

## Physics 420/580: Lab 1

Thursday, September 15, 2011

1. Log in using your University of Alberta CCID and password. After a brief startup sequence, the machine should present you with graphical Linux desktop.
2. Open a terminal window by selecting with the mouse the **Applications** ▷ **Accessories** ▷ **Terminal** menu item from the top-left corner of the screen. See what's in your home directory, and check that your default shell is set to BASH.

```
$ ls -F
Desktop/  public_html/
$ mkdir phys420_580
$ ls -F
Desktop/  phys420_580/  public_html/
$ cd phys420_580
$ env | grep SHELL
SHELL=/bin/bash
```

3. At the moment, you have local privileges but cannot access the internet. In order to authenticate your machine on the network, open another terminal and type the following. (You'll be asked for your password one more time.)

```
$ ssh -l CCID 10.0.0.1
```

4. If you're not already expert with Linux, BASH, and C++, try working through some of the examples and exercises from the tutorials posted on the class website. (Go to the class website

[http://www.ualberta.ca/~kbeach/phys420\\_580.html](http://www.ualberta.ca/~kbeach/phys420_580.html)

and follow the links from the first two items listed under **Learning Resources**.) If there is *anything* you do not understand, please ask questions of the TA. He's there to help.

5. Download the Lab 1 instructions and source code from the class website. You can either do this from a web browser or from the terminal using the `curl` command.

```
$ WEBPATH=http://www.ualberta.ca/~kbeach/phys420_580
$ curl $WEBPATH/docs/Lab1.pdf -O
$ curl $WEBPATH/src/Lab1.tar.gz -O
$ tar xzf Lab1.tar.gz
$ cd Lab1
```

The pdf instructions (this document) can be viewed onscreen using `evince` (or `acroread` or `xpdf`). The file ending in `.tar.gz` is an archived and compressed directory of files (sometimes called a *tarball*).

6. A *Lissajous figure* refers to a planar trajectory that is harmonic in two orthogonal directions. This is something you might have seen traced out on an oscilloscope. You can find more details here:

[http://en.wikipedia.org/wiki/Lissajous\\_curve](http://en.wikipedia.org/wiki/Lissajous_curve)

Write a C++ program that computes the quantities

$$x(t) = A \cos(at + \delta)$$

$$y(t) = B \cos(bt)$$

at  $100N$  equally-spaced points in the range  $0 < t < 2\pi N \times \max(1/a, 1/b)$  and outputs the results in three-column format  $t, x(t), y(t)$  to the standard output stream. Have your program require six command

line arguments: the first five interpreted as floating-point numbers (with the `atof` function, say) and used to set the values of  $A, B, a, b, \delta$ ; the sixth as an integer (with `atoi`) and assigned to  $N$ . The program output can then be written to a file via redirection (`>`) and viewed with `gnuplot`.

```
$ make lissajous
$ ./lissajous 2.6 1 3 2 0.5 2 > curve1.dat
$ ./lissajous 1 1 1.1 1.2 0 35 > curve2.dat
$ gnuplot
> plot "curve1.dat" using 2:3 with lines
> plot "curve2.dat" using 1:($2+$3) w l, 2*cos(0.05*x), -2*cos(0.05*x)
> quit
```

- Convince yourself that a Lissajous figure is closed iff  $a/b$  is a rational number.
- How does the ratio  $a/b$  control the shape of the curve?
- In the case  $a = b$ , how does the phase shift  $\delta$  effect the curve?
- Investigate the beats produced when the two sinusoidal components—with equal amplitudes and slightly different frequencies—are superimposed. In other words, plot  $z(t) = x(t) + y(t)$  versus  $t$  for  $A = B$  and  $|a - b| \ll 1$ . The result is a product of a slowly varying envelope function and a rapidly varying beat function:

$$\cos(\alpha t) + \cos(\beta t) = 2 \cos\left[\frac{1}{2}(\alpha + \beta)t\right] \cos\left[\frac{1}{2}(\alpha - \beta)t\right]$$

7. The Mandelbrot set consists of the bounded orbits of the complex-valued recurrence relation

$$z_{n+1} = z_n^2 + c, \quad z_0 = c \equiv x + iy$$

The set is typically visualized as a plot in the  $x$ - $y$  plane, with each point corresponding to an unbounded orbit coloured according to its rate of escape. For more information, take a look at this article:

[http://en.wikipedia.org/wiki/Mandelbrot\\_set](http://en.wikipedia.org/wiki/Mandelbrot_set)

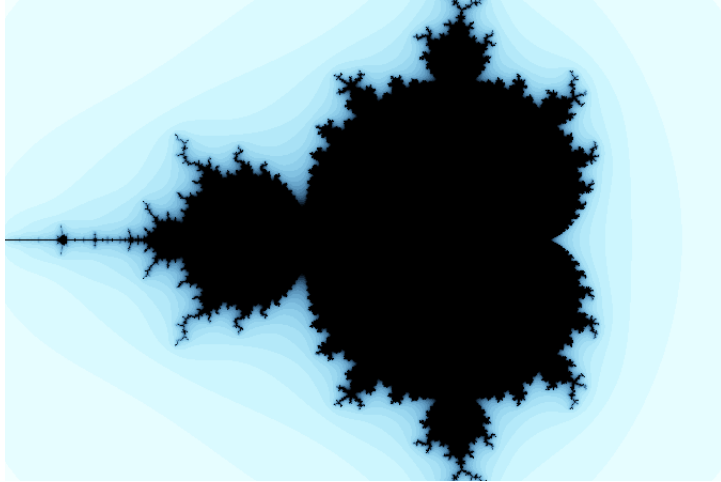
Write a C++ program that implements the following algorithm. Scan over a grid of  $c$  values such that its real and imaginary parts range over  $x \in [-2, 1]$  and  $y \in [-1, 1]$ . At each point, run the recurrence relation until  $|z_n| > R$  or  $n > N$ . I suggest the values  $R = 3$  and  $N = 500$ .

Output the escape counts  $n$  as a rectangular table of values to `stdout`, and then plot the Mandelbrot set using `gnuplot`:

```
$ make mandelbrot
$ ./mandelbrot > mandelbrot.dat
$ gnuplot
gnuplot> set pm3d map
gnuplot> splot "mandelbrot.dat" matrix
```

8. Take a look at the file `mandelbrot-png.cpp`, which includes sample code for constructing RGB bitmaps in the `png` format. Modify it so that it draws a Mandelbrot set into the file `out.png`.

```
$ make mandelbrot-png
$ ./mandelbrot-png
$ display out.png
$ convert out.png out.pdf
$ evince out.pdf
```



9. Before you leave, make sure to close out of your session from the **System** ▸ **Logout** menu.