DESVENDANDO A API DE DESENVOLVIMENTO DO MICROSOFT KINECT

Daniel Urio Mendes
Prof. Aurélio Faustino Hoppe

FURB – Universidade Regional de Blumenau
DSC – Departamento de Sistemas e Computação
2012/2

Roteiro

- Motivação
- Objeto de estudo
- Trabalhos relacionados
- Objetivo
- Requisitos
- Especificação
- Desenvolvimento
- Operacionalidade
- Experimentos
- Conclusão
- Extensões
- Demonstração

FURB

Motivação

- Kinect para Xbox 360
- Kinect for Windows SDK
- Tecnologia Recente
- Reconstrução de superfícies

FURB

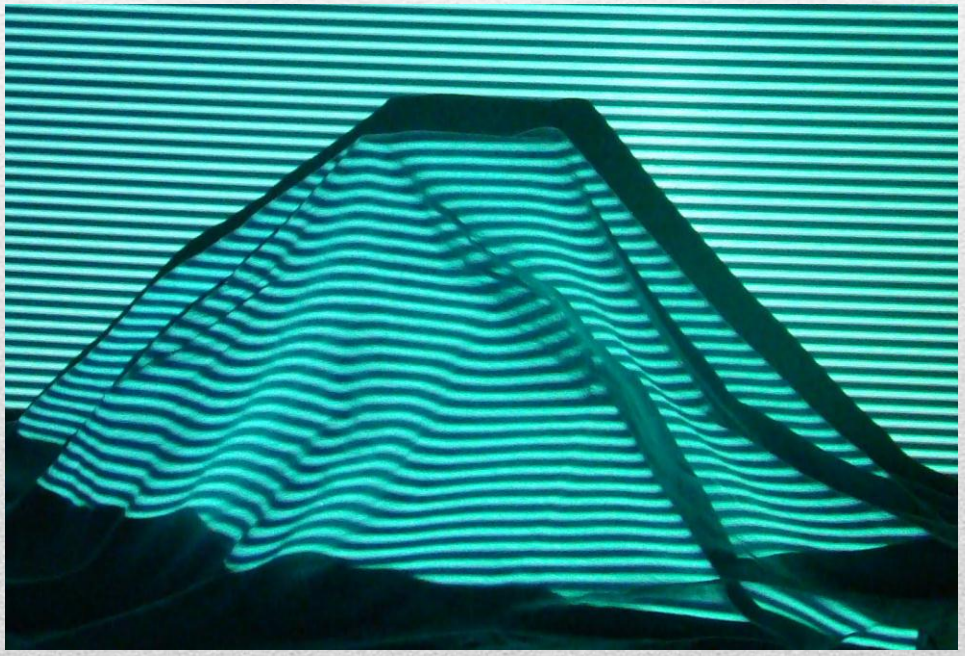
Kinect

- Funcionamento

FURB

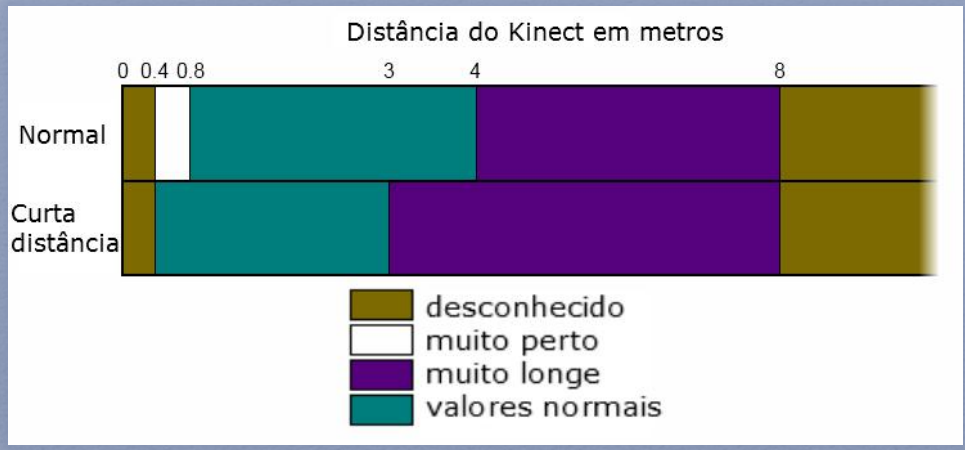
Kinect - Profundidade

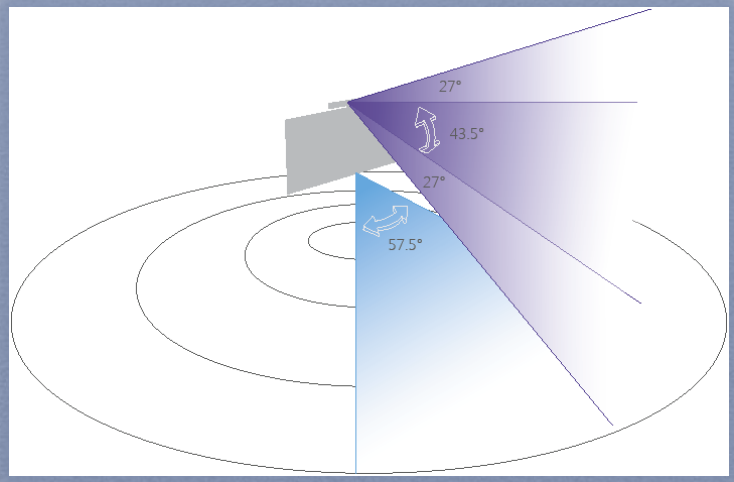
- Luz estruturada



Kinect - Limitações

- Dados de profundidade





- Ângulo de visão

FURB

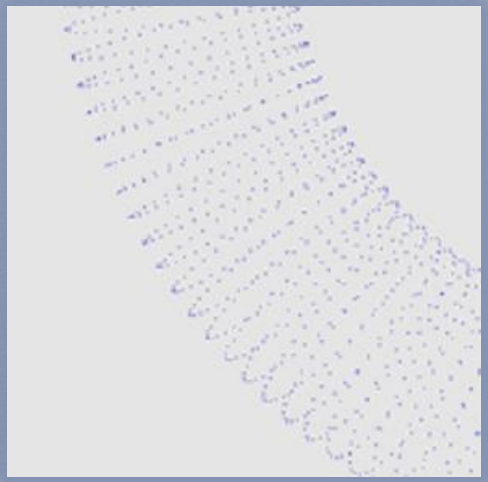
Kinect - SDK

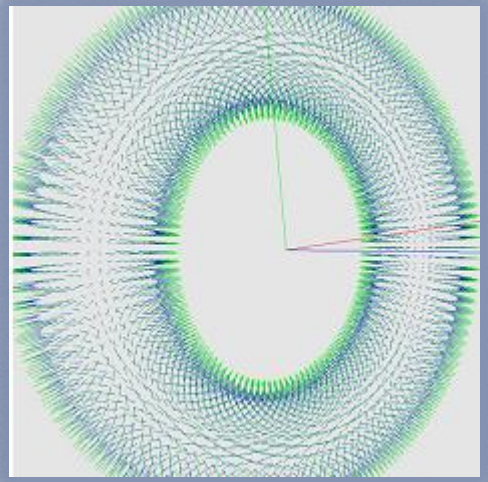
- Permite acessar os streams do Kinect
  - Imagem
  - Profundidade
  - Áudio
- Permite utilizar o skeletal tracking
- Alterar o ângulo do motor de inclinação
- Linguagens suportadas: C#, VB.NET e C++

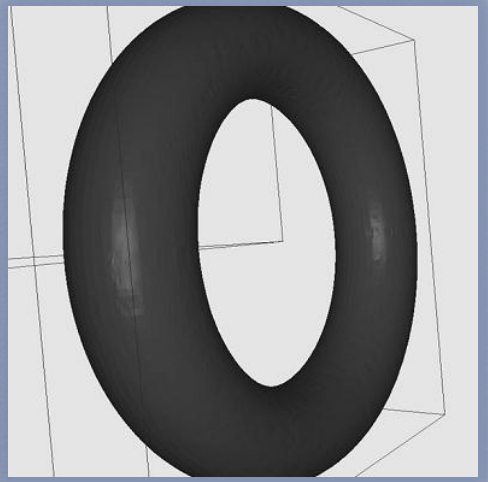
FURB

Reconstrução de superfícies

- Aquisição de dados
- Utilização de técnicas de reconstrução
- Geração de aproximação de superfície



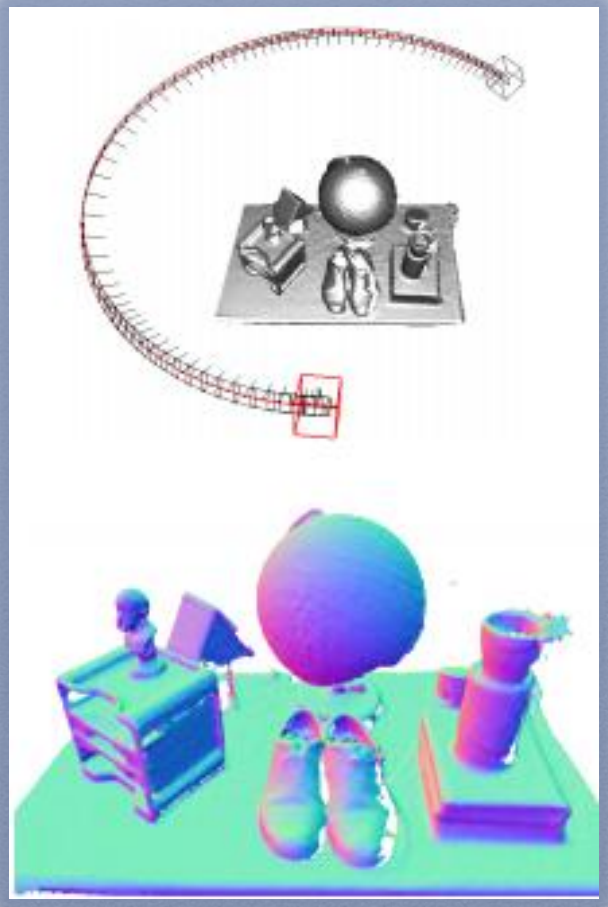


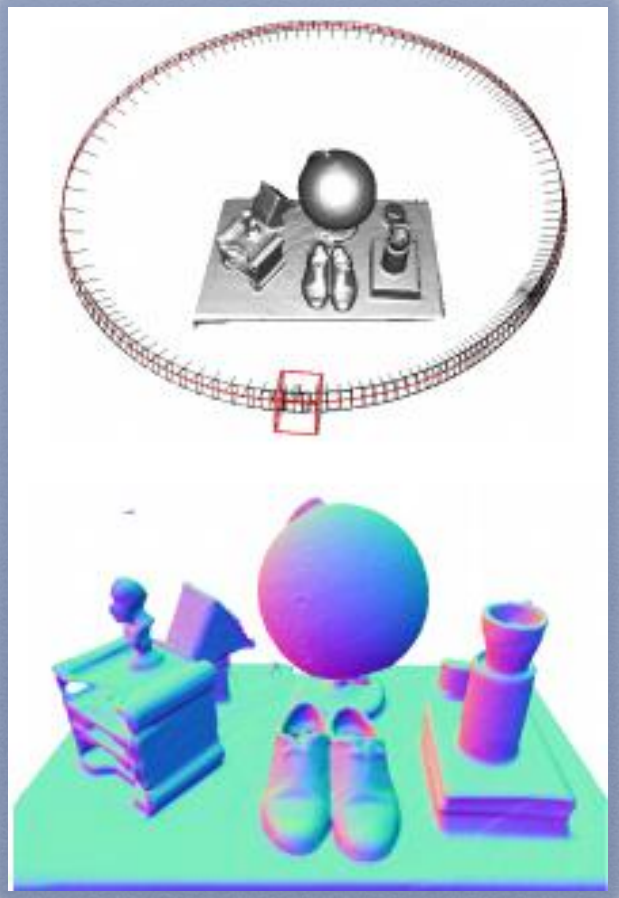


FURB

Reconstrução de superfícies

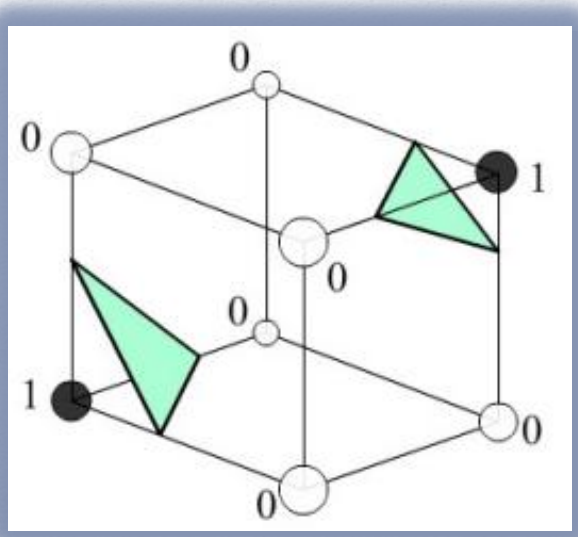
- Reconstrução de superfícies incremental

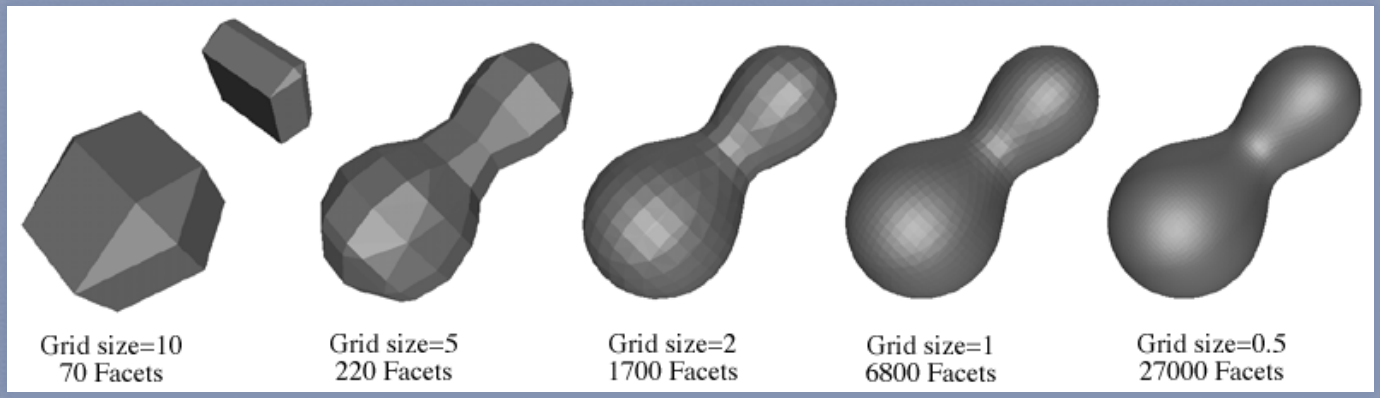




Reconstrução de superfícies

- Marching Cubes





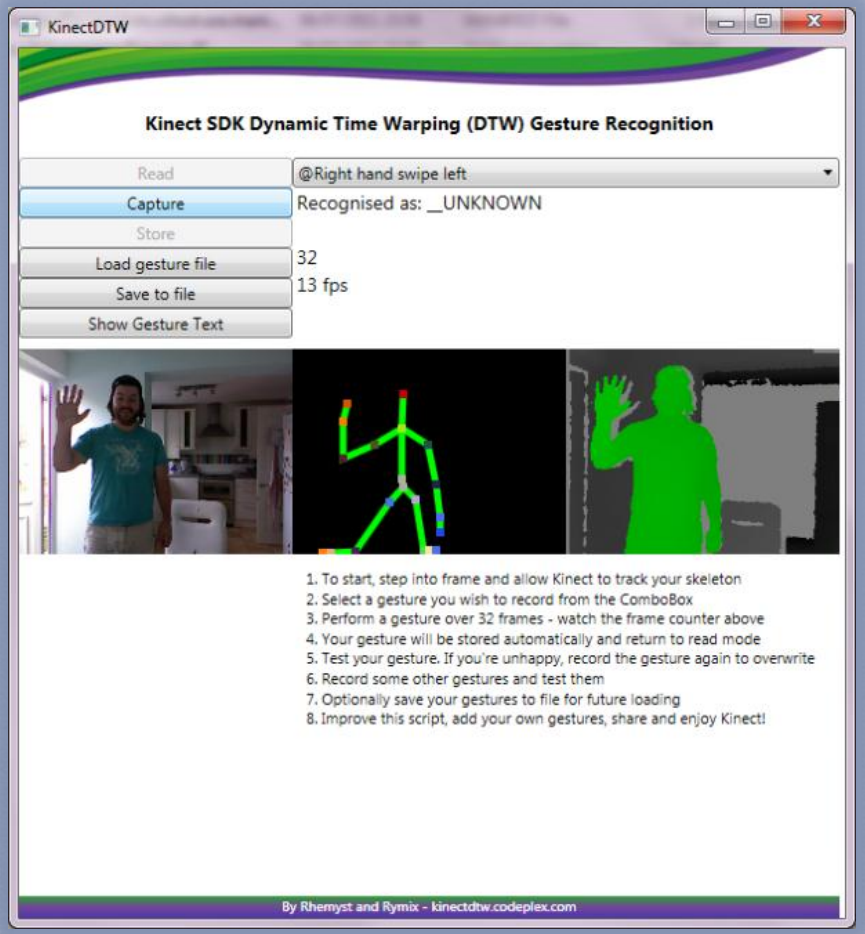
FURB

Trabalhos relacionados

- KinectFusion

Trabalhos relacionados

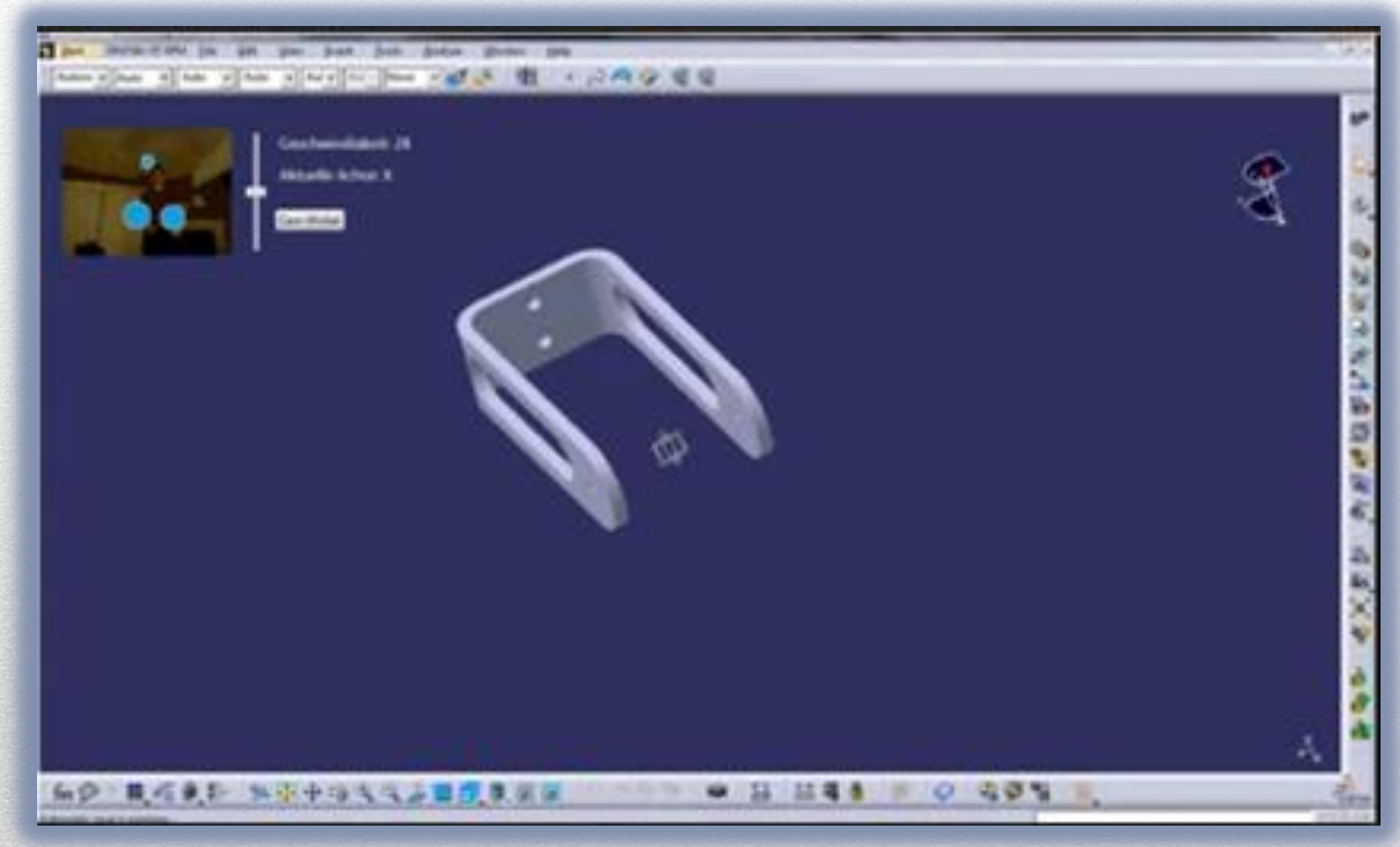
- Kinect SDK Dynamic Time Warping Gesture Recognition



FURB

Trabalhos relacionados

- KinectCAD



Objetivo

- Desenvolver uma aplicação que capture um ambiente real e gere o modelo tridimensional utilizando o Kinect

FURB

Requisitos

- Principais Requisitos Funcionais
  - fazer a reconstrução de superfície tridimensional de ambientes ou objetos estáticos dentro do limite de visibilidade do Kinect
  - disponibilizar uma interface para apresentar o mapeamento tridimensional

FURB

Requisitos

- Principais Requisitos Não Funcionais
  - ser implementado utilizando a linguagem C#
  - utilizar o ambiente de desenvolvimento Visual Studio 2010
  - utilizar o equipamento Kinect
  - utilizar o Kinect for Windows SDK
  - executar em Windows 7 64 bits

FURB

Diagrama de casos de uso

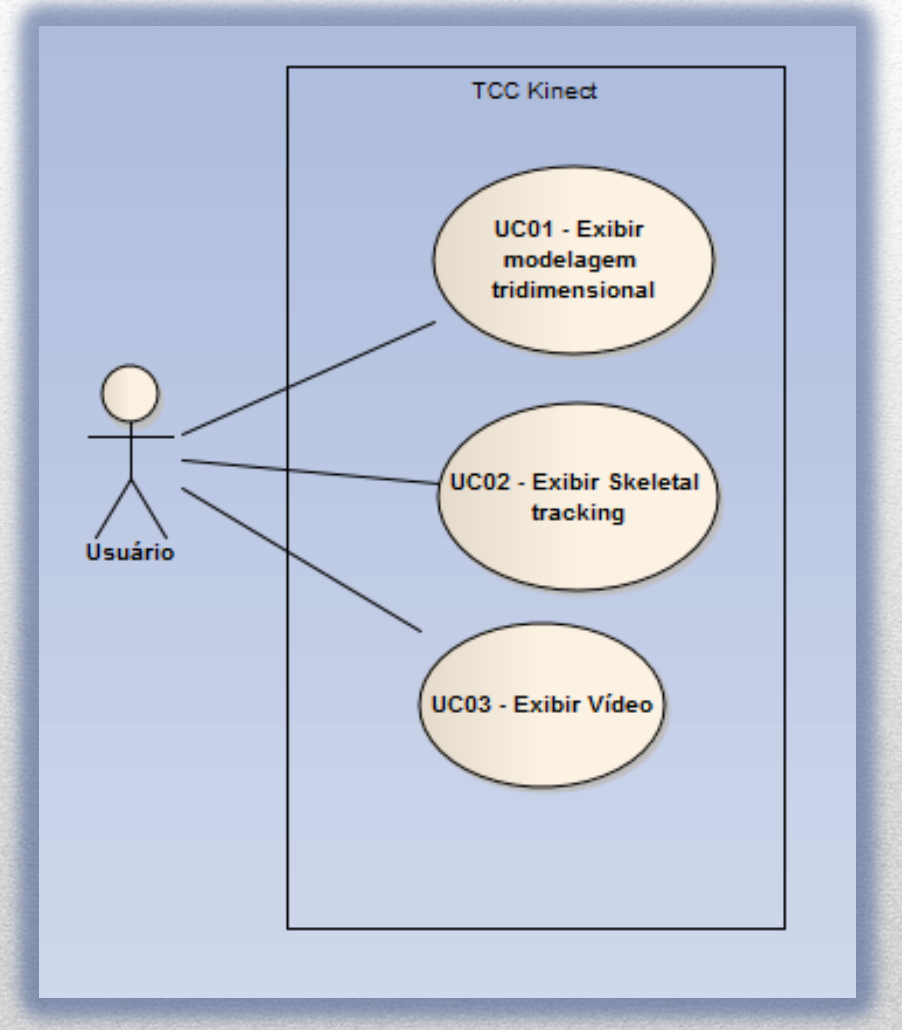
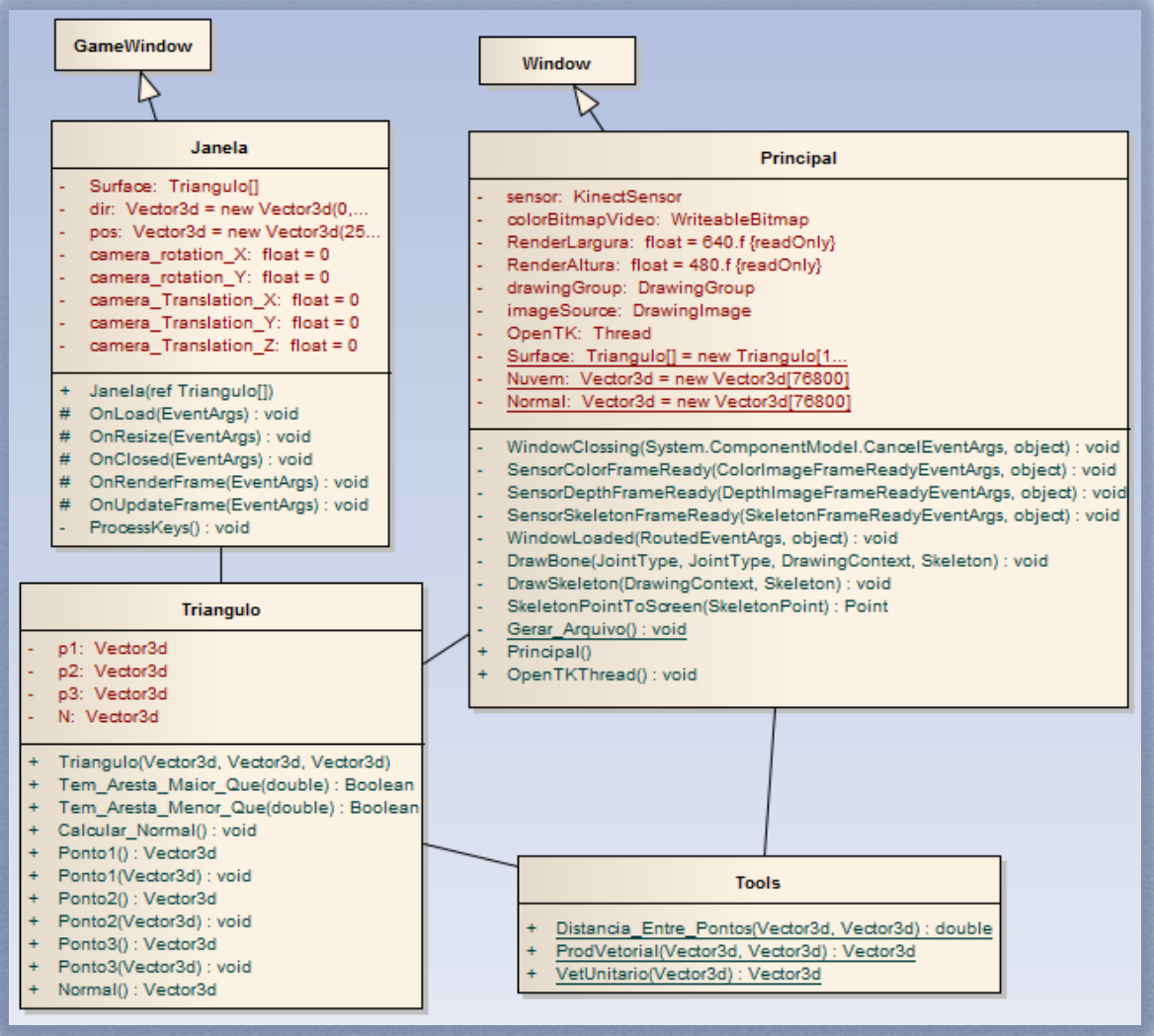


Diagrama de classes



FURB

Ferramentas utilizadas

- Microsoft Visual Studio 2010

- Kinect for Windows SDK

- OpenTK

FURB

Implementação

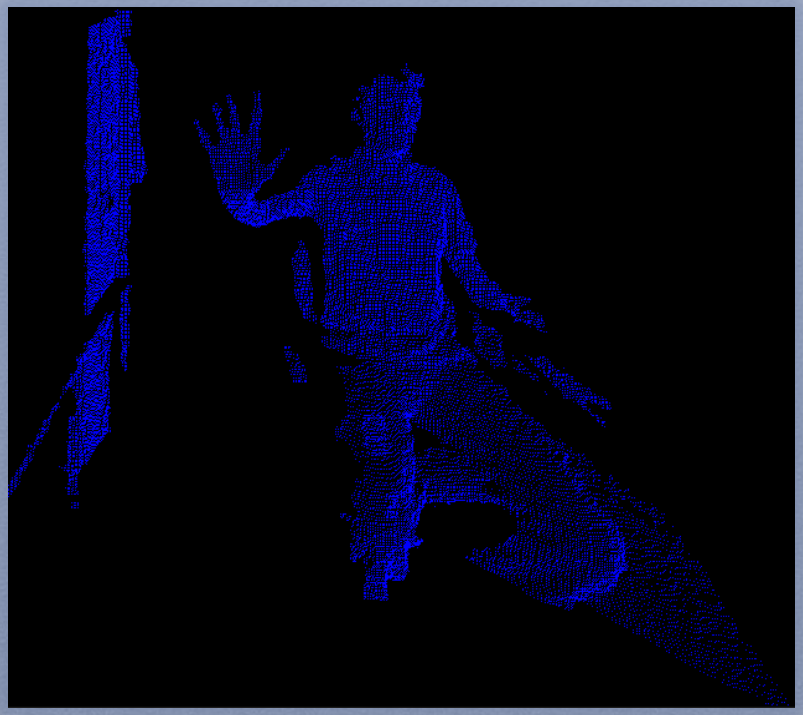
- Inicialização do Kinect
- Ativação dos streams
- Criação de thread para apresentar as janelas em paralelo

```
public void OpenTKThread()
{
    // Depth
    this.sensor.DepthStream.Enable(DepthImageFormat.Resolution320x240Fps30);
    this.sensor.DepthFrameReady += this.SensorDepthFrameReady;
    using (Janela janela = new Janela(ref Surface))
    {
        janela.Run(60);
        this.sensor.DepthFrameReady -= this.SensorDepthFrameReady;
    }
    this.sensor.DepthStream.Disable();
}
```

FURB

Implementação

- Capturando nuvem de pontos



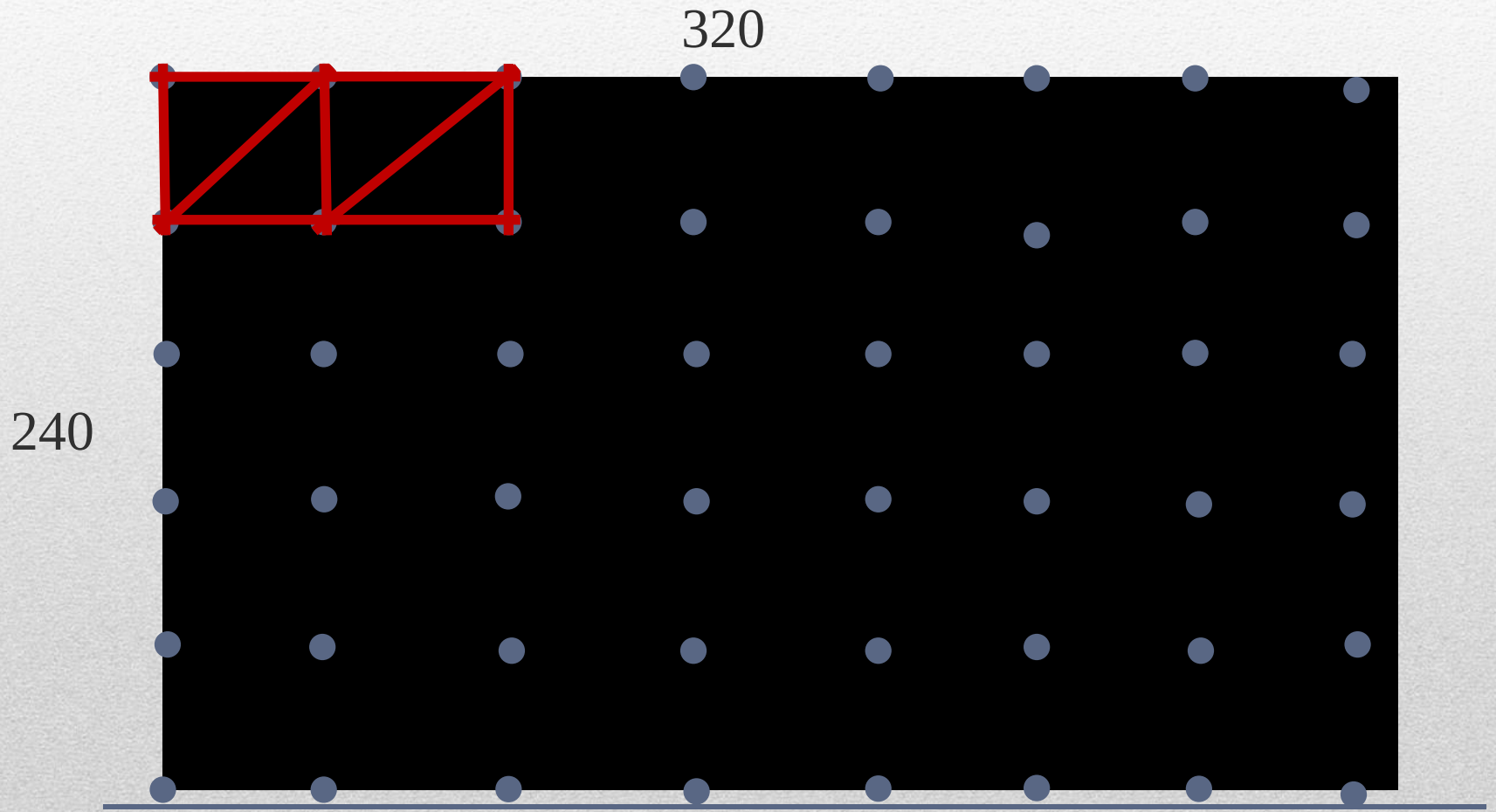
```
DepthImageFrame imageFrame = e.OpenDepthImageFrame();
if (imageFrame != null)
{
    short[] pixelData = new short[imageFrame.PixelDataLength];
    imageFrame.CopyPixelDataTo(pixelData);
    double temp = 0;
    int i = 0;
    for (int y = 0; y < 240; y++)
        for (int x = 0; x < 320; x++)
        {
            temp = ((ushort)pixelData[x + y * 320]) >> 3;

            Nuvem[i].X = (float)-((x - cx_d) * temp * fx_d);
            Nuvem[i].Y = (float)-((y - cy_d) * temp * fy_d);
            Nuvem[i].Z = (float)temp;

            i++;
        }
```

```
// Calibração
private double fx_d = 1.0 / 5.9421434211923247e+02;
private double fy_d = 1.0 / 5.9104053696870778e+02;
private double cx_d = 3.3930780975300314e+02;
private double cy_d = 2.4273913761751615e+02;
```

FURB

Implementação

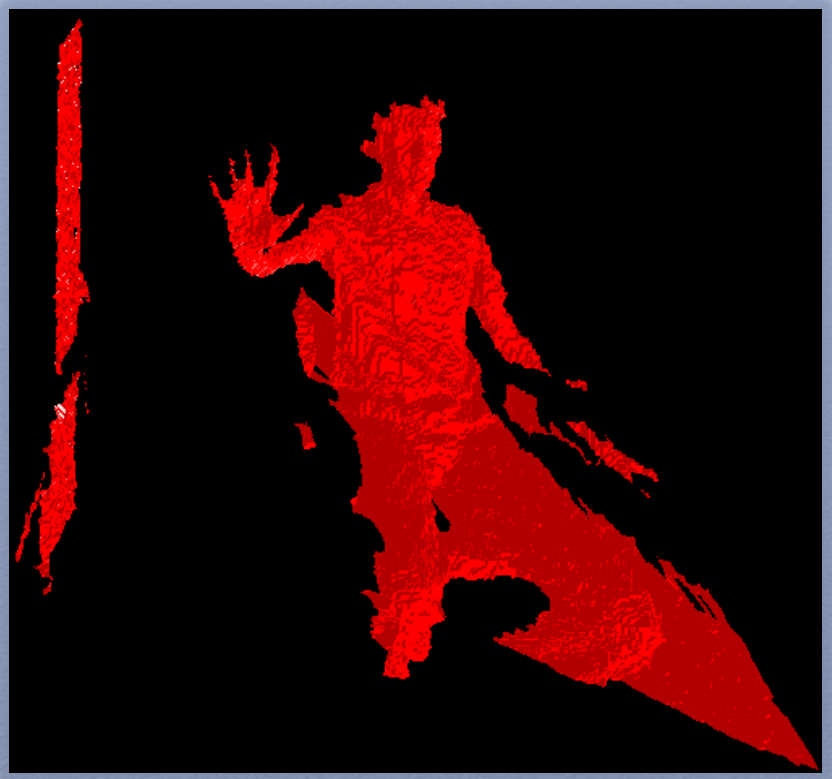
- Geração da superfície



Implementação

- Desenhando a superfície

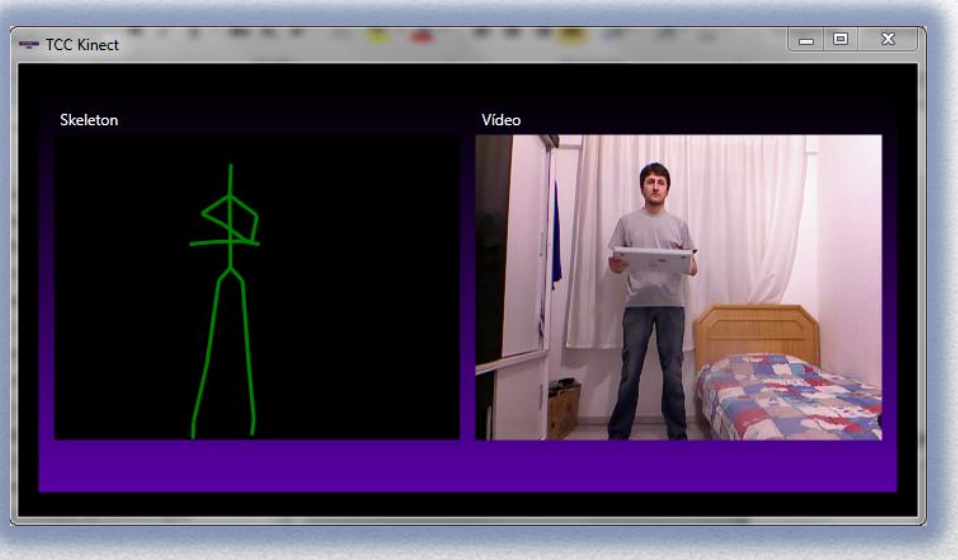
```
GL.Begin(BeginMode.Triangles);
foreach (Triangulo T in Surface)
{
    if (T != null)
    {
        GL.Normal3(T.Normal);
        GL.Color4(Color.Red); GL.Vertex3(T.Ponto1);
        GL.Color4(Color.Red); GL.Vertex3(T.Ponto2);
        GL.Color4(Color.Red); GL.Vertex3(T.Ponto3);
    }
}
GL.End();
SwapBuffers();
```

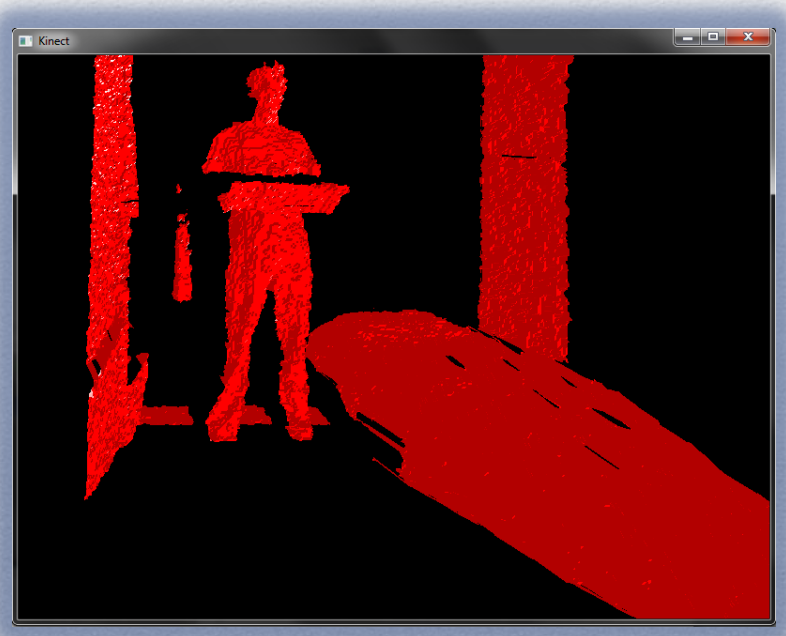


FURB

Aplicação

- Janelas da aplicação





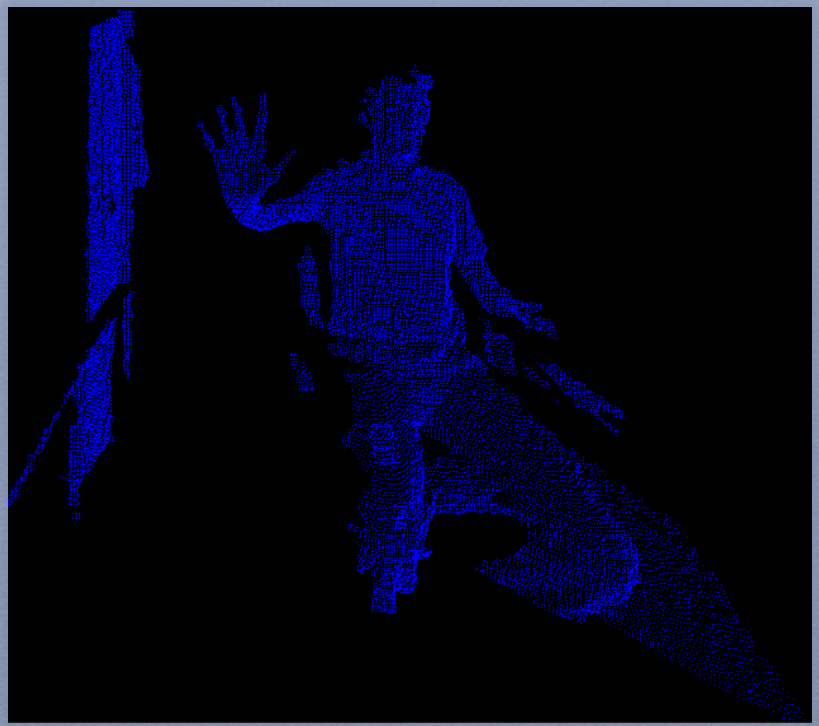
FURB

Experimentos

Reconstruir a superfície de uma pessoa

Utilizando a aplicação

Imagem real

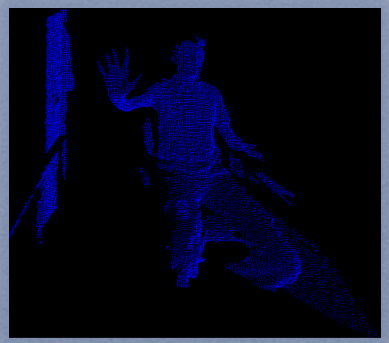


Nuvem de pontos

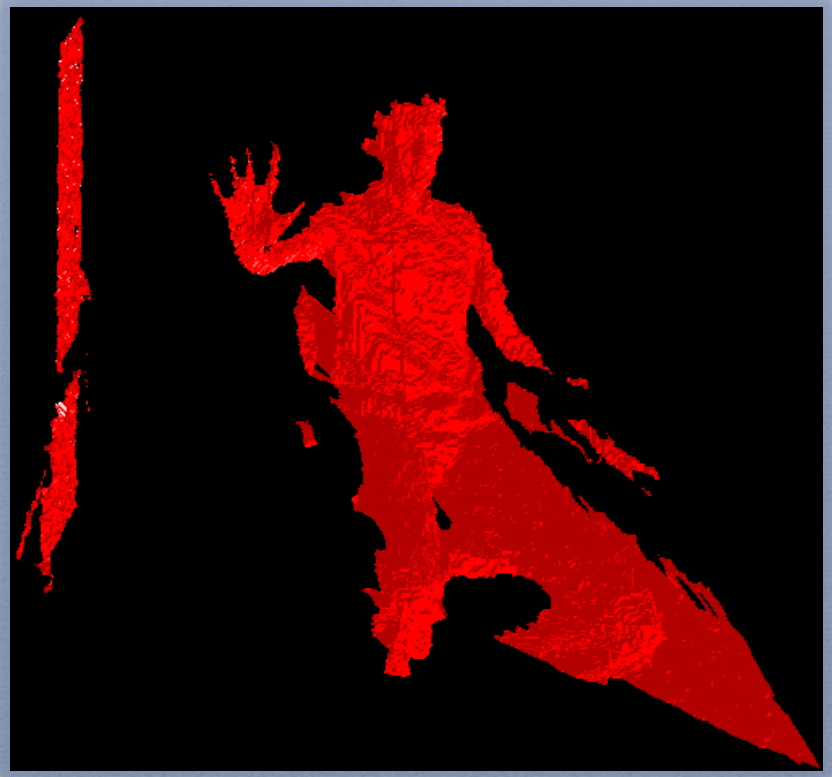
FURB

Experimentos

Imagem real



Nuvem de pontos



Superfície reconstruída

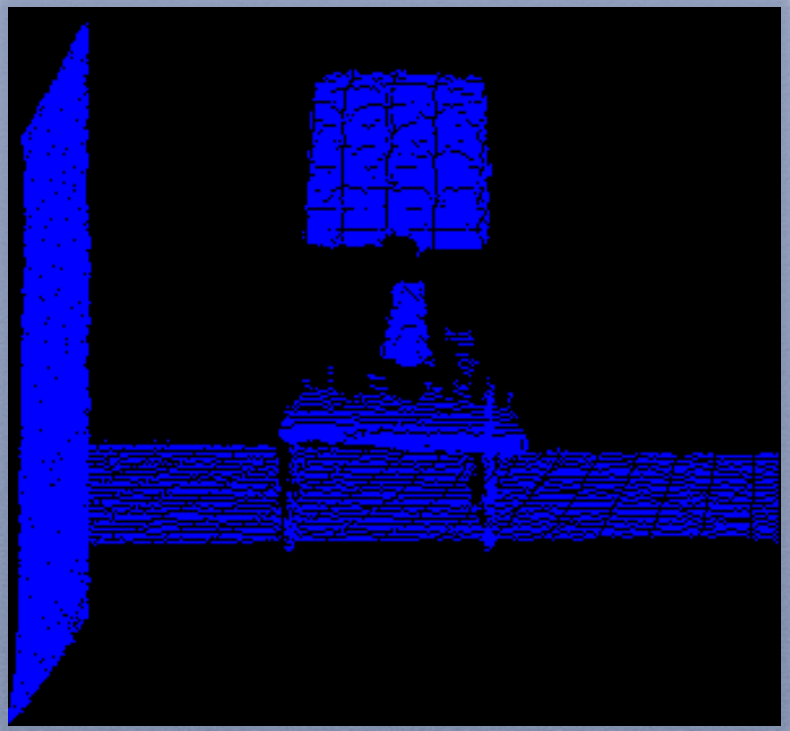
FURB

Experimentos

Reconstruir a superfície de objetos com superfície simples

Utilizando a aplicação

Imagem real

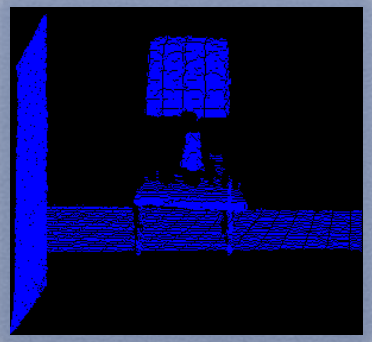


Nuvem de pontos

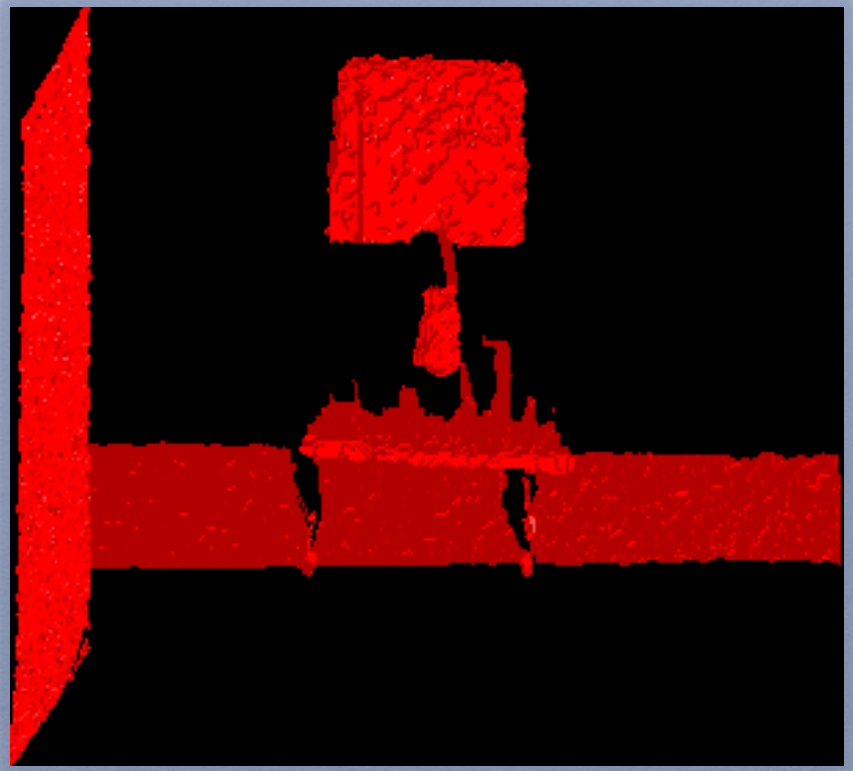
FURB

Experimentos

Imagem real



Nuvem de pontos



Superfície reconstruída

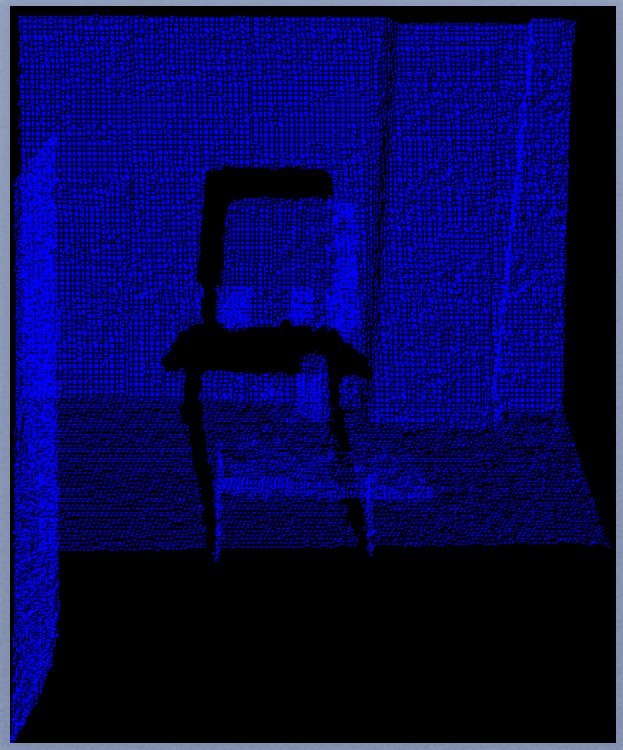
FURB

Experimentos

Reconstruir a superfície de objetos utilizando marching cubes

com executável de terceiros

Imagem real

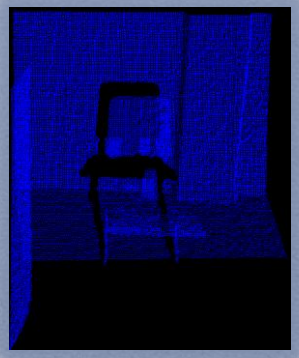


Nuvem de pontos

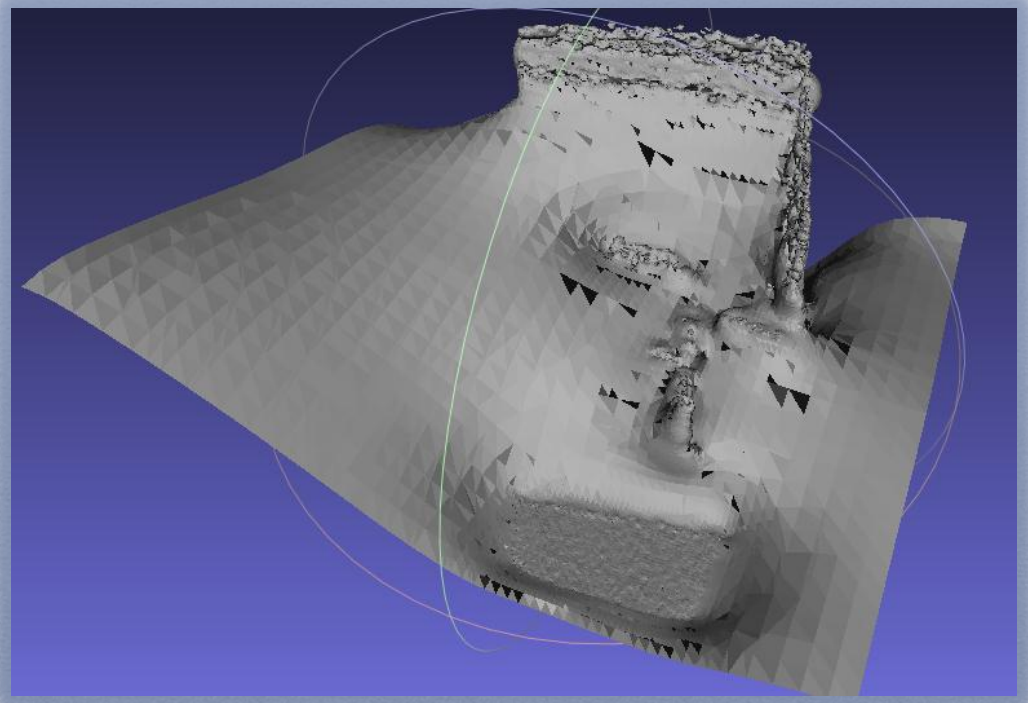
FURB

Experimentos

Imagem real



Nuvem de pontos



- Superfície reconstruída incorreta

FURB

Conclusão

- Os resultados obtidos foram bastante favoráveis, mas como não foi feita uma comparação da superfície com outras aplicações, não é possível dizer se a qualidade da reconstrução é boa ou não.

FURB

Extensões

- Fazer a superfície reconstruída com a cor real do objeto.
- Reconstrução incremental

FURB

DEMONSTRAÇÃO

FURB