corticalSim v2.0 code manual

corticalSim is a plant microtubule cortical array simulation tool created by **Simon Tindemans** and **Eva Deinum**. This document provides compilation instructions and details on the v2.0 release of *corticalSim*, by **Marco Saltini**, **Bas Jacobs**, and **Eva Deinum**.

Document revision history

18/05/2012	Initial version (v1.25)	ST
20/09/2012	v1.26	ST
19/09/2014	v1.28	ST
22/07/2024	v2.0 beta	MS

Code revision history

V2.0 (18 July 2024)

Implemented the nucleation mode "aster" (local density-dependent nucleation algorithm as described in Saltini & Deinum, 2024, https://doi.org/10.1101/2024.03.25.586463).
Implemented deflection for microtubules (as described in Jacobs et al., 2024, https://doi.org/10.1101/2024.04.04.588070).

V1.28 (19 September 2014)

See changes on bitbucket

V1.26 (20 September 2012)

Fixed bug in queue.cpp: `EventDescriptor::reinitialize`
Bug was found during a code review. As far as I can tell, it has not affected any previous simulation runs.

V1.25 (18 May 2012)

Implemented 'densityLimit' option.
Removed debug code for bundle extensions (included since v1.21b).
Changed default wall clock and memory limits to VERY_LARGE.
Fixed a bug where reaching the wall clock limit would not perform a final measurement.
Cleaned up comments.

V1.24 (20 Jan 2012)

With pre-seeded nucleation, nucleation events are also triggered on 'forbidden zones', but the events are discarded. This is analogous to regular nucleations.
General code cleanup.

V1.23 [beta] (4 Dec 2011)

Option to specify a density of 'pre-seeded' nucleation complexes that each nucleate a single microtubule.

V1.22 [beta] (25 Nov 2011)

Functionality to reduce the efficiency of nucleation complexes that are not bound to microtubules.

V1.21 [beta] (30 June 2011)

Bug fix/feature completion: when extending a trajectory, the program now searches for existing trajectories with the same properties. These exist when both trajectories are descendents of the same progenitor. This prevents overlapping bundles from being created, which can lead to counting problems for complex bundle collisions.
Removed option for compiling without intersection severing support. (#CROSS_SEV)
Removed commented out code snippets throughout and collected remaining warnings and 'to dos' at the top of corticalSim.h.

V1.20 (30 May 2011)

Implemented edge catastrophes and related options: *edgeCatastropheEnabled*, *edgeCatastropheSmooth*, *pCatRegularEdge* and *pCatSpecialEdge*.

V1.19 (10 May 2011)

Fixed statements leading to errors and warning on Visual C++ 2010

V1.18 (20 April 2011)

- Removed nucleation modes "chanLloyd", "chanLloydRandomPosition" and "chanLloydIsotropic"
- Enabled nucleation bias on free part of "ellipse" nucleation.

V1.17 (1 July 2010)

- Implemented the Ncollision bundle interaction mode
- Changed default values for microtubule dynamics and interaction
- Removed the cytrynbaum interaction mode and enabled 'zipFraction' for the 'zipFirst' interaction mode
- Remove unused/commented code
- Removed legacy parameter support
- Remove the Gscaling parameter
- Separated the parameters of the 'order' drawing command by spaces instead of tabs

Building the program

The code includes a make file (makefile) that can be used to compile the program. See the make file for configuration instructions for your platform and compiler. It is expected that the makefile is run in the code directory and that it has a subdirectory 'libs' with the libraries MersenneTwister.h and boost.

The provided make options are listed below. If you're building from an IDE, these should be specified within the IDE.

Command	Function
make debug	Build debug version of the code: corticalSim.debug[.exe]
make release	Build release version of the code: corticalSim.release[.exe]
make warning	As debug version, but with increased compiler warning sensitivity
make all	Compiles the previously used configuration. If no previous build information is found, 'release' is assumed.
make clean	Removes all expected object files
make cleanmore	Removes all object directories and clears the build flags.

The program has been successfully compiled and run under gcc (v11.4.0) on Linux Mint (v21.3), and gcc (v9.4.0) on Linux Mint (v20.2). The code makes use of C++ boost libraries. The relevant libraries can be found, for example, in the libboost-all-dev package for Ubuntu-based Linux distributions.