Universidade Estadual Paulista “Júlio de Mesquita Filho”
Faculdade de Ciências - Campus de Bauru
Departamento de Computação
Bacharelado em Ciência da Computação

Implementação de oclusão mútua por meio da utilização de sensor de profundidade para sistemas de estúdio virtual

Emerson Belancieri de Souza
RA: 151021971

Orientador:
Prof. Dr. Antonio Carlos Sementille

Bauru
2018

1 Introdução

Problema

A oclusão mútua entre objetos reais e virtuais é essencial em um sistema de estúdio virtual baseado em realidade aumentada.

Tradicionalmente o processo é realizado em um esquema de camadas, tratando os objetos como planos, e não considerando sua profundidade como contínua.

Outra limitação desse esquema é a necessidade de marcadores fiduciais presentes na cena capturada.

Oclusão mútua em camadas

Fonte: Aguilar (2017)

Objetivo Geral

Implementar um método de oclusão mútua utilizando sensores de profundidade, o qual poderá ser usado em estúdios virtuais baseados em realidade aumentada

Objetivos Específicos

- Implementar um método de calibração de um sensor de profundidade com uma câmera RGB externa em uma aplicação do motor de jogos Unity3D
- Implementar um método de oclusão mútua utilizando dados do sensor de profundidade em uma aplicação do motor de jogos Unity 3D
- Análise dos resultados do método de oclusão mútua por meio de experimentos

2 Fundamentação Teórica

Realidade Aumentada

Um sistema de realidade aumentada (RA) complementa o mundo real com objetos virtuais que parecem coexistir no mesmo espaço que o mundo real.

Esse sistema deve:

- Ser interativo;
- Executar em tempo real;
- Combinar objetos reais e virtuais em um ambiente real;
- Alinhar os objetos reais e virtuais corretamente.

Sensores de Profundidade

São dispositivos que empregam tecnologias para a medição da distância entre o sensor e um objeto no ambiente dentro de sua área de captura.

Atualmente existem três tipos distintos:

- Luz estruturada;
- Time-of-Flight (ToF);
- Estereoscópico.

Luz estruturada

Funcionamento:

1. Um padrão de luz conhecido é projetado no ambiente;
2. Uma câmera captura o padrão projetado;
3. A distância entre o sensor e os objetos é calculada baseando-se nas deformações no padrão capturado.

Características:

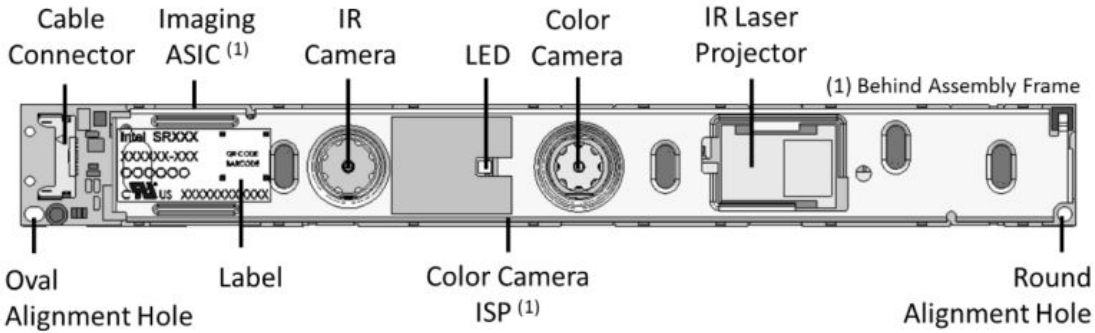
- Possui curto alcance;
- Depende de algoritmos de visão computacional para analisar as imagens capturadas;
- Facilmente afetado por outras fontes de luz do mesmo tipo utilizado para a projeção.

Luz estruturada - SR300

A câmera Intel RealSense SR300 é um exemplo de dispositivo que utiliza luz estruturada.

Composta por:

- Um projetor infravermelho;
- Uma câmera infravermelha;
- Uma câmera RGB.

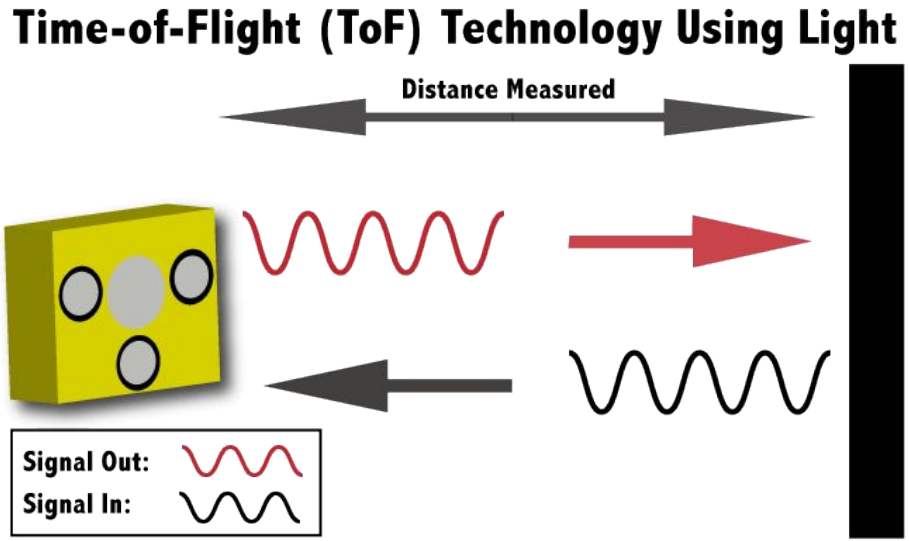


Fonte: Intel (2016)

Time-of-Flight

Funcionamento:

1. Um sinal é emitido no ambiente;
2. Após ser refletido pelos objetos no ambiente, o sinal é captado por receptores;
3. A distância entre o sensor e os objetos é calculada com base no tempo entre a emissão do sinal e a sua captura pelo receptor.



Fonte: Terabee (2018)

Time-of-Flight

Características:

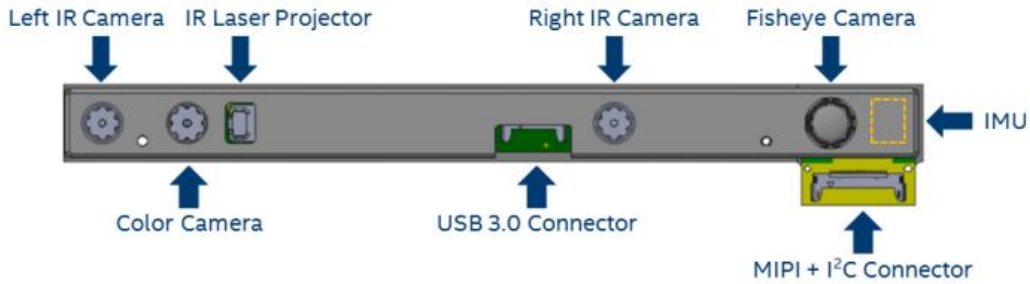
- Pode apresentar um grande alcance, de acordo com o sinal utilizado;
- A distância é determinada com uma fórmula matemática simples;
- Também afetado por outras fontes de sinal do mesmo tipo utilizado pelo sensor, porém em menor grau do que nos sensores de luz estruturada.

Time-of-Flight - ZR300

A câmera Intel RealSense ZR300 é um exemplo de dispositivo que utiliza Time-of-Flight.

Composta por:

- Um projetor infravermelho;
- Duas câmera infravermelhas;
- Uma câmera RGB;
- Uma câmera olho de peixe.



Fonte: Intel (2017)

Método Estereoscópico

Funcionamento:

1. O ambiente é capturado por dois ou mais ângulos;
2. Estima a profundidade dos objetos no ambiente utilizando algoritmos de visão computacional baseado nas diferenças entre as duas imagens capturadas;

Características:

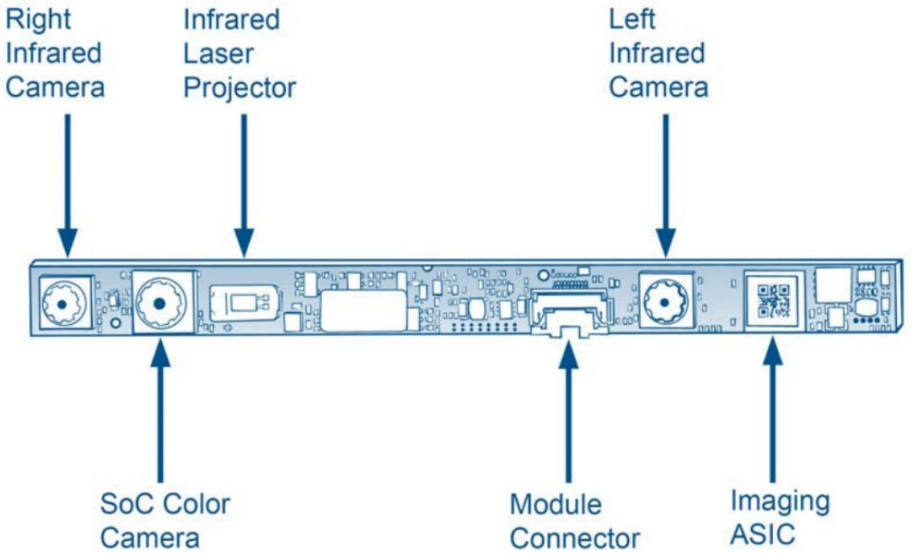
- Possui curto alcance;
- Depende de algoritmos de visão computacional para analisar as imagens capturadas;
- Facilmente afetado por outras fontes de luz do mesmo tipo utilizado para a projeção.

Método Estereoscópico - D400

As câmeras Intel RealSense da série D400 utilizam o método estereoscópico para o cálculo de profundidade.

Compostas por:

- Um projetor infravermelho;
- Duas câmera infravermelhas;
- Uma câmera RGB.



Fonte: Keselman et al. (2017)

Calibração de Câmeras

O processo para calibração de um sensor de profundidade com uma câmera é análogo a calibração de duas câmeras.

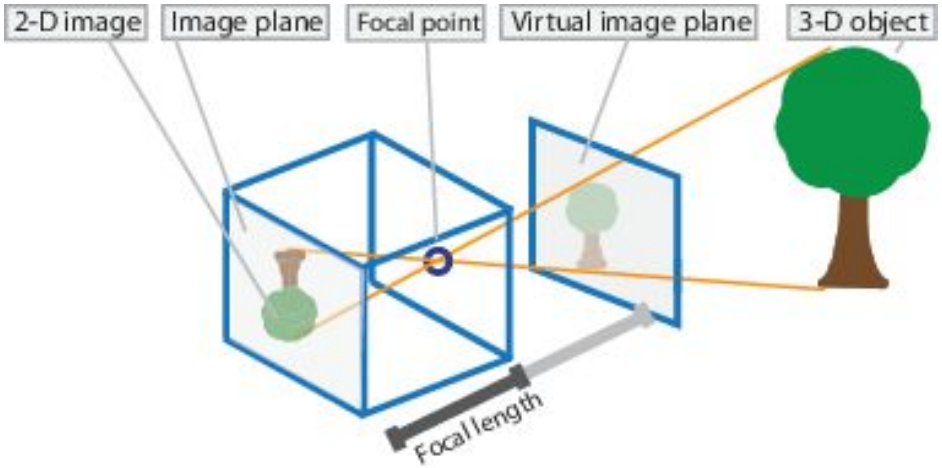
É necessário para a correta determinação da profundidade de cada ponto de uma imagem capturada pela câmera RGB, devido a diferença de posicionamento e resolução entre a câmera e o sensor.

Modelo de Câmera Pinhole

Modelo matemático mais utilizado para câmeras em visão computacional.

Apresenta uma formulação matemática da relação entre os objetos do ambiente e espaços de coordenadas.

Permite a transformação perspectiva de um ponto tridimensional de um objeto no ambiente para um ponto bidimensional no plano da imagem.



Fonte: Mathworks (2018)

Calibração Intrínseca

Consiste na estimação dos parâmetros intrínsecos da câmera, calculados através de experimentos e cálculos de imagens capturadas pela câmera.

Representados por uma matriz **K**, composta por:

- Distância focal: **fx** e **fy**;
- Ponto principal: **cx** e **cy**.

E um vetor **d**, composto por:

- Coeficientes de distorção tangencial: **k1** e **k2**
- Coeficientes de distorção radial: **p1**, **p2**, e **p3**

$$K = \begin{bmatrix} fx & 0 & cx \\ 0 & fy & cy \\ 0 & 0 & 1 \end{bmatrix}$$

$$d = \begin{pmatrix} p1 & p2 & k1 & k2 & p3 \end{pmatrix}$$

Calibração Extrínseca

Consiste na estimação dos parâmetros extrínsecos da câmera, que descrevem a posição relativa entre as câmeras e as suas diferenças angulares em relação ao ambiente capturado.

Representados por uma matriz de rotação **R**, que indica o quanto uma câmera está rotacionada em relação a outra. E um vetor de translação **t**, que indica a diferença da posição física das câmeras no ambiente.

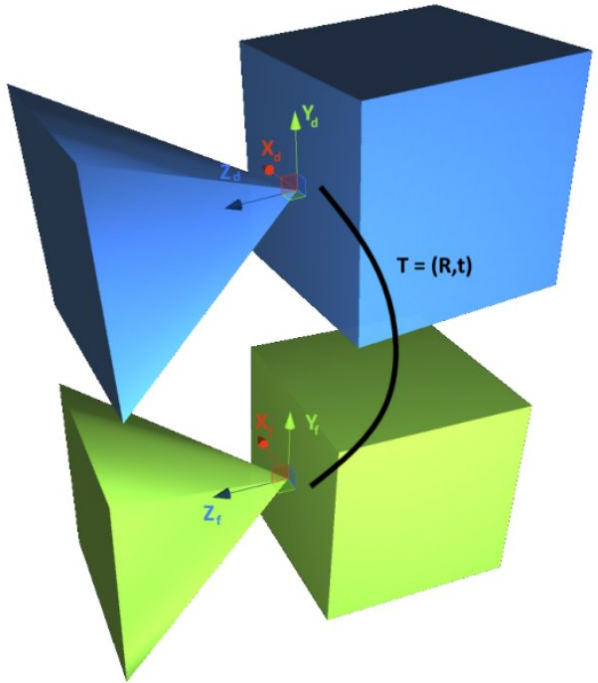
$$K = \left[\begin{array}{ccc|c} r11 & r12 & r13 & t1 \\ r21 & r22 & r23 & t2 \\ r31 & r32 & r33 & t3 \end{array}\right]$$

Processo de Calibração

Uma técnica clássica é descrita por Zhang (2000), a qual se baseia na análise de imagens capturadas em diferentes perspectivas de um padrão especial.

Adequada para uso sem conhecimentos especializados de visão computacional e geometria espacial.

Esse método de calibração é implementado pela biblioteca de código aberto para visão computacional OpenCV.



Fonte: Goussencout (2016)

Transformação perspectiva do modelo pinhole

Utilizando os parâmetros obtidos na calibração, a transformação perspectiva do modelo pinhole pode ser reproduzida da seguinte forma:

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} fx & 0 & cx \\ 0 & fy & cy \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r11 & r12 & r13 & t1 \\ r21 & r22 & r23 & t2 \\ r31 & r32 & r33 & t3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

3 Implementação

Método de calibração de câmeras

O método de calibração de câmeras é realizado nas seguintes etapas:

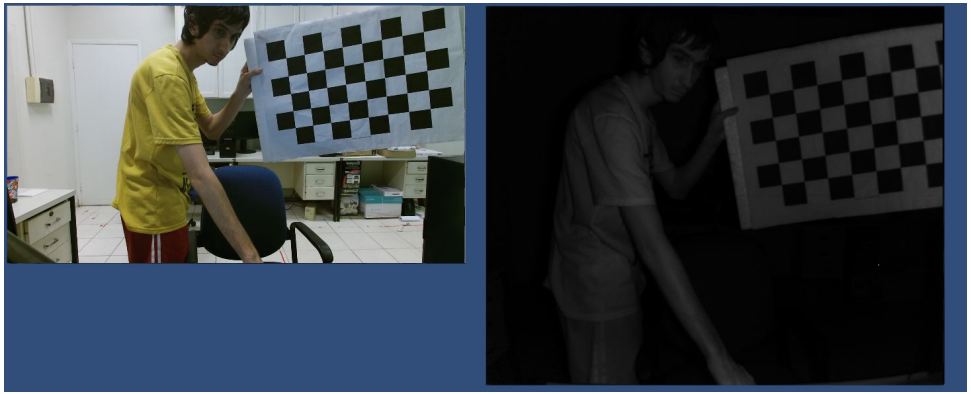
1. Captura de quadros correspondentes de cada câmera de um padrão especial conhecido;
2. Aplicar o algoritmo de calibração implementado na biblioteca OpenCV;
3. Armazenar os parâmetros resultantes do processo de calibração.

Captura de quadros correspondentes

Para a captura de quadros correspondentes da câmera e do sensor de profundidade foi desenvolvida uma aplicação no motor de jogos Unity3D.

As imagens da câmera e do sensor são exibidas em tempo real, e os quadros salvos com o pressionar de uma tecla.

Após a captura dos quadros ser finalizada o processo de calibração é executado automaticamente.



Fonte: Elaborada pelo autor.

Aplicação do algoritmo de calibração

A biblioteca OpenCV não possui integração com o motor de jogos Unity3D. Assim, foi necessário o desenvolvimento de um **Unity Native Plugin**.

O **Unity Native Plugin** é composto por:

- Uma biblioteca estática de código nativo (comumente uma DLL), contendo as funcionalidades da biblioteca externa que serão utilizadas;
- Um script da Unity3D que permite a outros scripts utilizarem as funções da biblioteca estática.

O script de gerenciamento da biblioteca estática requer algumas diretivas especiais, como no exemplo que segue:

```
using UnityEngine;
using System.Runtime.InteropServices;

class NativePluginScript : MonoBehaviour {
    // Carrega a biblioteca estatica
    [DllImport ("NomeDaBiblioteca")]

    // Define a funcao externa FooPluginFunction presente na bilioteca
    private static extern float FooPluginFunction ();
}
```

Resultado da calibração

Após o final da calibração, seu resultado é armazenado em arquivos no formato YAML (YAML Ain't Markup Language).

A sintaxe da saída segue o exemplo ao lado.

```
%YAML:1.0
---
calibration_time: "Wed Oct 24 10:23:14 2018"
image_width: 512
image_height: 424
board_width: 8
board_height: 5
square_size: 6.7000001668930054e-01
aspectRatio: 1.
flags: 2
camera_matrix: !!opencv-matrix
   rows: 3
   cols: 3
   dt: d
   data: [ 3.7078769283746763e+02, 0., 2.6497348294738594e+02, 0.,
     3.7078769283746763e+02, 2.0619283740189305e+02, 0., 0.,
distortion_coefficientes: !!opencv-matrix
   rows: 5
   cols: 1
   dt: d
   data: [ 1.4258293749502938e-01, -4.99087328392849303e-01,
     1.9888927394039485e-04, 2.24719293982993482e-03,
     6.9859282387934983e-01 ]
}
```

Método de oclusão mútua

Utilizando a matriz de transformação obtida na calibração das câmeras é possível projetar o mapa de profundidade na imagem da câmera RGB.

Utilizando o conjunto dessas informações é possível criar uma superfície virtual que emula o ambiente real capturado.

Essa superfície virtual é criada utilizando algoritmos de geração de malhas.

Mapa de profundidade

Os sensores de profundidade apresentam como saída uma matriz, chamada de mapa de profundidade.

Os sensores de profundidade atuais são afetados por um problema de sombreamento, que são seções em que o sensor não conseguiu obter a profundidade.

Essas sombras podem ser causadas por objetos presentes antes do plano **near** ou após o plano **far** do sensor.

Outros emissores infravermelhos e superfícies que fazem com que os raios emitidos não cheguem ao receptor são outras causas.

Fonte: Elaborada pelo autor.

Tratamento do mapa de profundidade

Um mapa de profundidade pode ser tratado para amenizar o problema do sombreamento. Levin, Lischinski e Weiss (2004), descrevem um método baseado em algoritmos de coloração de imagem. O qual utiliza informações da câmera RGB e das áreas próximas a sombra para estimar uma profundidade para as sombras.

Fonte: Gaspari (2015)

Geração de Malhas

A geração de malhas é um processo que consiste em transformar o mapa de profundidade em uma superfície virtual, e então aplicar adequadamente uma textura baseada na imagem da câmera RGB.

Existem diversos algoritmos para a realização desse processo, um deles sendo o algoritmo **Marching Cubes** (LORENSEN; CLINE, 1987).

O Kinect SDK 2.0 disponibiliza um algoritmo base para a geração de uma malha utilizando as nativas do motor de jogos Unity3D.

4 Experimentos

Ambiente de Desenvolvimento

O ambiente de desenvolvimento utilizado estava configurado com as seguintes aplicações e bibliotecas:

- Motor de jogos Unity3D 2018.2.8f1;
- Microsoft Kinect SDK 2.0;
- Plugin do Microsoft Kinect SDK 2.0 para a Unity3D;
- Ambiente de programação Microsoft Visual Studio 2017 (C++ e C#);
- Biblioteca de visão computacional OpenCV 3.4.3.

Equipamentos Utilizados

Os equipamentos utilizados no desenvolvimento do projeto foram disponibilizados pelo laboratório SACI da UNESP de Bauru. Sendo eles:

- Webcam Logitech C920;
- Microsoft Kinect v2;
- Intel RealSense ZR300;
- Intel RealSense SR300;
- Computador:
  - Microsoft Windows 10 Pro 64-bits;
  - Intel Core i7-4770 Quad-Core 3.4GHz;
  - Memória RAM: 16GB DDR3;
  - NVIDIA GeForce GTX 1060;
  - Placa mãe ASUS H81M-A/BR.

Fixação da webcam e do Kinect v2

A webcam Logitech C920 e o Microsoft Kinect v2 foram fixados fisicamente na estrutura de um tripé. Dessa forma sua posição relativa é preservada para o processo de calibração e o seu uso posterior.

Fonte: Avellar (2017)

RealSense ZR300

Os primeiros experimentos foram realizados utilizando a câmera Intel RealSense ZR300, a qual possui um sensor do tipo Time-of-Flight com dois emissores infravermelhos.

Porém o SDK oficial da câmera só está disponível para os sistemas operacionais baseados em linux, e a biblioteca de código aberto librealsense 1, não possuí integração com o motor de jogos Unity3D.

RealSense SR300

A câmera Intel RealSense SR300 utiliza um sensor de profundidade do tipo luz estruturada.

Essa câmera é suportada pela biblioteca librealsense 2, que se tornou o SDK oficial para as novas câmeras da linha RealSense, e fornece um plugin para integração com o motor de jogos Unity3D.

Porém o alcance do seu sensor de profundidade é muito pequeno, tornando seu uso inviável para um sistema de estúdio virtual por não cobrir uma cena completa.

SR300 - Resultado dos experimentos

Fonte: Elaborado pelo autor.

Kinect v2

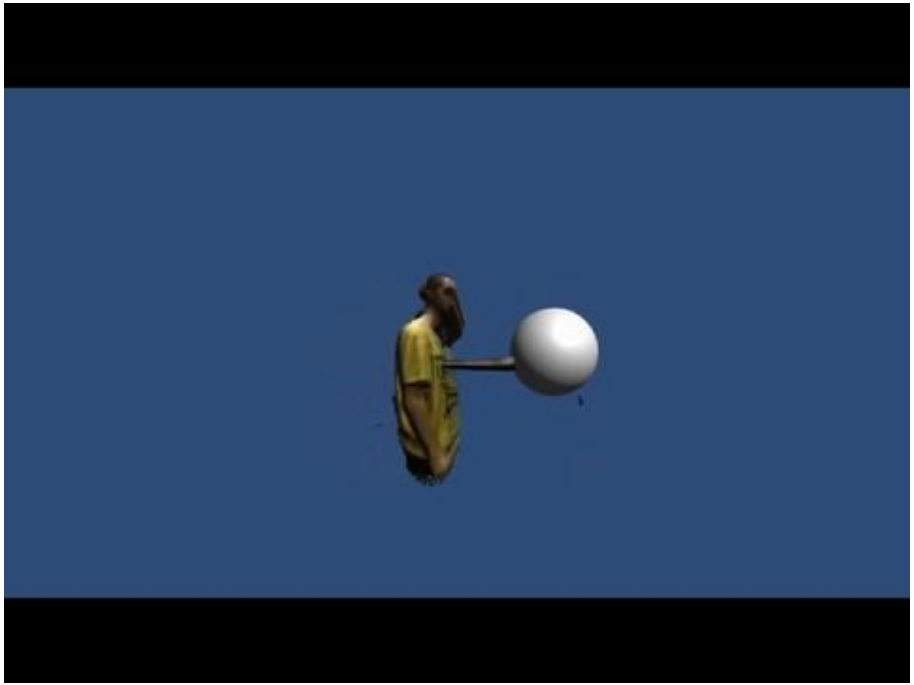
O Kinect v2 é equipado com um sensor de profundidade do tipo Time-of-Flight, e possui tanto um plugin para integração com o motor de jogos Unity3D quando um alcance suficiente para cobrir uma cena completa.

Porém, se trata dispositivo mais antigo, tendo uma resolução de apenas 512x424 pixels e utiliza apenas um receptor infravermelho.

Kinect v2 - Malha gerada (visão frontal)

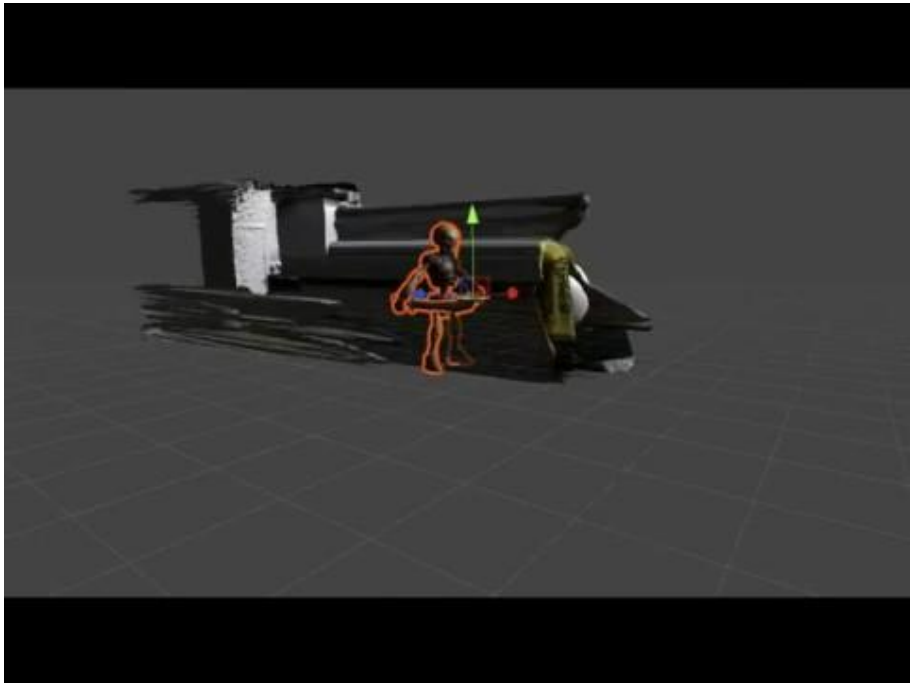
Fonte: Elaborada pelo autor.

Kinect v2 - Malha gerada (visão lateral)



Fonte: Elaborada pelo autor.

Kinect v2 - Visão lateral da malha



Fonte: Elaborada pelo autor.

5 Conclusão

Perspectiva Final

A etapa de integração da calibração de câmeras foi desafiadora, devido a necessidade de integração entre a Unity3D e o código nativo da OpenCV.

O uso do Kinect v2, um dispositivo com especificações não muito boas, o torna ineficaz em aplicações de reconstrução 3D.

O trabalho apresenta uma abordagem alternativa e mais realista para a oclusão mútua entre objetos reais e virtuais.

Com os devidos refinamentos o método pode ser aplicado em sistemas de estúdio virtual baseados em realidade aumentada.

Trabalhos Futuros

- Implementação do método de oclusão mútua utilizando dispositivos mais recentes como as câmeras Intel RealSense da série D400;
- Implementação de algoritmos de refinamento do mapa de profundidade;
- Implementação de algoritmos de geração de malhas melhores;
- Utilização de múltiplos sensores de profundidade.

Referências

AGUILAR, I. A. ARSTUDIO 2.0: um sistema de estúdio virtual para geração de conteúdo midiático baseado no motor de jogos Unity3D. Dissertação (Mestrado) — Universidade Estadual Paulista "Júlio de Mesquisa Filho", Faculdade de Arquitetura, Artes e Comunicação, Bauru, 2017. Disponível em: <http://hdl.handle.net/11449/151188>. Acesso em: 23 mar. 2018.

AVELLAR, G. M. N. Sistema para calibração de sensor de profundidade time-of-flight e câmera RGB externa. Dissertação (Mestrado) — Universidade Estadual Paulista "Júlio de Mesquisa Filho", Faculdade de Ciências, Bauru, 2017.

AZUMA, R.; BAILLOT, Y.; BEHRINGER, R.; FEINER, S.; JULIER, S.; MACINTYRE, B. Recent advances in augmented reality. IEEE Computer Graphics and Applications, v. 21, n. 6, p. 34–47, Nov 2001. ISSN 0272-1716.

Referências

GASPARI, T. de. Desenvolvimento de um método semiautomático para geração de ground truths de vídeos. Dissertação (Mestrado) — Universidade Estadual Paulista Julio de Mesquita Filho, Instituto de Biociências, Letras e Ciências Exatas, Bauru, 2015. Disponível em: <http://hdl.handle.net/11449/138444>. Acesso em: 25 out. 2018.

GIBBS, S.; ARAPIS, C.; BREITENEDER, C.; LALIOTI, V.; MOSTAFAWY, S.; SPEIER, J. Virtual studios: an overview. IEEE MultiMedia, v. 5, n. 1, p. 18–35, Jan 1998. ISSN 1070-986X.

GOUSSENCOURT, T. D. Système multimodal de prévisualisation “on set” pour le cinéma. Tese (Theses) — Université Grenoble Alpes, dez. 2016. Disponível em: <https://tel.archives-ouvertes.fr/tel-01592556>.

GOUSSENCOURT, T. D.; DELLAC, J.; BERTOLINO, P. A Game Engine as a Generic Platform for Real-Time Previz-on-Set in Cinema Visual Effects. In: Advanced Concepts for Intelligent Vision Systems (ACIVS). Catania, Italy: [s.n.], 2015. Disponível em: <https://hal.archives-ouvertes.fr/hal-01226184>.

Referências

INTEL. Product Datasheet. Intel RealSense Camera SR300. 2016. Disponível em: <https://www.intel.com/content/dam/support/us/en/documents/emergingtechnologies/intel-realsense-technology/realsense-sr300-datasheet1-0.pdf>. Acesso em: 25 out. 2018.

INTEL. Product Datasheet. Intel RealSense 3D Camera ZR300. 2017. Disponível em: <http://click.intel.com/media/ZR300-Product-Datasheet-Public-002.pdf>. Acesso em 25 out. 2018.

INTEL. Open Source Computer Vision Library. 2018. Disponível em: <https://github.com/opencv/opencv>. Acesso em: 25 out. 2018.

KESELMAN, L.; WOODFILL, J. I.; GRUNNET-JEPSEN, A.; BHOWMIK, A. Intel(r) realsense(tm) stereoscopic depth cameras. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). [S.l.: s.n.], 2017. p. 1267–1276. ISSN 2160-7516.

Referências

LEVIN, A.; LISCHINSKI, D.; WEISS, Y. Colorization using optimization. In: ACM. ACM transactions on graphics (tog). [S.l.], 2004. v. 23, n. 3, p. 689–694. Citado na página 24. LORENSEN, W. E.; CLINE, H. E. Marching cubes: A high resolution 3d surface construction algorithm. In: ACM. ACM siggraph computer graphics. [S.l.], 1987. v. 21, n. 4, p. 163–169.

MATHWORKS. What Is Camera Calibration? 2018. Disponível em: <https://www.mathworks.com/help/vision/ug/camera-calibration.html>. Acesso em: 25 out. 2018.

MILLERSON, G.; OWENS, J. Television Production. 14. ed. [S.l.]: Oxford: Elsevier, 2009. 423 p. ISBN 978-0-240-52078-0.

TERABEE. Technology overview. Time-of-Flight Principle. 2018. Disponível em: <https://www.terabee.com/time-of-flight-principle>. Acesso em 25 out. 2018.

UNITY3D. Unity Documentation. Native Plugins. 2018. Disponível em: <https://docs.unity3d.com/Manual/NativePlugins.html>. Acesso em: 25 out. 2018.

Referências

ZHANG, Z. A flexible new technique for camera calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence, v. 22, n. 11, p. 1330–1334, Nov 2000. ISSN 0162-8828.