[10.05.2011]

* Optimization vs. performance
* What to do when optimizing C code
  + Try using less local variables, and use instead global variables (due to stack management stuff)
  + Function calling
    - It takes a lot of time
    - Functions take time to push all the registers to the stack
    - Inline functions
      * Void whichever (){  
        stuff  
        }
      * Inline void whichever (){  
        stuff  
        }
        + The compiler actually replaces the function call with the code of the function
  + Loop optimization
    - Move constants outside the loop
    - Replace division with replacing it with multiplying by the inverse
    - When running for loops, instead of comparing every time, copy and paste the code to run multiple times before going back to the start of loop
      * For (i=0, i<100; i++){printf(“blah”);}  
        for (i=0;i<100; i+=5){printf(… 5 times}
  + Use look up tables, or the complete opposite
  + Accessing stuff in sequential order (from arrays) increases read times due to caching
  + Try to avoid testing (condition statements)
* Looked at the optimization of vanilla\_rescale and stuff
  + Valgrind –tool=callfring (a memory profiler)
    - A useful tool to figure what is slowing down a program (works on any executable)
    - Calloc takes time initializing everything to zero, use malloc