Você sabia que seu material didático é interativo e multimídia?
Ele possibilita diversas formas de interação com o conteúdo, a qualquer hora e de qualquer lugar. Mas na versão impressa, alguns conteúdos interativos são perdidos, por isso, fique atento! Sempre que possível, opte pela versão digital. Bons estudos!

Imprimir

Computação Gráfica e Processamento de Imagens

CGPI: Síntese de imagens

Unidade 3 - Seção 1

Esta webaula aborda os primeiros conceitos da síntese de imagens: o pipeline tradicional de visualização e o algoritmo de *ray casting*.

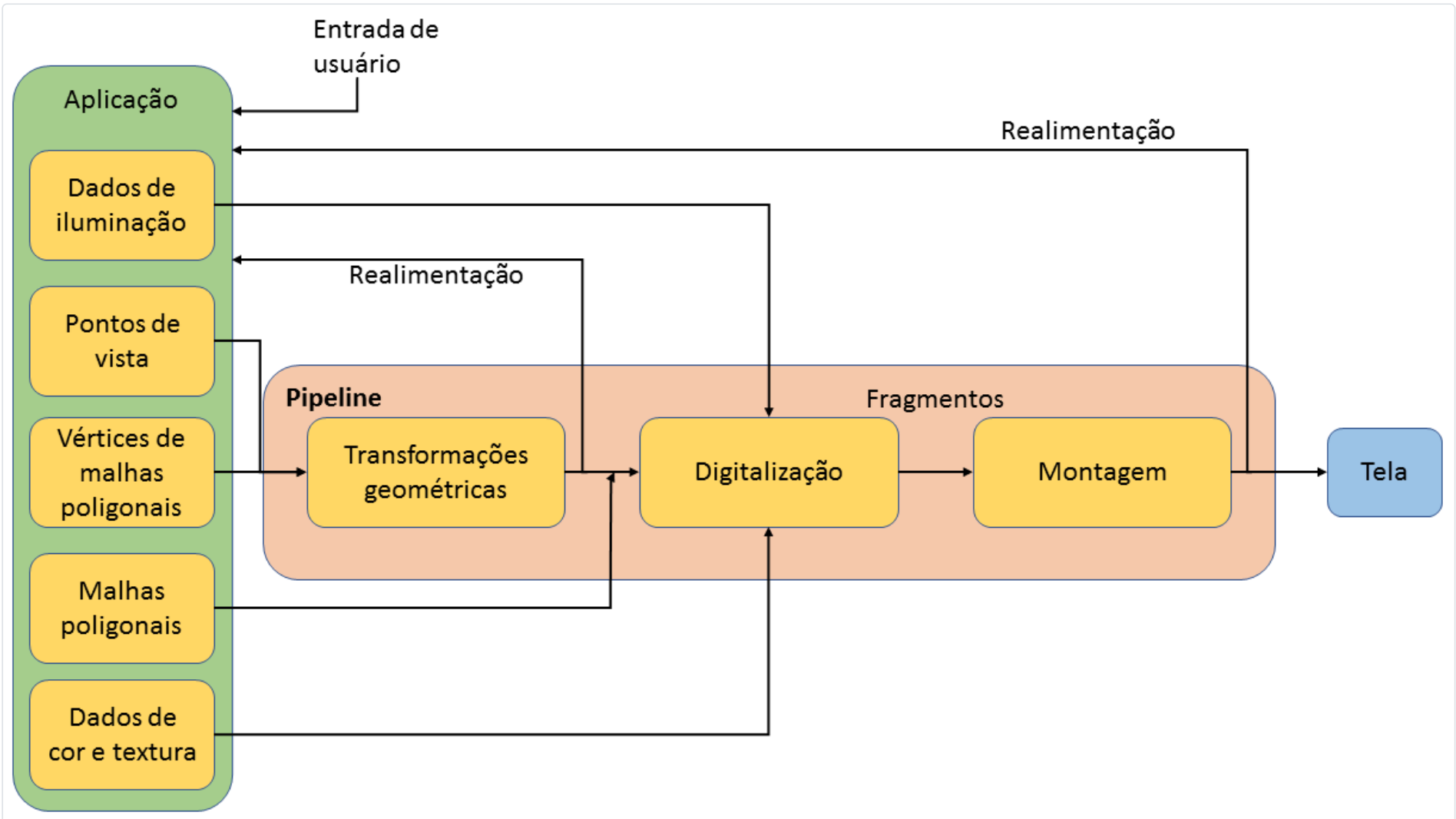
A síntese de imagens é a subárea da computação gráfica que produz imagens sintéticas a partir de modelos geométricos tridimensionais.

Pipeline de visualização

Entende-se por pipeline de visualização a sequência de operações e transformações que devem ser executadas para gerar uma imagem 2D a partir de um modelo 3D, considerando como ponto de vista uma das câmeras virtuais presentes no modelo.

A palavra pipeline é utilizada para indicar que se trata de um processo de vários estágios em que a saída de um estágio é a entrada de outro estágio na sequência. Muitas vezes o pipeline de visualização é também chamado de pipeline de renderização, um termo aportuguesado a partir do inglês “rendering pipeline”.

Veja a seguir, o pipeline tradicional de visualização, segundo HUGHES ET AL (2013).



Fonte: Retirada do LD.

• O modelo matemático é composto por cinco elementos: dados de textura, malhas poligonais, vértices das malhas poligonais, pontos de vista e dados de iluminação.

• As malhas poligonais são as descrições de como os vértices das malhas poligonais estão

conectados para formar as faces poligonais.

Ray *casting* e digitalização

Para se compreender o que deve ser feito para a digitalização de uma cena 3D, o primeiro passo após o estudo das transformações geométricas é o estudo dos efeitos de luz, cores e textura.

O princípio do *ray casting* é projetar uma linha reta, a partir do centro de projeção da tela, passando pelo pixel que se deseja determinar e em direção à cena, no sentido inverso do raio (*ray*) luminoso proveniente da fonte de luz.

Diversos objetos podem intersectar a linha reta seguida pelo raio, mas o *ray casting* busca o polígono mais próximo intersectado pelo raio. Do ponto intersectado o raio se reflete em direção à fonte de luz. O algoritmo de *ray casting* é um algoritmo de traçado de raios aplicável para o caso em que os objetos sejam opacos e recebem iluminação direta.

Para digitalizar toda a cena o algoritmo *ray casting* deve ser executado para cada pixel da imagem.
O código em Python a seguir apresenta esse algoritmo:

```
1
2  def raycastingP(f,model,d):
3      (xs,ys) = f.shape
4      ca = (0,0,-d)
5      d = 0
6      for y in range(ys):
7          for x in range(xs):
8              p = (x,y,0)
9              r = ray(ca,p)
10             for poly in model.getPolygons():
11                 pi = intersecao(r,poly)
12                 dp = distancia(ca,pi)
13                 if (dp < d):
14                     d = dp
15                     f(x,y) = getColor(pi,poly)
```

O código acima assume que todos os vértices do modelo já passaram por transformações geométricas e no momento da execução do algoritmo de *ray casting* encontram-se no sistema de coordenadas da tela de projeção.

Considere agora o código a seguir:

```
1
2  def raycastingP(f,model,d):
3      (xs,ys) = f.shape
4      ca = (0,0,-d)
5      d = 0
6      for y in range(ys):
7          for x in range(xs):
8              p = (x,y,0)
9              r = ray(ca,p)
10             for poly in model.getPolygons():
11                 pi = intersecao(r,poly)
12                 dp = distancia(ca,pi)
13                 if (dp < d):
14                     d = dp
15                     f(x,y) = getColor(pi,poly)
16
```

A diferença da primeira versão do algoritmo para a segunda pode parecer sutil e irrelevante, mas está fortemente relacionada à capacidade de processamento de cenas com mais ou menos detalhes.

Na primeira versão, o processamento de cada reta exige que esteja em memória todo o modelo tridimensional, enquanto que na segunda versão, o processamento de cada polígono exige que esteja em memória toda a matriz bidimensional da imagem digital. O processamento pixel a pixel exige muito mais memória do que o processamento triângulo a triângulo. O paralelismo, na segunda versão, está no número de polígonos, enquanto o da primeira versão está no número de pixels da imagem digital.

O ray casting considera que todos os polígonos da cena refletem diretamente a luz proveniente da fonte. Tal restrição impede a digitalização mais realística, pois desconsidera três outras formas de contribuição da fonte de luz para com o pixel projetado: reflexão indireta, sombra e refração.

Saiba Mais

A generalização do *ray casting* que considera luz indireta, sombras e refração, é denominada **ray tracing** (ANGEL E SHREINER, 2012). O *ray tracing* é o algoritmo de referência na atualidade.

Portanto, o algoritmo de *ray tracing* é uma extensão do algoritmo de ray casting. O ray casting considera todos os

objetos da cena como opacos e trata apenas a reflexão direta da luz sobre o objeto visualizado. O *ray tracing* considera também objetos transparentes, e trata a reflexão da luz sobre diversos objetos até chegar no observador. Trata ainda de sombras e refração.

Nesta webaula apresentou conceitos de síntese de imagem. As técnicas de síntese de imagem exploram de forma mais aprofundada o conhecimento de ótica. Pesquise mais e se aprofunde nesse conteúdo.