User's documentation

for the “robotflocksim” software

Contents

Preface

The main goal of this software is to present a coarse-grained framework for simulating autonomous drones through the numerical solution of a set of second order differential equations. The main features of this dynamical system can be found in [1].

Installation

After the user has downloaded the software from the repository, the necessary libraries should be installed. Any user with root entitlements can do this by running ./tools/install.sh from the main folder of the software:

```
./tools/install.sh
```

The pogram only works under Linux. Basically, the installer script mentioned avbove installs the following packages:

1. OpenGL, GLUT (freeglut3-dev package) for visualization
2. Python 2.x (python package), it is useful for statistical calculations
3. ILUT (libdevil-dev package) for obtaining a toolkit which is capable for saving images from OpenGl.

For testing a specific model, one should compile it, the process of the compilation depends on which algorithm does the user would like to test. For example, there is a basic model exists in the src/ directory called “algo_spring” (this is a very simple algorithm, the agents are connected with spring-like pairwise forces). With the following command, this algorithm can be compiled:

```
make spp_evol
```

In the Makefile, the list of existing algorithms can be checked (there is a label in the Makefile for each algorithm). After the compilation is ready, one can run the program by typing the following command:

```
./robotflocksim_main
```

If everything works, a visualization window should appear.

If one writes a “-novis” flag after this command, then the software starts without visualization and some statistical parameters will be saved into a specific directory (details will be discussed later).

```
./robotflocksim_main -novis
```

Input files and parameters

There are 5 different types of basic input files, which files can be loaded with specific flags. This files are responsible for the following features:

1. Parameters of flocking algorithm (default is “parameters/flockingparams.dat”) - some of these parameters can be changed during visualization. These parameters are the fundamentals of the functions of any algorithm, e.g. in the case of spring-like force terms, the spring constant and the equilibrium distance should be in this file.
2. Parameters of the flying robot (default is “parameters/unitparams.dat”) - these parameters can be changed during visualization. This file contains the value of time delay, noise levels, parameters of inertial effects, etc. For details about the model of a robot, see [1].
3. Initialization parameters (default is “parameters/initparams.dat”) - e.g. number of units, length of simulation, range of “collision zone”, et cetera.
4. Color setups (default is “config/default_colors.ini”) .
5. Setup file for details of output mode (default is “config/output_defaul.ini”) - details will be discussed later.

If the user starts the program simply with the command mentioned in the previous section, the default files listed above will be loaded. But of course one can load other files by using the following flags:

```
 ./robotflocksim_main -f flockingparams_1.dat -u unitparams_1.dat -c config/csabi_colors.ini -i initparams_2.dat
```

Flag	File
-f	Parameters of the flocking algorithm
-u	Parameters of the model of the robot
-i	Initialization parameters
-c	Colors

Output files

If the program has been started with “-novis” flag, it generates the following output files by default:

Name of output file	**Content of output file**
posandvel.dat	Time series of positions and velocities of every agents

innerstates.dat	Time series of inner states of every agents
CoM.dat	Centre of mass (time series)
dist_between_neighbours.dat	Time series of distance between neighbouring agents
dist_between_units.dat	Time series of average, standard deviation, minimum and maximum of distance between the robots
velocity.dat	Time series of statistics about velocities.
correlation.dat	Velocity correlation timeline

By default, these files will be present in a directory called output_default, but one can change this directory by using the -o flag:

```
./robotflocksim_main -o output_spp
```

In this case, the output files will be in the output_spp directory.

If the user starts the program with visualization (in other words: without the “novis” flag), then the output directory will contain two subdirectories: “frames” and “screenshots”. If the image output is enabled, screenshots and video frames can be saved into these directories. One can enable image output mode by adding a flag (pngout=true) to the make command (!) :

```
make spring pngout=true
```

With the “-outputconf” flag, one can select a configuration file, where the output modes are listed.

```
./robotflocksim_main -outputconf config/config.ini
```

In this case, the structure of the config.ini file should be similar with this, if the user would like to save only the trajectories and the distance between neighboring units:

```
# Inner states and trajectories
SaveTrajectories=true
SaveInnerStates=false
# Order parameters (general)
SaveDistanceBetweenUnits=false
SaveDistanceBetweenNeighbours=timeline
SaveVelocityStatistics=timeline
SaveCorrelation=stat
SaveCoM=stat
```

In the case of the first two items (trajectories and inner states) the user can select between

saving the timeline (true) or not (false). The other order parameters have 4 output options: *false*, *timeline, stat* and *steadystat*. The “timeline” option will save the complete timeline, while with the “stat” and “steadystat” option, the user can save time-averages instead of full timelines.

The “steadystat” option is capable of saving a time-average with the exclusion of the first part of the measurement. “First part” is defined via a timestamp which can be set up in the initparams file (e.g. StartOfSteadyState=30 will set up this to 30 sec.)

User interactions during visualization

In the following sheet, the enabled mouse- and keyboard functions are listed:

Event (Mouse of Keyboard)	Feature
F3	Switching between 2D and 3D visualization
F5	Saving parameters (both the flocking algo's params and the robot models' params will be saved)
F7	Displaying communication graph (agents which are closer to each other than the communication range will be connected with an edge)
F11	Screenshot, if it is enabled (one should enable it during compilation).
ALT + F11	Saving video frames (video recording can be stopped by pressing F11 again)
Insert	Displaying GPS fluctuations around the agents, If the visualization mode is 2D (“GPS Ghosts”)
1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f	Choosing parameter to change (For details, see the menu system on the right)
m	Switching between the two sets of changeable parameters (flocking algo parameters and unit model parameters)
Space	Pause
Right, left, up and down arrows	Moving the camera to the corresponding direction
+ / -	Increasing / decreasing the value of the actually selected parameter
PageUp, PageDown	Same, with larger step size
Home, End	Same, with even larger step size
Mouse Scroll	Zoom
ALT + Mouse Scroll	Rotating the camera up or down, (if 3D visualization is on)
SHIFT + Mouse scroll	Rotating the camera left or right (is 3D visualization is on)

F12	Reset situation (positions will be redistributed on the visible territory, velocities will be set to zero)
ALT + F12	Reset situation (positions will be redistributed on the original starting area, velocities will be set to zero)

References

[1] - “Flocking algorithm for autonomous flying robots” - Cs. Virágh et al., Bioinspiration & Biomimetics, 9/2, 0250