基礎プログラミングおよび演習 レポート ＃06

2120029 政野玄空

2021年11月14日

1　構想・計画・設計

美しいと思う絵を生成するということで美しいと思うものをまず考えた。結果猫を描くことに決定した。猫の全体像を描くのは非常に難易度が高いので顔面だけを描くことにした。その後猫のサンプル写真として知り合いの飼っているノルウェージャンフォレストキャットの写真を選んだ。美しい。

まず顔面を中心に描くこととしてど真ん中の座標、耳を置く位置、鼻、口、ひげの大まかな座標を決めた。その後実装しながら細かい調整を行い完成した。

2　プログラムコード

```
Pixel = Struct.new(:r, :g, :b)
$img = Array.new(200) do
  Array.new(300) do Pixel.new(255,255,255) end
end
def pset(x, y, r=0, g=0, b=0, a=0.0)
  if x < 0 || x >= 300 || y < 0 || y >= 200 then return end
  $img[y][x].r = ($img[y][x].r * a + r * (1.0 - a)).to_i
  $img[y][x].g = ($img[y][x].g * a + g * (1.0 - a)).to_i
  $img[y][x].b = ($img[y][x].b * a + b * (1.0 - a)).to_i
end
def writeimage(name)
  open(name, "wb") do |f|
    f.puts("P6\n300 200\n255")
    $img.each do |a|
      a.each do |p| f.write(p.to_a.pack("ccc")) end
    end
  end
```

```
end
def fillcircle(x, y, rad, r=0, g=0, b=0, a=0.0)
  j0 = (y-rad).to_i; j1 = (y+rad).to_i
  i0 = (x-rad).to_i; i1 = (x+rad).to_i
  j0.step(j1) do |j|
    i0.step(i1) do |i|
      if (i-x)**2+(j-y)**2<rad**2
        if block_given? then yield(i,j) else pset(i,j,r,g,b,a) end
      end
    end
  end
end
def filldonut(x, y, r1, r2, r=0, g=0, b=0, a=0.0)
  j0 = (y-r1).to_i; j1 = (y+r1).to_i
  i0 = (x-r1).to_i; i1 = (x+r1).to_i
  j0.step(j1) do |j|
    i0.step(i1) do |i|
      d2 = (i-x)**2+(j-y)**2
      if r2**2 <= d2 && d2 <= r1**2
        if block_given? then yield(i,j) else pset(i,j,r,g,b,a) end
      end
    end
  end
end
def fillrect(x, y, w, h, r=0, g=0, b=0, a=0.0)
  j0 = (y-0.5*h).to_i; j1 = (y+0.5*h).to_i
  i0 = (x-0.5*w).to_i; i1 = (x+0.5*w).to_i
  j0.step(j1) do |j|
    i0.step(i1) do |i|
      if block_given? then yield(i,j) else pset(i,j,r,g,b,a) end
    end
  end
end
def fillellipse(x, y, rx, ry, r=0, g=0, b=0, a=0.0)
  j0 = (y-ry).to_i; j1 = (y+ry).to_i
  i0 = (x-rx).to_i; i1 = (x+rx).to_i
  j0.step(j1) do |j|
    i0.step(i1) do |i|
      if ((i-x).to_f/rx)**2 + ((j-y).to_f/ry)**2 < 1.0
```

```
        if block_given? then yield(i,j) else pset(i,j,r,g,b,a) end
      end
    end
  end
end
def fillrotellipse(x, y, rx, ry, theta, r=0, g=0, b=0, a=0.0)
  d = (if rx > ry then rx else ry end)
  j0 = (y-d).to_i; j1 = (y+d).to_i
  i0 = (x-d).to_i; i1 = (x+d).to_i
  j0.step(j1) do |j|
    i0.step(i1) do |i|
      dx = i - x; dy = j - y;
      px = dx*Math.cos(theta) - dy*Math.sin(theta)
      py = dx*Math.sin(theta) + dy*Math.cos(theta)
      if (px/rx)**2 + (py/ry)**2 < 1.0
        if block_given? then yield(i,j) else pset(i,j,r,g,b,a) end
      end
    end
  end
end
def filltriangle(x0, y0, x1, y1, x2, y2, r = 0, g = 0, b = 0, a = 0.0)
  if block_given?
    fillconvex([x0, x1, x2, x0], [y0, y1, y2, y0]) do |x,y| yield(x,y) end
    fillconvex([x0, x2, x1, x0], [y0, y2, y1, y0]) do |x,y| yield(x,y) end
  else
    fillconvex([x0, x1, x2, x0], [y0, y1, y2, y0], r, g, b, a)
    fillconvex([x0, x2, x1, x0], [y0, y2, y1, y0], r, g, b, a)
  end
end
def fillconvex(ax, ay, r=0, g=0, b=0, a=0.0)
  xmax = ax.max.to_i; xmin = ax.min.to_i
  ymax = ay.max.to_i; ymin = ay.min.to_i
  ymin.step(ymax) do |j|
    xmin.step(xmax) do |i|
     if isinside(i, j, ax, ay)
       if block_given? then yield(i,j) else pset(i,j,r,g,b,a) end
     end
    end
  end
```

```
end
def isinside(x, y, ax, ay)
  (ax.length-1).times do |i|
    if oprod(ax[i+1]-ax[i],ay[i+1]-ay[i],x-ax[i],y-ay[i])<0
      return false
    end
  end
  return true
end
def oprod(a, b, c, d)
  return a*d - b*c;
end
def fillline(x0, y0, x1, y1, w, r=0, g=0, b=0, a=0.0)
  dx = y1-y0; dy = x0-x1; n = 0.5*w / Math.sqrt(dx**2 + dy**2)
  dx = dx * n; dy = dy * n
  if block_given?
    fillconvex([x0-dx, x0+dx, x1+dx, x1-dx, x0-dx],
               [y0-dy, y0+dy, y1+dy, y1-dy, y0-dy]) do |x,y| yield(x,y) end
  else
    fillconvex([x0-dx, x0+dx, x1+dx, x1-dx, x0-dx],
               [y0-dy, y0+dy, y1+dy, y1-dy, y0-dy], r, g, b, a)
  end
end

fillellipse(150,100,132,82)
fillellipse(150,100,130,80,255,255,255)
fillellipse(150,60,100,40,245,222,179)
filltriangle(79, 3, 59, 61, 122, 61)
filltriangle(214, 3, 189, 61, 242, 61)
filltriangle(80, 5, 60, 60, 120, 60,255,182,193)
filltriangle(215, 5, 190, 60, 240, 60,255,182,193)
fillrotellipse(90, 100, 10, 20, -30)
fillrotellipse(210, 100, 10, 20, 30)
filltriangle(130, 120, 170, 120, 150, 150,255,182,193)
filltriangle(150, 150, 110, 160, 180, 160,255,182,193)
fillline(10, 120, 100, 145, 1)
fillline(10, 150, 100, 150, 1)
fillline(10, 180, 100, 155, 1)
fillline(200, 145, 290, 120, 1)
```

```
fillline(200, 150, 290, 150, 1)
fillline(200, 155, 290, 180, 1)
writeimage("t.ppm")
```

3　プログラムの説明

サンプルからメソッドをコピーして持ってくる。その後まず顔の輪郭を描写する。その後顔の全体を描写する。次に耳となる三角を生成する。輪郭も同時にコピペして少し外側にでるように調整する。次に目玉を 2 つ描画する。330 度で調整すると思ったとおりにならず、マイナスを利用したらいい感じになった。次に鼻を描画する。次に口を描画する。最後にひげを 6 本薄い長方形を利用し描画する。いままでの命令を t.ppm に書き込む。(プログラムのどの部分が何をしているかを説明する)

4　生成された絵

猫の顔面の絵。参考にした絵の美しさとは程遠いが猫に見えるので間違いなく美しいと言える。

5　考察

最初に大まかな設計をし、色は何でもいいのでわかるようにわけて画像を生成しながら微調整を重ねていったのが功を奏して短い時間で完成できたと考えられ

る。ただ先に同等のドット絵などを準備していればもっと早く完成できた可能性がある。

6　アンケート

6.1　Q1：画像が自由に生成できるようになりましたか。

できた。

6.2　Q2：画像をうまく生成する「コツ」は何だと思いましたか。

先にドット絵を生成するツールなどをつかったりして写真を取り込んでおけば座標がかんたんに取れると思った。

6.3　Q3：リフレクション (今回の課題で分かったこと) ・感想・要望をどうぞ。

猫の美しさを表現するのはとても困難だった。