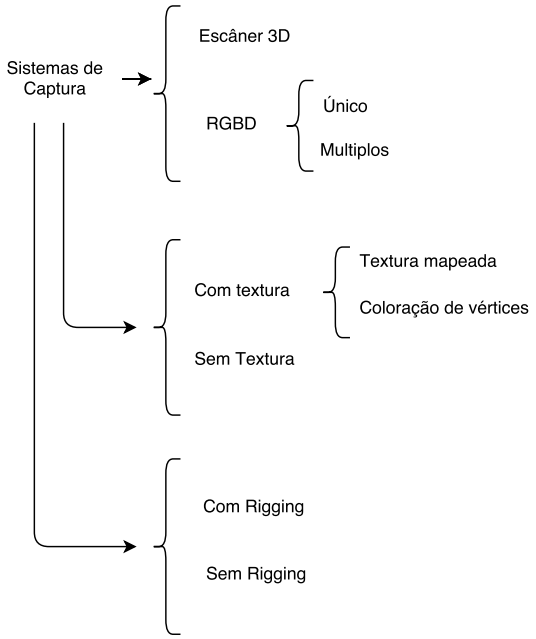

Figura 4: Classificação dos sistemas de captura

dos utilizando o *Software Development Kit* (SDK) da *Microsoft*[7] ou *Open Natural Interaction (OpenNI)*[8] da *Prime Sense*. Alguns desses *softwares* possuem integração entre si ou com *softwares* de modelagem, ou motores gráficos como Unity[9] e outros. As subseções a seguir serão relacionados a cada programa individualmente, como foram selecionados e suas especificações.

4.1 *Kinect Studio v1.8.0 (Kinect for Windows)*

Em 2012, a *Microsoft* disponibilizou um pacote de ferramentas chamado *Kinect Software Development Kit (SDK)*, contendo diferentes aplicações utilizando o dispositivo para diferentes fins [33]. As aplicações estão implementadas em C# e C++.

Cada aplicação trabalha com informações especificas conseguidas através do *Kinect* como: cores; profundidade; segmentação e esqueleto. Podem ser utilizadas para geração de malha, texturização e animação. O *software* não é aberto, não oferecendo acesso a códigos fonte e afins, foi criado para demonstrar como o dispositivo pode ser utilizado e permite que usuários utilizem as aplicações para desenvolvimento próprio.

4.2 *NITE – OpenNI*

Desenvolvido pela *PrimeSense*, o *OpenNI*, até antes de sua versão 2, era um *framework* de código aberto (suas antigas versões podem ser encontradas na internet), desenvolvido para trabalhar com sensores 3D em aplicações. A *Apple* comprou a *PrimeSense*, descontinuando o *OpenNI* e fechando o site do *framework*[10]. Felizmente, os códigos das ultimas versões encontram-se disponíveis no Github (acesível em https://github.com/PrimeSense/Sensor).

O NITE é um *middleware* usado pela interface *OpenNI*, desenvolvido pela *PrimeSense*. Apesar de possuir código fechado, é gratuito e pode ser usado comercialmente. Ele é responsável por identificar os usuários na imagem e rastrear seus movimentos, além de prover uma API que detecta gestos. Possui dois modos básicos de operação: um permite rastrear mãos, com detecção de gestos em particular, e outro permite rastrear o corpo todo (esqueleto), dando informações sobre as principais juntas do corpo [9]. *OpenNI* é multiplataforma, suportando múltiplas linguagens de programação para a escrita de aplicações que utilizam interação natural, utilizando vozes ou gestos interagindo com sistemas computacionais [25].

4.3 *Smart Body – Virtual Human ToolKit (VHTK)*

SmartBody[11] é uma plataforma ou biblioteca de animação de personagens desenvolvido na Universidade do Sul da Califórnia (USC). O *software* possui ferramentas para se trabalhar com personagens (humanoides) 3D em tempo real, tais como locomoção, sincronização labial, expressão facial e emoção, animação, gestos, *rigging* automático, entre outros.

Possui integração com os motores de jogos *Unity*, *Ogre*, Panda 3D e *Irrlicht*, podendo ser compilado em *Windows*, *Linux* e *OSx*, bem como possui portabilidade para *Android*, *IOS* e *Flash*. É escrito na linguagem de programação C++ e pode ser controlado utilizando a linguagem de programação Python. Possui licença para uso comercial apenas, sendo o código acessível e podendo ser utilizado para uso acadêmico e não profissional. *SmartBody* faz parte de um *software* maior feito utilizando *Unity 3D* chamado *Virtual Human ToolKit* (VHTK)[12], que também pode ser usado para fins não comerciais. Ao utilizar o *Kinect*, o programa pode trabalhar utilizando tanto o *SDK* da *Microsoft* quanto o da *PrimeSense (OpenNI)*, variando em algumas pastas específicas, devido ao fato do *OpenNI* não funcionar se instalado junto com o *SDK Kinect For Windows*. O VHTK é uma coleção de módulos, ferramentas, e bibliotecas designadas para auxílio e suporte em pesquisa e desenvolvimento de humanos virtuais.

4.4 *Shapify*

O software *Shapify*[13] é um programa licenciado de escaneamento para geração de bonecos 3D feitos em impressoras 3D, por um determinado preço dependendo da qualidade do boneco e do dispositivo utilizado, e outros fatores. O escaneamento pode ser feito por múltiplas câmeras, dando um resultado melhor ou feito em casa usando câmeras de baixo custo, como o *Kinect*.

Por se tratar de um programa fechado e sem uma forma de extrair o modelo gerado, o *software* acaba servindo apenas para mostrar que esta etapa de geração do personagem com textura é possível, de maneira rápida e eficiente. O uso da tecnologia por trás do *Shapify* pode ser encontrado em [13].

4.5 *Skanect*

Skanect[14] é um programa de fácil utilização, compatível com *Kinect* e outros dispositivos de baixo custo, e conseguindo gerar modelos rapidamente, permitindo que o usuário modifique o modelo da forma que desejar. Também permite colorir o modelo usando as cores conseguidas através do dispositivo ou através de outro *software*, podendo ser instalado em *Windows* e *Mac*. Para o primeiro deve-se ter instalado o kit de ferramentas do *Kinect SDK for Windows*, enquanto o segundo deve usar a última versão do *OpenNI*.

O programa gera os mapeamentos de textura em .STL e .OBJ, permite edição da malha em outros softwares e permite escanear pessoas, objetos e ambientes. O *Skanect* é um *software* fechado, mas sua versão gratuita permite ao usuário exportar o modelo 3D gerado, desde que não ultrapasse 5.000 (cinco mil) polígonos, e a sua utilização não deve visar fins lucrativos, a menos que se possua a licença.

[7] https://www.microsoft.com/en-us/download/details.aspx?id=40276
[8] https://github.com/PrimeSense/Sensor
[9] https://unity3d.com/
[10] https://www.theguardian.com/technology/2013/nov/25/why-did-apple-buy-primesense-for-a-key-technology-itll-deploy-within-a-year
[11] http://smartbody.ict.usc.edu/
[12] https://vhtoolkit.ict.usc.edu/
[13] https://www.shapify.me/
[14] http://skanect.occipital.com/

4.6 *ReconstructMe*

ReconstructMe[15] é um *software* de fácil uso para escaneamento, em tempo real, de um objeto, pessoa ou ambiente fechado e geração de malha tridimensional texturizada através de uma câmera RGBD, é licenciado mas possibilita uso para teste, permitindo a exportação do modelo em STL, 3DS, PLY e OBJ, o modelo gerado vem com uma marca d'água do *software* no caso de uso não licenciado.

O programa gera os modelos em nuvem de pontos (STL) ou malha 3D (OBJ), texturizadas, permitindo a edição em programas de modelagem e outros, porém o arquivo gerado pelo programa, sem estar licenciado, gera também objetos aleatórios. O programa permite também gerar modelos para impressão 3D e reconhece outros sensores além do *Microsoft Kinect*.

4.7 *Fast Avatar Capture*

Fast Avatar Capture[16] é um *software* desenvolvido na Universidade da Califórnia do Sul que escaneia um pessoa em quatro poses diferentes e então reconstrói a malha unindo os pontos de todas as poses [26]. A reconstrução é feita em várias etapas: captura, alinhamento das nuvens de pontos, reconstrução da malha e aplicação da textura. Na fase de captura são salvas imagens de cada pose, as quais são utilizadas após a reconstrução da malha para colorir os vértices.

O programa possui resultados bons e exporta o resultado para o formato de aquivo .PLY. Possui código fonte fechado mas é gratuito para a utilização em pesquisas, já para fins comerciais é necessário entrar em contato com o autor.

4.8 *Pinocchio*

Pinocchio[17] faz apenas o *rigging* de uma malha fechada, possui código fonte aberto, porém não exporta o resultado para um formato que possa ser lido em *softwares* de edição, como o *Blender*.

4.9 *RadiantRigXL*

Assim como o *Pinocchio* o *RadiantRigXL*[18] cuida apenas da parte de *rigging* da malha. O *software* é um *plugin* para o *Blender* e possui código fonte aberto. Pode ser utilizado dois modelos de esqueleto presentes no *plugin Rigify*, um com detalhamento no rosto e outro sem.

4.10 Discussão da Sessão

Como a ferramenta *Shapify* possui código fonte proprietário e não existe uma forma de exportar o modelo capturado é a ferramenta mais limitada entre as testadas. O *Kinect Studio*, possui vários exemplos de capacidades usando o dispositivo. Este não permite mudanças no código fonte, mas é livre para utilizar em quaisquer aplicações. O escaneamento deve ser realizado lentamente, com a câmera um pouco distante do alvo, o objeto é escaneado e a malha é gerada em tempo real, permitindo exportá-la para os formatos de arquivo .STL e .OBJ. Também possui outra aplicação que consegue as cores do alvo escaneado mas não é possível aplicá-la na malha gerada. Para fazer isto deve-se integrar as duas aplicações.

Sem uma ferramenta de captura não é possível escanear usando apenas o *SmartBody*, porém este possui integração com outras ferramentas, como os motores de jogos *Unity3D* e *Ogre*, e consegue realizar o *rigging* automático de modelos humanoides. Porém o resultado obtido pelo *SmartBody* não pode ser aberto no *Blender*, *software*, este, que será usado no restante do processo de *rigging* e geração do mapa de textura, o que impede seu uso.

O *Skanect* para *Windows* necessita do *SDK Kinect*, possui uma interface amigável e gera a malha 3D com cor, podendo editá-la no próprio *Skanect* ou em outros programas. Este consegue adquirir a malha e a texturização da mesma. Porém, pela restrição da quantidade de polígonos (cinco mil por modelo) para uso não pago, o *Skanect* gera modelos muito simples sendo impossível o reconhecimento de pessoas nos modelos 3D gerados. O *software* é fácil de instalar e possui uma interface simples, na qual é possível regular medidas de altura, largura e profundidade para o escaneamento, enquanto se configura a ferramenta é apresentada uma prévia do que está sendo captado pela câmera RGBD.

O escaneamento é rápido, ou seja, depois de configurada a área de captura, o programa aguarda alguns segundos e então inicia o processo, só termina quando o usuário apertar o botão para finalizar. Para maior precisão da ferramenta, é recomendado que alguém mova o dispositivo ao redor do alvo, seja ele objeto, pessoa ou cena, mas é possível que o mesmo (pessoa ou objeto) gire 360º para capturar todos os lados e reconstruir o objeto tridimensional. O resultado, como malha 3D, é mostrado no próprio software após encerrada a captura, mostrando também quantos vértices e triângulos o modelo gerado possui.

O *Fast Avatar Capture* possui bons resultados, porém não faz o *rigging* automático. Também possui código fonte proprietário, o que impossibilita modificações no *software* e integração total com outros *softwares*. E tem como vantagens a exportação do modelo para um formato conhecido e modelo texturizado.

Já os *softwares* para o *rigging* automático: o *Pinocchio* apresenta bons resultados porém não pode ser utilizado por não exportar o esqueleto em um formato conhecido; o *RadiantRigXL* parecia uma solução boa para a geração do *rigging* porém gerou resultados ruins em algumas das malhas testadas.

Após a análise das ferramentas, chegou-se a seis critérios de avaliação, como pode ser observado na Tabela 1. O primeiro critério de avaliação é se o *software* possui código aberto, pois neste caso é possível estudá-lo e alterá-lo de acordo com alguma necessidade. Algumas das ferramentas que não possuem código aberto permitem utilizar suas aplicações de diferentes formas, mas sem alteração livre.

O segundo critério avalia se é gerado o modelo 3D ou uma nuvem de pontos após o escaneamento e com quantos polígonos/pontos. No caso de ser gerado, é avaliado se o modelo contém muitos erros ou buracos e se é possível reconhecer a forma humana da pessoa escaneada, ou seja, se não houve muitas deformações no modelo geométrico. Entendem-se por erros em modelos 3D, além de buracos na malha, vértices duplicados, intersecção de faces, faces com número de lados diferentes entre si, entre outros problemas.

O terceiro critério verifica se a ferramenta, após o escaneamento, permite a edição do modelo 3D gerado. O quarto critério analisa se o modelo ou nuvem de pontos gerado já possui informação de cor, ou seja, já tem como saída a malha com textura, uma vez que com o dispositivo *Kinect* é possível, durante o escaneamento, conseguir informações de cores RGB. O quinto critério, verifica se a ferramenta permite a exportação para formatos conhecidos e que são utilizados pela maioria das ferramentas de modelagem ou motores de jogos como o OBJ, DAE, FBX, entre outros. Por fim, é avaliado o critério de *rigging* automático. Ou seja, se a ferramenta avaliada consegue incorporar um esqueleto ao modelo geométrico capturado.

A análise destes critérios aplicados às ferramentas podem ser visto na Tabela 1.

Os únicos *softwares* encontrados que possuem código aberto não conseguem fazer o escaneamento e gerar o modelo. *SmartBody*, que não faz o escaneamento permite a edição de um modelo já pronto dentro do próprio *software* e exporta nos formatos desejados, já que o objetivo principal é trabalhar com animações e sincronização labial de personagens humanoides. O *Shapify* realiza o escaneamento, gera o modelo 3D texturizado e detalhado, porém não permite a edição e nem exportação do modelo 3D. O

[15] http://reconstructme.net/

[16] http://smartbody.ict.usc.edu/fast-avatar-capture-software-download

[17] http://www.mit.edu/ ibaran/autorig/pinocchio.html

[18] https://4colorgrafix.net/bpy/

	Código aberto	Geração de malha 3D / nuvem de pontos	Modelo editável	Texturização	Exportação em formatos conhecidos	Esqueleto (*Rigging*)
KINECT for Windows	✗	✓	✓	✓	✓	✓
Fast Avatar Capture	✗	✓	✓	✓	✓	✗
SmartBody	✓	✗	✓	✗	✓	✓
Shapify	✗	✓	✗	✓	✗	✗
SKANECT	✗	✓	✓	✓	✓	✗
ReconstructMe	✗	✓	✓	✓	✓	✗
RadiantRigXL	✓	✗	✗	✗	✓	✓
Pinocchio	✓	✗	✗	✗	✗	✓

Tabela 1: Tabela de análise das ferramentas

Skanect, em sua versão gratuita, permite a exportação de um modelo muito simples, com restrição de cinco mil triângulos e apresenta uma perda significativa de cores e falhas na textura, limitando muito a malha, sendo impossível o reconhecimento da pessoa escaneada após o processo. O *ReconstructMe* e o *Kinect Studio* conseguem gerar o modelo 3D ou nuvem de pontos, exportar nos formatos requeridos, adquirir informações de cores, gerar textura, além de permitir edição. Porém, em relação ao *rigging*, estes possuem um esqueleto muito simplificado e código fonte proprietário. O *Fast Avatar Capture* não possui *rigging* automático. Já o *Pinocchio* e o *RadiantRigXL* apenas fazem o *rigging* do modelo. Portanto, nenhuma das soluções encontradas consegue isoladamente ou de forma combinada executar todo o processo de captura do modelo 3D com textura e *rigging*.

5 sKanner3D: captura completa

O processo do sKanner3D pode ser observado na Figura 1 e este possui vários requisitos, dos quais se destacam:

- Utilizar apenas um único *Kinect* parado: assim o sKanner3D se torna mais acessível e qualquer pessoa com um *Kinect* pode utilizá-lo;
- *Software* livre/gratuito: todos os *softwares* utilizados durante o processo devem ser livres/gratuitos, para que ao final o sKanner3D também seja livre;
- Textura mapeada: a maioria dos *softwares* de modelagem como o *Blender* não utilizam cor no ponto em seu renderizador, apenas textura mapeada. O que se deseja é criar uma imagem do mapa de textura do modelo para a renderização e maior detalhamento;
- Encontrar o posicionamento das juntas do esqueleto na malha escaneada, gerar o esqueleto e realizar o *rigging* de forma automática;
- Sem necessidade de Internet, para tornar o processo *offline*;
- Todo o processo deve ser realizado de modo automatizado.

O sKanner3D busca atender a estes requisitos utilizando-se de três etapas sequenciais, como mostra a Figura 1: captura (Criação da malha 3D), em que um humano real é posicionado em frente a uma câmera RGBD e é utilizado o *Fast Avatar Capture* para gerar o modelo 3D; textura (Geração do mapa de textura), utilização de um *script* em python para *Blender* que gera um arquivo de textura 2D; geração e associação do esqueleto (*rigProcessing*, geração do esqueleto e *rigging*), que serão explicados nas subseções a seguir. Os processos para o mapeamento de textura e *rigging* são independentes, o que significa que não dependem um do outro para serem executados, somente da malha gerada no processo de captura. Estas três etapas produzem como resultado final um avatar com *rigging* e um arquivo de textura.

5.1 Captura

A ferramenta para captura e reconstrução de um modelo escaneado com o *Microsoft Kinect* que mais se adequou ao sKanner3D foi o *Fast Avatar Capture* [26], pois atende à restrição de apenas um *Kinect* parado. O processo de captura do *Fast Avatar Capture* é realizado obtendo 4 *scans*, um em cada visão (frente, lateral direita, costas, lateral esquerda), sempre virando 90º no sentido anti-horário. Após estas capturas é feito o pós-processamento, em que os *scans* são alinhados, a malha é reconstruída e é colocada cor em cada ponto, utilizando a técnica de coloração de vértice.

A iluminação deve ser adequada pois pode afetar a qualidade da malha, causando várias manchas brancas na textura. Outro aspecto que pode afetar a qualidade da malha é a pose. No momento da captura é necessário ficar sempre na mesma pose pois caso contrário o *software* terá dificuldades na hora do alinhamento dos *scans* e a malha final pode conter deformidades. Após vários testes no *Fast Avatar Capture* a pose com melhores resultados para a captura foi com os braços abertos em um ângulo de aproximadamente 45º, sempre alinhados com o tronco e as mãos fechadas. Foram definidas marcações no chão para auxiliar a pessoa a ser escaneada a permanecer nas melhores posições para captura, conforme a Figura 5. Também foi definida uma padronização nas distâncias que a pessoa escaneada deve ficar em relação ao *Kinect* como mostra a Figura 6. Devido ao espaço de profundidade da captura do *Fast Avatar Capture*, que varia de 90 cm a 160 cm, estas padronizações se fazem necessárias para que todo o corpo seja capturado pela câmera.

A Figura 5 se relaciona com a Figura 6 através das posições 1 (frontal), 2 (lateral direita), 3 (costas) e 4 (lateral esquerda) e descreve a rotação em 90º que a pessoa deve realizar para a captura. A padronização da distância dos pés e dos braços permite uma posição confortável e aproximadamente idêntica para cada pose, visto que quando a pessoa gira pode ocasionar um leve desvio em seu posicionamento, principalmente da abertura dos braços. As distâncias 1 e 3 são diferentes, pois devem levar em consideração o volume do corpo em relação à profundidade de captura, tendo como referência a ponta dos pés da pessoa. Já as posições 2 e 4, são idênticas, mas diferentes das distâncias 1 e 3, visto que neste caso deve-se levar em consideração a abertura do braço da pessoa que está sendo escaneada. Em todas as posições a altura do Kinect é igual.

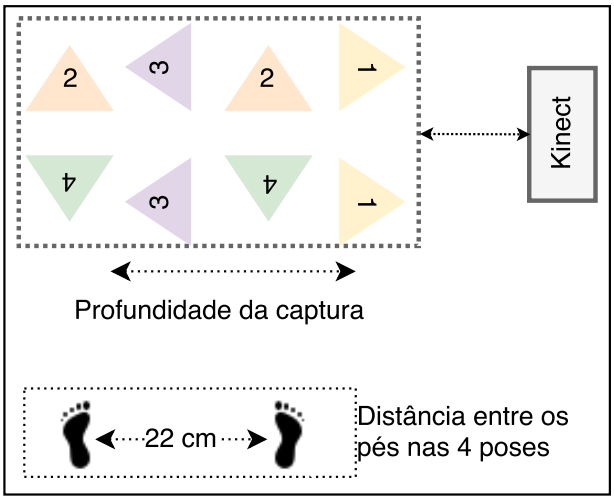

Figura 5: Marcações no chão para a captura

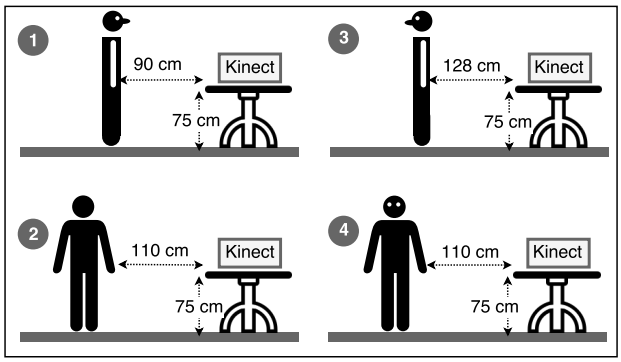

Figura 6: Distância para captura no *FastAvatarCapture* (até a ponta dos pés)

5.2 Textura

Após o processo de captura, a malha gerada não possui um mapa de textura, mas cada ponto guarda uma informação de cor. Então é feito um processo para geração do mapa de textura a partir da cor dos pontos. O processo é feito utilizando um *script* desenvolvido em python para o *Blender* que realiza as funções de *Smart UV Project* e *Bake*. A primeira função cria o mapa de textura, ainda sem cor e a segunda coloca a cor no mapa a partir da cor de cada ponto da malha, como mostra a Figura 7.

Figura 7: A esquerda o modelo sem textura, no centro o mapa de textura ainda sem cor, e na direita o mapa de textura com as cores

5.3 *Rigging*

Primeiramente utilizou-se o *software RadiantRigXL*, porém devido a alguns resultados insatisfatórios no posicionamento do esqueleto decidiu-se buscar uma outra solução. Foi criado um *script* chamado *rigProcessing* desenvolvido em *Processing*[19] que busca encontrar pontos para posicionar o esqueleto em uma malha com determinada pose.

A ideia básica do *rigProcessing* é mapear os pontos da malha 3D em uma matriz 2D (plana), encontrar as coordenadas de todos os 28 pontos necessários para posicionamento do esqueleto e exportar as coordenadas desses pontos. Em seguida outro *software*, nesse caso foi utilizado o *Blender*, cria um esqueleto no padrão utilizado e importa as coordenadas encontradas anteriormente para posicionar o esqueleto na malha e então une os dois.

Todos os pontos utilizados no processo foram obtidos analisando o esqueleto dos modelos do *software* SAPECO [8], que foram gerados no *software MakeHuman*[20]. Porém, as proporções antropométricas são de crianças da faixa etária de sete a dez anos, que também foram base para o SAPECO e portanto esta solução está focada nesta faixa de idade.

[19]https://processing.org/
[20]http://www.makehuman.org

A primeira parte do método utilizado no *rigProcessing* é importar a malha no formato .ply ASCII, gerada pelo processo anterior, e mapear os pontos em uma matriz 2D com dimensões 800*x*400, como pode ser visto na Figura 8. A primeira componente é tratada como X, a coordenada horizontal na Figura 8, e a segunda componente é referida como Y e representa a coordenada vertical na figura. O mapeamento do ponto 3D para a matriz 2D é feito utilizando as equações 1 e 2:

Com o ponto original $P(pX, pY, pZ)$ e seu mapeamento na matriz $pM(pmX, pmY)$

$$pmX = pX \times 350 + 400 \quad (1)$$

$$pmY = pY \times 350 + 200 \quad (2)$$

Ao final do processo é utilizada a mesma equação para transformar os pontos encontrados para as coordenadas originais.

O processo para encontrar os pontos inicia localizando a região referente a cabeça, fazendo uma varredura na matriz e verificando se os pontos estão em uma determinada zona em Y. Cada linha com coordenada X é analisada, então quando começa a existir pontos isso representa o começo da cabeça, e quando for encontrado uma linha com pontos fora desta zona então é determinado o fim da cabeça. Utilizando o ponto médio da cabeça em Y, percorre a matriz em X até quando não encontrar mais pontos, esse representa o meio das pernas. Então, utilizando os pontos da cabeça e do meio das pernas como referência, coloca todos os pontos do meio do corpo como cabeça, pescoço e vários pontos referentes à coluna. Partindo do ponto referente ao meio das pernas são encontrados os limites do tronco e então é desenhado um retângulo que representa esta região. Em seguida, utilizando o retângulo é calculada a posição dos ombros, partindo do canto próximo ao ombro avança em Y até coincidir com pontos na matriz. Para garantir que nenhum ponto fique fora da malha nesta região e que os ombros fiquem alinhados é calculada a posição de cada ombro e então limita-se X com o maior valor dos dois. Com o ponto do ombro é lançada uma reta que ricocheteia nas bordas do braço, até chegar ao extremo da mão esquerda.

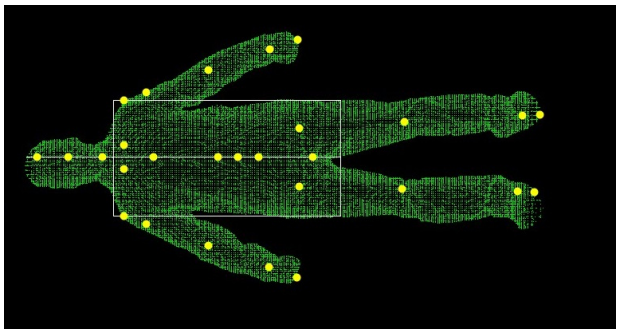

Figura 8: Pontos utilizados no rigProcessing.

Com os pontos do ombro e da mão são posicionados todos os outros pontos do braço esquerdo utilizando a proporção dos segmentos utilizados nos modelos do SAPECO. O método para o braço direito é o mesmo que para o braço esquerdo.

Na sequência, é posicionado o ponto do início da perna esquerda, com Y igual a média entre o ponto do meio das pernas e o limite do tronco naquela direção, já o X utilizado é um pouco menor do que o fim do tronco em X. Partindo do ponto de início da perna é traçada uma linha até o mais próximo do pé, portando se deslocando em Y quando possível. Essa linha não necessariamente chega até o pé, pois como a malha não é perfeita podem existir espaços não marcados na matriz 2D.

Para solucionar esse problema é deslocada uma posição em X e é traçada uma nova reta. Quando o *software* não conseguir avançar em Y nem em X então esse ponto representa o pé esquerdo. Da mesma forma que os braços, são utilizados os pontos do início da perna esquerda e do pé esquerdo para posicionar todos os outros pontos da perna esquerda, também utilizando a proporção dos segmentos observada nos modelos do SAPECO. O processo para a perna direita é o mesmo que para a perna esquerda. Em seguida são posicionados os pontos referentes ao peitoral, com base nos limites do tronco.

A última fase do *rigProcessing* é encontrar a coordenada Z de cada ponto, que é a dimensão desconsiderada no mapeamento para a matriz 2D e que representa a profundidade. Para cada ponto pM encontrado na matriz, nas coordenadas da matriz 2D, é transformado novamente para as coordenadas originais e é feita uma busca nos pontos originais da malha. É verificado se cada ponto da malha pX está mais próximo que uma distância d do ponto pM. Para todos os $pX's$ que estão dentro da região desejada, com distância menor que d do ponto pM, é armazenado o maior Z e o menor Z nessa região de pontos. Com isso, é feita a média entre o menor Z e o maior Z obtidos. Isso garante que o esqueleto seja posicionado no meio da malha. Para os pontos dos pés e dos calcanhares essa regra não se aplica, visto que o osso entre estes pontos não devem ficar com Z no meio da região encontrada. Desta forma, o ponto do calcanhar utiliza o menor Z encontrado na busca e o ponto do pé utiliza o maior Z encontrado.

As coordenadas encontradas são exportadas para um arquivo .CSV, que é utilizado como entrada em um *script* para *Blender*, que posiciona um esqueleto padrão utilizando as coordenadas de entrada e então junta o esqueleto a malha, como mostra a Figura 9.

Figura 9: Esqueleto posicionado utilizando os pontos encontrados no rigProcessing.

6 Resultados e Discussão

O processo do sKanner3D permite capturar o modelo de uma pessoa utilizando apenas uma única câmera RGBD parada gerando um modelo geométrico resultante com textura mapeada e esqueleto, tudo de forma automática e integrada. Todo o processo é fruto da união de vários *softwares*, alguns já prontos e outros desenvolvidos para suprir as demandas existentes. Possui bons resultados nas condições adequadas: iluminação, vestimenta e pose. Todo o processo dura aproximadamente 3 minutos e meio, utilizando um computador com processador Intel Core i5-4570 (2.6GHz), 16GB de memória RAM, placa de vídeo NVidia GForce GTX 780 Ti 3GB, sistema operacional Windows 7 Professional (64bits) e um único dispositivo *Microsoft Kinect v1.0*. O processo demanda os seguintes tempos:

- *Fast Avatar Capture* (*scans*): 40 segundos, 10 segundos por *scan*;
- *Fast Avatar Capture* (pós-processamento): 2 minutos e 20 segundos;
- *rigProcessing*: 1 segundo;
- Posicionamento e união do esqueleto a malha: 10 segundos;
- Mapeamento da textura: 10 segundos.

Para atingir estes tempos, a malha gerada pelo processo de captura foi configurado para obter aproximadamente 50000 vértices e é exportada para um arquivo no formato .ply com a informação de cor. O *rigProcessing* utiliza este arquivo para realizar a discretização e exporta um arquivo .csv com a posição das juntas. Após isso, a ferramenta *Blender* importa o arquivo .ply e o .csv produzindo como resultados o mapa de textura no formato .png e uma malha com *rigging* no formato .dae, os quais podem ser abertos na maioria das ferramentas de modelagem/animação ou motor de jogos.

O tamanho da malha capturada interfere nos tempos alcançados, podendo ser mais rápido em malhas menores, porém perde-se a qualidade do modelo escaneado ou não se consegue realizar todo o processo caso a malha esteja aberta. Em modelos com malhas maiores ganha-se qualidade da textura mas, aumenta-se consideravelmente o tempo que o sKanner3D utiliza para realizar todo o processamento.

O diferencial do sKanner3D para os outros *softwares* de captura e *rigging* analisados é que ele possui um processo totalmente automatizado, que gera o modelo geométrico do avatar completo com textura e esqueleto. As Figuras 10 e 11 mostram: à esquerda os resultados obtidos pela captura; o modelo texturizado ao centro e; à direita, o *rigging*, utilizando duas pessoas escaneadas com altura diferentes.

Figura 10: Exemplo 1 do processo completo do sKanner3D

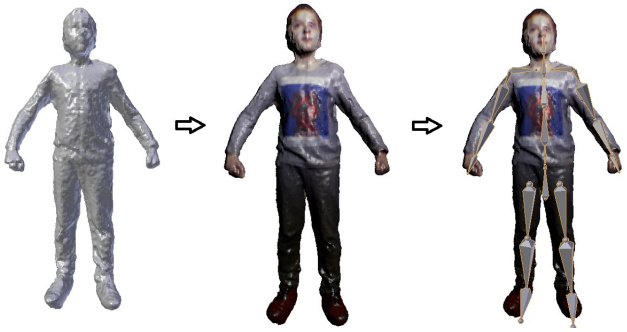

Figura 11: Exemplo 2 do processo completo do sKanner3D

6.1 Limitações

Durante os testes no processo do sKanner3D observaram-se algumas limitações, como mostram as Figuras 12 e 13:

- O modelo deve ser escaneado na mesma posição e com a mão fechada. Devido ao fato de que o dispositivo *Kinect* tem uma baixa resolução em comparação aos escâneres a *laser* tradicionais e quando o modelo é escaneado com a mão aberta ou com o braço muito perto do corpo, o processo de reconstrução fica prejudicado. Registre-se que foi utilizado o modelo v1 do *Kinect* e, como o modelo v2 tem mais precisão e resolução, acredita-se que este problema será menor se usada esta última versão do dispositivo;
- A captura com os braços muito próximos ao corpo prejudica o *rigProcessing* pois gera dificuldades para encontrar as mãos do modelo. Isto significa que a movimentação deve ser acompanhada e corrigida por uma pessoa acompanhando o processo;
- A iluminação do ambiente também interfere na texturização do modelo, criando manchas no corpo. A falta de iluminação pode gerar modelos muito escuros. Isto significa que é desejável um ambiente com boa iluminação difusa;
- Outra limitação é quanto ao tipo de roupa utilizada pela pessoa durante a captura, o que impacta nos processos subsequentes. O mais adequado são roupas justas ao corpo, visto que pessoas escaneadas com roupas mais largas tendem a induzir um erro de posicionamento das juntas do esqueleto.

Figura 12: A imagem à esquerda apresenta uma falha na reconstrução no braço esquerdo; a da direita, exibe um esqueleto posicionado de forma incorreta.

7 Conclusão

Humanos virtuais capturados utilizando câmeras RGBD de baixo custo, como o *Microsoft Kinect* v1.0, podem ser utilizados em diversas aplicações, principalmente, quando estes possuem uma textura e esqueleto permitindo, por exemplo, uma futura animação.

Muitos estudos e ferramentas focam em tarefas de captura, textura e *rigging* de um modelo como processos isolados ou não contemplam todos eles. Este artigo apresentou um procedimento para geração de modelos tridimensionais a partir de pessoas escaneadas, geração de mapa de textura e *rigging* de modo automatizado e integrado. Os mesmos podem ser utilizados para qualquer propósito, como por exemplo, animação e importação em qualquer ferramenta de modelagem ou motores de jogos.

Figura 13: Manchas brancas na textura decorrentes da iluminação incorreta.

Assim, os modelos capturados pelo sKanner3D podem ser imediatamente manipulados pelo esqueleto através de ferramentas de modelagem ou animação. Também são adequados para prototipação e análises, visto que o modelo consegue representar a pessoa capturada utilizando um dispositivo de baixo custo, antes de se investir numa solução de maior precisão e detalhamento.

Apesar de possuir algumas limitações (baixa resolução da câmera RGBD; ser sensível à iluminação e roupa utilizada; ser restrito ao posicionamento da pessoa) é um procedimento simples e eficaz que utiliza apenas uma única câmera RGBD e produz resultados satisfatórios. Para obter o código do *rigProcessing* basta contactar um dos autores.

Espera-se que o procedimento apresentado aqui, sirva como reflexão na área de computação gráfica e possa fomentar a criação de mais e novas técnicas baseadas no processo completo a partir de corpos escaneados para produção de personagens.

7.1 Trabalhos Futuros

Como trabalhos futuros pretende-se utilizar o ASM (*Active Shape Model*) [5] para a identificação das juntas e *rigging* e então, realizar este processo independente da pose.

Também pretende-se melhorar o processo de texturização, visto que durante a captura são geradas quatro imagens, uma de cada visão, e em seguida essa informação é transferida para malha através da técnica de coloração de vértice para depois ser mapeada para uma arquivo de textura, o que reduz a qualidade da textura. Assim, espera-se desenvolver um método que utilize de forma direta as 4 imagens para gerar o mapa de textura com o intuito de melhorar a qualidade da textura do modelo.

Agradecimentos

Os autores gostariam de agradecer à UDESC, Universidade do Estado de Santa Catarina, pela bolsa de iniciação científica e à CAPES, Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, pela bolsa de estudos disponibilizada através do Programa de Demanda Social. Também à CNPQ, Conselho Nacional de Desenvolvimento Científico e Tecnológico, pelo apoio parcial ao projeto e à FAPESC, Fundação de Apoio à Pesquisa Científica e Tecnológica do Estado de Santa Catarina, pela disponibilização dos *Kinects*.

REFERÊNCIAS

[1] D. S. Alexiadis, D. Zarpalas, and P. Daras. Real-time, full 3-d reconstruction of moving foreground objects from multiple consumer depth cameras. *IEEE Transactions on Multimedia*, 15(2):339–358, Feb 2013.

[2] I. Baran and J. Popović. Automatic rigging and animation of 3d characters. *ACM Transactions on Graphics (TOG)*, 26(3):1–8, July 2007.

[3] M. B. Boquet. Automatic and guided rigging of 3D characters. Master's thesis, Universitat Politècnica de Catalunya, 2008.

[4] H. Braun, V. J. Cassol, R. Hocevar, F. P. Marson, and S. R. Musse. Crowdvis: A framework for real time crowd visualization. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, SAC '13, pages 989–995, 2013.

[5] T. Cootes, C. Taylor, D. Cooper, and J. Graham. Active shape models-their training and application. *Computer Vision and Image Understanding*, 61(1):38 – 59, 1995.

[6] R. Fabio et al. From point cloud to surface: the modeling and visualization problem. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, XXXIV-5/W10:1–11, 2003.

[7] A. Feng, D. Casas, and A. Shapiro. Avatar reshaping and automatic rigging using a deformable model. In *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games - SA '15*, MIG '15, pages 57–64, 2015.

[8] E. P. Ferrari. *Construção e validação de um instrumento digital para avaliação da imagem corporal infantil*. PhD thesis, Programa de Pós-Graduação em Ciências do Movimento Humano, Universidade do Estado de Santa Catarina, 2016.

[9] B. B. Gnecco, D. R. C. Dias, G. Brasil, and M. Guimarães. Desenvolvimento de interfaces naturais de interação usando o hardware kinect. *Tendências em Realidade Virtual e Aumentada. Niteroi-Rj: Sociedade Brasileira de Computação-SBC*, pages 37–46, 2012.

[10] P. Haeberli and M. Segal. Texture mapping as a fundamental drawing primitive. In *Fourth Eurographics Workshop on Rendering*, pages 1–12, 1993.

[11] P. S. Heckbert. Survey of texture mapping. *IEEE Computer Graphics and Applications*, 6(11):56–67, Nov 1986.

[12] C. Heindl, S. C. Akkaladevi, and H. Baue. Photorealistic Texturing of Human Busts Reconstructions. In *Proceedings of the 7th International Conference on 3D Body Scanning Technologies*, pages 225–230, Lugano, Switzerland, 2016.

[13] H. Li, E. Vouga, A. Gudym, L. Luo, J. T. Barron, and G. Gusev. 3d self-portraits. *ACM Transactions on Graphics (TOG)*, 32(6):187:1–187:9, 2013.

[14] C. Lovato, U. Castellani, and A. Giachetti. *Automatic Segmentation of Scanned Human Body Using Curve Skeleton Analysis*, pages 34–45. Springer, Berlin, Heidelberg, 2009.

[15] Y. Ma, J. Zheng, and J. Xie. Foldover-free mesh warping for constrained texture mapping. *IEEE Transactions on Visualization and Computer Graphics*, 21(3):375–388, March 2015.

[16] N. Magnenat-Thalmann. *Modeling and simulating bodies and garments*. Springer Science & Business Media, 2010.

[17] N. Magnenat-Thalmann, H. Seo, and F. Cordier. Automatic modeling of virtual humans and body clothing. *Journal of Computer Science and Technology*, 19(5):575–584, 2004.

[18] N. Magnenat-Thalmann and D. Thalmann. *Handbook of Virtual Humans*. John Wiley & Sons, 2006.

[19] A. Mao, H. Zhang, Y. Liu, Y. Zheng, G. Li, and G. Han. Easy and fast reconstruction of a 3d avatar with an rgb-d sensor. *Sensors*, 17(5):1–21, 2017.

[20] F. A. C. Modesto et al. Humanos virtuais e avatares. In *Fundamentos e Tecnologia de Realidade Virtual e Aumentada*, volume 8, pages 79–97. Sociedade Brasileira de Computação, 2006.

[21] J.-C. Nebel. *Soft Tissue Modeling from 3D Scanned Data*, pages 85–97. Springer, Boston, MA, 2001.

[22] J. F. Oliveira, D. Zhang, B. Spanlang, and B. F. Buxton. Animating scanned human models. *Journal of WSCG*, 11(1-3):1–9, 2003.

[23] V. Orvalho, P. Bastos, F. I. Parke, B. Oliveira, and X. Alvarez. A facial rigging survey. In *Eurographics (STARs)*, pages 183–204, 2012.

[24] R. Pagés, D. Berjón, and F. Morán. Automatic system for virtual human reconstruction with 3d mesh multi-texturing and facial enhancement. *Signal Processing: Image Communication*, 28(9):1089–1099, 2013.

[25] E. S. Santos, E. A. Lamounier, and A. Cardoso. Interaction in augmented reality environments using kinect. In *2011 XIII Symposium on Virtual Reality*, pages 112–121, May 2011.

[26] A. Shapiro, A. Feng, R. Wang, H. Li, M. Bolas, G. Medioni, and E. Suma. Rapid avatar capture and simulation using commodity depth sensors. *Computer Animation and Virtual Worlds*, 25(3-4):201–211, 2014.

[27] A. T. d. Silva. Geração automática de populações de personagens virtuais. Master's thesis, Universidade do Vale do Rio dos Sinos - UNISINOS, 2005.

[28] D. Thalmann and S. R. Musse. *Crowd simulation*. Springer, London, 2 edition, 2013.

[29] J. Tong, J. Zhou, L. Liu, Z. Pan, and H. Yan. Scanning 3d full human bodies using kinects. *IEEE Transactions on Visualization and Computer Graphics*, 18(4):643–650, April 2012.

[30] R. C. C. Vieira, C. A. Vidal, and J. B. Cavalcante-Neto. Manipulação corporal de personagens virtuais por deformações de medidas antropométricas. In *2010 XII Symposium on Virtual Reality*, pages 102–111, 2010.

[31] A. Weiss, D. Hirshberg, and M. J. Black. Home 3d body scans from noisy image and range data. In *2011 International Conference on Computer Vision*, pages 1951–1958, Nov 2011.

[32] N. Werghi. Segmentation and modeling of full human body shape from 3-d scan data: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(6):1122–1136, Nov 2007.

[33] Z. Zhang. Microsoft kinect sensor and its effect. *IEEE MultiMedia*, 19(2):4–10, Feb 2012.