Enseignement AS

Bibliothèque CGP
Application cours Anim et Simulation

damien.rohmer@polytechnique.edu

GTAS – 06/07/2022

CGP – Computer Graphics Programing library

Bibliothèque de code C++ orientée apprentissage pour faciliter la création, l'animation et l'interaction avec des scènes 3D.

Principe KISS

https://imagecomputing.net/cgp

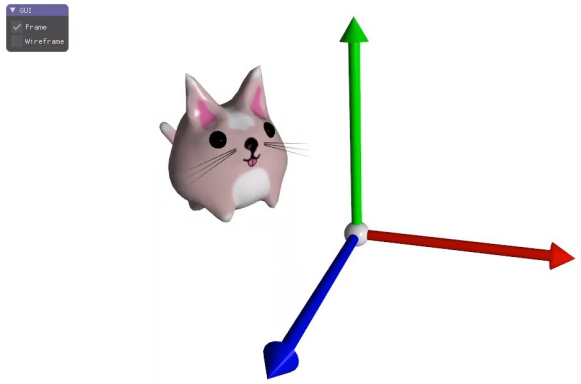
```
mesh_drawable shape;

void initialize() {
  ...
  mesh cat_model = mesh_load_file_obj("assets/cat.obj");
  shape.initialize_data_on_gpu(cat_model);
}

void display_frame() {
  ...
  float t = timer.t;

  shape.model.translation = {2*r*cos(t),r*abs(sin(t)),0};
  shape.model.rotation.from_axis_angle({0,1,0}, cos(2*t));

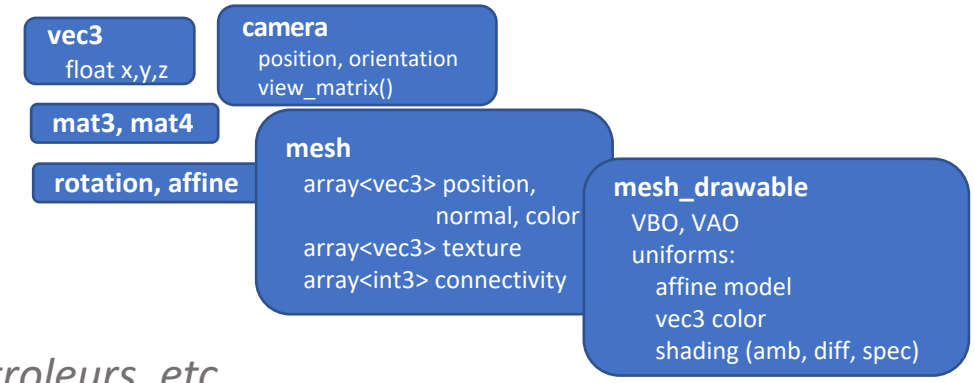
  draw(shape, environment);
  ...
}
```



Utilise

- *GLFW (fenêtre)*
- *GLAD (OpenGL)*
- *ImGUI (interface)*

CGP – Computer Graphics Programing library



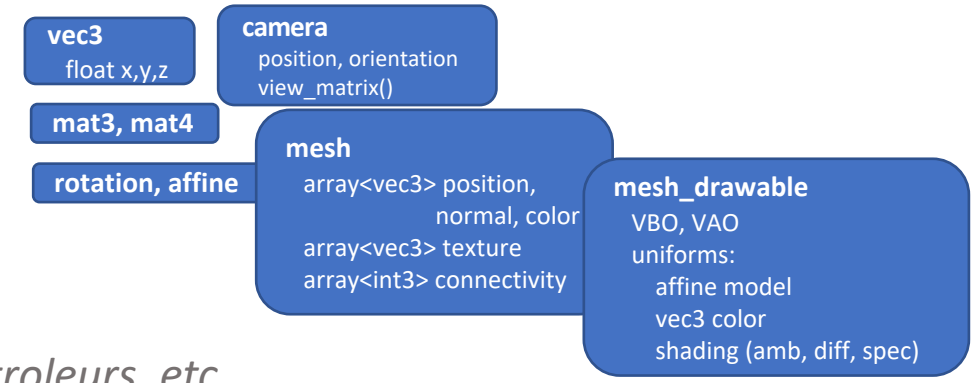
En pratique**:** Ensemble de

- **structures géométriques** simples
  vecteurs, matrices, maillage, transformations affines/rotations, etc.
- **d'aides à l'affichage** utilisant OpenGL
  function "draw", conteneur VBO/uniforms maillage, cameras & controleurs, etc.

CGP – Computer Graphics Programing library

En pratique**:** Ensemble de

- **structures géométriques** simples
  vecteurs, matrices, maillage, transformations affines/rotations, etc.
- **d'aides à l'affichage** utilisant OpenGL
  function "draw", conteneur VBO/uniforms maillage, cameras & controleurs, etc.

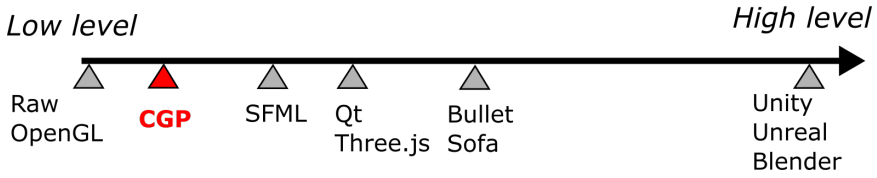


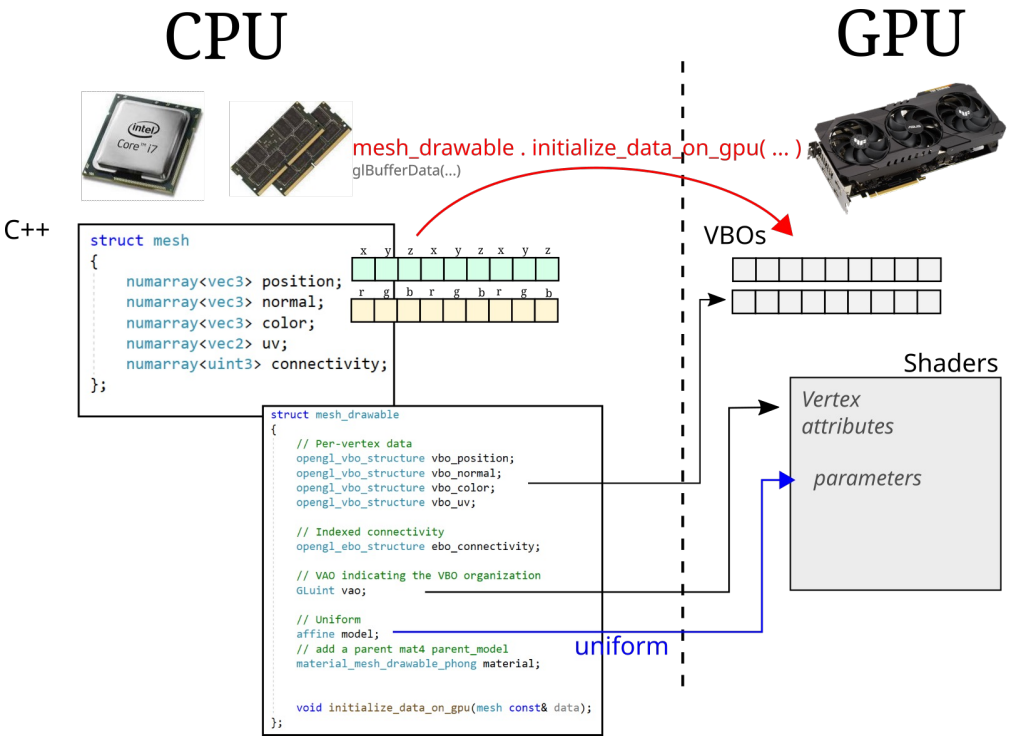
Objectifs:

- Gagner du temps / OpenGL brut
- Restant bas niveau

  Préserver la logique/compréhension du graphique temps-réel

  Préparation données contigues mémoire (envoie sur GPU)
  Affichage par shader
  Paramétrage de l'affichage par variables (/uniforms)





Exemples: Afficher un cube

```
mesh_drawable cube;
```

Initialisation

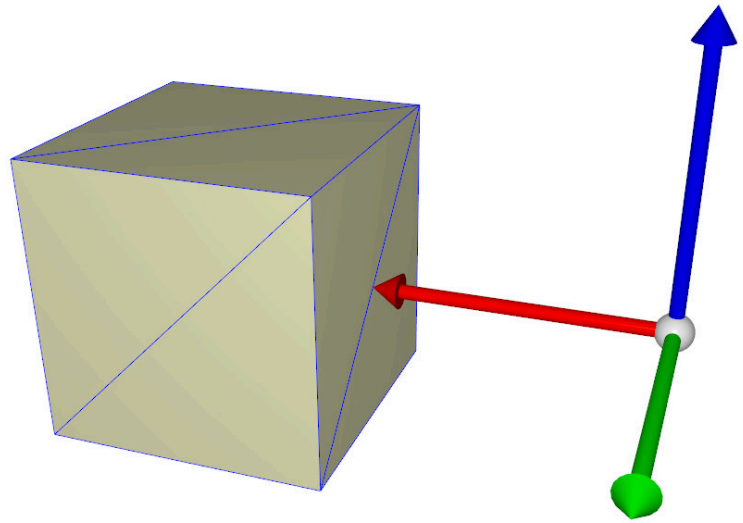
```
void initialize() {
   ...
  mesh cube_model = mesh_primitive_cube();
  shape.initialize_data_on_gpu(cube_model);
}
```

Affichage

```
void display_frame() {
   ...

  shape.model.translation = {tx,ty,tz};


  draw(shape, environment);
  draw_wireframe(shape, environment);
  ...
}
```



Exemples: Modifier couleur/model

```
mesh_drawable cube;
```

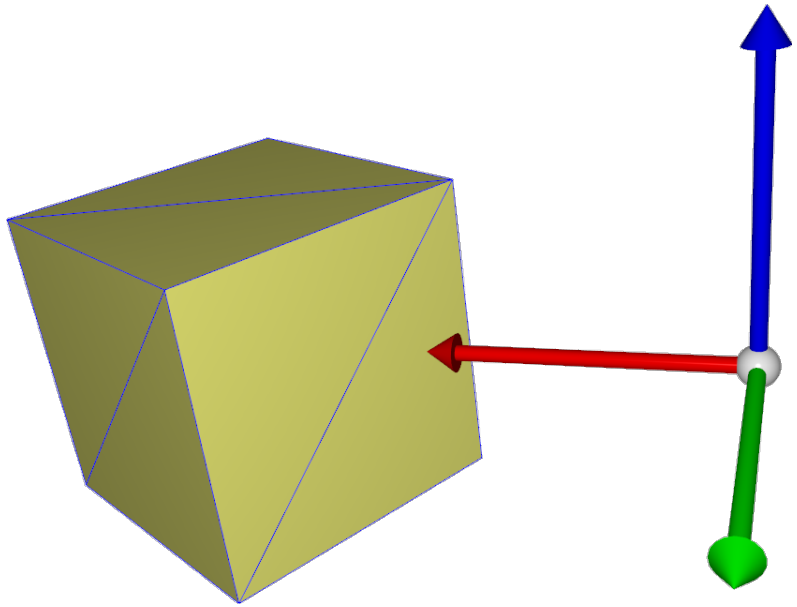
Initialisation

```
void initialize() {
   ...
  mesh cube_model = mesh_primitive_cube();
  shape.initialize_data_on_gpu(cube_model);
}
```

Affichage

```
void display_frame() {
   ...
  shape.material.color = {1,1,0.5};
  shape.model.translation = {tx,ty,tz};
  shape.model.rotation.from_axis_angle({0,0,1 }, -Pi/6 );

  draw(shape, environment);
  draw_wireframe(shape, environment);
  ...
}
```

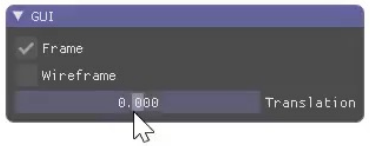


Exemples: Interaction

```
mesh_drawable cube;
```

Initialisation

```
void initialize() {
  ...
  mesh cube_model = mesh_primitive_cube();
  shape.initialize_data_on_gpu(cube_model);
}
```

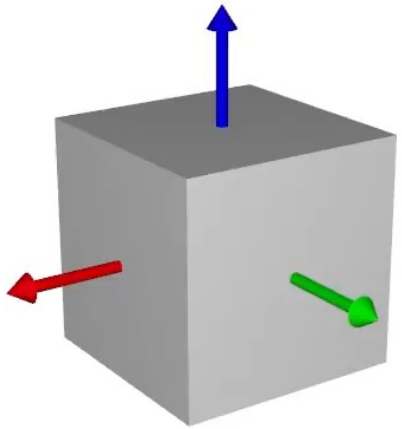


Affichage

```
void display_frame() {
  ...

  shape.model.translation = {tx,ty,tz};

  draw(shape, environment);
  draw_wireframe(shape, environment);
  ...
}
```



Interaction

```
void display_gui() {
  ImGui::SliderFloat("Translation", &shape.model.translation.x, -2.0f, 2.0f);
}
```

Guidelines

Orientée apprentissage:

1. Simple d'utilisation pour des scènes 3D basiques (débutant OpenGL/3D)

Concept simple => code simple (ex. Afficher maillage), calculs syntaxe lisible

---

```
rotation_transform R;

vec3 z = R * p + cross(a,b);

meshVar.position[5] = z;
```



~~R.clone().mul(p).add(a.clone().cross(...));~~



Guidelines

Orientée apprentissage:

1. **Simple d'utilisation** pour des scènes 3D basiques (débutant OpenGL/3D)

   Concept simple => code simple (ex. Afficher maillage), calculs syntaxe lisible

2. **Code source lisible** et compréhensible trivialement (débutants C++)

   Structures minimalistes: pas d'état cachés, paramètres accessibles

```
struct vec3 {
  float x, y, z;
}
```



```
template<typename T, qualifier Q>
struct vec<3, T, Q>{
 typedef T value_type;
 typedef vec<3, T, Q> type;
 typedef vec<3, bool, Q> bool_type;
 #if GLM_CONFIG_XYZW_ONLY
 T x, y, z;
 GLM_SWIZZLE_GEN_VEC_FROM_VEC3_COMP(T, Q, x, y, z)
 #endif//GLM_CONFIG_SWIZZLE
 union {
  struct{ T x, y, z; };
  struct{ T r, g, b; };
  struct{ T s, t, p; };
 typename detail::storage<3, T, detail::is_aligned<Q>::value>::type data;
 };
 ...
```



Guidelines

Orientée apprentissage:

1. **Simple d'utilisation** pour des scènes 3D basiques (débutant OpenGL/3D)

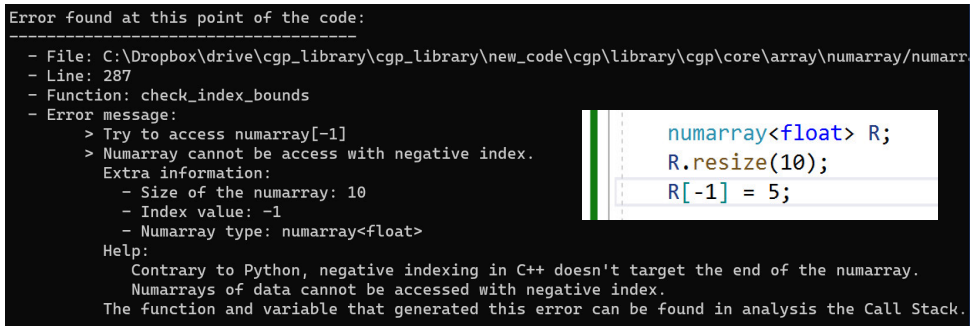
   Concept simple => code simple (ex. Afficher maillage), calculs syntaxe lisible

2. **Code source lisible** et compréhensible trivialement (débutants C++)

   Structures minimalistes: pas d'état cachés, paramètres accessibles

3. **Facile à compiler et debugger** (débutants C++)

   Vérification tailles/accès par défaut, pas de manipulation directs de pointeurs





Guidelines

Orientée apprentissage:

1. **Simple d'utilisation** pour des scènes 3D basiques (débutant OpenGL/3D)

   Concept simple => code simple (ex. Afficher maillage), calculs syntaxe lisible

2. **Code source lisible** et compréhensible trivialement (débutants C++)

   Structures minimalistes: pas d'état cachés, paramètres accessibles

3. **Facile à compiler et debugger** (débutants C++)

   Vérification tailles/accès par défaut, pas de manipulation directs de pointeurs

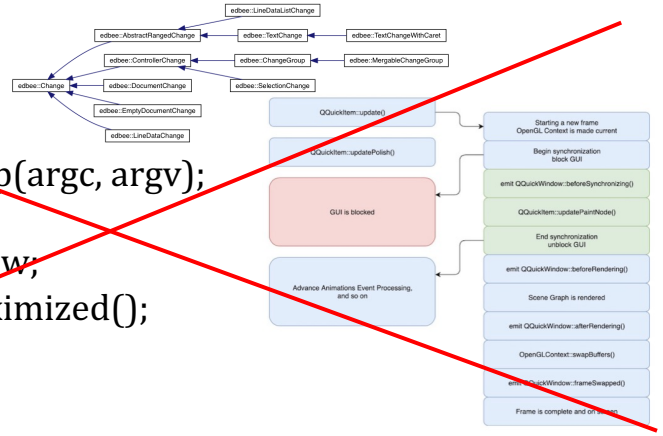
4. **Aide sans imposer de framework** (projets plus avancés)

   Compatible avec appels OpenGL directs.

```
while (!glfwWindowShouldClose(window))
{
   glClear(GL_DEPTH_BUFFER_BIT);
   glBufferData(...);
   glGenFrameBuffer(...);
   draw(mesh, environment);
   ...
```

```
QGuiApplication app(argc, argv);

GLWindow glWindow;
glWindow.showMaximized();
app.exec();
```



Utilisation: Computer Animation & Simulation

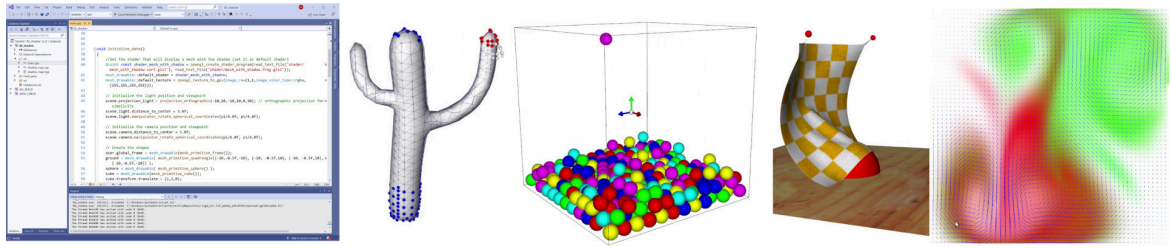
Durée: 9 séances 4h

14h cours/TD, 14h TP, 8h projet

Etudiants: x40, M1/M2

Ingé. Polytechnique, Masters IPP, AI-ViC, EuroTech: Background variable

Principe: 1 séance = 1 theme (basique) en cours et à coder en TP



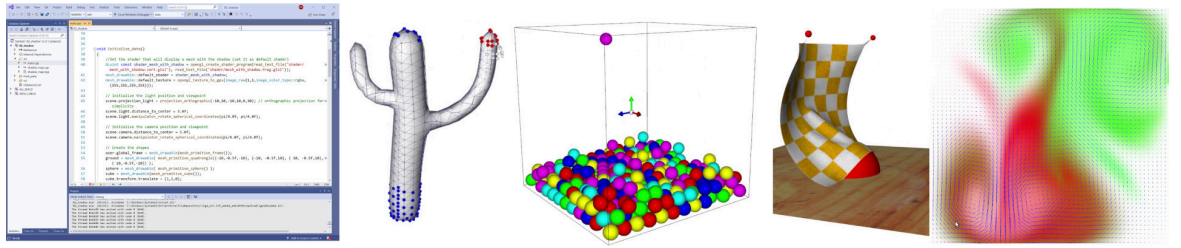
Utilisation: Computer Animation & Simulation

Durée: 9 séances 4h

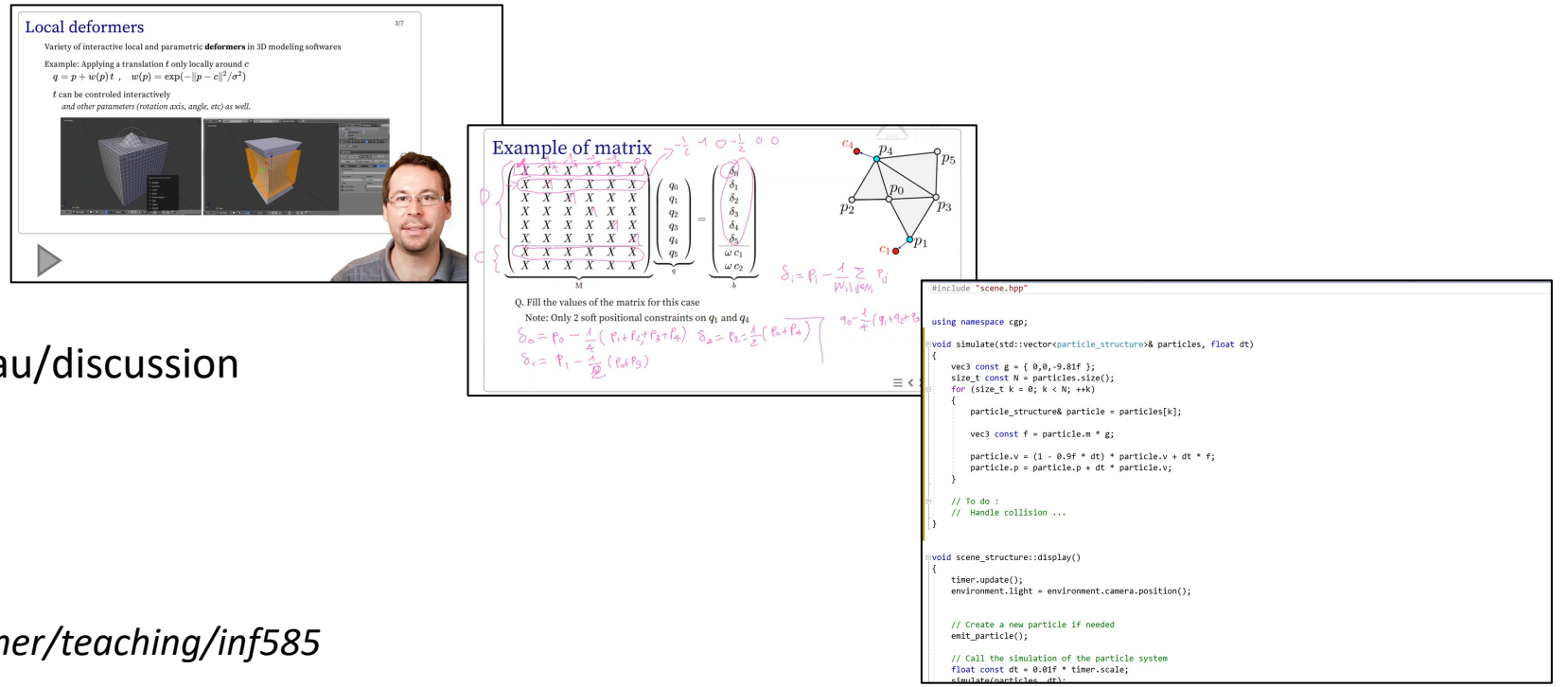
14h cours/TD, 14h TP, 8h projet

Etudiants: x40, M1/M2

Ingé. Polytechnique, Masters IPP, AI-ViC, EuroTech: Background variable



Principe: 1 séance = 1 theme (basique) en cours et à coder en TP



Format:

a) 1h cours (vidéo)

Disponible sur YouTube

b) 1h exercises interactifs tableau/discussion

Uniquement présentiel

c) 2h TP pratique

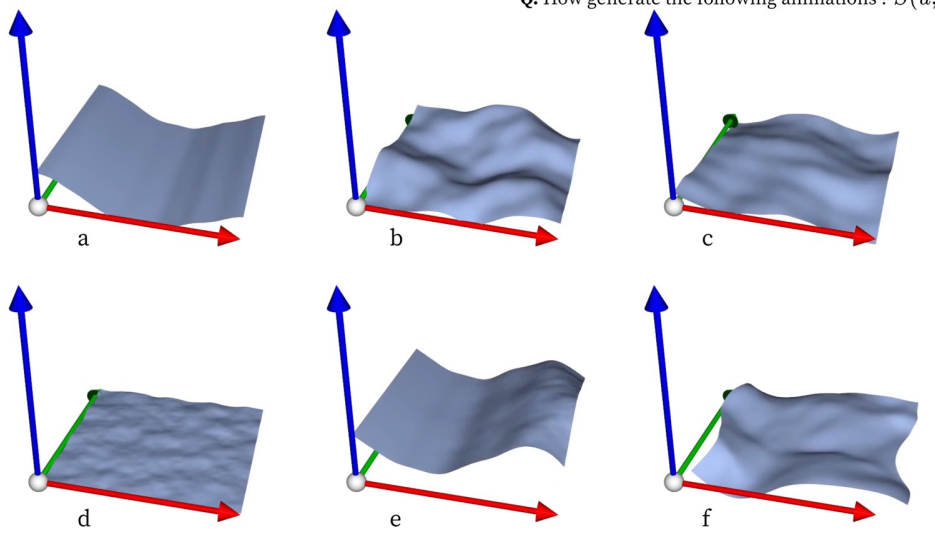
Videos (& slides) + TP disponible en ligne:
https://imagecomputing.net/damien.rohmer/teaching/inf585

Exemples d'exercices interactifs

Perlin Noise - Animation

Reference surface function: $(u, v) \in [0, 1]^2$, $f(u, v) = (u, v, 0)$

Q. How generate the following animations ? $S(u, v, t) = ...$

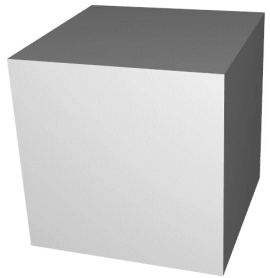


11/11

Deformers example cases

4/7

Q. How to model this interactive deformer ?



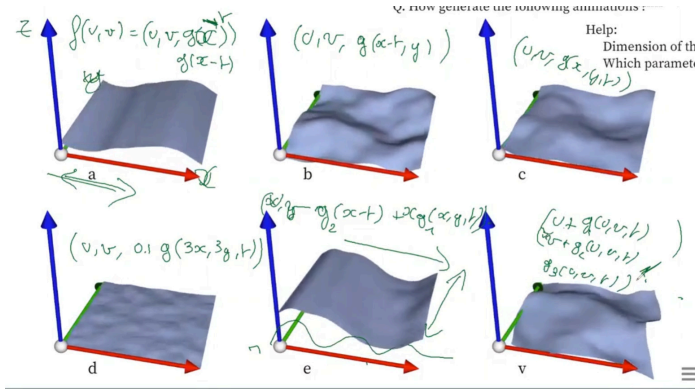
Input:

- c: Selected position on the surface
- t_{2D}: 2D displacement coordinates of the mouse on screen while dragging
- r: Circle radius

Computation:

For all positions of the surface p

...



For all positions:
dist = ||p-c||
w = exp(-dist^2/r^2)
R = rotation(axis, angle)
- axis = w * tr.x
- angle = normal(c) / or CameraO* (0,0,1)
p' = R*(p-c)+c

Thèmes

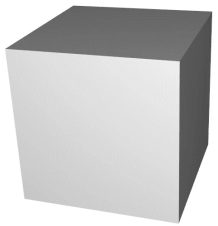
Anim géométrique

Procedural



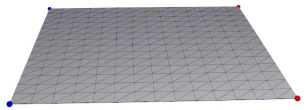
$$p(t) = p_0 + (r\cos(t), r\sin(t), t - t_0)$$

Deformeurs Interactifs



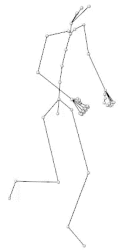
$$p(t) = \mathrm{p} + \mathrm{e}^{(\mathrm{p}-\mathrm{p_0})^2}\mathrm{t}$$

Laplacian/ARAP



$$E = \sum_{i=0}^{N-1} \|\delta(q_i) - \delta(p_i)\|^2 + \omega \sum_{i \in C} \|q_i - c_i\|^2$$

Skinning

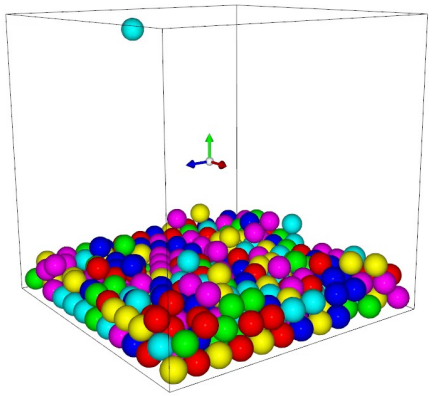




$$p = \left(\sum_{k=0}^{N-1} \alpha_k \, \mathrm{M}_k\right) p^0$$

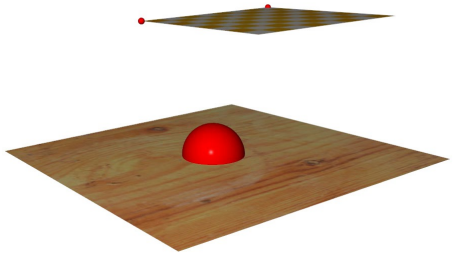
Simulation

Collision sphères



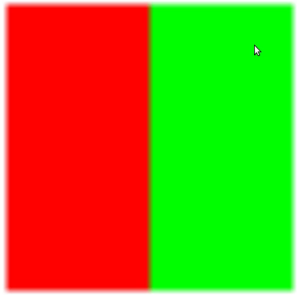
$$j = 2\frac{m_1 m_2}{m_1 + m_2}(v_2 - v_1) \cdot u$$

Masse ressorts



$$k\,(\|p(t)\| - l^0)\,\frac{p(t)}{\|p(t)\|}$$

Stable fluids



$$\Delta q = div(w)$$

SPH



$$A(p_i) = \sum_j A(p_j)\, m_j/\rho_j\, W_h(\|p_i - p_j\|)$$

Conclusion

Tout est disponible en ligne: *https://imagecomputing.net/damien.rohmer/teaching*

- Vidéos/slides de cours (YouTube + html)
- Enoncés + codes des TP

  Code complet solutions sur demande
- Bibliothèque CGP (indépendent) – Open Source

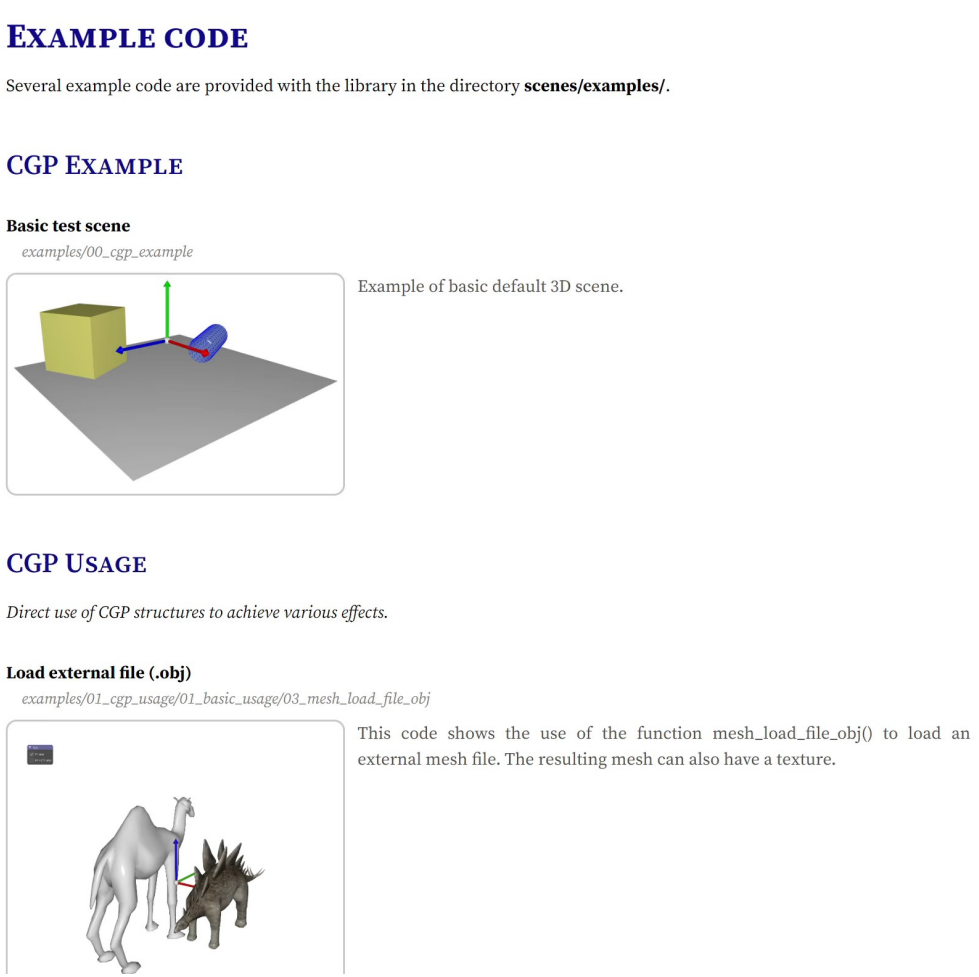
CGP encore en travail:

- Changements majeures d'API d'une année à l'autre

  Equilibre: simplicité du code / facilité utilisation / compréhension
- Manque de documentation

  Une fois stabilisée

  Pour l'instant: énoncés + code exemples
- Limité à des modèles animation/simulation simples



EXAMPLE CODE

Several example code are provided with the library in the directory **scenes/examples/**.

CGP EXAMPLE

Basic test scene

examples/00_cgp_example

Example of basic default 3D scene.

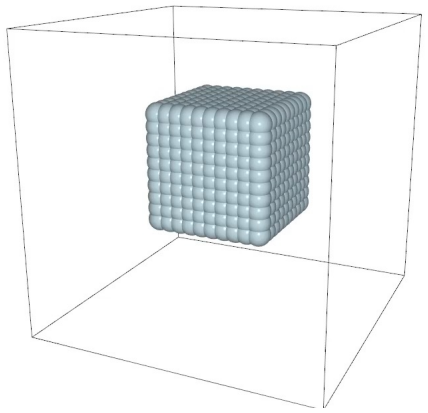
CGP USAGE

Direct use of CGP structures to achieve various effects.

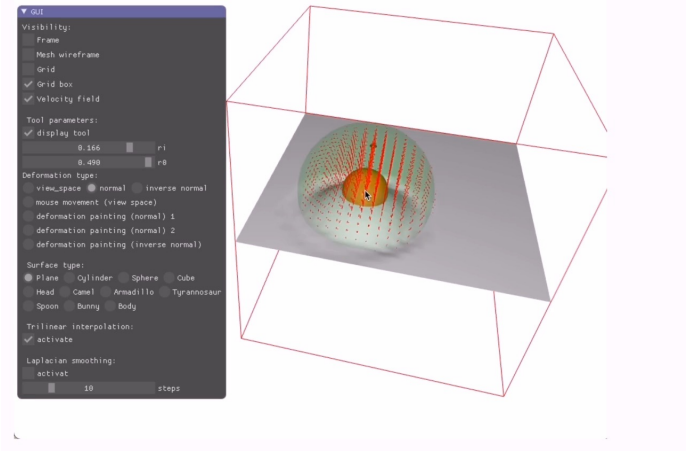
Load external file (.obj)

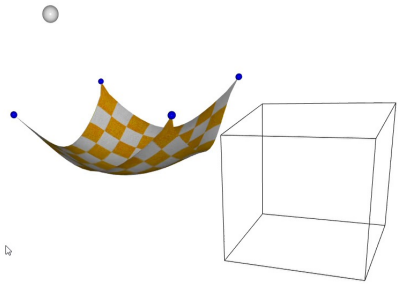
examples/01_cgp_usage/01_basic_usage/03_mesh_load_file_obj

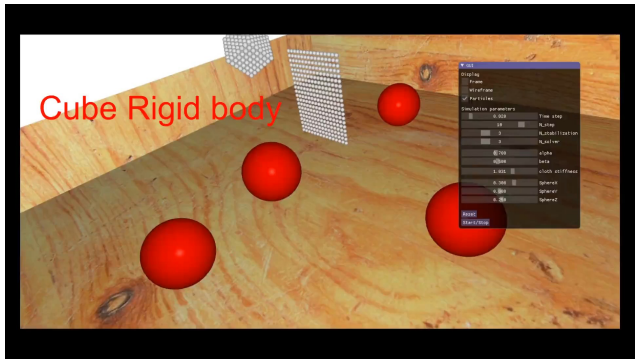
This code shows the use of the function mesh_load_file_obj() to load an external mesh file. The resulting mesh can also have a texture.

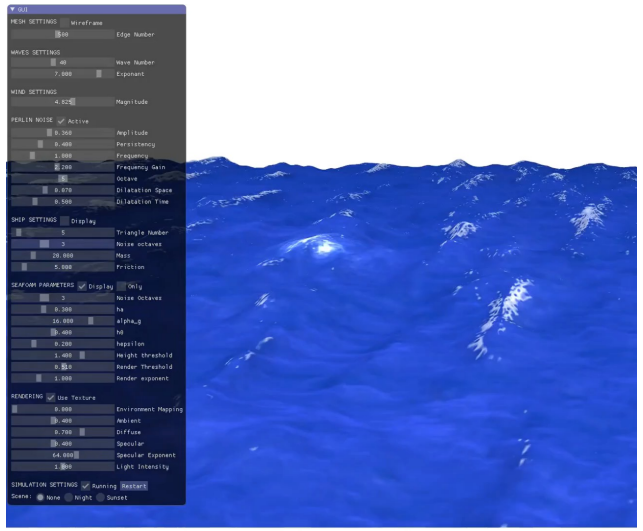


Merci









https://imagecomputing.net/damien.rohmer/teaching