## Source code and Linux binaries for virtual creatures

Here they are. Note: this material, including the source code, is (c) The University of Birmingham (and quite possibly the University of Portsmouth). It is made available for academic and educational purposes only.

## How to run the binaries

The binaries are Linux executables that should run on any x86 machine with Linux (and OpenGL) installed. *However,* if you have a 64 bits machine, you won't be able to use the creature data files provided, so you'll have to run the programs to evolve your own. The main programs are:

- `ssga`: Runs a two-population genetic algorithm to evolve fighting creatures. Saves the best creature of each population at every generation. Also creates files "best0.dat" and "best1.dat" that contain the current best creatures of each population. Note: despite the name, the algorithm is not an SSGA, but simply Sim's algorithm extended with a "sliding archive". Note that this program is expected to be run for days or weeks on ened, though early results emerge in a matter of hours (especially if you reduce evaluation time). Please note that by default, the program will make use of box-targeted sensors, which is obviously a waste as no box is used in this sensor. See the README file for how to address this (basically, you just need to comment/uncomment a small section).
- `readbest`: When run without arguments, retrieve creatures from files "best0.dat" and "best1.dat" in the current directory and displays a competition between them. This program allows you to visualise the current results of evolution. You can also specify other creature data files as arguments (i.e. `readbest file1.dat file2.dat`). **NOTE: If readbest dies immediately with messages like "ODE Message 2: mass must be > 0 in dMassCheck()" or other, this simply menas that the provided data files are not compatible with your ODE build / hardware configuration.** To overcome this, simply re-compile ssga (see below) and run ssga until it has generated both a best0.dat and a best1.dat file. Then re-compile readbest and run it. It should work.
- `grab` and `readgrab`: similar to `ssga` and `readbest`, but for the box-grabbing task rather than fighting creatures. Note: the `grab` program implements a 2-strikes-out algorithm, with a window of 10 previous results (i.e. any results obtained before the 10 latest confrontations are forgiven).
- `sphere`: Runs an Evosphere simulation - a population of free-living creatures interacting, reproducing and (hopefully) evolving on a spherical mini-planet. At regular interval, this program store some statistics for the current interval in "stats.txt". Several individuals are also saved to disk: the "best" of each species (and of the entire simulation) according to various metrics (total number of kids, lifetime, etc.). In addition, the two creatures with the highest number of kids in the simulation are stored in files "best0.sphere.dat" and "best1.sphere.dat". The entire population is also stored in a separate file ("pop.sphere.dat").
- `readspheric`: An equivalent to `readbest` for Evosphere. Without arguments, retrieve creatures from files "best0.sphere.dat" and "best1.sphere.dat" in the current directory and runs a competition between them (by dropping them close to each other on a sphere).
- `visual`: Retrieves the population stored in the "pop.sphere.dat" file in the current directory, and displays it visually. Note: run it as "`./visual -noground -noshadows`" to avoid having the ground plane dividing the sphere in half.

Basically, when unzipping the file, just type "`./readbest`" for instant gratification. **NOTE: If readbest dies immediately with messages like "ODE Message 2: mass must be > 0 in dMassCheck()" or other, this simply menas that the provided data files are not compatible with your ODE build / hardware configuration.** To overcome this, simply re-compile and run ssga until it has generated both a best0.dat and a best1.dat file.

## How to compile from source

If you want to compile from source, you must follow this procedure:

1. Download the source code for the ODE physics simulation package (you may want to compile it once, just to check that it runs smoothly).

2. *Optional, but STRONGLY recommended:* Copy this file (drawstuff.cpp, also included in the zip archive) into the $ODE/drawstuff/src/ directory, replacing the original file of the same name that comes with ODE. If you don't do that, then you will always need to have the texture files (.ppm files included in the zip archive and in the ODE distribution) in the current directory, or wherever the "fn.path_to_textures" variable (set in `animat.h`) points to, even if you don't use textures. Also, when running the Evosphere visual, the sphere will be divided in two by a giant plane (though this plane is transparent when looked at from below); if you use this modified file, calling your visualisation program with arguments "-noground -noshadows" will get rid of that plane.

3. Compile ODE. (./configure, make, etc.) NOTE: ODE v0.5 gives faster, smaller executable than ODE 0.9 (smaller by a factor of 10!) Apparently there's a lot of unneeded ovrhead there.

4. Compile the `ssga` program, by using (something like) the following command:

   ```
   gcc -Wall -fno-rtti -fno-exceptions -O1 -I ~/ode-0.8/include/
   ./ssga.cpp ~/ode-0.8/ode/src/libode.a ~/ode-
   0.8/drawstuff/src/libdrawstuff.a -L/usr/X11R6/lib -L/usr/X11/lib -
   L/usr/lib/X11R6 -L/usr/lib/X11 -lX11 -lGL -lGLU -lm -o ssga
   ```

   Of course, you will need to change the "~/ode-0.5" bit to whichever directory you installed ODE in. Same thing for X11, OpenGL, Glut, etc. Note that the gcc options are not necessary, those are just the ones I use. Apply the same command to compile the `readbest` program, just changing "ssga" with "readbest" in the entire string.

5. Run `"./readbest"` to ensure that everything runs smoothly. **Read the comments above (in bold) regarding readbest if it seems to crash instantly.**

6. If you want to modify the code, be sure to have a look at the `README` file!

7. Enjoy!