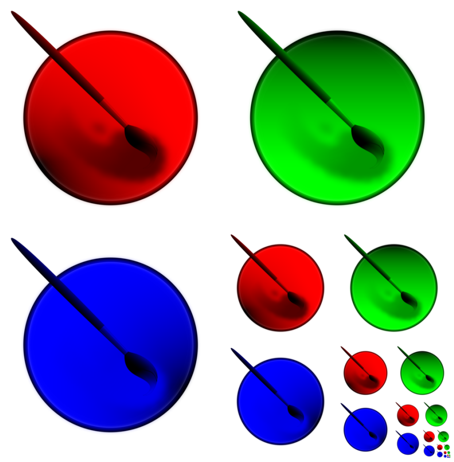
Solution du devoir 1

Auteur : Alexandre Blondin Massé

1. (15 points) Écrivez un programme qui prend en entrée une image RGB de format $2^d \times 2^d$ et qui produit une image de format $2^{d+1} \times 2^{d+1}$ qui stocke les canaux rouge, vert et bleu de l'image pour les dimensions $2^i \times 2^i$, où $i = 0, 1, \ldots, d$.

   Votre programme doit être un script shell qui fait appel aux programmes `convert` et `montage` de ImageMagick ou un programme dans votre langage préféré (par exemple Python) qui fait appel à une bibliothèque qui interface avec ImageMagick (par exemple, pgmagick).

   Ainsi, on s'attend à obtenir l'image de droite suivante lorsqu'on utilise comme entrée le logo de Krita (image de gauche):



Solution:

Le script shell suivant (nommé `mipmap`) permet de produire l'image.

```
#!/bin/sh

make_image() {
    local d="$1"
    if [ "$d" -ne 1 ]; then
        local dd=$((d / 2))
        make_image "$dd"
        convert $image -resize "${dd}x${dd}"\
            -channel GB -evaluate set 0 +channel "$dir/r.png"
        convert $image -resize "${dd}x${dd}"\
            -channel RB -evaluate set 0 +channel "$dir/g.png"
        convert $image -resize "${dd}x${dd}"\
            -channel RG -evaluate set 0 +channel "$dir/b.png"
        montage "$dir/r.png" "$dir/g.png" "$dir/b.png" "$dir/$dd.png" \
```

```
            -geometry +0+0 -tile 2x2 "$dir/$d.png"
    fi
}

image="$1"
w="$(convert $image -ping -format "%w" info:)"
h="$(convert $image -ping -format "%h" info:)"

dir="$(mktemp -d)"
convert $image -resize 1x1 "$dir/1.png"
make_image "$w"
cp "$dir/$w.png" "./$image.mipmap.png"
```

Il suffit de lancer la commande

```
$ ./mipmap krita-logo.png
```

2. (30 points) Dans cette question, on s'intéresse au calcul de l'intersection entre une droite et un cercle.

   Remarque: Ne confondez par *cercle* et *disque*. Un *cercle* est une courbe alors qu'un disque est une surface.

   (a) (10 points) Donnez le pseudocode d'une fonction

   fonction RACINES(a, b, c : réels) : ensemble de réels

   qui retourne toutes les solutions de l'équation $ax^2 + bx + c = 0$.
   Remarque. Le nombre de solutions est 0, 1 ou 2.

   Solution: Il suffit d'abord de calculer le discriminant $\delta = b^2 - 4ac$. Ensuite, on retourne les résultats suivants selon la valeur de δ. En pseudocode, on obtient donc

   fonction RACINES(a, b, c : réels) : ensemble de réels
       $\delta \leftarrow b * b - 4 * a * c$
       si $\delta < 0$ **alors**
           retourner $\emptyset$
       sinon si $\delta = 0$ **alors**
           retourner $\{-b/(2 * a)\}$
       sinon
           retourner $\{(-b + \sqrt{\delta})/(2 * a), (-b - \sqrt{\delta})/(2 * a)\}$
       fin si
   fin fonction

   (b) (10 points) Soit C un cercle de centre A et de rayon r. Soit D une droite de vecteur directeur $\vec{u}$ passant par le point P_0. Rappelons qu'un point P se trouve sur C si et seulement si

   $$\|P - A\|^2 = r^2 \tag{1}$$

et que P se trouve sur D si et seulement s'il existe un réel t tel que

$$P = P_0 + t\vec{u} \tag{2}$$

Montrez que P se trouve simultanément sur C et sur D si et seulement s'il existe un réel t tel que $P = P_0 + t\vec{u}$ et

$$\left[\|\vec{u}\|^2\right] t^2 + [2(P_0 - A) \cdot \vec{u}]\, t + \left[\|P_0 - A\|^2 - r^2\right] = 0. \tag{3}$$

Indice. Vous pouvez utiliser les propriétés suivantes du produit scalaire et de la norme sans les démontrer:

- (carré de la norme) $\|\vec{u}\|^2 = \vec{u} \cdot \vec{u}$
- (commutativité) $\vec{u} \cdot \vec{v} = \vec{v} \cdot \vec{u}$
- (distributivité) $(\vec{u} + \vec{v}) \cdot \vec{w} = \vec{u} \cdot \vec{w} + \vec{v} \cdot \vec{w}$

Solution: Il suffit de substituer P par $P_0 + t\vec{u}$ dans l'équation (1). On obtient

$$\begin{array}{lll}
 & \|P_0 + t\vec{u} - A\|^2 = r^2 & \\
\Rightarrow & \|(P_0 - A) + t\vec{u}\|^2 = r^2 & \\
\Rightarrow & ((P_0 - A) + t\vec{u}) \cdot ((P_0 - A) + t\vec{u}) = r^2 & \text{(carré de la norme)} \\
\Rightarrow & (P_0 - A) \cdot (P_0 - A) + 2(P_0 - A) \cdot t\vec{u} + t^2(\vec{u} \cdot \vec{u}) = r^2 & \text{(distributivité et commutativité)} \\
\Rightarrow & \|P_0 - A\|^2 + [2(P_0 - A) \cdot \vec{u}]t + \|\vec{u}\|^2 t^2 = r^2 & \text{(carré de la norme)} \\
\Rightarrow & \left[\|\vec{u}\|^2\right] t^2 + [2(P_0 - A) \cdot \vec{u}]\, t + \left[\|P_0 - A\|^2 - r^2\right] = 0 &
\end{array}$$

tel que voulu.

(c) (10 points) Déduire des deux questions précédentes le pseudocode d'une fonction

fonction INTERSECTION(C : cercle, Δ : droite) : ensemble de points

qui retourne l'ensemble des points qui se situent à l'intersection du cercle C de centre A de rayon r, et de la droite Δ passant par le point P_0 de vecteur directeur $\vec{u}$.

Remarque 1. Vous pouvez supposer que les opérations habituelles sur les points et sur les vecteurs (addition, soustraction, produit scalaire, norme, etc.) sont disponibles.

Remarque 2. Vous pouvez utiliser la notation $C.A$, $C.r$, $\Delta.P_0$ et $\Delta.\vec{u}$ pour accéder aux différents paramètres du cercle et de la droite.

Solution: Il suffit de calculer les coefficients a, b et c du polynôme de degré 2 en t obtenu à la sous-question précédente, puis ensuite d'appeler la fonction

RACINES() pour trouver les valeurs de t possibles. Ensuite, on récupère les points en substituant chaque valeur de t dans l'équation (2). Le pseudocode correspondant est celui-ci:

```
fonction INTERSECTION(C : cercle, Δ : droite) : ensemble de points
    a ← ‖Δ.u⃗‖²
    b ← 2 ∗ (Δ.P₀ − C.A) · Δ.u⃗
    c ← ‖Δ.P₀ − C.A‖² − C.r²
    T ← RACINES(a, b, c)
    retourner {Δ.P₀ + t ∗ Δ.u⃗ | t ∈ T}
fin fonction
```

3. (20 points) Écrivez un script Blender qui génère un maillage ayant la forme d'une grille comme celle illustrée à la figure 1.

   Les paramètres à prendre en compte sont les suivants:

   - Le nombre de rangées r;
   - Le nombre de colonnes c;
   - L'épaisseur des barreaux b;
   - L'épaisseur des trous t.

   Remarque. Il n'y a aucune contrainte sur l'utilisation d'opérateurs disponibles dans Blender (par exemple une extrusion).

Solution: Le script suivant produit le résultat attendu:

```python
import bpy
from math import pi, cos, sin

def xyuv_to_v(x, y, u, v):
    r"""
    Returns the vertex index from x, y, u and v.

    - `x`: the column number
    - `y`: the row number
    - `u`: the column offset
    - `v`: the row offset
    """
    return 4 * ((c + 1) * y + x) + 2 * u + v

# Parameters
r = 4
c = 6
b = 0.2
t = 0.8
o = b + t
w = o * c + b
h = o * r + b

# Vertices and faces
```

```
verts = [(x * o + u * b - 0.5 * w, 0.5 * b, y * o + v * b - 0.5 * h) \
        for y  in range(r + 1) for x in range(c + 1) \
        for (u, v) in [(0,0), (0,1), (1,0), (1,1)]]
small_squares = \
    [(xyuv_to_v(x, y, 0, 0), xyuv_to_v(x, y, 0, 1), \
      xyuv_to_v(x, y, 1, 1), xyuv_to_v(x, y, 1, 0))
          for y  in range(r + 1) for x in range(c + 1)]
horizontal_rectangles = \
    [(xyuv_to_v(x, y,     1, 0), xyuv_to_v(x + 1, y, 0, 0), \
      xyuv_to_v(x + 1, y, 0, 1), xyuv_to_v(x, y,     1, 1))
          for y  in range(r + 1) for x in range(c)]
vertical_rectangles = \
    [(xyuv_to_v(x, y,     0, 1), xyuv_to_v(x, y, 1, 1), \
      xyuv_to_v(x, y + 1, 1, 0), xyuv_to_v(x, y + 1, 0, 0))
          for y  in range(r)     for x in range(c + 1)]
faces = small_squares + horizontal_rectangles + vertical_rectangles

# Model
mesh_data = bpy.data.meshes.new('Grid')
mesh_data.from_pydata(verts, [], faces)
mesh_data.update()
obj = bpy.data.objects.new('Grid', mesh_data)
scene = bpy.context.scene
scene.objects.link(obj)

# Extrude
scene.objects.active = obj
bpy.ops.object.mode_set(mode='EDIT')
bpy.ops.mesh.select_all(action='SELECT')
bpy.ops.mesh.extrude_region_move(
        MESH_OT_extrude_region={"mirror":False},
        TRANSFORM_OT_translate={
            "value":(0, 0, b),
            "constraint_axis":(False, False, True),
            "constraint_orientation":'NORMAL',
            "mirror":False,
            "proportional":'DISABLED',
            "proportional_edit_falloff":'SMOOTH',
            "proportional_size":1,
            "snap":False,
            "snap_target":'CLOSEST',
            "snap_point":(0, 0, 0),
            "snap_align":False,
            "snap_normal":(0, 0, 0),
            "gpencil_strokes":False,
            "texture_space":False,
            "remove_on_cancel":False,
            "release_confirm":False})
bpy.ops.object.mode_set(mode='OBJECT')

# Save
bpy.ops.wm.save_as_mainfile(filepath='grid.blend')
```

4. (35 points) Soient

- C un point de $\mathbb{R}^3$;
- $\vec{u}, \vec{v} \in \mathbb{R}^3$ deux vecteurs unitaires orthogonaux, c'est-à-dire que $\|\vec{u}\| = \|\vec{v}\| = 1$ et $\vec{u} \cdot \vec{v} = 0$, puis
- $R > 0$ un nombre réel.

Considérez la fonction vectorielle

$$\vec{r}(t) = C + (R\cos t)\vec{u} + (R\sin t)\vec{v}, \quad 0 \leq t < 2\pi.$$

(a) (5 points) En utilisant votre logiciel préféré, dessinez la courbe paramétrée par $\vec{r}(t)$ lorsque $C = (0,0,0)$, $R = 1$, $\vec{u} = (1,0,0)$ et $\vec{v} = (0,1,0)$.

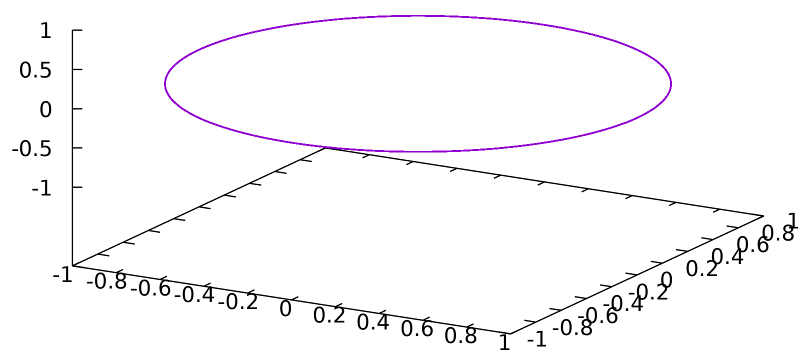
Solution: Dans ce cas, la fonction vectorielle devient

$$\vec{r}(t) = (\cos u, \sin u, 0).$$

En utilisant le script Gnuplot suivant

```
#!/usr/bin/gnuplot
set terminal pdf
set output "cercle-1.pdf"
set parametric
set urange [-4*pi:4*pi]
set samples 1000
set nokey
splot cos(u),sin(u),0
```

on produit un cercle centré en $(0,0,0)$ dans le plan $z = 0$.



(b) (5 points) Même question en prenant $C = (1,1,1)$, $R = 2$, $\vec{u} = (1,1,1)/\sqrt{3}$ et $\vec{v} = (1,-1,0)/\sqrt{2}$.

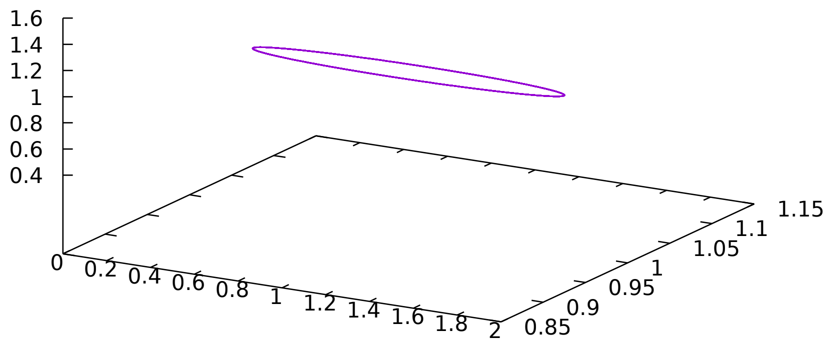
Solution: Dans ce cas, la fonction vectorielle devient

$$\vec{r}(t) = \left(1 + \frac{\cos u}{\sqrt{3}} + \frac{\sin u}{\sqrt{2}}, 1 + \frac{\cos u}{\sqrt{3}} - \frac{\sin u}{\sqrt{2}}, 1 + \frac{\cos u}{\sqrt{3}}\right).$$

Le script Gnuplot correspondant est

```
#!/usr/bin/gnuplot
set terminal pdf
set output "cercle-2.pdf"
set parametric
set urange [-4*pi:4*pi]
set samples 1000
set nokey
splot 1+1*cos(u)/sqrt(3)+1*sin(u)/sqrt(2),1+1*cos(u)/sqrt(3)-1*cos(u)/sqrt(2)
    ,1+1*cos(u)/sqrt(3)
```

qui produit aussi un cercle.



(c) (10 points) Montrez que la courbe décrite par la fonction vectorielle $\vec{r}(t)$ se trouve sur la sphère de centre C de rayon R.

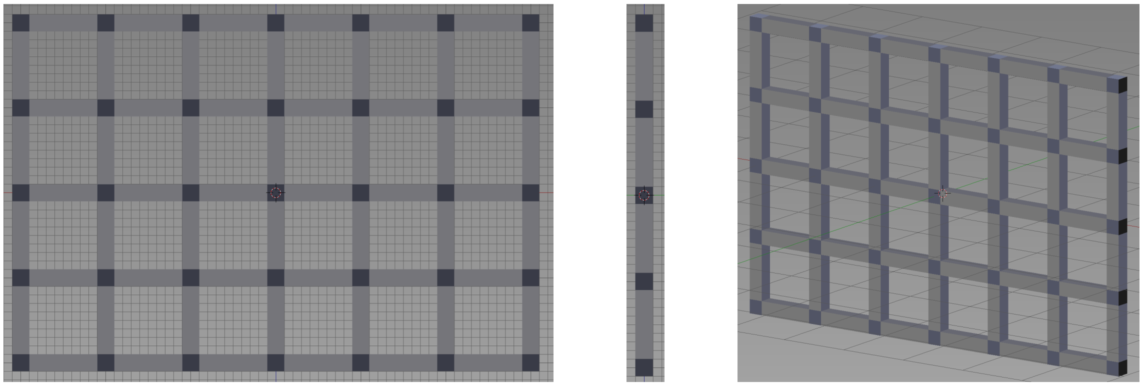
Indice. Il suffit de montrer que $\|\vec{r}(t) - C\|^2 = R^2$ pour tout $t \in \mathbb{R}$.

Solution: Pour tout $t \in \mathbb{R}$, on a

$$\begin{aligned}
\|\vec{r}(t) - C\|^2 &= \|C + (R\cos t)\vec{u} + (R\sin t)\vec{v} - C\|^2 \\
&= \|(R\cos t)\vec{u} + (R\sin t)\vec{v}\|^2 \\
&= [(R\cos t)\vec{u} + (R\sin t)\vec{v}] \cdot [(R\cos t)\vec{u} + (R\sin t)\vec{v}] \\
&= (R^2\cos^2 t)(\vec{u}\cdot\vec{u}) + (R^2\cos t\sin t)(\vec{u}\cdot\vec{v}) + (R^2\sin^2 t)(\vec{v}\cdot\vec{v}) \\
&= R^2\cos^2 t + R^2\sin^2 t \\
&= R^2(\cos^2 t + \sin^2 t) \\
&= R^2,
\end{aligned}$$

tel que voulu.

(d) (10 points) Montrez que la courbe décrite par la fonction vectorielle $\vec{r}(t)$ se trouve dans le plan passant par le point C de vecteur normal $\vec{u} \times \vec{v}$.



(a) (b) (c)

Figure 1: Maillage en forme de grille avec les paramètres $r = 4$, $c = 6$, $b = 0.2$ et $t = 0.8$. (a) Vue de face. (b) Vue de côté. (c) Vue en angle.

Indice. Il suffit de montrer que $(\vec{r}(t) - C) \cdot (\vec{u} \times \vec{v}) = 0$ pour tout $t \in \mathbb{R}$.

Solution: Pour tout $t \in \mathbb{R}$, on a

$$\begin{aligned}
(\vec{r}(t) - C) \cdot (\vec{u} \times \vec{v}) &= (C + (R\cos t)\vec{u} + (R\sin t)\vec{v} - C) \cdot (\vec{u} \times \vec{v}) \\
&= ((R\cos t)\vec{u} + (R\sin t)\vec{v}) \cdot (\vec{u} \times \vec{v}) \\
&= ((R\cos t)\vec{u} \cdot (\vec{u} \times \vec{v})) + ((R\sin t)\vec{v} \cdot (\vec{u} \times \vec{v})) \\
&= 0,
\end{aligned}$$

tel que voulu.

(e) (5 points) En déduire une description textuelle de ce qu'est la courbe décrite par $\vec{r}(t)$ peu importe C, $\vec{u}$, $\vec{v}$ et R.

Solution: Comme la courbe obtenue est continue, se trouve à l'intersection d'une sphère et d'un plan et qu'elle commence et termine au même point ($\vec{r}(0) = \vec{r}(2\pi)$), on en déduit qu'il s'agit d'un cercle, dont le centre est en C, le rayon est r et dont les axes sont donnés par $\vec{u}$ et $\vec{v}$.