Assignment 7

Computer Graphics

Edoardo Riggio (edoardo.riggio@usi.ch)

November 10, 2021

1 Exercise 1

In order to compute the pixels that need to be coloured, we first need to transform the global coordinates into screen coordinates. To do so, we use the following formula:

$$p_{ixs} = p_{ix} \cdot \frac{8}{\tan \frac{3}{8} \cdot \pi}$$

$$p_{iys} = p_{iy} \cdot \frac{8}{\tan \frac{3}{8} \cdot \pi}$$

Where p_{ixs} and p_{iys} are the points in screen coordinates, p_{ix} and p_{iy} are the points in global coordinates, $\frac{3}{8} \cdot \pi$ is the camera opening divided by 2, and 8 is the width of the screen divided by 2 – rounded up to the nearest integer. By using this function, we can finally compute the values of the points.

In the following computations, the results are floored. Moreover, in the case that the z value of the global point is not 1, then p_{ix} and p_{iy} are divided by z. Finally, I assume the point $(0, 0)$ to be at the center of the screen, rather than at the top-left corner.

$$p_{1xs} = \frac{1}{4} \cdot \frac{8}{\tan \frac{3}{8} \cdot \pi} = 0$$

$$p_{1ys} = \frac{1}{4} \cdot \frac{8}{\tan \frac{3}{8} \cdot \pi} = 0$$

$$p_{1s} = (0, 0)$$

$$p_{2xs} = 2 \cdot \frac{8}{\tan \frac{3}{8} \cdot \pi} = 6$$

$$p_{2ys} = 1 \cdot \frac{8}{\tan \frac{3}{8} \cdot \pi} = 3$$

$$p_{2s} = (6, 3)$$

After converting the points, we can use the following pseudocode in order to compute the pixels of the line.

```
x = p1.x
y = p1.y
dx = p2.x - p1.x
dy = p2.y - p1.y
f = -2 * dy + dx

for i in range(0, dx+1):
  setPoints(x,y)
  x += 1
  if f < 0:
    y += 1
    f += 2 * dx
  f -= 2 * dy
```

Which will output the following pixels:

$$(0,0),(1,0),(2,1),(3,1),(4,2),(5,2),(6,3)$$

In the case of the segment between p_1 and $p'2$, we have that $p'2$ has a negative z value. Because the screen has $z = 1$, this means that the segment intersects the screen and goes behind it. For this reason we need to find the intersection between the segment and the plane $z = 1$. By using the calculator, I've found out the intersection to be the point $(2, 1, 1)$. This means that the points to be coloured are the same as the ones in the previous part of the exercise.

2 Exercise 2

The pseudocode used to compute the midpoint algorithm for the circle is the following:

```
function midpoint_circle(r):
  x = 0
  y = -r

  f = 1 - r

  while y < x:
    setPoint(x,y)
    setPoint(-x,y)
    setPoint(x,-y)
    setPoint(-x,-y)
    setPoint(y,x)
    setPoint(-y,x)
    setPoint(y,-x)
    setPoint(-y,-x)
    x += 1

    if f > 0:
      y += 1
      f += 2 * x - 2 * y + 1

    f += 2 * x + 1

    if (|y| < x):
      exit loop
```

Here we use a similar approach as the one used in the midpoint algorithm for the line. Inside of the while loop – which computes only the points of $\frac{1}{8}$ of the circle, and then uses symmetric properties of the circle in order to compute the points in the other $\frac{1}{8}$s of the circle – we check if f is greater, equal or smaller than 0. This is because we can recognize three different cases:

1. f $>$ 0: In this case the midpoint is outside of the circle;
2. f $=$ 0: In this case the midpoint is on the circle;
3. f $<$ 0: In this case the midpoint is inside of the circle.

In this algorithm the decision value (f) is computed by using the formula of the circle:

$$x^2 + y^2 - r^2$$