AN ESSAY ABOUT THE ELLIPSE
v 0.1.5

PANAGIOTIS TSIBERIS

March 2015

Abstract

This is an essay about the ellipse and the calculation of the elliptical arc, by considering the fact that an ellipse is always a circle projection. We will use Python and C for the calculations and AutoCAD for the drawings. I am not making any claims here about finding something new, as I saw many similar approaches on the net when I was writing that essay.

1. Introduction

As it is known, an ellipse may be regarded as the projection, onto a vertical plane, of a circle that has revolved around an imaginary horizontal axis passing through its center.

Let us imagine a circular ring disposed vertically in space. At two points of the perimeter of the ring, begin two horizontal axes. By holding both axes, we rotate the circular ring as below:

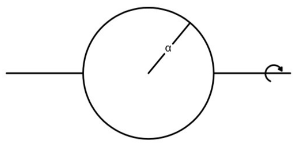

Figure 1: A rotating circle

The ellipse is then the horizontal projection of the circular ring on a vertical plane:

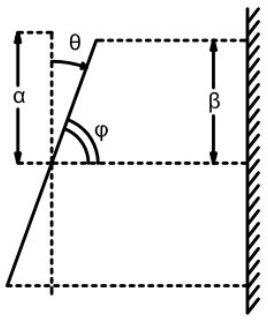

Figure 2: Side view of a projected rotating circle

Where θ (theta) is the rotation angle and β (beta) is the semi-minor axis of the ellipse. The semi-major axis is α (alpha) which is the circle's radius also. Last, φ equals to (90 - θ).

2. Drawing the ellipse

First we will draw the unit circle, approximately using AutoCAD, with the help of the Python programming language. For simplicity reasons we will focus on the first quadrant of the circle – and later of the ellipse. The following python code calculates, for angles from 0 to 90 degrees, the corresponding sine and cosine:

```
from math import sin, cos, radians
for x in range(91):
      a = sin(radians(x))
      b = cos(radians(x))
      print str(b)+","+str(a)
```

Listing 1: First python code to compute sin and cosine of a unit circle

The partial output of the above program follows:

```
1.0,0.0
0.999847695156,0.0174524064373
0.999390827019,0.0348994967025
0.998629534755,0.0523359562429
0.99756405026,0.0697564737441
...
0.0697564737441,0.99756405026
0.0523359562429,0.998629534755
0.0348994967025,0.999390827019
0.0174524064373,0.999847695156
6.12323399574e-17,1.0
```

Listing 2: Partial output of the first python code

It is clear that the last cosine is zero and not 6.12323399574e-17. This happens due to binary floating point representation. That said, the “Decimal” python module looks promising.

If we now use AutoCAD's polyline command and copy/paste the whole above python output, the result would be the first quadrant of (an approximation of) the unit circle:

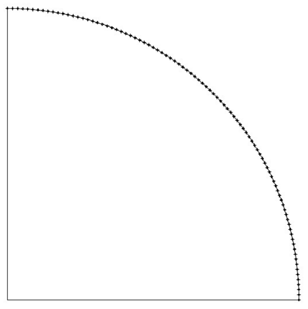

Figure 3: First quadrant of the unit circle

If the rotating angle of the circular ring mentioned earlier is theta (θ), then to draw an ellipse in a similar way of that of the unit circle, we have to multiply every cosine of **Listing 2** with the cosine of (90 – θ). So a python program that computes the “coordinates” of a “unit ellipse” with a rotation angle, say 55 degrees, would be:

```
from math import sin, cos, radians
theta = 55
c = sin(radians(90-theta))
for x in range(91):
     a = sin(radians(x)) * c
     b = cos(radians(x))
     print str(b)+","+str(a)
```

Listing 3: Second python code to compute coordinates of a “unit ellipse”

the partial output would be:

```
1.0,0.0
0.999847695156,0.0168577301087
0.999390827019,0.0337103251894
0.998629534755,0.0505526517786
0.99756405026,0.0673795795403
...
0.0697564737441,0.963572879523
0.0523359562429,0.964602058514
0.0348994967025,0.965337410374
0.0174524064373,0.965778711107
6.12323399574e-17,0.965925826289
```

Listing 4: Partial output of the second python code

and the AutoCAD output would be:

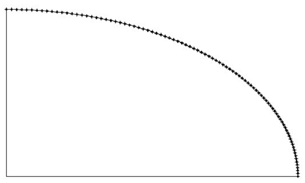

Figure 4: First quadrant of the “unit ellipse”

3. Ellipse equations

The parametric equations of the first quadrant of an ellipse (in a general form – not the unit one) should be:

$$x=\alpha*\cos(n) \quad (1)$$

$$y=\alpha*\sin(n)*\sin(\varphi) \quad (2)$$

where n ∈ [0,90], φ equals to 90-θ and α is the circle's radius.

From the above equations we get:

$$(1)\Rightarrow n=\arccos\left(\frac{x}{\alpha}\right) \quad (3)$$

$$(2)\wedge(3)\Rightarrow y=\alpha*\sin\left(\arccos\left(\frac{x}{\alpha}\right)\right)*\sin(\varphi) \quad (4)$$

Also, from **Figure 2** we get:

$$\beta=\alpha*\sin(\varphi)\Rightarrow\sin(\varphi)=\frac{\beta}{\alpha} \quad (5)$$

thus we get:

$$(4)\wedge(5)\Rightarrow y=\sin\left(\arccos\left(\frac{x}{\alpha}\right)\right)*\beta \quad (6)$$

for x ∈ [0,1] (first quadrant) with α ≠ 0.

Finally, if we're talking about the “unit ellipse” (a=1) we get:

$$y=\sin(\arccos(x))*\beta \quad (7)$$

4. Perimeter of the ellipse approximation

Given the parametric equations (1) & (2) mentioned earlier, we will figure out the perimeter (C) of an ellipse (an approximation that is) with the Pythagorean theorem. So if n ∈ [1,90] (first quadrant), α equals to 1 (unit circle) and theta (θ) is the rotation angle, we get:

$$C=4*\sum_{n=1}^{90}\sqrt{\left[\sin(90-\theta)*(\sin(n)-\sin(n-1))\right]^2+(\cos(n-1)-\cos(n))^2} \quad (1)$$

A python program to compute the above quantity follows:

```
from math import sin, cos, radians, hypot
theta = 60
c = sin(radians(90-theta))
cir = 0
for n in range(1,91):
      y = c * (sin(radians(n))-sin(radians(n-1)))
      x = cos(radians(n-1))-cos(radians(n))
      z = hypot(x,y)
      cir += z
print cir * 4
```

Listing 5: Third python code to compute the perimeter of an ellipse

the output of the above program is `4.84416262571`.

If we want more accuracy then we should increment the angle (n) not by 1 at a time but by something smaller. Here's where the C language comes forth; because it's faster. The following program computes the perimeter of the previous ellipse more accurately (the angle increments by 0,001):

```
#include <stdio.h>
#include <math.h>
main()
{
      long double i, n, m, x, y, r, z, p, b;
      i = 0.001;
      n = 0.001;
      m = 90.0;
      r = 2 * M_PI / 360;
      p = 0.0;
      short int a = 30;
      b = sin(a*r);
      while (n <= m)
      {
            x = cos((n-i)*r) - cos(n*r);
            y = b * (sin(n*r) - sin((n-i)*r));
            z = sqrt(pow(x,2) + pow(y,2));
            p += z;
            n += i;
```

```
        }
        p *= 4;
        printf("%1.16Lf\n",p);
}
```

Listing 6: First C code to compute the perimeter of an ellipse

the output of the above program is `4.8441542970422806.`

5. Calculation of the elliptical arc

To calculate the arc for a given angle of a (unit) ellipse we need to find the corresponding angle of the unit circle:

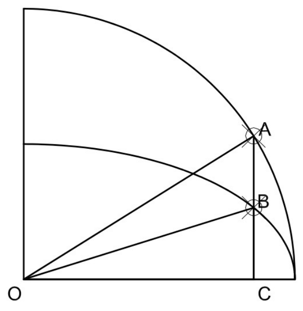

Figure 5: The given ∠BOC angle of the ellipse and the corresponding ∠AOC circle angle

For the above triangle BOC, the law of sines gives:

$$\frac{OC}{\sin(\angle OBC)}=\frac{BC}{\sin(\angle BOC)}=\frac{OB}{\sin(\angle BCO)} \quad (1)$$

Also, for the above triangle AOC, the law of sines gives:

$$\frac{OC}{\sin(\angle OAC)}=\frac{AC}{\sin(\angle AOC)}=\frac{OA}{\sin(\angle ACO)} \quad (2)$$

From the triangles AOC and BOC we have:

$$\angle OAC=90-\angle AOC \quad (3)$$

$$\angle OBC=90-\angle BOC \quad (4)$$

Last, we know that:

$$BC=AC*\sin(90-\theta) \quad (5)$$

where θ is the rotation angle.

Thus we have:

$$(1)\wedge(4)\wedge(5)\Rightarrow OC=\frac{\sin(90-\angle BOC)}{\sin(\angle BOC)}*AC*\sin(90-\theta)\quad(6)$$

$$(2)\wedge(3)\Rightarrow OC=\frac{\sin(90-\angle AOC)}{\sin(\angle AOC)}*AC\quad(7)$$

$$(6)\wedge(7)\Rightarrow tan(\angle AOC)=\frac{\tan(\angle BOC)}{\sin(90\text{-}\theta)}\quad\Rightarrow$$

$$\Rightarrow\angle AOC=\arctan\left(\frac{\tan(\angle BOC)}{\sin(90\text{-}\theta)}\right)\quad(8)$$

where θ can't be 90 degrees.

So every time we want to calculate the arc for a given elliptical angle, we need to find the corresponding circle angle first and then use equation (1) of **chapter 4** or any of the code mentioned there. That's because there is no direct equation to find an elliptical arc – or a perimeter.

For instance, let's assume that the given elliptical angle (∠BOC) is 35 degrees and the rotation angle of the ellipse is 60 degrees. The following python routine computes the corresponding circle angle:

```
from math import sin, tan, atan, radians, degrees
ellipse_angle = 35
rotation_angle = 60
b = tan(radians(ellipse_angle))
c = sin(radians(90-rotation_angle))
a = degrees(atan(b/c))
print a
```

Listing 7: Fourth python code to compute the corresponding circle angle.

the output of the above program is `54.4703551333`.

Now we can use the C code of **chapter 4** to compute the elliptical arc. Instead of `m = 90.0;` we write `m = 54.4704;` which is a reasonable approach. Also, the variables `i` & `n` become `0.0001`.

And the output would be `2.4789205506457636`.

Finally, to compute the elliptical arc between two angles, we follow the above procedure twice and subtract the two quantities.

Software used
Python 2.7.3
GCC 4.6.3
Geany 0.2.1
Code::Blocks 10.05
LibreOffice 3.5.7.2
Ubuntu 12.04 LTS
&
AutoCAD 2007
MS Windows XP

Panagiotis Tsiberis
Forestry & Environmental Scientist
Forest Maps Department
Forestry Commission of Samos
Greece

e-mail: tsiberis@gmail.com
webpage: http://takira.freehosting.net/

Changelog
v 0.1.1 & 1.2: Fixed syntactical errors
v 0.1.3: Changed `double` & `int` to `long double` & `short int` in the C program.
v 0.1.4: Corrected the last 3 lines of chapter 5 – those referring to the C code output.
v 0.1.5: Added that in equation (8) of chapter 5, the θ angle can't be 90 degrees.