Vision tridimensionelle

TP 0 : Geométrie projective et Mathematica

Soit le modele de camera suivant (en geometrie projective):

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = M \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

ou M est une matrice 3 x4

Illustrez ce modele de camera en 3D.

Initialisation

Cette section initialise Mathematica.

Pour ne pas avoir certains message d'erreur ennuyants....

```
Off[General::spell1];
```

Fonctions de base

Dans cette section, on declare les fonction de base qui sont utilisee souvent.

Homogenize (passe de projectif a euclidien)

```
h[v_] := Delete[v, -1] / v[[-1]];
```

```
h[{x, y, h}]
```

$$\left\{\frac{x}{h}, \frac{y}{h}\right\}$$

```
h[{x, y, z, h}]
```

$$\left\{\frac{x}{h}, \frac{y}{h}, \frac{z}{h}\right\}$$

Extraire une sous−matrice

```
sousmatrice[m_, r1_, r2_, c1_, c2_] := Take[#, {c1, c2}] & /@ Take[m, {r1, r2}];

m = Partition[Table[i, {i, 1, 16}], 4]; m // MatrixForm
```

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{pmatrix}$$

```
sousmatrice[m, 1, 3, 2, 4] // MatrixForm
```

$$\begin{pmatrix} 2 & 3 & 4 \\ 6 & 7 & 8 \\ 10 & 11 & 12 \end{pmatrix}$$

```
sousmatrice[m, 1, 3, 1, 3] // MatrixForm
```

$$\begin{pmatrix} 1 & 2 & 3 \\ 5 & 6 & 7 \\ 9 & 10 & 11 \end{pmatrix}$$

```
sousmatrice[m, 1, 3, 4, 4] // MatrixForm
```

$$\begin{pmatrix} 4 \\ 8 \\ 12 \end{pmatrix}$$

Matrice de changement d'echelle 3D (4x4)

```
scaleMatrix[v_] := DiagonalMatrix[{v, v, v, 1}];

scaleMatrix[{sx_, sy_, sz_}] := DiagonalMatrix[{sx, sy, sz, 1}];

scaleMatrix[a] // MatrixForm
```

$$\begin{pmatrix} a & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
scaleMatrix[{a, b, c}] // MatrixForm
```

$$\begin{pmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Matrice de translation 3D (4x4)

```
transMatrix[v_] := Transpose[{{1, 0, 0, 0}, {0, 1, 0, 0}, {0, 0, 1, 0}, Append[v, 1]}];

transMatrix[{a, b, c}] // MatrixForm
```

$$\begin{pmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Matrice de rotation arbitraire 3D (4x4)

```
rotMatrix[a_, u_] := Block[{m, s, x, y, z, nu}, (
    {x, y, z} = u / Sqrt[u.u];
    s = {{0, -z, y}, {z, 0, -x}, {-y, x, 0}};
    nu = {{x}, {y}, {z}};
    m = nu.Transpose[nu] + Cos[a] (IdentityMatrix[3] - nu.Transpose[nu]) + Sin[a] s;

    Join[Append[#, 0] & /@ m, {{0, 0, 0, 1}}]
    )];

{rotMatrix[a, {1, 0, 0}] // MatrixForm,
    rotMatrix[a, {0, 1, 0}] // MatrixForm,
    rotMatrix[a, {0, 0, 1}] // MatrixForm}
```

$$\left\{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \text{Cos}[a] & -\text{Sin}[a] & 0 \\ 0 & \text{Sin}[a] & \text{Cos}[a] & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \begin{pmatrix} \text{Cos}[a] & 0 & \text{Sin}[a] & 0 \\ 0 & 1 & 0 & 0 \\ -\text{Sin}[a] & 0 & \text{Cos}[a] & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \begin{pmatrix} \text{Cos}[a] & -\text{Sin}[a] & 0 & 0 \\ \text{Sin}[a] & \text{Cos}[a] & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}\right\}$$

Matrice de parametres internes (passe de camera −> image)

Centre de l'image (cx,cy), distance focal (fx,fy)

```
internMatrix[cx_, cy_, fx_, fy_] :=
  {{fx, 0, cx, 0}, {0, fy, cy, 0}, {0, 0, 1, 0}, {0, 0, 0, 1}};
```

```
internMatrix[128, 128, 256, 256] // MatrixForm
```

$$\begin{pmatrix} 256 & 0 & 128 & 0 \\ 0 & 256 & 128 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Projection d'un point 3D vers un point 2D

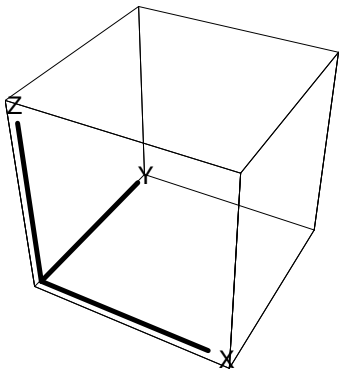
```
projete[{x_, y_, z_}, m_] := h[Take[m.{x, y, z, 1}, 3]];
```

Creation d'une systeme d'axe (x,y,z)

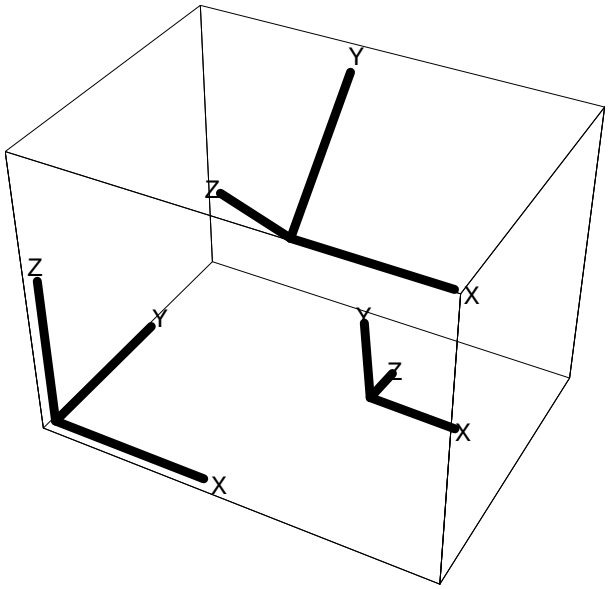
```
axes[] := axes[IdentityMatrix[4]];

axes[t_] := Module[{p}, (
            p = {{0, 0, 0, 1}, {1, 0, 0, 1}, {0, 1, 0, 1}, {0, 0, 1, 1}};
            q = (h[t.#]) & /@ p;
            r = (h[t.scaleMatrix[{1.1, 1.1, 1.1}].#]) & /@ p;
            Graphics3D[{
            Thickness[0.015],
            Line[q[[{1, 2}]]],
            Line[q[[{1, 3}]]],
            Line[q[[{1, 4}]]],
            Text["X", r[[2]]],
            Text["Y", r[[3]]],
            Text["Z", r[[4]]]
            }]
        )];

Show[axes[]];
```



```
Show[axes[],
    axes[transMatrix[{1, 1, 1}].rotMatrix[Pi / 4, {1, 0, 0}]],
    axes[
   transMatrix[{2, 0, 1}].rotMatrix[Pi / 4, {0, 1, 1}].scaleMatrix[{0.5, 0.5, 0.2}]]];
```



■ Fonctions principales

Dans cette section, on declare les fonctions les plus importantes

■ Geometrie de camera

Affiche la geometrie de la camera, pour une taille d'image connue et une matrice de passage m

On "deprojete" les coins de l'image, et l'origine est directement la translation de im.
La distance entre le plan de projection et l'origine est donne par f.

```
geomCamera[imgx_, imgy_, f_, m_] := Module[{orig, coins, cs, im}, (
            im = Inverse[m];
            orig = Flatten[sousmatrice[im, 1, 3, 4, 4]];
            coins =
     {{0, 0, f, 1}, {f * imgx, 0, f, 1}, {f * imgx, f * imgy, f, 1}, {0, f * imgy, f, 1}};
            cs = Take[im.#, 3] & /@ coins;
            Graphics3D[{Line[{orig, #}] & /@ cs, Polygon[cs]}]
            )];
```

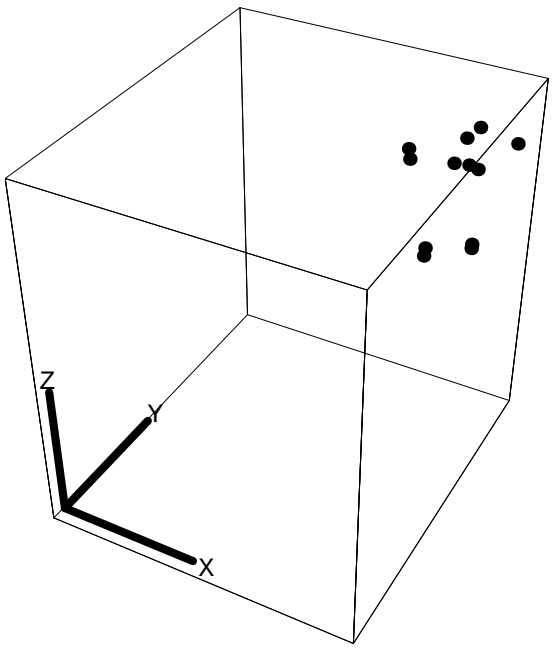
Test

Creation de quelques points 3D a projeter

```
p = Table[{
            Random[Real, {15, 25}],
            Random[Real, {15, 25}],
            Random[Real, {15, 25}]}, {i, 1, 12}];
p // MatrixForm
```

$$\begin{pmatrix} 16.9568 & 24.4798 & 20.4461 \\ 20.0054 & 18.3882 & 24.9934 \\ 19.5727 & 19.9254 & 15.2726 \\ 17.236 & 24.5664 & 18.4409 \\ 16.8502 & 15.3549 & 23.5009 \\ 18.7117 & 17.8665 & 22.4397 \\ 21.1956 & 22.1918 & 22.3071 \\ 17.1364 & 18.0223 & 15.6435 \\ 20.3503 & 17.6566 & 22.5761 \\ 20.6381 & 16.9621 & 17.6632 \\ 18.0034 & 15.7128 & 16.6895 \\ 15.4271 & 18.437 & 22.2719 \end{pmatrix}$$

```
s0 = Show[axes[scaleMatrix[10]], Graphics3D[{PointSize[0.025], Point /@ p}]]
```



- Graphics3D -

Fabrique une matrice camera m test

Pas de rotation, pas de translation
Seulement une image 256x256 (angle de vue 90 degres)

```
campos = {2, 2, 2};

mext = N[rotMatrix[Pi / 4, {1, -1, 0}].transMatrix[-campos]]; mext // MatrixForm
```

$$\begin{pmatrix} 0.853553 & -0.146447 & -0.5 & -0.414214 \\ -0.146447 & 0.853553 & -0.5 & -0.414214 \\ 0.5 & 0.5 & 0.707107 & -3.41421 \\ 0 & 0 & 0 & 1. \end{pmatrix}$$

```
mint = internMatrix[100, 100, 200, 200]; mint // MatrixForm
```

$$\begin{pmatrix} 200 & 0 & 100 & 0 \\ 0 & 200 & 100 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
m = mint.mext; m // MatrixForm
```

$$\begin{pmatrix} 220.711 & 20.7107 & -29.2893 & -424.264 \\ 20.7107 & 220.711 & -29.2893 & -424.264 \\ 0.5 & 0.5 & 0.707107 & -3.41421 \\ 0. & 0. & 0. & 1. \end{pmatrix}$$

Inverse de m : image −> camera −> scene

```
im = Inverse[m]; im // MatrixForm
```

$$\begin{pmatrix} 0.00426777 & -0.000732233 & 0.146447 & 2. \\ -0.000732233 & 0.00426777 & 0.146447 & 2. \\ -0.0025 & -0.0025 & 1.20711 & 2. \\ 0. & 0. & 0. & 1. \end{pmatrix}$$

Projete tous les points sur le plan z=1 de la camera

```
pcam = Append[projete[#, m], 1] & /@ p;
N[pcam] // MatrixForm
```

$$\begin{pmatrix} 101.582 & 148.954 & 1. \\ 108.799 & 99.1312 & 1. \\ 142.292 & 144.891 & 1. \\ 109.694 & 157.719 & 1. \\ 99.79 & 89.5853 & 1. \\ 111.195 & 105.697 & 1. \\ 119.228 & 125.079 & 1. \\ 129.743 & 136.766 & 1. \\ 119.536 & 102.461 & 1. \\ 142.23 & 115.855 & 1. \\ 134.12 & 115.973 & 1. \\ 92.6036 & 113.173 & 1. \end{pmatrix}$$

"Deprojete" les points images vers la scene

```
pcamscene = (Take[im.Append[#, 1], 3]) & /@ (10 * pcam);
pcamscene // MatrixForm
```

$$\begin{pmatrix} 6.70908 & 9.07765 & 7.80766 \\ 7.38188 & 6.89849 & 8.87281 \\ 8.47622 & 8.60619 & 6.89148 \\ 6.99106 & 9.39235 & 7.38574 \\ 7.0673 & 6.55706 & 9.33669 \\ 7.43608 & 7.16116 & 8.64876 \\ 7.63698 & 7.92952 & 7.96339 \\ 8.00013 & 8.35129 & 7.40836 \\ 7.81572 & 6.96199 & 8.52114 \\ 8.68617 & 7.36745 & 7.61894 \\ 8.33921 & 7.43185 & 7.81874 \\ 6.58789 & 7.61633 & 8.92666 \end{pmatrix}$$

```
Show[s0, Graphics3D[{PointSize[0.015], Point /@ pcamscene}],
    Graphics3D[Line[{campos, #}] & /@ p],
    geomCamera[200, 200, 10, m],
    PlotRange -> All, ViewPoint -> {-0.75, 1, 0.2}];
```