MTI835

Transformations différents API 3D

V1.7

Université du Québec
École de technologie supérieure
Département de génie logiciel et des TI

Transformations – Unity

- Système main gauche
- Vector3.right (X)
- Vector3.up (Y)
- Vector3.forward (Z)
- Vector3
  - aussi back, down, left

Transformations – Unity

- Translation

```
void Update () {
    ...
  transform.Translate(
    new Vector3(1.0f, 0.0f, 0.0f) * Time.deltaTime,
    Space.World);

  transform.Translate(
    Vector3.forward * Time.deltaTime, Space.Self);
  ...
}
```

Transformations – Unity

- Rotation

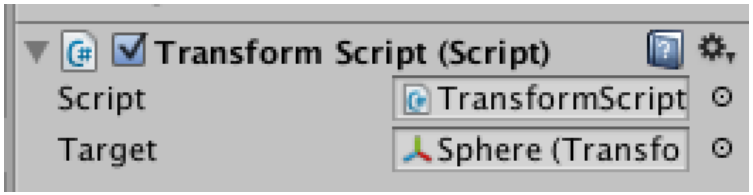
```
void Update () {
    ...
  transform.Rotate(
    new Vector3(1.0f,0.0f,0.0f)*Time.deltaTime*20,
    Space.World);
  // Euler angles in degrees

  transform.Rotate(Vector3.up * Time.deltaTime * 20,
                   Space.Self);
  ...
}
```

Transformations – Unity

- Rotation – orientation

```
public Transform target;
void Update () {
    ...
  transform.LookAt(target);
  ...
}
```

- Inspector view
  - glisser – déposer l'objet cible de Hierarchy view vers Inspector view



Three.js – Translation

- Object3D
  - .position : THREE.Vector3

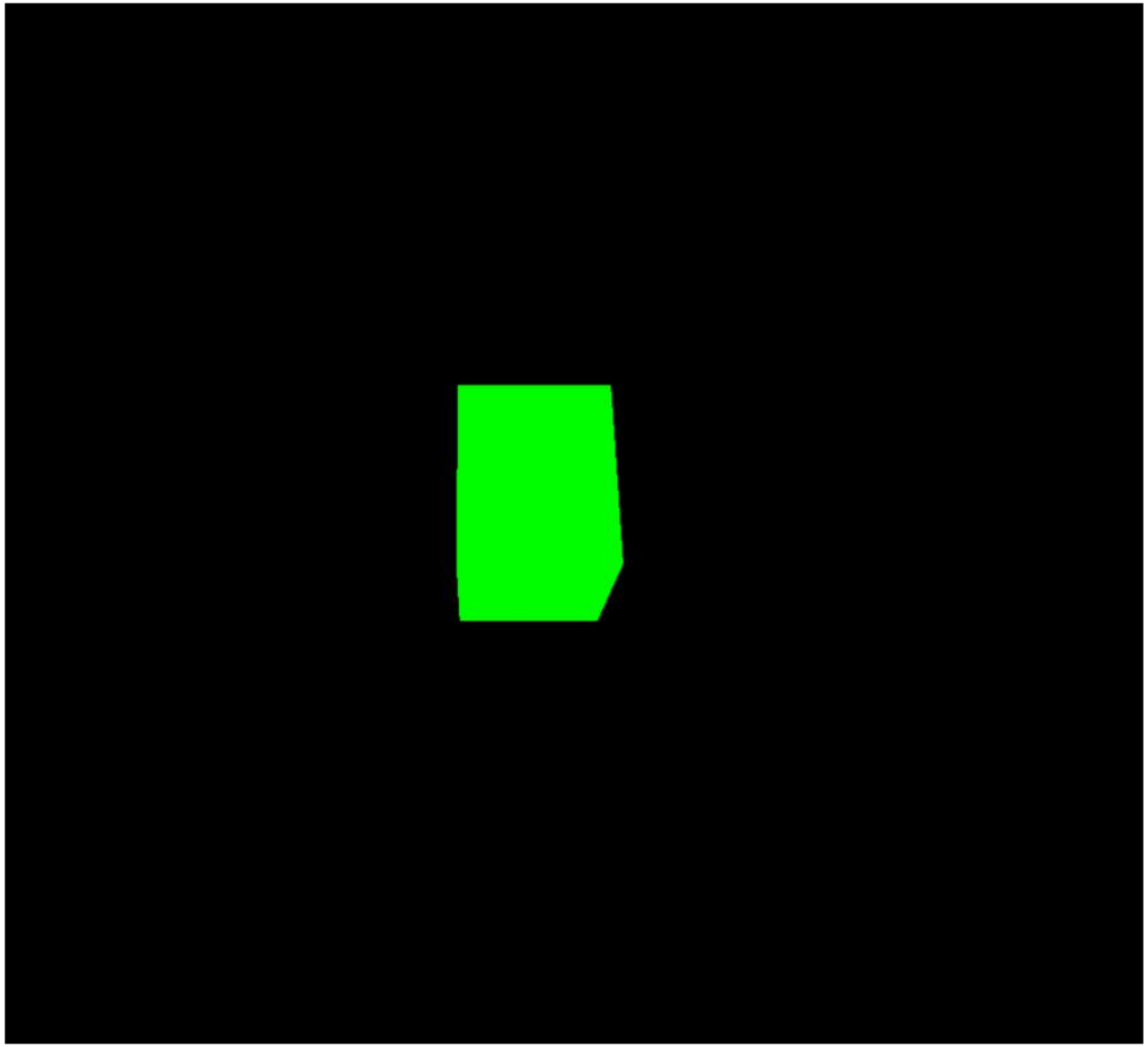
Three.js - Rotation

- Object3D
  - .rotation : THREE.Euler
  - .rotateX (rad : Float)
  - .rotateY (rad : Float)
  - .rotateZ (rad : Float)
  - .rotateOnAxis (axis : Vector3, angle : Float)
  - .lookAt (vector : Vector3)
    - (0, 0, 1) = vecteur vers l'avant
- Attention : angles an radians

Three.js – Changement d'échelle

- Object3D
  - .scale : THREE.Vector3

threejs_transformation.html



libigl – Eigen – Transform

- Transform

```
Transform<double, 3, Affine> t =
   Transform<double,3,Affine>::Identity();
```

libigl – Eigen – Translation

- translate()

```
Transform<double, 3, Affine> t =
  Transform<double,3,Affine>::Identity();

t.translate( Vector3d( tx, ty, tz ) );
```

libigl – Eigen – Rotation

- rotate()

```
Transform<double, 3, Affine> t =
  Transform<double,3,Affine>::Identity();

t.rotate(
  AngleAxisd( degrees / 180 * M_PI,
              Vector3d::UnitX() ) );
```

libigl – Eigen – Changement d'échelle

- scale()

```
Transform<double, 3, Affine> t =
  Transform<double,3,Affine>::Identity();

t.scale( Vector3d( sx, sy, sz ) );
```

libigl-example-transformations