UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE TECNOLOGIA

Luiz Felipe Rosa da Silveira

Além das Lentes do KINECT-v2

(Um Estudo Exploratório do Hardware de Captura de Movimentos)

Limeira, 2020

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE TECNOLOGIA

Luiz Felipe Rosa da Silveira

Além das Lentes do KINECT-v2
(Um Estudo Exploratório do Hardware de Captura de Movimentos)

Trabalho de Conclusão de Curso apresentado à Faculdade de Tecnologia da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Prof. Dr. Marco Antonio Garcia de Carvalho

Este exemplar corresponde à versão final do Trabalho de Conclusão de Curso defendido por Luiz Felipe Rosa da Silveira e orientado pelo Prof. Dr. Marco Antonio Garcia de Carvalho.

Limeira, 2020

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE TECNOLOGIA

Luiz Felipe Rosa da Silveira

Além das Lentes do KINECT-v2
(Um Estudo Exploratório do Hardware de Captura de Movimentos)

Banca Examinadora:

Prof. Dr. Marco Antonio Garcia de Carvalho (Orientador)
Faculdade de Tecnologia (FT/UNICAMP)

Prof. Dr. Celmar Guimarães da Silva
Faculdade de Tecnologia (FT/UNICAMP)

Prof. Dr. Plínio Roberto Souza Vilela
Faculdade de Tecnologia (FT/UNICAMP)

Limeira, 2020

Dedico este trabalho aos meus pais, pelo constante apoio durante todo o meu percurso acadêmico e por me incentivarem a ir sempre em busca dos meus sonhos.

AGRADECIMENTOS

Agradeço a Deus por mais esta conquista.

Aos meus pais Cláudia e Omar, pelo amor, carinho e apoio durante toda a minha jornada acadêmica e, principalmente, nos momentos de dificuldades, me dando enorme força e energia para concretização deste trabalho da melhor forma possível.

À minha irmã Ana Carolina, pelo diálogo, amizade e apoio nos momentos de frustração e saudade de casa e durante a realização deste trabalho.

Ao Professor Dr. Marco Antonio Garcia de Carvalho, pela oportunidade dada de aprofundamento dos estudos na área da Computação Gráfica, excelente orientação desde as primeiras disciplinas que lecionou, enorme dedicação, paciência e compreensão, sempre incentivando para obter os melhores resultados.

Não querendo esquecer ninguém, agradeço a todos os meus familiares, professores e amigos que me apoiaram, participaram direta ou indiretamente na minha jornada acadêmica. E a todos os que colaboraram de alguma maneira no processo de execução deste trabalho.

Por fim, gostaria de agradecer à UNICAMP, por me proporcionar uma formação ampla e sólida, em ambiente de excelência acadêmica, inovação e pesquisa tecnológica.

Muito obrigado!

“Tenha coragem de seguir seu coração e intuição. Eles já sabem o que você realmente deseja. Todo resto é secundário”.

- Steve Jobs

RESUMO

Animação consiste em, de alguma forma, dar vida a algo que não tem; num processo em que imagens ou fotogramas são mostrados sequencialmente de modo a criarem a ilusão de uma imagem em movimento. Com o surgimento dos computadores eletrônicos, as técnicas de animação tiveram um enorme avanço, com o desenvolvimento da animação digital, permitindo maior velocidade e eliminação de tarefas mecânicas e repetitivas nos processos produtivos. Novos dispositivos e técnicas de captura de movimento como sensores de detecção e reprodução de movimentos naturais permitiram a elaboração de animações cada vez mais reais. No entanto, essas tecnologias nem sempre são acessíveis, e demandam altos investimentos em equipamentos e equipes especializadas para seu correto funcionamento. Daí a importância de se procurar tecnologias acessíveis e de baixo custo, que possam ser utilizadas em aplicações com o uso de captura de movimentos. Com esse propósito, neste trabalho foram explorados os recursos técnicos do KINECT-V2, que é um dispositivo para captura de movimentos de baixo custo, lançado em 2014 pela Microsoft, mas que utiliza tecnologias sofisticadas, na elaboração de uma aplicação prática inspirada numa “caixa de brinquedos”, desenvolvida em C# no motor gráfico Unity 3D, em que a interação homem-máquina é feita através dos movimentos do usuário captados pelo sensor, que permite ao usuário entrar no universo de cada personagem 3D e controlar seus movimentos.

Palavras-chave: Captura de movimento, Mocap, Kinect, Animação por computador.

ABSTRACT

Animation consists of, in some way, giving life to something that has no life. It is the process in which images or frames are shown sequentially, at a certain speed, creating the illusion of a moving image. With the emergence of electronic computers, animation techniques had a huge advance, with the development of digital animation, allowing greater speed and elimination of mechanical and repetitive tasks in the production processes. New devices and techniques for capturing motion such as sensors for detecting and reproducing natural movements allowed the development of increasingly real animations.. However, these technologies are not always accessible, and require high investments in equipment and specialized teams for their correct functioning. That is why it is important to look for accessible and low-cost technologies that can be used in applications with the use of motion capture. For this purpose, in this work the technical resources of KINECT-V2 were explored, which is a low cost motion capture device, launched in 2014 by Microsoft, but which uses sophisticated technologies, in the elaboration of a practical application inspired by a “box toys ”, developed in C # on the Unity 3D graphics engine, in which a human-machine interaction is made through the user's movements captured by the sensor, which allows the user to enter the universe of each 3D character and control their movements.

Keywords: Motion Capture, MoCap, Kinect, Computer Animation.

LISTA DE FIGURAS

SUMÁRIO

1. INTRODUÇÃO

Este capítulo tem por objetivo a introdução do tema deste trabalho, apresentar as motivações e justificativas, seus objetivos e a organização e sequência em que o texto será discorrido.

1.1. Motivação e Justificativa

A motivação para a realização deste projeto surgiu das infinitas possibilidades de aplicações de captura de movimento em filmes, jogos, etc. Porém, nesses tipos de produção, geralmente é necessário dispor de equipamentos com tecnologia de ponta e estruturas físicas que demandam altos investimentos, o que, às vezes inviabiliza a execução de aplicações.

Daí a importância de se buscar tecnologias acessíveis e que não necessitem de grande investimento, que possam ser utilizadas em aplicações com o uso de captura de movimentos, sem depender de sofisticadas infraestruturas de suporte ou a utilização de marcadores ópticos para rastreamento.

Com esse propósito, neste trabalho foi testado o KINECT-V2, da Microsoft, que é um dispositivo para captura de movimentos de baixo custo, lançado em 2014 pela Microsoft, mas que utiliza tecnologias sofisticadas e não necessita de uso de marcadores reflexivos para rastreamento. Esse dispositivo inicialmente foi desenvolvido como um acessório para consoles de videogame, para proporcionar maior interação física entre usuários e jogos, em uma combinação de software e hardware criados pela Microsoft. Em sua versão original, o hardware incluiu uma tecnologia de chipset da empresa israelense PrimeSense, que desenvolveu um sistema que consiste em um projetor de raios infravermelhos e uma câmera e um micro-chip especial que gera uma grade a partir da qual pode verificar a localização de um objeto próximo em 3 dimensões. Através de sequência de algoritmos o dispositivo processa as imagens captadas, resultando em detecção de segmentos corporais sem a necessidade de marcadores (Shotton et al., 2011).

1.2. Objetivo

O objetivo deste trabalho de conclusão de curso é o de realizar um estudo exploratório sobre o hardware de captura de movimento – KINECT v2, da Microsoft, bem como a realização de uma aplicação interativa onde o usuário pode usar seus movimentos captados pelo sensor, transferindo-os, em tempo real, para as personagens virtuais, utilizando ferramentas e softwares de baixo custo, ou de uso livre.

1.3. Organização do Texto

Este trabalho foi organizado e dividido em 6 capítulos, sendo este o primeiro deles. Os demais estão divididos como apresentado a seguir:

O capítulo 2 refere-se à Fundamentação Teórica. dos principais processos de Captura de Movimentos (MoCap) e suas tecnologias atualmente utilizadas em aplicações em animação digital de filmes, jogos etc. bem como abordamos algoritmos de classificação e agrupamento.

O capítulo 3 descreve os materiais e métodos utilizados no desenvolvimento dos trabalhos. Neste capítulo abordamos ainda os recursos técnicos do dispositivo KINECT v2, suas especificações e detalhes técnicos, e como se processam os dados captados pelo sensor de movimento por algoritmos do sistema.

O capítulo 4 refere-se ao desenvolvimento da aplicação interativa do sensor de captura de movimentos pelo KINECT v2, acoplado a um computador PC, e seus resultados.

Por fim, o capítulo 5 traz as conclusões.

2. FUNDAMENTAÇÃO TEÓRICA

Este capítulo descreve um resumo do embasamento teórico deste trabalho, abordando temas como breve história da Animação, as diferentes tecnologias de Captura de Movimentos (MoCap): captura de movimento óptico (com e sem marcadores), captura de movimento magnético e captura de movimento mecânico; e algoritmos de classificação e agrupamento.

2.1. Animação

Segundo Williams (2016), animação é o processo em que imagens ou fotogramas são manipulados de forma a parecerem um filme em movimento. Geralmente, o efeito da animação é alcançado por uma rápida sucessão de imagens sequenciais que diferem minimamente uma da outra. Quando os fotogramas são ligados entre si e o filme resultante é visto a uma velocidade de 16 ou mais imagens por segundo, tem-se a ilusão de movimento contínuo.

As técnicas de animação passaram por constante evolução e aperfeiçoamento, através de diferentes invenções e métodos de produção.

Após a invenção de aparelhos ópticos como o *phenakistoscopio* (1832) e o *praxinoscópio* (1877), as exibições das primeiras animações se tornaram possíveis. Após essas descobertas, foram produzidas animações desenhadas, quadro a quadro, que eram fotografados um a um separadamente e compilados em filmes. Esse procedimento foi aperfeiçoado ao longo do tempo, com adoção de recursos como desenhos em celulóides transparentes, que possibilitaram o desenvolvimento de animação em camadas.

Em 1915, Max Fleisher desenvolveu um dispositivo chamado “rotoscópio”, em que imagens de um filme eram projetadas em uma superfície de vidro, sobre a qual se podia redesenhar à mão sobre alguns quadros, para inserir animações. O

dispositivo podia ainda ser utilizado para copiar os movimentos de um modelo-referência já filmado. Essa técnica foi amplamente utilizada por diversos estúdios na produção de desenhos animados na década de 1930. Walt Disney também explorou essa técnica em estudos de movimento humano e animal, que foram utilizados em seus filmes. Pode-se considerar que a rotoscopia foi precursora da moderna "captura de movimento digital".

O desenvolvimento da animação digital aumentou bastante a velocidade do processo de criação, eliminando tarefas mecânicas e repetitivas. Um marco nos filmes de animação foi "Toy Story" (1995), que foi o primeiro longa-metragem inteiramente realizado com imagens geradas por computador.

A maioria dos filmes de animações, hoje em dia, é feita com imagens geradas por computador, produzidas com recursos de captura de movimentos por sensores e efeitos visuais 3D, com efeitos muito realistas.

Quanto ao uso de animações nos jogos eletrônicos, estas começaram a surgir com a popularização de máquinas de jogos eletrônicos 2D na década de 80; e na década de 90 surgiram os jogos com animações 3D, que incluíam variáveis de movimentos animados dos personagens.

Do ano 2000 até hoje, várias gerações de consoles para jogos foram lançadas, com a introdução de animações mais complexas, fluídas e realistas para acompanhar a evolução na complexidade das personagens e com mais possibilidades para o jogador.

Atualmente na produção de jogos eletrônicos são utilizados equipamentos sofisticados de computação gráfica e sensores de captura de movimentos, obtendo efeitos muito realistas em 3D que dão ao jogador a sensação de estar vivenciando realmente todas as situações do jogo.

2.2. Captura de Movimento (MoCap)

A **Captura de Movimento (Motion Capture ou MoCap)** é uma técnica de computação gráfica para gravação de movimentos, que funciona por meio de sensores que capturam e registram, através de um escaneamento corporal por câmeras especiais, todas as ações e movimentos corporais realizados por atores humanos, ou animais. Essas ações e movimentos são convertidos de forma digital e transpostos para personagens virtuais de animação digital 3D, que assim, ganham movimentação natural e realística.

Existem vários métodos para realização de captura de movimento, sendo que cada tecnologia tem seus pontos fortes e fracos, não havendo uma única tecnologia de captura de movimento que seja perfeita para todos os usos possíveis (SILVA,1997). Apresentamos a seguir os principais sistemas utilizados para captura de movimento: Sistema Óptico (com e sem marcadores), Sistema Mecânico e Sistema Magnético.

Hoje já existem sistemas híbridos que misturam os pontos positivos de cada um desses três sistemas fundamentais, e várias tecnologias híbridas oferecem a possibilidade de fazer captura facial junto com os movimentos do corpo.

2.2.1. Captura de Movimento Óptico:

2.2.1.1. Com Marcadores

A captura de movimento óptico é a tecnologia mais comumente usada atualmente. É uma técnica que captura dados digitalmente, capazes de transformar o movimento da vida real em formato digital. A captura de movimento óptico é amplamente utilizada em muitos campos, como animação, efeitos especiais, jogos. Essa técnica imprime uma melhor “sensação” de vida no personagem de animação.

A tecnologia de captura de movimento óptico utiliza dois ou mais ângulos de câmeras especiais, para criar a sensação de ambiente 3D; e emprega marcadores refletivos. Geralmente são utilizadas esferas feitas ou cobertas de material reflexivo que são colados em pontos-chave do corpo do ator, para que o software possa identificar sua posição em 3D. Quando o mesmo marcador é

rastreado por mais de uma câmera, ele fornece detalhes de todos os três eixos, oferecendo maior precisão.

Os tipos mais utilizados de marcadores são: Refletor e LED de pulso. (Figuras 1 e 2)

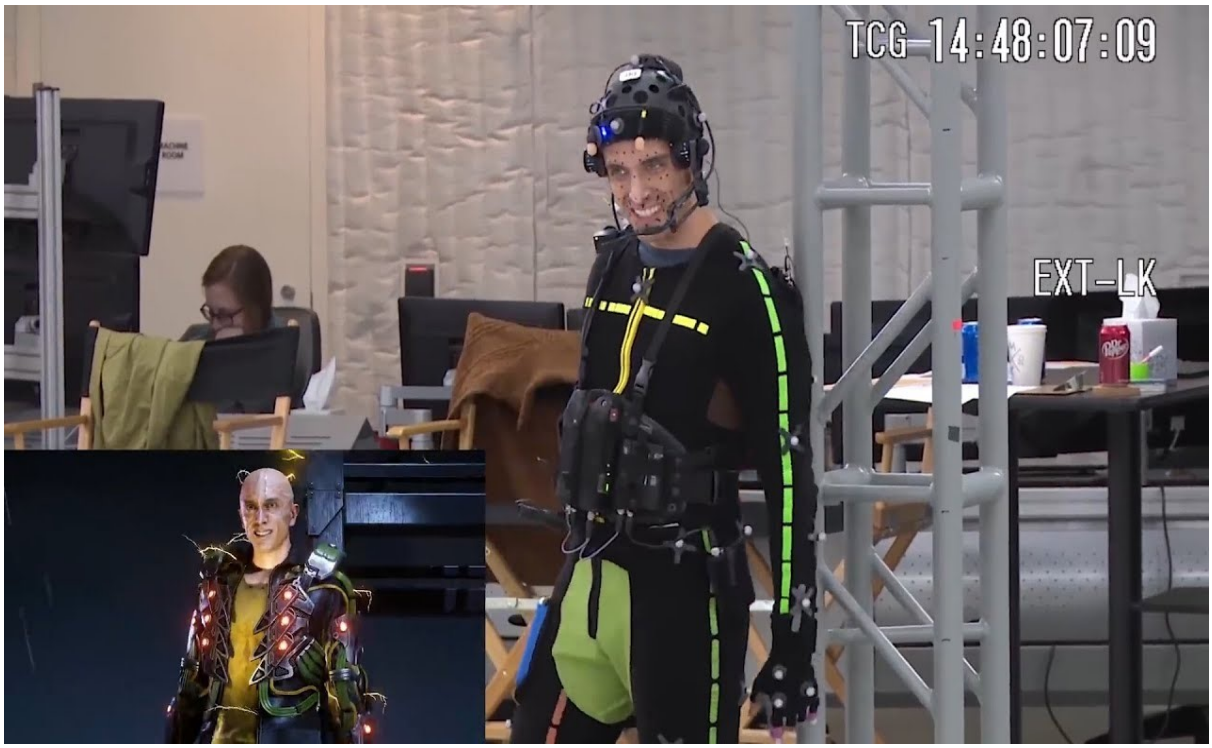


Figura 1 - Cena de vídeo divulgado pelo estúdio *Insomniac Games*, de uma sessão de MoCap para o jogo *Spider Man*, disponível para *Playstation 4.*
Fonte: (https://www.youtube.com/watch?v=lbmuQGJ6WHs)

Há prós e contras dessas tecnologias. Como pontos positivos, os atores sentem-se à vontade para agir, pois o traje não é pesado e os marcadores são leves; podem trabalhar em grandes áreas, ao mesmo tempo em que são possíveis mais performances; dados limpos e detalhados são capturados. Por outro lado, os pontos negativos são o alto custo dos equipamentos, que dependendo da sofisticação da produção, podem ultrapassar o valor de U$ 100.000,00; os marcadores podem ficar ocultos por outros objetos ou atores, e são propensos a interferências leves.

Figura 2 - Sessão de Mocap realizada para o filme Planeta dos Macacos: A Revolta onde diversos pontos são rastreados, desde o ator (que será representado como um macaco),arma e partes do cavalo. Fonte: (https://thevirtualassist.net/visual-effects-movies-hollywood/)

Ainda nessa técnica de captura de movimento ótico com marcadores, a captura de detalhes e sutilezas das expressões faciais ou dos dedos das mãos também é referida como **captura de performance**. É utilizada para registrar os movimentos mais complexos em um rosto humano e capturar graus mais elevados de emoção. (Figura 3)

Figura 3 – Captura de movimentos faciais: marcadores colocados em pontos específicos na face
Fonte: (https://en.wikipedia.org/wiki/Motion_capture#/media/File:Motion_capture_facial.jpg)

2.2.1.2. Sem Marcadores

Os sistemas de captura de movimentos ópticos sem marcadores, não exigem que os participantes usem equipamentos especiais para rastreamento de seus movimentos. Algoritmos especiais de computador são projetados para permitir que o sistema analise múltiplos fluxos de entrada óptica e identifique formas humanas, quebrando-as em partes constituintes para rastreamento. O sistema de imagem óptica é responsável por converter a luz da área focada em imagem digital que o computador de rastreamento pode processar. Dependendo do projeto do sistema de rastreamento óptico, o sistema de imagem óptica pode variar de tão simples quanto uma câmera digital padrão a tão especializado quanto um telescópio astronômico. O software de processamento de imagem deve ser capaz de extrair a imagem alvo de seu fundo e calcular sua posição, através de algoritmos de processamento de imagens projetados para esse fim. O KINECT-v2, da Microsoft, é um exemplo de sensor de captura de movimento ótico sem a necessidade de uso de marcadores reflexivos. Apesar do dispositivo ter sido descontinuado, ainda é possível de ser adquirido em sites, como Amazon.com, a um valor aproximado de US$ 100.

2.2.2. Captura de Movimento Magnético:

Sistemas de captura de movimento magnético caracterizam-se pela maior velocidade de processamento dos dados capturados. Nesse sistema um conjunto de sensores receptores é posicionado nas articulações (pontos chave) no corpo de um ator. Os Receptores medem a posição e orientação dos pontos-chave em relação a uma antena transmissora, unidade de controle eletrônico, que emite um sinal de pulso e correlaciona seus locais relatados dentro do campo. Essas unidades são conectadas em rede com o computador host, que usa um driver de software para representar essas posições no espaço 3D. Esses sensores denotam as informações posicionais e rotacionais dos marcadores.

Essa tecnologia apresenta pontos positivos por ser relativamente mais barata que a captura óptica; maior precisão dos dados, as posições são absolutas (não existem problemas de oclusão). No entanto, as desvantagens desse tipo de sistema são os diversos fios que conectam os receptores à antena e restringem os movimentos complexos e rápidos dos atores; alcance limitado; distorção magnética à

medida que a distância aumenta; propenso a interferências por objetos de metal próximos, ou estruturas prediais.

2.2.3. Captura de Movimento Mecânico:

Nos sistemas de Captura de Movimento Mecânico, o ator usa um conjunto de tiras de metal em forma humana, como um esqueleto básico articulado, preso às costas (Figura 4). Cada articulação possui sensores que indicam a posição e orientações em altas taxas de amostragem. Também podem ser utilizados luvas, braço mecânico ou modelos articulados, que são usados para o enquadramento das teclas. Os sistemas mecânicos possuem pontos positivos, como: por serem equipamentos de medida absoluta, não são afetados por campos luminosos ou magnéticos; a captura é feita em tempo real; seu campo de trabalho é ilimitado e depende apenas do comprimento dos cabos de transmissão; permitem a captura de movimento de múltiplos atores numa mesma sessão. No entanto há desvantagens, pois, devido ao movimento dos atores, é comum ocorrerem quebras no esqueleto usado para a captura, e os atores são orientadas a não realizarem movimentos bruscos ou amplos para evitar a quebra do equipamento; os movimentos não são realistas; as posições absolutas não são conhecidas, mas são calculadas a partir das rotações; a tecnologia não tem consciência do solo, portanto não pode haver salto, além dos dados dos pés tenderem a deslizar. Como exemplo de sistema de captura de movimento mecânico podemos destacar o sistema Gypsy 7 da empresa Animazoo (Figura 4) com preço médio em torno de U$ 8000.

Figura 4 – Sistema de captura de movimentos mecânico Gypsy-7
Fonte: http://www.cgspeed.com/2010/07/animazoo-gypsy-7-announced-now-under.html

2.3. Algoritmos de Aprendizado: Classificação e de Agrupamento

Esta seção consiste na fundamentação teórica dos algoritmos de aprendizado presentes no processamento das imagens captadas pelo sensor Kinect-v2. Estes são algoritmos de classificação e clusterização (agrupamento), que analisam sequencialmente os dados captados e, a partir dos dados processados, resultam no esqueleto final inferido.

2.3.1. Algoritmos de Classificação

Algoritmos de Classificação são tipos de algoritmos de aprendizagem supervisionados, isto é, são algoritmos de “machine learning” em que existe uma base de dados previamente rotulada e classificada, para treinamento de modelo de predição, com a finalidade de predizer e classificar dados não-rotulados, a partir dos parâmetros aprendidos pela base rotulada de dados dos modelos.

Para classificar os dados captados, o Kinect emprega o algoritmo de Florestas Aleatórias, que consiste na junção de múltiplas Árvores de Decisão.

2.3.1.1. Árvore de Decisão

De acordo com Horning (2011), Árvores de Decisão são modelos preditivos supervisionados, que fazem uso de um conjunto de regras para chegar a um resultado alvo. Possuem formato de fluxograma e demonstram visualmente as condições e as probabilidades para se chegar a resultados. Cada nó interno de uma Árvore de Decisão é considerado como um teste para os dados de entrada, a ponto que esses dados terão saídas diferentes, de acordo com a satisfação de testes.

Segundo Gama (2004), é uma técnica baseada em dividir e conquistar, em que um problema é quebrado em subproblemas mais simples. O critério utilizado para realizar as partições é o da função utilidade de uma característica para a classificação. Seguindo esse critério, atribui-se um determinado ganho de informação a cada atributo, a partir do quanto esse atributo é capaz de separar

dados. O atributo escolhido como atributo teste para o corrente nó é aquele que possui o maior ganho de informação. (Figura 5).

```
INPUT: S, where S = set of classified instances
OUTPUT: Decision Tree
Require: S ≠ ∅, num_attributes > 0
 1: procedure BUILDTREE
 2:     repeat
 3:         maxGain ← 0
 4:         splitA ← null
 5:         e ← Entropy(Attributes)
 6:         for all Attributes a in S do
 7:             gain ← InformationGain(a, e)
 8:             if gain > maxGain then
 9:                 maxGain ← gain
10:                 splitA ← a
11:             end if
12:         end for
13:         Partition(S, splitA)
14:     until all partitions processed
15: end procedure
```

Figura 5 – Pseudocódigo do Algoritmo de Árvore de Decisão:
Fonte: https://www.kdnuggets.com/2016/10/decision-trees-concise-technical-overview.html

2.3.1.2. Random Forest

Random Forest, ou Floresta Aleatória, é um algoritmo de aprendizado supervisionado. Segundo seu idealizador, Leo Breiman, uma floresta aleatória consiste da combinação de T árvores de decisão, em que cada árvore é preenchida com elementos aleatórios da base total de dados. Nesse método, depois que um determinado número de árvores são geradas, cada uma lança um voto para uma classe do problema, considerando um vetor de entrada. Então, a classe mais votada (moda) será escolhida na predição do classificador. O algoritmo de Random Forest tem finalidade de adicionar aleatoriedade ao modelo, no processo de criação de suas árvores. Ao invés de procurar pela melhor característica ao fazer a partição de nodos, ele busca a melhor característica em um subconjunto aleatório das características. (Figura 6)

Algorithm 1: Pseudo code for the random forest algorithm

To generate c classifiers:
for i = 1 to c **do**
  Randomly sample the training data D with replacement to produce D_i
  Create a root node, N_i containing D_i
  Call BuildTree(N_i)
end for

BuildTree(N):
if N contains instances of only one class **then**
  return
else
  Randomly select x% of the possible splitting features in N
  Select the feature F with the highest information gain to split on
  Create f child nodes of N, N_1,..., N_f, where F has f possible values (F_1, ... , F_f)
  for i = 1 to f **do**
    Set the contents of N_i to D_i, where D_i is all instances in N that match
    F_i
    Call BuildTree(N_i)
  end for
end if

Figura 6- Pseudocódigo do Algoritmo de Random Forest:
Fonte:. https://www.semanticscholar.org/paper/Integration-of-Rules-from-a-Random-Forest-Sirikulviriya-Sinthupinyo/d63537ef1be0c9f71fac53805705ae78de959cf2

2.3.2. Algoritmos de Agrupamento (Clusterização)

Algoritmos de agrupamento consistem em algoritmos de aprendizado não-supervisionado, onde não existe rotulamento prévio da base de dados, e seu principal objetivo é agrupar objetos similares em um grupo, e objetos distintos em grupos diferentes, a partir de uma variável para comparação.

2.3.2.1. Algoritmo Mean Shift

Mean Shift é um algoritmo de agrupamento que objetiva inferir o ponto médio de uma região agrupada (clusterizada), a partir de uma função de densidade, dentro de sua região de interesse. No algoritmo Mean Shift, o centro da região de interesse tende a se deslocar ao ponto de maior densidade dentro de sua região de interesse. Após algumas iterações, o algoritmo identifica o lugar com maior densidade da região total, e só pára a execução quando não achar nenhum outro ponto com centro de massa maior que o atual (Figura 7).

Pseudocódigo

- Defina uma região de interesse circular d-dimensional de raio E;
- Para os dados dentro da região, calcule a média de distância entre todos os pontos, onde as distâncias forem menores, será considerado o novo centro da região de interesse (distâncias menores entre os pontos significam regiões mais adensadas) ;
- O algoritmo pára quando a bola estiver centrada em uma região mais densa que as vizinhas (máximo local).

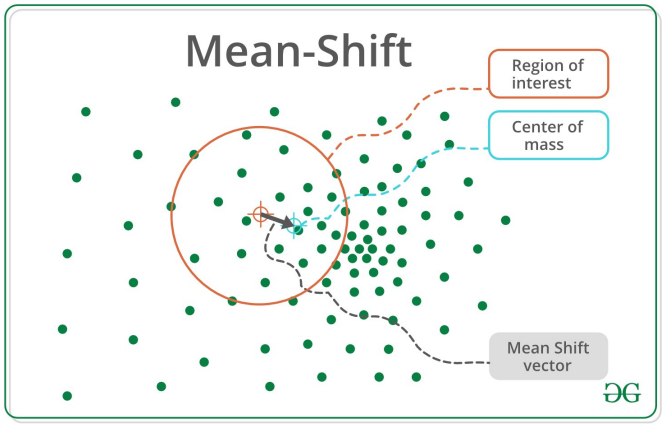


Figura 7 – Funcionamento do Algoritmo de Mean Shift
Fonte: https://www.geeksforgeeks.org/ml-mean-shift-clustering/

Dada a fundamentação teórica apresentada neste capítulo, passaremos, a seguir, à descrição dos materiais e métodos utilizados para o desenvolvimento da aplicação.

3. MATERIAIS E MÉTODOS

O objetivo principal do presente trabalho foi o de explorar os recursos técnicos do sensor de captura de movimentos KINECT-v2, da Microsoft, para sua utilização na realização de uma aplicação de animação por computação gráfica, de cunho interativo, em que os movimentos físicos do usuário são transferidos em tempo real para um personagem-avatar.

A determinação das ferramentas para a elaboração deste projeto foi precedida de pesquisa das opções existentes no mercado, que fossem compatíveis entre si e apresentassem as funcionalidades necessárias para efetuar todas as operações pretendidas.

Diante disso, os softwares escolhidos foram: Microsoft Kinect Studio, Blender 3D, o motor gráfico Unity 3D, Modelos 3d obtidos de websites com bibliotecas livres para uso não comercial; a IDE Visual Studio 2017; e a plataforma Adobe Mixamo; adiante referidos, sendo que o fator determinante na escolha foi o fato de esses softwares serem gratuitos. Foi utilizado também: o plugin Kinect v2 MS SDK for Unity, que permite uma interface integrando o dispositivo Kinect com o Unity 3D.

Para a conexão do Kinect no computador foi necessário adquirir ainda o adaptador para Windows – Microsoft Xbox One Kinect Adapter for Windows, que, apesar de sua fabricação já ter sido descontinuada, ainda é distribuído online. O computador utilizado para testes de software e realização dos projetos possui como principais especificações de configuração: Placa de Vídeo de 6GB, Processador i7 e 16GB de Memória RAM.

3.1. Sensor KINECT v2:

Figura 8 – Imagem do sensor Kinect para XBoxOne
Fonte: https://upload.wikimedia.org/wikipedia/commons/f/f6/Xbox-One-Kinect.jpg

O Kinect-V2, da Microsoft (Figura 8) é, basicamente, composto de:

- Câmera grande angular, que possui detecção de vídeo em RGB, combinando vermelho, verde e azul para formar as imagens captadas em uma resolução de 1920 x 1080 a 30 quadros por segundo (fps);
- Sensores de profundidade 3D, resolução 512 x 424, utilizando a tecnologia "tempo de vôo" para determinar os recursos do ambiente e o movimento de determinados objetos; consegue rastrear sem luz visível usando um sensor de IR (infravermelho) ativo. Devido à natureza dos sensores, o Microsoft Kinect “enxerga” tanto em ambiente iluminado, quanto no escuro (Figura 9). A faixa de detecção do sensor de profundidade é ajustável; o software é capaz de calibrar automaticamente o sensor com base na jogabilidade e no ambiente físico do jogador, acomodando a presença de móveis ou outros obstáculos. Possui um campo de visão abrangente, podendo rastrear até 6 esqueletos de uma só vez. Também pode detectar a frequência cardíaca, expressões faciais, a posição e orientação de 25 articulações individuais, incluindo polegares (Figura 10), a velocidade dos movimentos do jogador e acompanhar gestos realizados;
- Quatro microfones multi-vetoriais autodirecionáveis (Figura 9), capazes de captar comandos de voz e de isolar o ruído ambiente da fala do jogador.

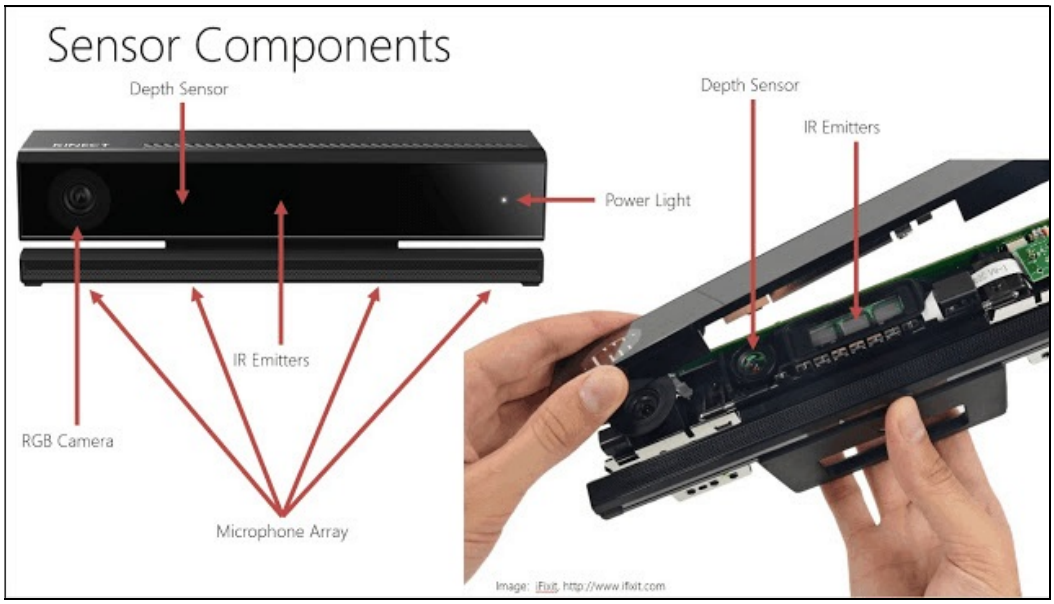


Figura 9 – Componentes do Sensor Kinect V2
Fonte: https://www.infraready.co.uk/shop/sls-kinect-for-windows-v2-sls/

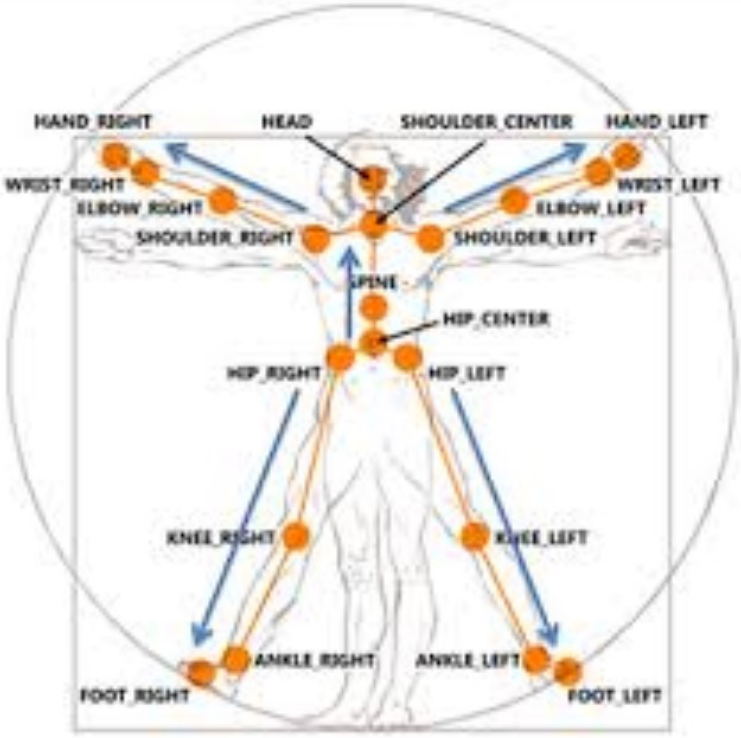


Figura 10 – Mapeamento das articulações captadas pelo Kinect v2
Fonte: https://www.inf.pucrs.br/~smusse/CG/PDFs2014_1/Kinect.pdf

3.1.1. Comparação dos hardwares do Kinect v2 X Kinect v1

		Kinect for windows v1	Kinect for windows v2
Color	Resolution	640×480	1920×1080
	fps	30fps	30fps
Depth	Resolution	640×480	512×424
	fps	30fps	30fps
Sensor		Structured light	Time of flight
Range		1.2 ~ 3.5*m*	0.5 ~ 4.5*m*
Joint		20 joint / people	25 joint / people
Hand state		Open / closed	Open / closed / Lasso
Number of Apps		Single	Multiple
Body Tracking		2 people	6 people
Body Index		6 people	6 people
Angle of View	Horizontal	62 degree	70 degree
	Vertical	48.6 degree	60 degree
Tilt Motor		Yes	No
Aspect Ratio		4:3	6:5
Supported OS		Win 7, Win 8	Win 8
USB Standard		2.0	3.0

Figura 11 – Comparativo dos aspectos técnicos do Kinect1 e Kinect2
Fonte: (http://zugara.com/how-does-the-kinect-2-compare-to-the-kinect-1)

O reconhecimento facial, rastreamento de movimento e resolução do Kinect-v2 são mais precisos que com o Kinect-v1, com melhoria na capacidade de resolução da câmera grande angular. O Kinect-v2 usa a tecnologia "tempo de voo" para determinar os recursos e movimentos de determinados objetos (Wasenmuller & Stricker, 2016). O Kinect-v2 pode processar 2 gigabytes de dados por segundo, o USB 3 fornece banda larga quase 10 vezes mais rápida para a transferência de dados; possui campo de visão 60% maior e pode detectar e rastrear 25 articulações dos corpos de 6 pessoas, incluindo os polegares; o Kinect-v1 consegue rastrear 20 articulações de 2 pessoas. O Kinect-v2 é capaz de detectar batimentos cardíacos, expressões faciais e pesos nos membros, além de dados biométricos significativos. Tem precisão no rastreamento de dedos, esticar e encolher as mãos e braços. Diante dos dados técnicos (Figura 11), verifica-se que a tecnologia do Kinect-v2 é bem mais poderosa e complexa do que a primeira geração do Kinect, razão pela qual ele foi escolhido para a realização deste trabalho.

O Kinect para Xbox One V2 foi descontinuado em outubro de 2017, sendo substituído pelo Microsoft Kinect Azure, desenvolvido para integrar projetos na plataforma de computação em nuvem Azure, e ser utilizado em soluções de inteligência artificial. No entanto, o sensor Kinect-v2 continua sendo empregado em projetos acadêmicos, em pesquisas de seus possíveis usos nas diversas áreas como, robótica, segurança, medicina, realidade virtual, entre outras.

3.1.2. Como se dá o processamento de imagens pelo Kinect-v2:

O processo de captura de movimentos com o Kinect ocorre em três etapas (Shotton et al., 2011).

-Identificação do usuário no ambiente utilizando dados de profundidade;

-Inferência de suas partes corporais através da aplicação de Random Forest e

-Inferência da posição de suas articulações através da aplicação de Mean-Shift.

Esses três processos ocorrem sequencialmente e na respectiva ordem, uma vez a cada frame (200 fps), ou seja, múltiplas computações são realizadas por segundo, frame por frame, passando a impressão de acompanhamento da movimentação ao longo do tempo.

O processamento de todos os dados das 3 etapas da captura de movimentos ocorre na própria Unidade de Processamento do Kinect, em seu SoC (System on a Chip).

Primeiramente, a identificação do usuário ocorre através dos dados captados através do seu módulo de profundidade, composto pelo Emissor de Raio Infravermelho e pela Câmera Infravermelha (Figura 12).

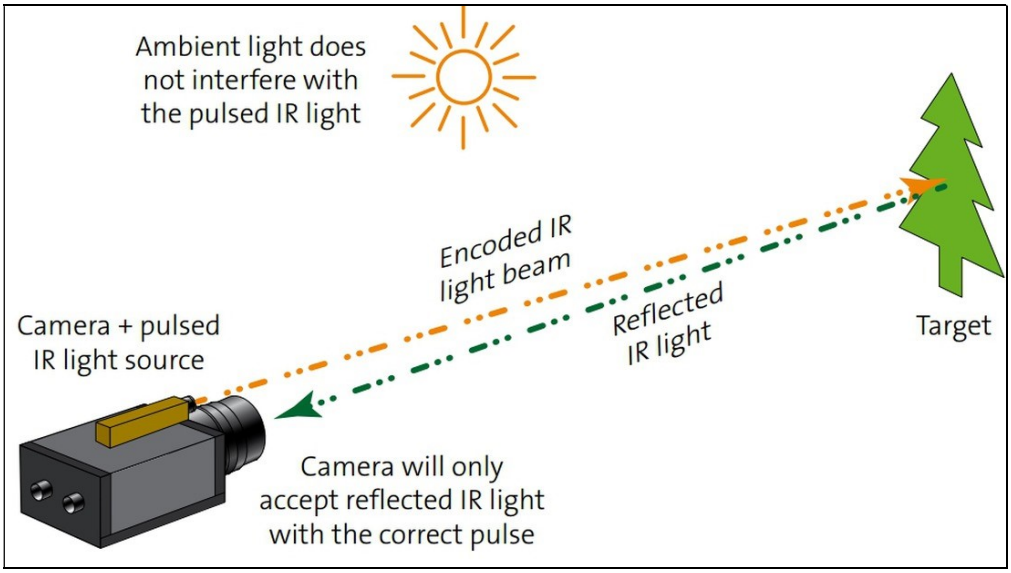


Figura 12 - Funcionamento de uma câmera de profundidade.
https://www.stemmer-imaging.com/fr-ch/donnees/cameras-3d-time-of-flight-cameras/

Os dados de profundidade são obtidos através da tecnologia de Tempo-de-Vôo, que é um método para detectar distâncias entre um sensor e um objeto alvo a partir do tempo que leva para o raio Infravermelho emitido refletir no objeto e voltar para o sensor.

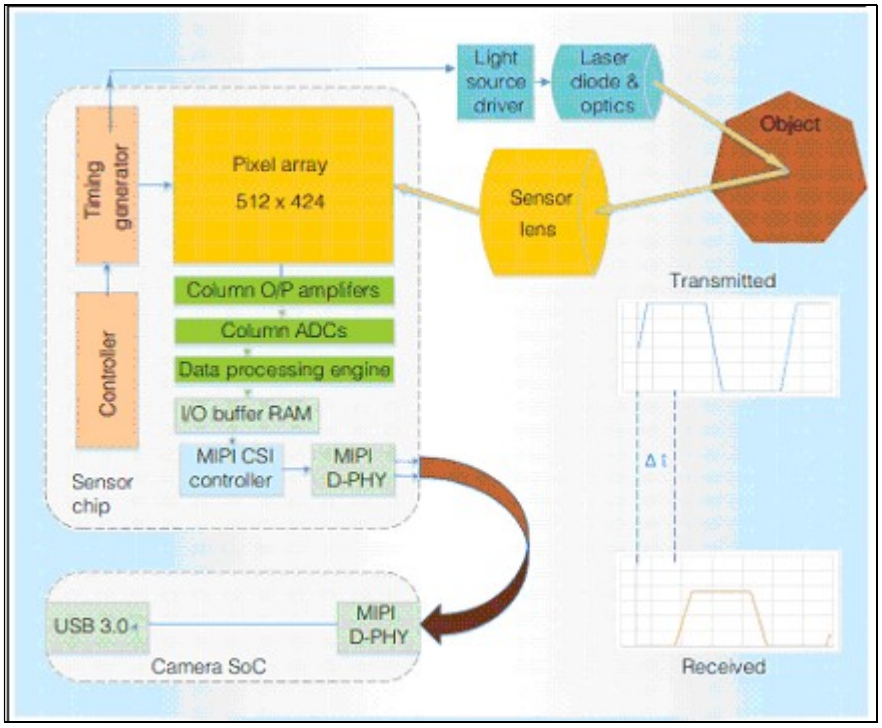


Figura 13 -.Funcionamento de um Sistema Sensor baseado em Tempo-de-Vôo
Fonte: https://www.semanticscholar.org/paper/The-Xbox-One-System-on-a-Chip-and-Kinect-Sensor-Sell-O'Connor/862387d3f044935ae1e6f6c95eb15a0676d04316 (adaptado pelo autor)

O sistema Tempo-de-vôo dispara ondas quadráticas de luzes (Figura 13), ou seja, que fogem do modelo senoidal, e variam em uma frequência intervalada constante entre os valores máximos e mínimos, dentro do campo de visão do Kinect.

A partir das imagens capturadas pela câmera e sensores do Kinect, são computadas as coordenadas de cada pixel no espaço 2D, como segue:

Um ponto de cor descreve um ponto 2D na imagem colorida. A posição no espaço de cores é uma localização de linha e de coluna de um pixel na imagem, onde x = 0, y = 0 é o pixel no canto superior esquerdo da imagem colorida e x = 1919, y = 1079 (largura- 1, altura-1) corresponde ao canto inferior direito.

Espaço de profundidade é o termo usado para descrever uma localização 2D na imagem de profundidade, equivalente à localização de linha/ coluna de um pixel, onde x é a coluna e y é a linha. Haja visto que a resolução do sensor de profundidade é de 512 x 424, Tem-se que x=0, y=0 corresponde ao canto superior esquerdo da imagem e x=511, y=423 (largura-1, altura-1) é o canto inferior direito da imagem.

Conforme documentação da Microsoft do dispositivo Kinect v2, o espaço da câmera se refere ao sistema de coordenadas 3D usado pelo Kinect. O sistema de coordenadas é definido da seguinte forma:

- A origem (X=0, Y=0, Z=0) está localizada no centro do sensor IR (infravermelhos) no Kinect;
- o valor de X refere-se à posição horizontal de um pixel, e cresce à esquerda do sensor;
- o valor de Y refere-se à posição vertical de um pixel em relação à posição do sensor, e cresce de acima da posição do sensor (A inclinação do sensor influencia a direção de Y)
- Z cresce na direção para qual o sensor está voltado;
- 1 unidade = 1 metro.

Na Figura 14, a seguir, é possível observar que a imagem formada é composta por ruído, em razão do intervalo de tempo entre um disparo e outro, de feixes de raios infravermelhos.

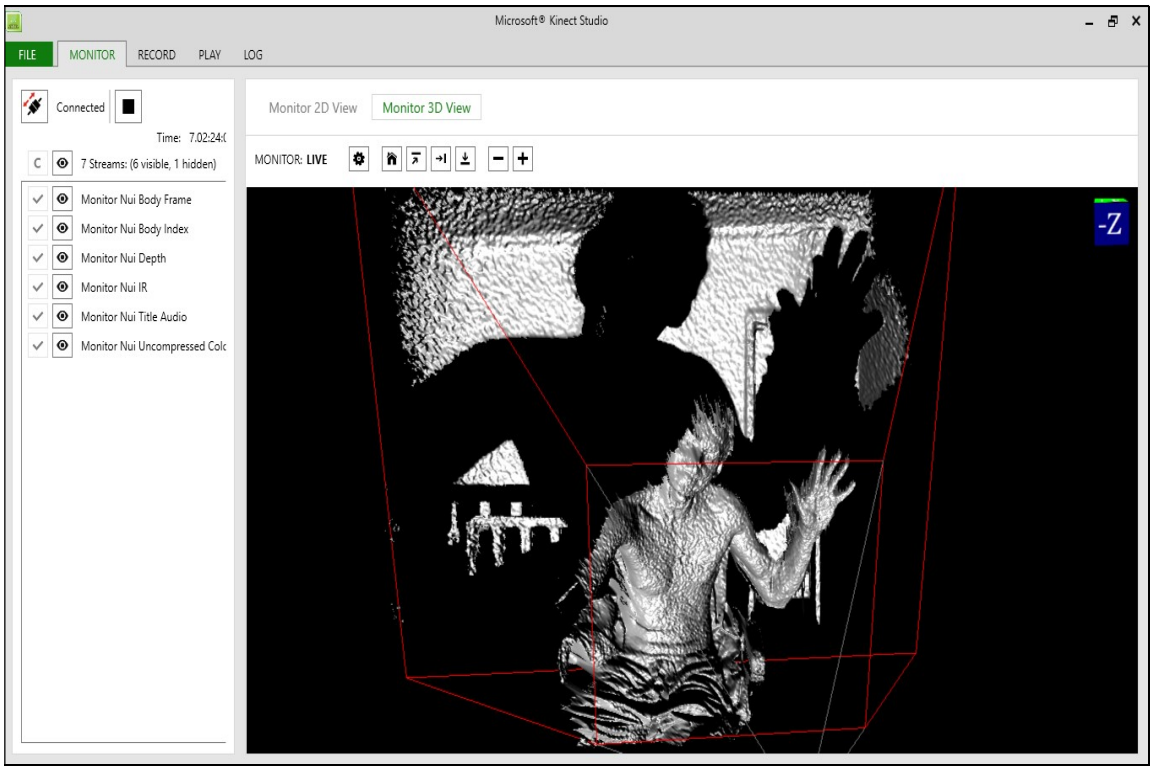


Figura 14 -Imagem captada pelo módulo de Profundidade (Nota-se o detalhamento na reconstrução dos ambientes apenas onde houve reflexão dos raios infravermelhos).

O sistema calcula a profundidade, a partir da análise de cada pixel Infravermelho refletido no ambiente e captado pela Câmera IR. O tempo que leva para um mesmo pixel Infravermelho ser emitido, refletido e captado pela câmera Infravermelha é o que constitui o Delta "t", que é proporcional ao dobro da distância da câmera até o objeto (ida e volta). O cálculo da distância é feito a partir do valor da velocidade da luz no ar: 299.792.458 m/s, normalizado em aproximadamente 1 cm em 33 picosegundos. No caso, a equação para definição da profundidade de cada pixel é dada por:

$$d = \frac{Ct}{2}$$

Figura 15 – Equação para cálculo da profundidade.
Fonte: (Shotton, 2011), imagem adaptada pelo autor.

Na equação acima, "d" é a Profundidade a descobrir, "c" equivale à velocidade da luz e "t" é a variável referente ao intervalo de tempo Delta Time. Ao final do cálculo, o numerador é dividido por 2, uma vez que a distância obtida corresponde à distância de ida e volta.

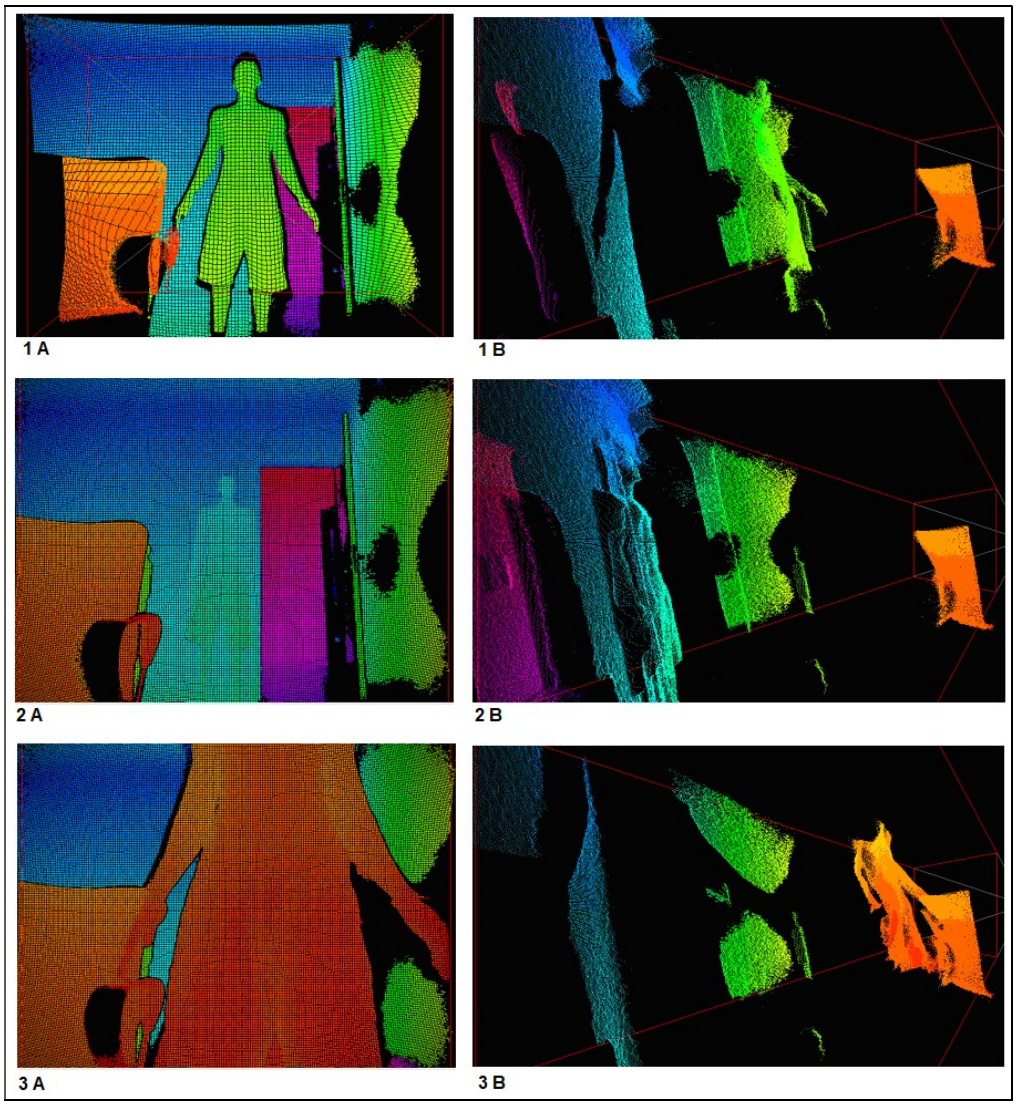
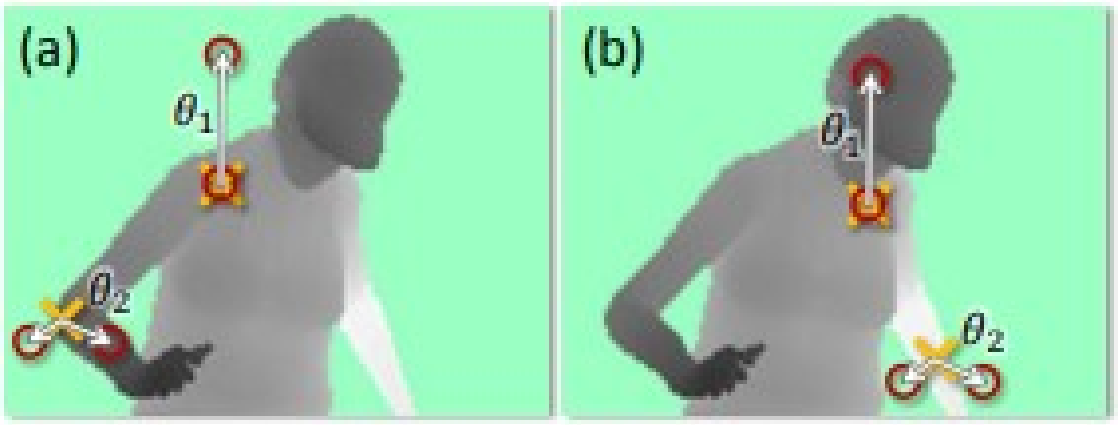


Figura 16 – Testes do sensor em 3 níveis de profundidade, representados em escala de cor RGB A=frente e B=lateral; Nível-1: autor a 1,5m do sensor; Nível-2: autor a 3,0m do sensor; e Nível-3: autor a 0,5m do sensor

A partir da imagem de profundidade (Figura 16), o Kinect passa para a próxima etapa do processo (lembrando que o conjunto de processamentos realizado sequencialmente pelo Kinect ocorre múltiplas vezes por segundo, e o processamento que ocorreu em um frame anterior não tem relação com o frame seguinte a ser computado, uma vez que os processamentos sequenciais são frame a frame). . Para essa etapa, o algoritmo de Random Forest é aplicado para realizar a classificação dos pixels da imagem e inferir sua posição no corpo, ou, se está fora do corpo, junto a pivôs escolhidos para a comparação de profundidade entre os pixels (Figura 17), que funcionam como características (features) a serem usadas no processo de classificação em cada Decision Tree que compõe a Random Forest.



.

Figura 17 – Comparação de profundidade “theta” entre um pixel “x” (cruzes amarelas) e um pivô “u” (círculos vermelhos)
Fonte: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/BodyPartRecognition.pdf

O modelo de predição foi construído a partir de uma base de dados com mais de centenas de milhares de “frames” retirados de sessões de gravação de movimento e imagens geradas através de Computação Gráfica, onde os atores tinham as partes do corpo identificadas (pintadas) para treinamento de algoritmo. Para treinamento, diversas árvores são geradas a partir de nós aleatórios, constituídos de amostragem de “frames” aleatórios igualmente distribuídos.

Os “features” são construídos a partir de pixels aleatórios dos diferentes “frames” da base de dados, e comparados com o pixel do nó de entrada a partir da equação da Figura 18.

$$f_\theta(I,\mathbf{x}) = d_I\left(\mathbf{x}+\frac{\mathbf{u}}{d_I(\mathbf{x})}\right) - d_I\left(\mathbf{x}+\frac{\mathbf{v}}{d_I(\mathbf{x})}\right),$$

Figura 18 – Equação para definição de features através da diferença de profundidade entre um frame x e um pivô com distância delta T de x.
Fonte : Shotton, 2011

Após a etapa de inferência das partes do corpo, a próxima etapa consiste em identificar a posição das articulações, e isso é feito através do algoritmo Mean-Shift, um algoritmo de agrupamento por densidade, junto à adição de pesos nos pixels para a normalização do resultado.

O estimador de densidade consiste no produto da probabilidade do pixel em questão estar contido na parte do corpo inferida, juntamente com a área de

superfície que o pixel em análise ocupa. Quanto maior o peso, maior o seu centro de influência e maior a probabilidade de esse pixel ser uma das articulações do corpo. Ao final do processamento de todos os pixels, o Kinect conecta as articulações adjacentes, conforme hierarquia apresentada na Figura 19, formando os ossos do esqueleto. A ligação entre uma “articulação pai” (parent joint) e uma “articulação filho”(child joint) resulta em um osso do esqueleto.

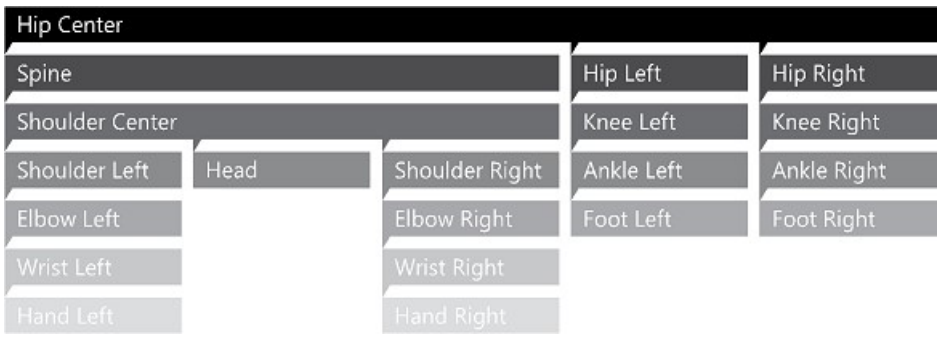


Figura 19 –Hierarquia de ramificações que formam o esqueleto
https://social.msdn.microsoft.com/Forums/getfile/512551

O processo de obtenção do esqueleto do Kinect é dado a partir do conjunto de técnicas apresentado anteriormente. O SDK desenvolvido pela Microsoft disponibiliza o acesso a todas as funcionalidades do Kinect via código, com exceção dos modelos de treinamento. Para a realização da transferência do esqueleto gerado pelo Kinect a partir das imagens de profundidade, foi utilizado o motor gráfico e ambiente Unity 3D. Através do plugin desenvolvido por Rumen Filkov, foi possível acessar os dados de posição relativa dos joints captados pelo Kinect e renderizá-los num espaço 3D.

3.2. Demais Ferramentas e Softwares utilizados:

3.2.1. Kinect Studio

Kinect Studio é uma ferramenta do Kinect v2 SDK 2.0 que permite, a pré-visualização e monitoramento dos movimentos e dados captados pelos sensores do Kinect; a gravação e reprodução dos mesmos em uma extensão padrão do SDK (.xef); manipular a linha do tempo e realizar cortes rápidos, e alternar entre as perspectivas das gravações entre 2D e 3D.

Sua interface é basicamente composta por 4 abas principais: Monitoramento, Gravação, Reprodução e Log. A aba de Monitoramento é designada para reproduzir a visão do Kinect no momento, e para isso ocorrer, o sensor precisa estar conectado à maquina. Na aba de Gravação, são feitas as gravações dos trechos que estão sendo monitorados pela aba anterior, os arquivos gravados são salvos na extensão (.XEF). A aba de Reprodução permite reproduzir os arquivos (.xef) que foram previamente gravados, e a aba Log, permite realizar anotações sobre cada gravação.

Na pré-visualização disponibilizada pelo Kinect Studio, é possível monitorar dados de cor, profundidade e de rastreamento do corpo, captados pelos diferentes sensores, bem como a visualização do esqueleto gerado pelo Kinect (Figura 20), formado por 25 *joints* (articulações), que, aos pares, formam ossos virtuais entre os joints, e os estados da mão do usuário, que variam entre: aberto, fechado e em descanso.

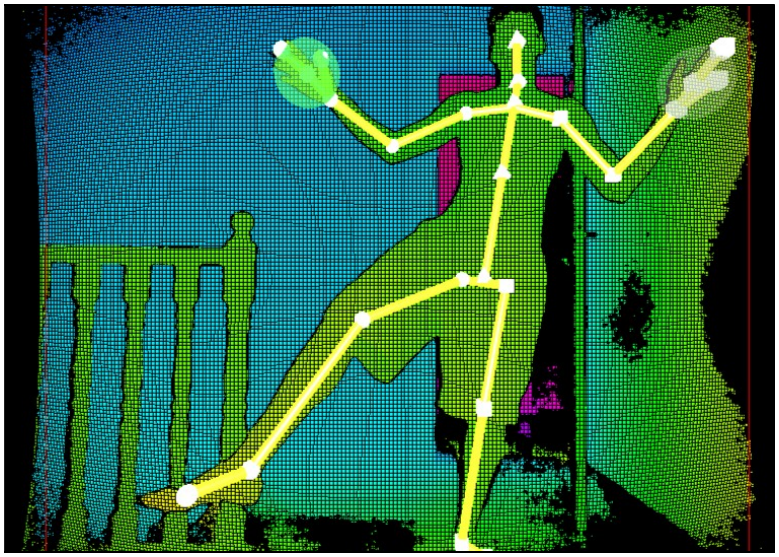


Figura 20 –.Exemplo de squeleto gerado com o processamento de captura de imagem pelo Kinect v2 - Fonte: O autor (2020)

3.2.2. Blender 3D

O software Blender 3D, (BLENDER..., 1998) desenvolvido e distribuído pela Blender Foundation, é um software de código livre voltado ao pipeline 3D, para modelagem, animação, renderização, rigging, simulações físicas e texturização. (Figuras 21 e 22).

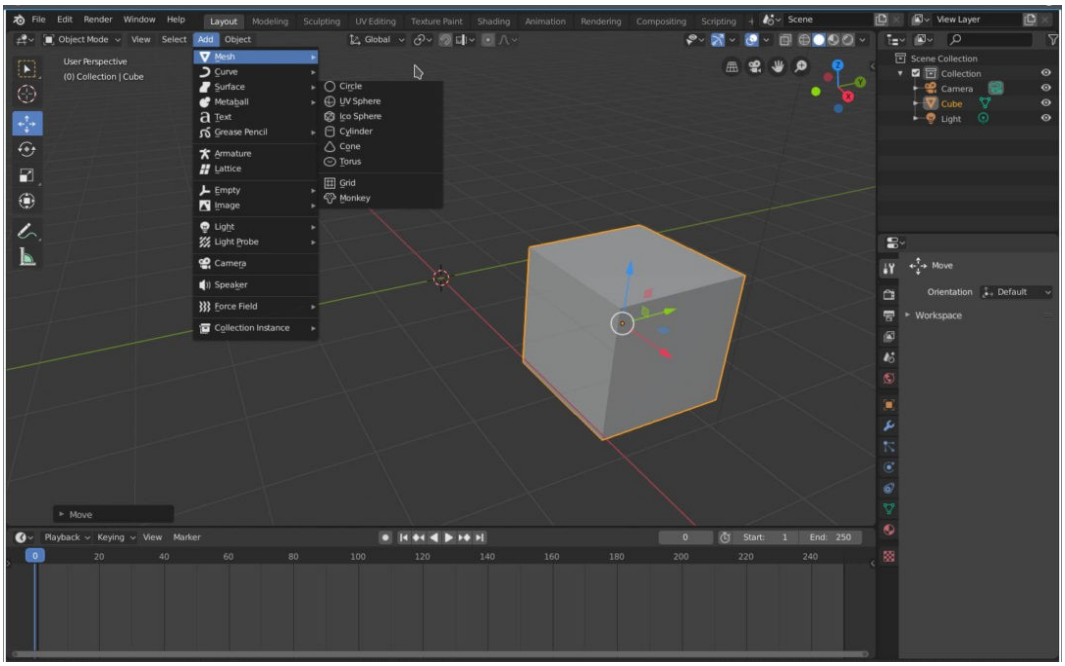


Figura 21 - Interface Padrão do Blender3D Fonte: O autor (2020)

Atualmente se encontra na sua versão 2.80 e conta com ferramentas equivalentes aos seus concorrentes diretos, Autodesk 3ds Max e ZBrush, que possuem licenças pagas para o seu uso (motivo pelo quais foram opções descartadas).

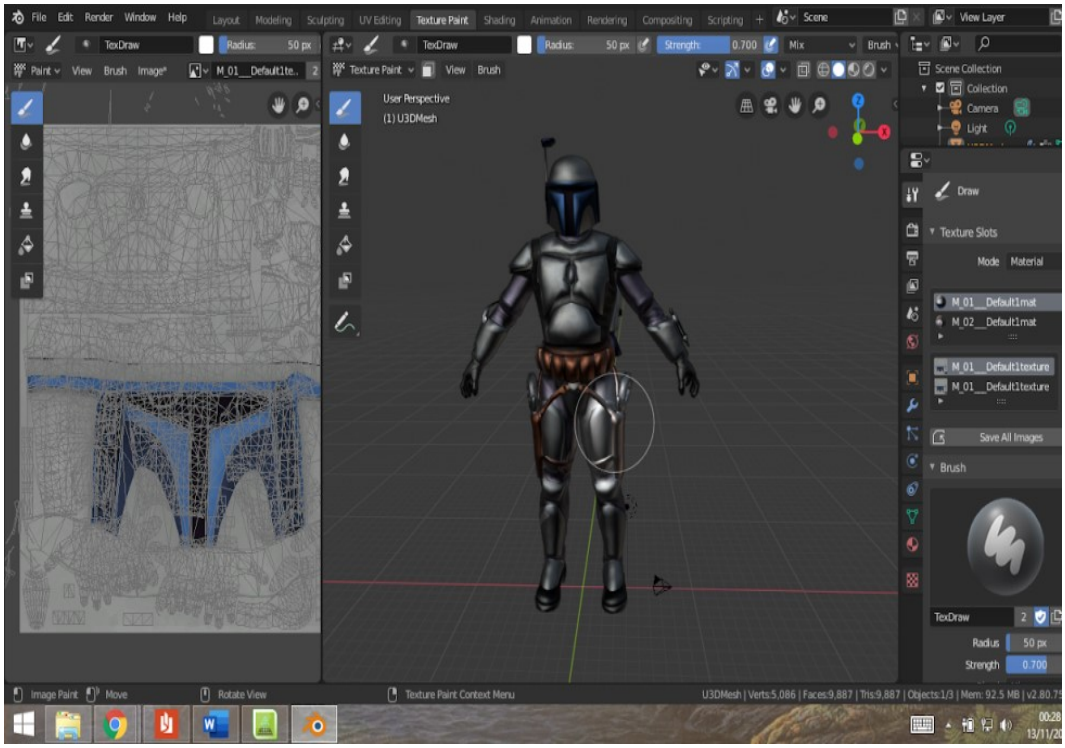


Figura 22 - Aba de Texturização do Blender - Fonte: O autor (2020)

3.2.3. Unity 3D

Desenvolvido e mantido pela Unity Technologies, o Unity 3D (UNITY..., 2005) é o motor gráfico (engine) mais popular do mundo (Figura 23), com milhões de jogos desenvolvidos para plataformas móveis, desktop e consoles, razão pela qual há um grande número de recursos disponíveis para uso sem custo, além de diversos plugins que garantem sua integração com frameworks externos, como TensorFlow da Google para Machine Learning, e com dispositivos externos, como aparelhos móveis, câmeras, óculos de Realidade Virtual, smart-watches e ferramentas de Realidade Aumentada. Um "*motor gráfico*" consiste em um conjunto de ferramentas para auxiliar e facilitar o desenvolvimento de jogos, com recursos que auxiliam na criação de funções gráficas e opções para acrescentar variáveis de física aos objetos, entre outras características.

Figura 23 – Interface Unity 3D (motor gráfico)
Fonte: https://apkpure.com/br/3d-unity-manual-reference/com.unity3d.manualforpcmac.learngameengine

3.2.4. K2-asset Kinect-v2 MS SDK for Unity

Trata-se de plugin (K2-ASSET..., 2014) que provê interface para uso da biblioteca de funcionalidades do Kinect dentro do motor gráfico de jogos Unity 3D. Ferramenta desenvolvida e comercializada por Rumen Filkov, com licença liberada para Estudantes, Livre para uso acadêmico, com permissão concedida pelo desenvolvedor para uso neste trabalho, via email, após pedido.

3.2.5. Download de Modelos 3D fbx grátis

Modelos 3D foram obtidos de websites com bibliotecas livres para uso não comercial: p3dm.ru; free3d.com; turbosquid.com; cgtrader.com. Foram baixados gratuitamente para uso na aplicação, personagens fictícios: heróis, vilões e figuras de filmes de cinema: o Homem de Ferro; Capitão América; Homem-Aranha; Thor; Wolverine; Hulk; Thanos; Darth Vader; Stormtrooper; C3PO e R2D2; Mínions e Bob Esponja.

3.2.6. Visual Studio 2017

O Visual Studio 2017 é IDE para codificação completa que possui extensões para as mais diversas linguagens de programação.

3.2.7. Adobe Mixamo

A plataforma Adobe Mixamo (MIXAMO..., 2015) é ferramenta online gratuita utilizada para fazer upload de personagens 3D. Fornece serviço de modelagem automática online conhecido como Auto-Rigger, que aplica aprendizado de máquina para entender onde estão os membros de um modelo 3D e inserir um "esqueleto", ou rig no modelo. O sistema também disponibiliza modelos 3D para download e sequências de animação que funcionam com personagens manipulados com o AutoRigger da plataforma Mixamo.

4. DESENVOLVIMENTO E RESULTADOS

Este capítulo apresenta os resultados obtidos no desenvolvimento do trabalho, em que realizamos aplicação prática para verificação do desempenho do Kinect v2 na captura de movimentos físicos e respectiva transferência para um avatar virtual, utilizando as ferramentas e métodos citados na Seção 3.

O processo iniciou com a obtenção do esqueleto virtual, pelo Kinect v2, a partir do conjunto de recursos e algoritmos utilizados pelo dispositivo, conforme apresentado na Seção 3.1.2. Para a transferência do esqueleto gerado pelo Kinect, foi utilizado o motor gráfico *"Unity 3D"*, bem como efetuada a importação do plugin *"Kinect v2 MS SDK for Unity"* (desenvolvido por Rumen Filkov) como interface para acesso aos dados processados pelo Kinect no ambiente do Unity 3D. O arquivo *"KinectAvatarsDemo",* do plugin, possui scripts pré-programados com a finalidade de demonstrar como transferir os dados de posição e rotação das articulações (joints) captadas pelo Kinect, para um avatar de esqueleto correspondente. (Figura 24).

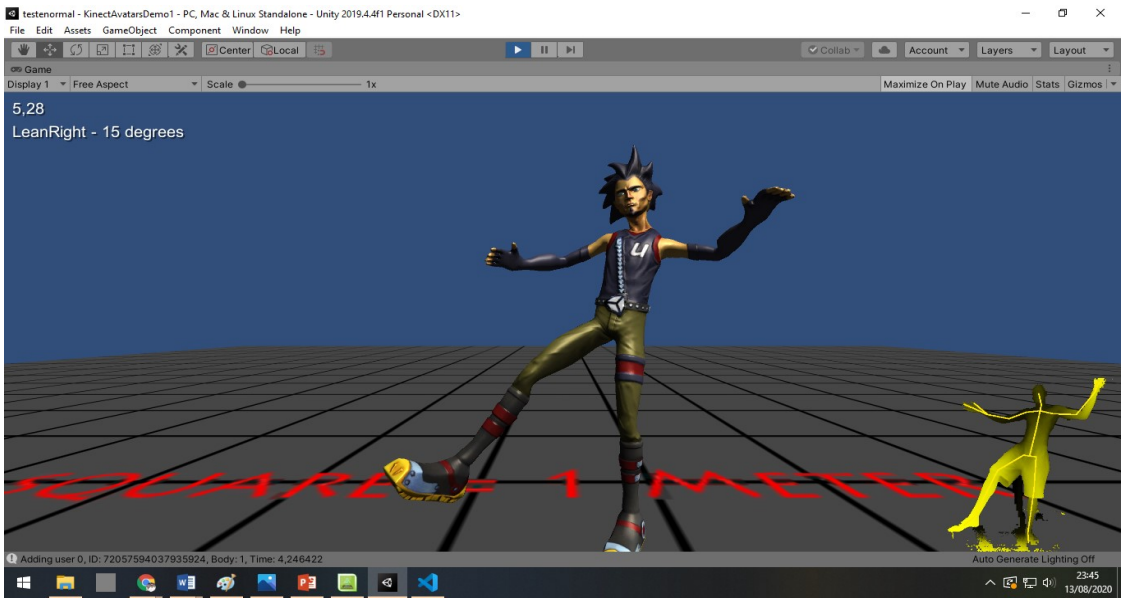


Figura 24: Personagem – Avatar padrão disponibilizado no arquivo KinectAvatarsDemo.
Do lado direito observa-se área de profundidade com o esqueleto inferido pelo Kinect v2
Fonte: O autor (2020)

Foram utilizados 12 modelos 3D de personagens fictícios, de diferentes estaturas e conformações, os quais foram baixados de bibliotecas livres citadas na Seção 3.2.5. Através da ferramenta "Blender" todas as texturas foram importadas e anexadas aos devidos lugares nos personagens, dando cor e aparência a eles. Em seguida, foi necessário unificar os modelos compostos por partes do corpo

modulares, em um único objeto 3D. Após, foi utilizada a plataforma online gratuita Adobe Mixamo para anexar um esqueleto proporcional aos tamanhos e formatos dos avatares, gerando automaticamente através da detecção de pontos chave na malha 3D do personagem. Esses esqueletos possuem hierarquia similar ao do Kinect, com a exceção dos ossos da mão, que não são mapeados pelo Adobe Mixamo. Após a importação, na Unity, de personagem com estrutura esquelética correspondente à do Kinect, foram feitas transferências dos dados de joints captados pelo Kinect, para o novo esqueleto através do script AvatarsController (Figura 25).

```
Algoritmo para transferência
//Declaração de todos os joints que vão receber os dados do Kinect

public GameObject joints = [
public GameObject Hip_Center,
public GameObject Spine_1,
...
...
public GameObject Ankle_Right,
public GameObject Foot_Right,
];

//Ao inicializar
Start(){

//Mapeia os joints do personagem para receberem os dados dos joints
captados pelo kinect
MapBones()

}

//Método Update é executado a cada frame
Update(){
UpdateAvatar(){
//para todo bone no array de bones
for (var boneIndex = 0; boneIndex < bones.Length; boneIndex++)
        {
                //Transformação de Rotação
                TransformBone(UserID, joint, boneIndex){
                        //Pega a orientação do joint captado
pelo kinect
                        Quaternion jointRotation =
kinectManager.GetJointOrientation(userId, iJoint);


    //calcula nova orientação do osso ao multiplicar a orientação do joint
captada, pela rotação inicial dele.
Quaternion newRotation = jointRotation * initialRotations[boneIndex];


};
}
```

Figura 25– Algoritmo para transferência de dados do esqueleto
Fonte: Script AvatarsController do plugin *Kinect v2 MS SDK for Unity"*

A transferência de movimentos baseia-se no acompanhamento de uma articulação (joint) raiz, com atualização da orientação de cada "joint filho", a partir dos dados captados pelo sensor Kinect-v2 (ver Figura 19). A rotação dos joints funciona da mesma forma que no corpo humano, sendo que um "osso filho" acompanha a movimentação do "osso pai", sem necessariamente ter a sua rotação alterada. Esses ossos receberão do Kinect v2 os dados de nova orientação das articulações, a cada frame, em tempo real; resultando na transferência de movimento captado e na ilusão da animação a partir da construção de poses ao longo do tempo.

Figura 26– Tela inicial da aplicação elaborada - Fonte: O autor (2020)

A aplicação se passa dentro de um “lobby” em um cenário de caixa de brinquedos, onde 12 personagens 3D estão à disposição para uso (Figura 26). As entradas de controle são feitas através de movimentos e ações da mão do usuário. Quando a mão se fecha, o Kinect reconhece e permite a adição de ações em uma função OnHandGripOn(), e com isso foi adicionada a funcionalidade de agarrar um personagem para selecioná-lo, e ao soltá-lo com a função OnHandGripOff() e mantendo a mão aberta, um timer de 5 segundos é disparado, ao fim do qual carrega a cena virtual referente ao personagem selecionado. Cada cena permite o uso de Mocap com personagens por 15 segundos e ao seu fim, retorna ao “lobby” inicial.

Em razão da compatibilidade entre as hierarquias de esqueletos geradas pelo Mixamo e Kinect-v2, os procedimentos de transferência esqueletal foram bem sucedidos, ou seja, a adaptação dos movimentos do usuário à estrutura do personagem ocorreu de forma esperada, mesmo em personagens de diferentes tamanhos e proporções anatômicas. (Figuras 27 e 28).

Figura 27: Personagens de proporções diferentes do usuário, porém com a mesma organização do esqueleto. Fonte: O autor (2020)

Figura 28: Personagens de proporções semelhantes ao usuário mantendo a organização do esqueleto. Fonte: O autor (2020)

Em casos específicos, observamos que os movimentos captados apresentaram falhas em ocasiões em que os algoritmos do Kinect não conseguiram inferir um esqueleto corretamente. Na figura 29 é possível notar o movimento de chute sendo reproduzido de forma incorreta, uma vez que o pé do usuário chegou a ficar muito próximo do sensor, obstruindo parte do campo de visão do Kinect. No canto inferior direito encontra-se a imagem de profundidade gerada pelo sensor. Nesse caso, os feixes de raios infravermelhos foram refletidos num plano muito próximo ao sensor, obstruindo o campo de visão em relação aos elementos mais distantes no espaço físico:

Figura 29: Falha de captação de imagem em movimento. Apesar da alta taxa de atualização, movimentos bruscos e muito rápidos podem confundir o sensor. Fonte: O autor (2020)

A rotação dos pés foi outro ponto que apresentou comportamento instável em certos momentos. No caso, o fato de os dois pés estarem apoiados no chão diminui a distância de profundidade entre os dois elementos (pé e chão), reduzindo a eficácia da análise dessa região pelos algoritmos do Kinect (Figura 30).

Figura 30: Comportamento instável na captação da rotação dos pés - Fonte: O autor (2020).

Após o seu desenvolvimento, essa aplicação prática foi compilada em formato executável (.exe), compatível com sistemas Windows e Mac. Para sua utilização, é necessário uso do Kinect v2, junto de seu adaptador e o SDK do Kinect v2, versão oficial desenvolvida pela Microsoft. Como se trata de resultado audiovisual, esse trabalho acompanha um vídeo suplementar demonstrativo.

5. CONCLUSÃO

No desenvolvimento desse trabalho pudemos explorar o recursos técnicos do Kinect v2, da Microsoft, um sensor de movimentos de baixo custo, originalmente desenvolvido para acompanhar consoles de vídeo game, e analisar seu comportamento em aplicação de captura de movimentos físicos de uma pessoa, e a sua reprodução em personagem-avatar virtual.

Foram abordados também neste trabalho, fundamentos relacionados com os sistemas de captura de movimento mais utilizados e empregados em animações computadorizadas de filmes ou jogos, o que proporcionou uma maior compreensão sobre os processos envolvidos nesses tipos de produção virtual.

Verificamos que o dispositivo Kinect v2, originalmente desenvolvido para uso juntamente com o videogame XBox, e, apesar de possuir tecnologia do ano 2014, pode ser utilizado acoplado a um computador e realizar performances sofisticadas, apenas utilizando poucas ferramentas de adaptação que não exigem muito investimento.

Devido aos seus recursos e tecnologia de sistema baseado em algoritmos que processam os dados de imagens captados pelo dispositivo, o Kinect se mostrou eficiente na função de capturar movimentos e transferi-los para um personagem-avatar virtual. Em determinadas situações, no entanto, foi observado que ele apresentou algumas falhas na detecção de partes do corpo do usuário, em poses próximas ao sensor, ou quando partes do corpo saíram momentaneamente do seu campo de visão, sendo que essas ocorrências foram rapidamente corrigidas pelo próprio dispositivo, nos frames seguintes.

Concluímos assim, que o sensor Kinect v2 é uma alternativa econômica que poderia ser mais utilizada no desenvolvimento de aplicações em diversas áreas, como: entretenimento, medicina, segurança, robótica e visão computacional.

REFERÊNCIAS

Referências Bibliográficas:

WILLIAMS, Richard. *Manual da Animação*. São Paulo: Senac São Paulo, 2016.

WASENMULLER, Oliver, STRICKER, Didier, *Comparison of Kinect V1 and V2 Depth Images in Terms of Accuracy and Precision*, German Research Center for Artificial Intelligence (DFKI), 2016, Disponível em: (https://www.dfki.de/fileadmin/user_upload/import/8767_wasenmuller2016comparison.pdf)

SHOTTON, Jamie, Body Part Recognition from Single Depth-Frame, disponível em: (https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/BodyPartRecognition.pdf)

Referências de artigos e/ou matérias de sites:

JONAS JUNIOR*, Por trás das câmeras – O que é Captura de Movimento?*, Cinéfilos anônimos, 2017, Disponível em: (http://www.cinefilosanonimos.com.br/por-tras-das-cameras-o-que-e-captura-de-movimento/)

TEDDYKGAMING, *Behind The Scenes - Marvel's Spider-Man, 2018*, Disponível em: (https://www.youtube.com/watch?v=lbmuQGJ6WHs)

Nolan North Motion Capture ‘Uncharted 4’, *И*гровое сообщество ps4land, 2015, Disponível em: (https://www.youtube.com/watch?v=L2f5ExMWvF0)

The Virtual Assist, *How Hollywood makes High Quality Block Buster VFX Movies so Fast? – Part 2*, 2017, Disponível em: (https://thevirtualassist.net/visual-effects-movies-hollywood/)

Brito, Allan, Conheça os diferentes tipos de captura de movimento para animação e jogos, 2014, Disponível em: (https://www.allanbrito.com/2014/11/17/conheca-os-diferentes-tipos-de-captura-de-movimento-para-animacao-e-jogos/)

MOTION Capture - Introdução à Tecnologia. In: SILVA, Fernando Wagner. Motion Capture - Introdução à Tecnologia. [S. l.], 1997. Disponível em: https://www.visgraf.impa.br/Projects/mcapture/publ/mc-tech/. Acesso em: 12 ago. 2020.

Kinect, Wikipedia.org, Disponível em (https://en.wikipedia.org/wiki/Kinect)

Suporte da Microsoft, *Perguntas frequentes sobre Privacidade e segurança online no Kinect para Xbox 360*, Disponível em: (https://support.microsoft.com/pt-br/help/4482932/xbox-360-kinect-for-xbox-360-privacy-and-online-safety-faq)

Microsoft OEM Kinect Adapter for Windows, disponível em: (https://www.amazon.com/Microsoft-OEM-Kinect-Adapter-Windows/dp/B00NMSHT7E)

BLENDER. 2.8. ed. [S. l.]: Blender Foundation, 1998. Disponível em: https://www.blender.org. Acesso em: 26 jul. 2020.

K2-ASSET Kinect v2 Examples with MS-SDK for Unity. [S. l.]: Rumen Filkov, 2014. Disponível em: https://rfilkov.com/2014/08/01/kinect-v2-with-ms-sdk/. Acesso em: 20 maio 2020.

MIXAMO. [S. l.]: Adobe Systems, 2015. Disponível em: https://www.mixamo.com/#/. Acesso em: 15 jun. 2020.

DIHL,Leandro, Kinect for Windows sensor and SDK, disponível em (https://www.inf.pucrs.br/~smusse/CG/PDFs2014_1/Kinect.pdf)

https://social.msdn.microsoft.com/Forums/en-US/1c068899-cd0d-437d-9a95-dbf2eac42487/kinect-v20-get-hierarchical-joint-orientation-c?forum=kinectv2sdk

APÊNDICE A - Permissão concedida, via e-mail, por Rúmen Filkov, desenvolvedor do plugin “Kinect-v2 MS SDK for Unity” (K2-asset), para utilização gratuita neste trabalho com fins acadêmicos.

Rumen Filkov
para mim

inglês > português Traduzir mensagem

Hello Luiz Felipe,

Of course, you have my permission to use the K2-asset as a base for your graduation project. No problem at all.

Greetings,
Rumen