

Computer Graphics

Computer Graphics

From mathematics ...



... to the screen

E. Galin
Université Lyon 1

Computer Graphics

Introduction
Mathematics
Modeling
Color and Texturing
Shading
Realistic Rendering
Acceleration

Computer Graphics

Pipeline de création

Pipeline de génération

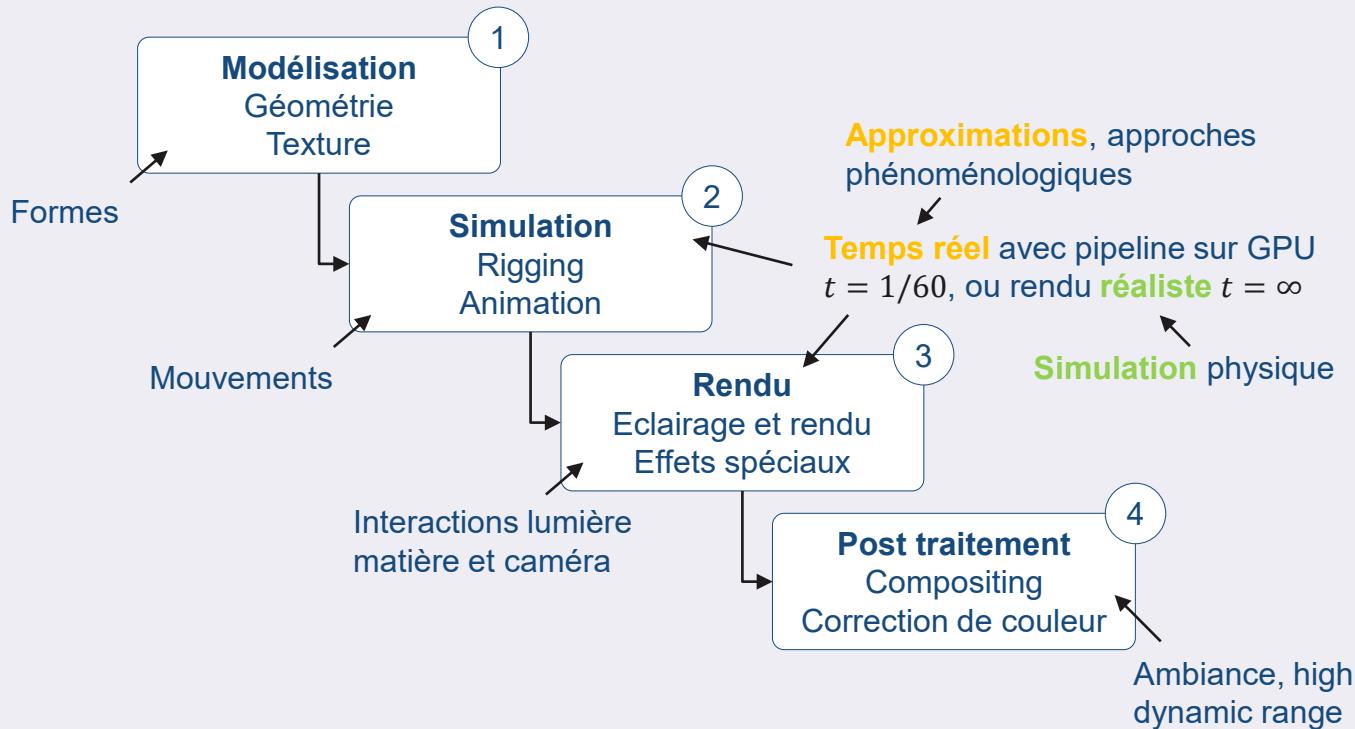
Introduction

Temps réel

Rendu réaliste

Etapes

Modélisation, animation, rendu, post traitement
Processus parfois partiellement entremêlés



eric.galin@liris.cnrs.fr
<http://liris.cnrs.fr/~egalin>

Fundamentals of Computer Graphics, Steve Marschner, Peter Shirley, ISBN 9780367505035, 2021 by A K Peters/CRC Press

Computer Graphics

Rasterization



Modeling
Geometry
Materials

A screenshot from a video game, likely The Witcher 3, showing a character standing on a wooden bridge over a river in a snowy, mountainous environment. Several callout boxes point to specific visual effects:

- An arrow points to a large, dark, jagged rock formation in the background with the text "Animated texture".
- An arrow points to the sky with the text "Sky sphere".
- An arrow points to a cluster of trees in the distance with the text "Transparent billboards".
- An arrow points to the character's arm with the text "Articulated body".
- An arrow points to the character's hand with the text "Animated sprites".
- An arrow points to the ground with the text "Snow particles".
- An arrow points to a wooden fence in the foreground with the text "Ambient".
- An arrow points to the water in the river with the text "Faked reflections".
- A large callout box in the bottom right corner contains the text "Rendering" in bold blue, followed by "Lighting" and "Camera" in regular blue.

The scene is set during sunset or sunrise, with warm light reflecting off the snow and clouds.

Vidéo

Pipeline de rendu par rastérisation

Introduction

Temps réel

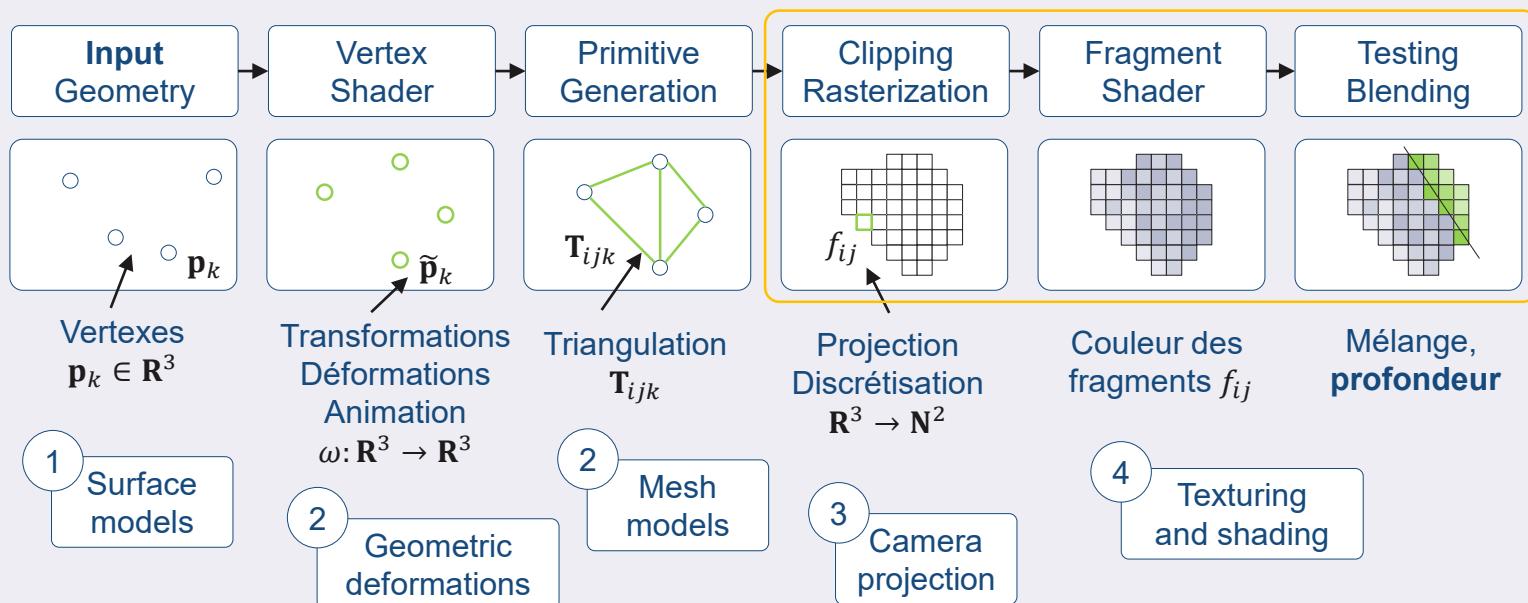
Rendu réaliste

Principe

Transformation, projection et discréétisation d'un modèle **continu** dans l'espace image **discret**

Modèles, animations et déformations doivent être adaptés au temps réel

Les ombres, la visibilité, l'éclairement sont **approximés**



eric.galin@liris.cnrs.fr
<http://liris.cnrs.fr/~egalin>

Depth Buffer

Introduction

Temps réel

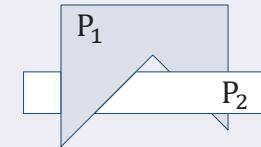
Rendu réaliste

Algorithmes de classement de profondeur

Algorithme du peintre : affichage dans un **ordre** de distance décroissante au point d'observation

Méthode de subdivision [Warnock 1969]

On ne peut **pas** trier des polygones ou des triangles



Depth Buffer

Discrétisation de tous les polygones \tilde{T} selon une grille de pixels E

Algorithme parallèle pour les pixels p_{ij} de chaque triangle

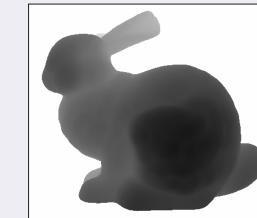
```
Initialize depth of pixels  $z_{ij}$  to  $\infty$ 
For all triangles  $T$                                 z buffer
    Project  $T$  onto screen
    For all pixels  $p_{ij}$  of  $\tilde{T}$                   Triangle in E
        Compute depth of  $p_{ij}$ 
        If  $z_T < z_{ij}$  then
            Set  $z_{ij} = z$ 
            Compute color  $c_{ij}$ 
        Otherwise
            Nothing to update
```

Distance to E

Hidden

∞	4	∞	∞	∞	∞	∞
∞	∞	4	4	3	3	∞
∞	∞	3	4	3	∞	
∞	∞	∞	3	3	∞	
∞	∞	∞	∞	2	∞	∞
∞	∞	∞	∞	2	∞	∞
∞						

1	4	11	11	∞	∞
10	10	4	4	3	3
9	9	3	4	3	∞
8	9	9	3	3	∞
∞	∞	∞	2	∞	∞
∞	∞	∞	∞	∞	∞



Computer Graphics

Ray Tracing



Vidéo

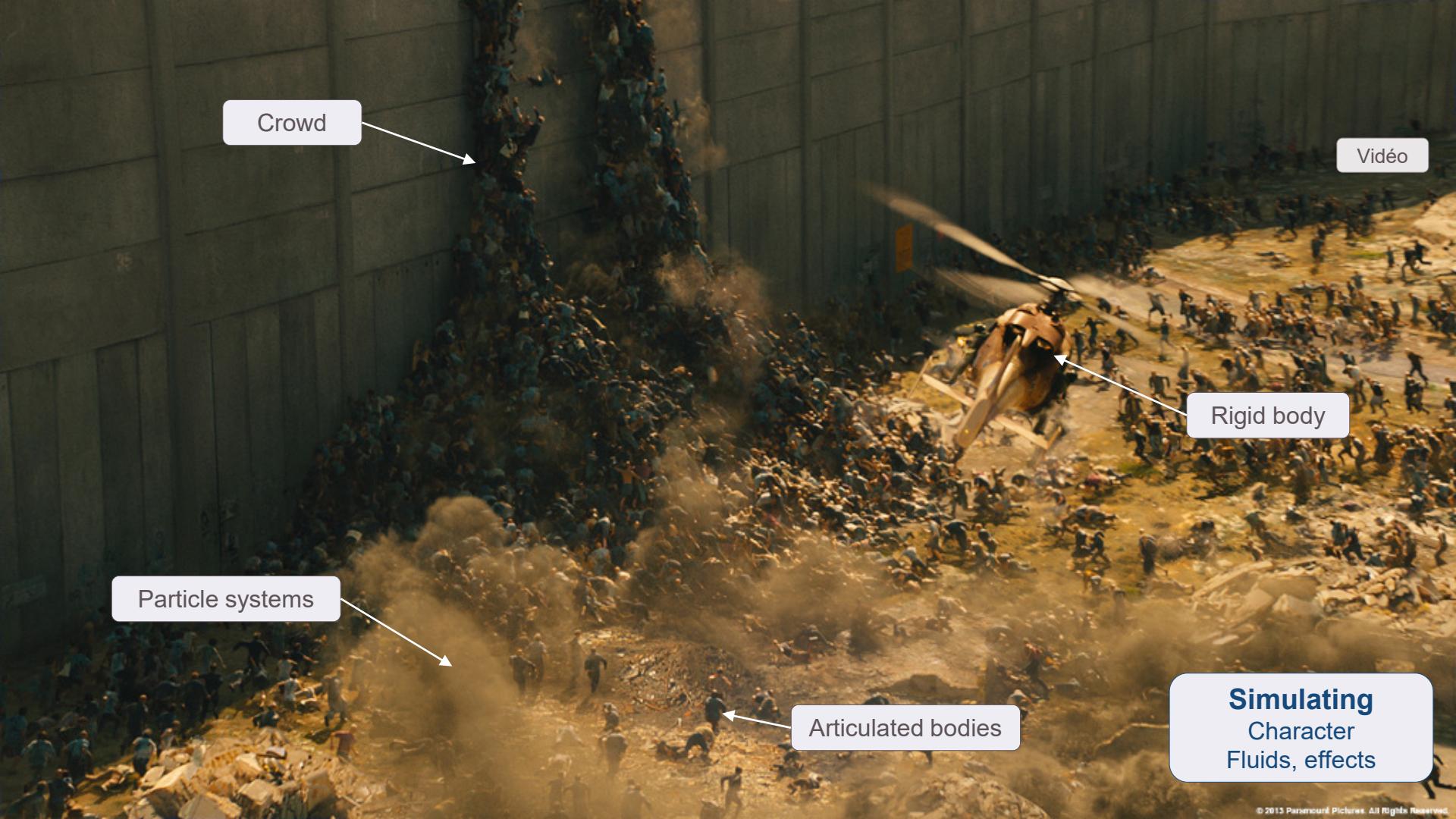
Volumes

Materials

Surfaces

Fog density

Modeling
Geometry
Materials



Crowd

Vidéo

Rigid body

Particle systems

Articulated bodies

Simulating
Character
Fluids, effects



Particle systems

Surface reflection

Radiosity

Rendering
Lighting
Camera

Rendu réaliste

Introduction

Temps réel

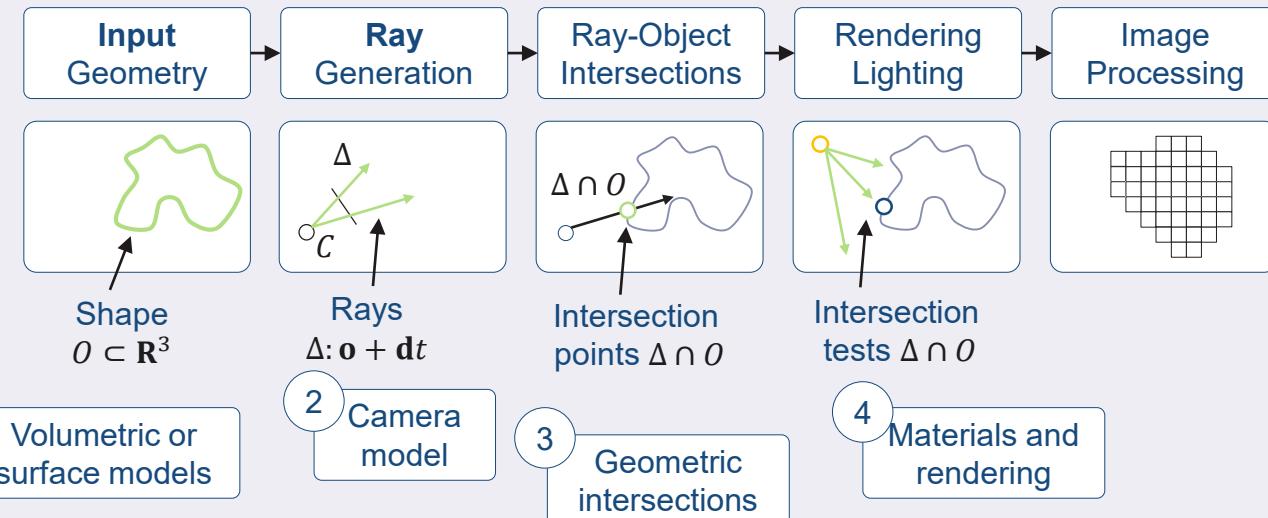
Rendu réaliste

Principe

Lancer de rayons depuis l'espace image **discret** vers le modèle **continu**

Modèles, déformations, textures plus complexes

Les ombres, la visibilité, l'éclairage plus réalistes



Université Claude Bernard Lyon 1

eric.galin@liris.cnrs.fr

<http://liris.cnrs.fr/~egalin>

Master

Lancer de rayon

Introduction

Temps réel

Rendu réaliste

Algorithme

Lancer un rayon primaire Δ_{ij} pour chaque pixel p_{ij} depuis l'observateur E

Calculer la plus proche intersection en calculant $\Delta_{ij} \cap T_k$ pour tous les triangles T_k

Déterminer la couleur c_{ij} selon le modèle d'éclairage

```
Initialize pixel color
For all pixels  $p_i$  on screen
  Compute ray  $\Delta_{ij}$ 
  For all triangles  $T$ 
    Intersect  $\Delta_{ij} \cap T$ 
    Check nearest
    Compute color  $c_{ij}$ 
```

Objects in \mathbb{R}^3

Visible from E

Recursive intersections



eric.galin@liris.cnrs.fr

<http://liris.cnrs.fr/~egalin>