# PHYS405
# Advanced Computational Physics
# Parallel Computing

## Assignment # 8 (and LAST)
## Due: Friday, December 10, 2010

*Purpose:* Learn the CUDA language.
*Note:* Please identify all your work.

Scattering experiments consist in sending in particles from an asymptotic region (where the potential is zero, or a non-zero constant) towards a scattering force field and observing how the particles get deflected. In PHYS305, we solved for the classical trajectory of a particle incoming onto a three hills potential and escaping back to infinity. The code *scattering.c*, linked below, solves this problem. The scattering angle (angle the trajectory makes with the x-axis) was plotted as a function of the impact parameter (y-coordinate for large negative x-distance). This system was found to be chaotic, i.e., the scattering angle was extremely sensitive to the initial impact parameter.

The scattering angle also depends sensitively on the initial energy of the projectile. This was illustrated in PHYS305 by producing a 2-D color image of the scattering angle. The program *survey_scattering.c* (linked below) calculates the scattering angle on a grid of initial impact parameter values and projectile energies. When piped into *pixelplot.py* it produces a color image (scattering angle) versus y and E.

*Syntax:*

*gcc survey_scattering.c -lm -o survey_scattering*

*./survey_scattering 120 120 ¿ angle*

*cat angle — python pixelplot.py 0.0 0.7 0.005 0.07 120 120*

Your task is to **parallelized** the code *survey_scattering.c* for the **GPU**.

1. adapt the code to run on the GPU.
2. instrument the sequential code and the CPU code for timing ( *gettimeofday()* ).
3. compare the wall times using the sequential and GPU codes to produce the data to plot images of increasing size in pixels, 64 x 64, 128 x 128, 512 x 512, 1024 x 1024
4. comment on the effect of using *float* (single precision)

---

*Hints*

- You may want to first modify survey_scattering.c into a modified sequential code since we know that
  - CUDA runs fast in single precision (question above), requiring to transform all *double* into *float* and specifying the constants accordingly
  - For simplicity survey_scattering.c defaults all parameters using globally defined variables - however CUDA does not allow the use of such variables - you have to modify the code
- You could tile the image in CUDA in terms of lines of equal energies (1-D blocks) or 2-D blocks
- Check your CUDA code with an image size 120x120 (or at least not a multiple of the number of CUDA threads in the CUDA blocks)

---