Instruction Manual

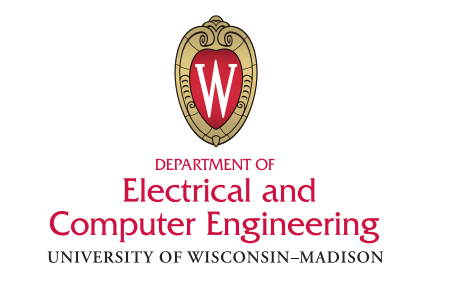
**Carbon Nanotube Mesh Monte Carlo Simulator**

Version: 1.00

Alexander Gabourie

Amirhossein Davoody

Prof. Irena Knezevic



Contents

[Introduction 2](#_Toc426728675)

[Usage 3](#_Toc426728676)

[Installation 3](#_Toc426728677)

[Initialization 4](#_Toc426728678)

[Configuration File 4](#_Toc426728679)

[Runtime 5](#_Toc426728680)

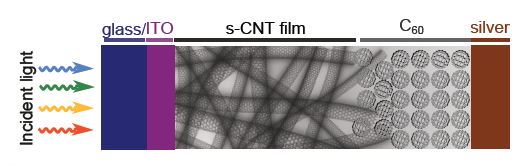
[Exit 6](#_Toc426728681)

[References 6](#_Toc426728682)

# Introduction

The National Academy of Engineering, concerned about the increasing energy demands as well as the environmental cost of meeting them, has determined that development of an economic solar panel is a grand engineering challenge of the 21st century. To address the challenge, much effort has been placed towards finding a suitable light-absorbing material. In recent years, carbon nanotubes (CNTs) have become a material candidate due to more effective separation techniques.[1] Researchers now have the ability to separate optically absorptive, semiconducting nanotubes from others. With a wide range of possible band gaps due to different nanotube chiralities, CNTs can be used to create widely tunable photovoltaic cells. The optical versatility, ultrafast exciton and charge transport, and chemical stability of CNTs has been the impetus of much recent research.[2]

While research on bilayer donor/acceptor heterojunction photovoltaics has shown that exciton dissociation at the semiconducting CNT (s-CNT) and C60 junction [fig. 1] is very efficient, the diffusion of the exciton to the junction has been determined to be a bottleneck in increasing the overall process efficiency.[2] Amirhossein Davoody, a member of Professor Knezevic’s research group at the the University of Wisconsin-Madison, has undertaken a project to develop the theory of exciton diffusion in CNTs and use his theoretical framework to computationally simulate exciton dynamics in a CNT mesh. Before development of the CNT Mesh Generator began, he had already found success calculating exciton transfer rates between CNTs with arbitrary chirality, angle, and separation.



**Fig. 1** Structure of a CNT-based bilayer donor/acceptor heterojunction photovoltaic device.[2]

The purpose of this project is to bridge the two research fronts: Carbon nanotube mesh generator, and exciton transition rates. The CNT mesh acts as the media for a Monte Carlo simulation and the exciton transition rates act as the mechanisms for exciton movement in the simulation. The amalgmation of the two projects will allow a better comparison between older exciton diffusion techniques and Amirhossein’s updated technique. Additionally, since the Monte Carlo simulation will be run on a randomly distributed carbon nanotube network, which better imitates real carbon nanotube meshes, it will allow the connection between the theoretical findings in a more direct and meaningful way to experiment. This software is the tool to accomplish these tasks.

The manual provides the information necessary to understand how to use the software and how the software works. This software is dependent on results from both the Carbon Nanotube Mesh Generator program and the exciton transition rate tables. The proceedings operate under the assumption that the external information is available and in the correct location. Discussion of the external information will be provided when deemed necessary by the author.

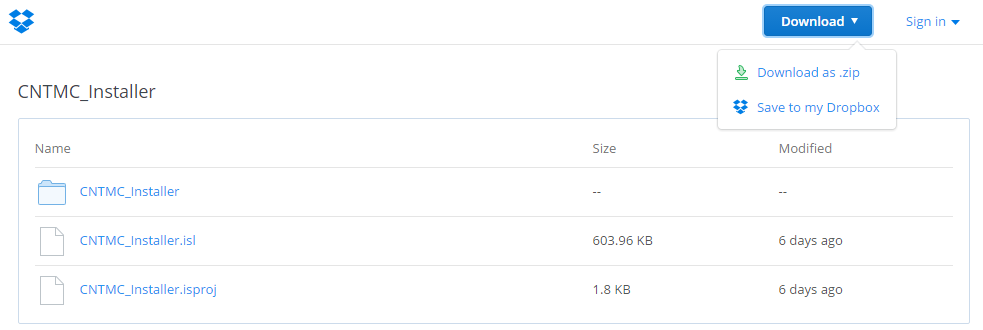
# Usage

## Installation

The current set-up for the software does not have a website for automatic updates. However,it does have a Dropbox link to the most recent deployed version of the software. The link is:

<https://www.dropbox.com/sh/900pnvuqqiqds5y/AADmWEKqXcbm_PWAYmuEv6RMa?dl=0>

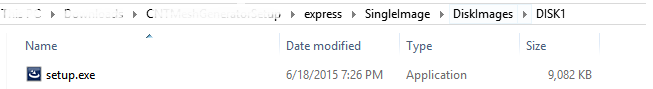
After clicking on the link you will come to a screen that looks like the image below. Click on the “Download” button and select “Download as .zip.”



This will download all the necessary program files. Extract the .zip to a directory of your choosing and open that directory. From that directory navigate to the following path:

*.\express\singleimage\diskimages\disk1*

At this location, a setup executable will be located as seen in the picture below.



Click on “setup.exe” and follow the on screen instructions to install. The installer may ask to download and install some Microsoft redistributables. Please install them as they are necessary components to the software.

## Initialization

One can run the program using either the desktop shortcut or the command line. If running the software from the desktop shortcut, a window will appear and ask for the path of the results folder. The results folder is the output folder from the CNT mesh generator software that this program uses to run the Monte Carlo simulation. Type the full path to the desired results folder. If running from the command line, there is the option to include the path to the results folder as a command line argument. If no argument is provided, the user will get the same prompt as seen when using the desktop shortcut.

### Configuration File

The configuration file itself can be edited to change the simulation parameters. This section is focused on explaining each of the editable sections.

Before describing the sections, the user should be aware of acceptable inputs for units in the configuration files. The acceptable terms are listed below:

-> mm, millimeter

-> um, micrometer

-> nm, nanometer

-> pm, picometer

-> A, angstrom

For this program, all nodes before *numberExcitons* should be kept the same. Changing their values only introduces errors in the Monte Carlo simulation as they are used to quickly characterize the mesh. For proper program usage, only change the nodes listed below.

***numberExcitons:***The number of excitons in the injection contact. This is the starting number of

excitons. As the simulation progresses and excitons leave the injection contact, the contact will be replenished with excitons up to this number. Must be an integer greater than zero.

***regionLength:*** The length of the regions in the direction that diffusion is to be observed. The

device area of the mesh is broken into equal segments of this length and the number of excitons per region is counted for the output files. Literal simulation value may be larger than specified to ensure equally spaced regions. The value must be greater than 0 and less than the <xdim> parameter. (Default: segmentLength)

***segmentLength:*** The CNTs are broken up into segments that are used to hold excitons. This

parameter controls the lengths of the segments. Literal simulation value may be larger than specified to ensure equally spaced segments for each CNT. The value must be greater than 0 and less than the <Lmin> parameter. (Default: 10 nm)

***segmentSeparation:*** The Monte Carlo simulation builds a table of possible transition locations

for each segment. In an effort to reduce the memory and computation requirements of the simulation, not all of the possible locations are included due to negligible probabilities. This parameter chooses the maximum separation distance between two segments for them to be included in each other’s transition tables. The value must be greater than 0. (Default: 30 nm)

***numberTimeSteps:*** The number of time steps the simulation is to run. The value must be an

integer greater than or equal to 0. If the number of time steps is equal to 0, then auto-completion is assumed to be wanted.

***percentFreeFlightTimeAboveDeltaT:***In Monte Carlo simulations, the free flight time is chosen

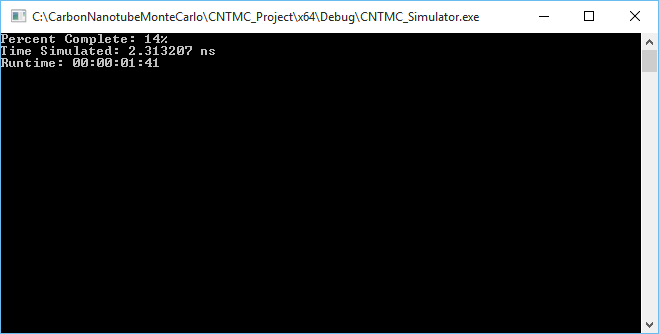
randomly based on an expression to be discussed later. This expression allows the free flight time, in theory, to be infinity. This means that no matter what interval exciton statistics are taken, there is a chance that an exciton’s free flight time may be larger. This parameter allows the user to choose what percent of exciton free flights will be greater than the interval that exciton statistics are taken. The value must be an integer greater than 0 but less than 100.

***autoComplete:*** This node holds four items relevant to the auto complete algorithm. The

<enable> node is a boolean specifying whether or not the auto-complete algorithm will run or the number of time steps dictates its run time. The value must be “true” or “false.” The <threshold> node specifies how strict the ending condition is. The derivative of the total number of excitons divided by the maximum derivative of the total number of excitons (completion parameter) is compared to this threshold value. The larger the number of excitons in the simulation, the smaller the threshold can get for auto completion.The value must be greater than 0 and less than 1. The <numBelowThreshold> node specifies how many times the completion parameter must be below the threshold for the simulation to finish. A larger value enforces a stricter total exciton count stability in the simulation before finishing. The value must be an integer greater than 0. The <numToAverage> node specifies how many time steps are averaged together before calculating the completion parameter. Larger values yield less noisy plots of the completion parameters over time. The value must be an integer greater than 0.

## Runtime

While the simulation is running, the user has very limited input, but can still watch the console to see run progress. Since these simulations can take a long time, the console displays two or three metrics while running to indicate the simulation’s progress. An example can be seen in the console screenshot below.



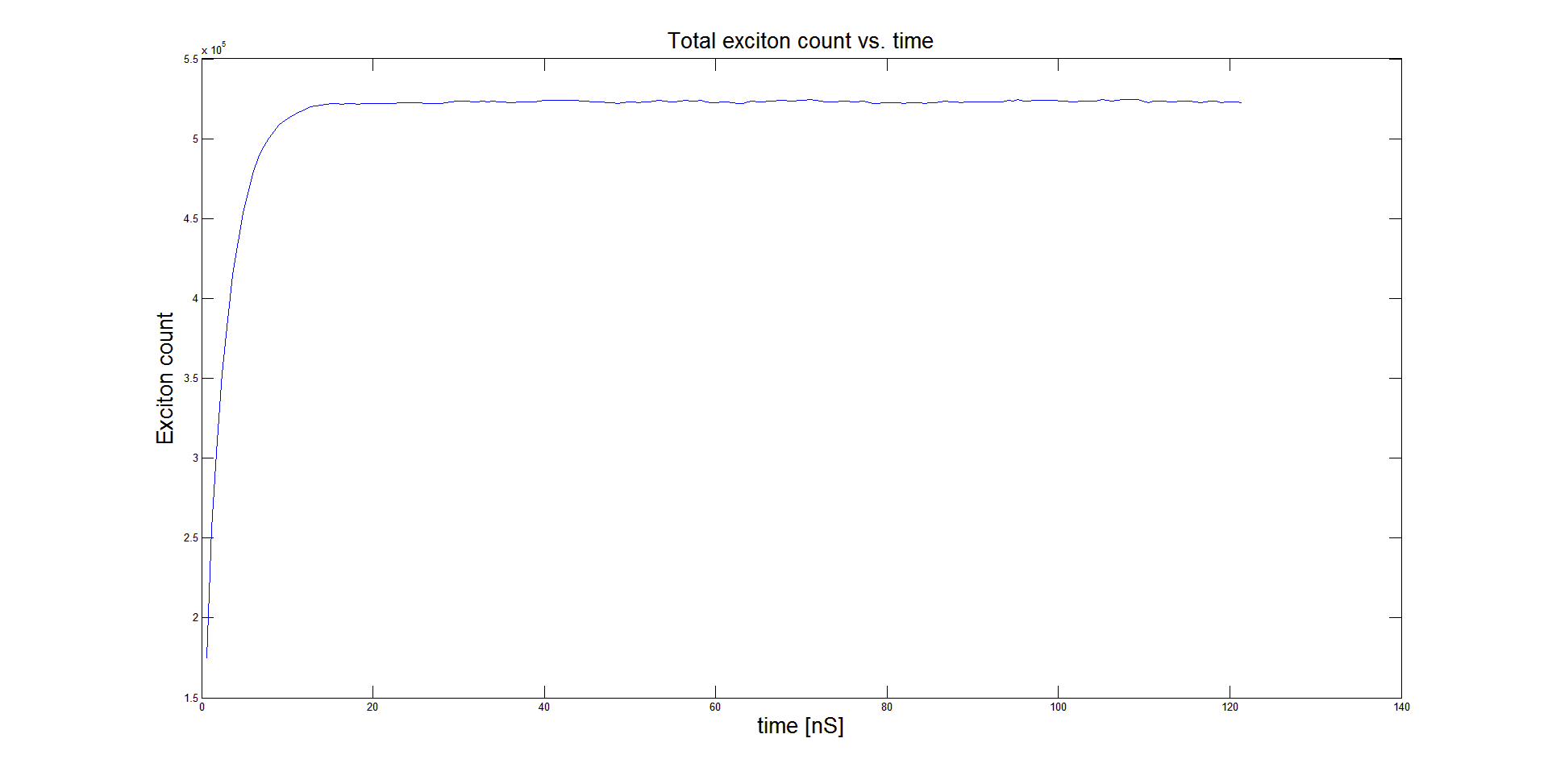
If the simulation is set to run a specific number of time steps, a “Percent Complete” statistic is shown to indicate total progress. All simulations show the simulated time and the run time. The simulated time is how many nanoseconds have been simulated. The run time is how long the simulation has run in days, hours, minutes, and seconds. These parameters are not updated frequently and only provide a general sense of the simulation progress.

The user does have one input option. If the user, for example, decides that they have chosen too long of a run time or are certain the simulation has run long enough, then they can end it manually. To end the simulation manually, press **[shift + q]**. This is the same key combination that the CNT mesh generator uses to end its simulation as well.

Lastly, it is in the user’s best interest to minimize the use of the computer running this software. The software uses multi-threading and as a result 100% of the CPU time if available. Light computer usage results in much faster run times.

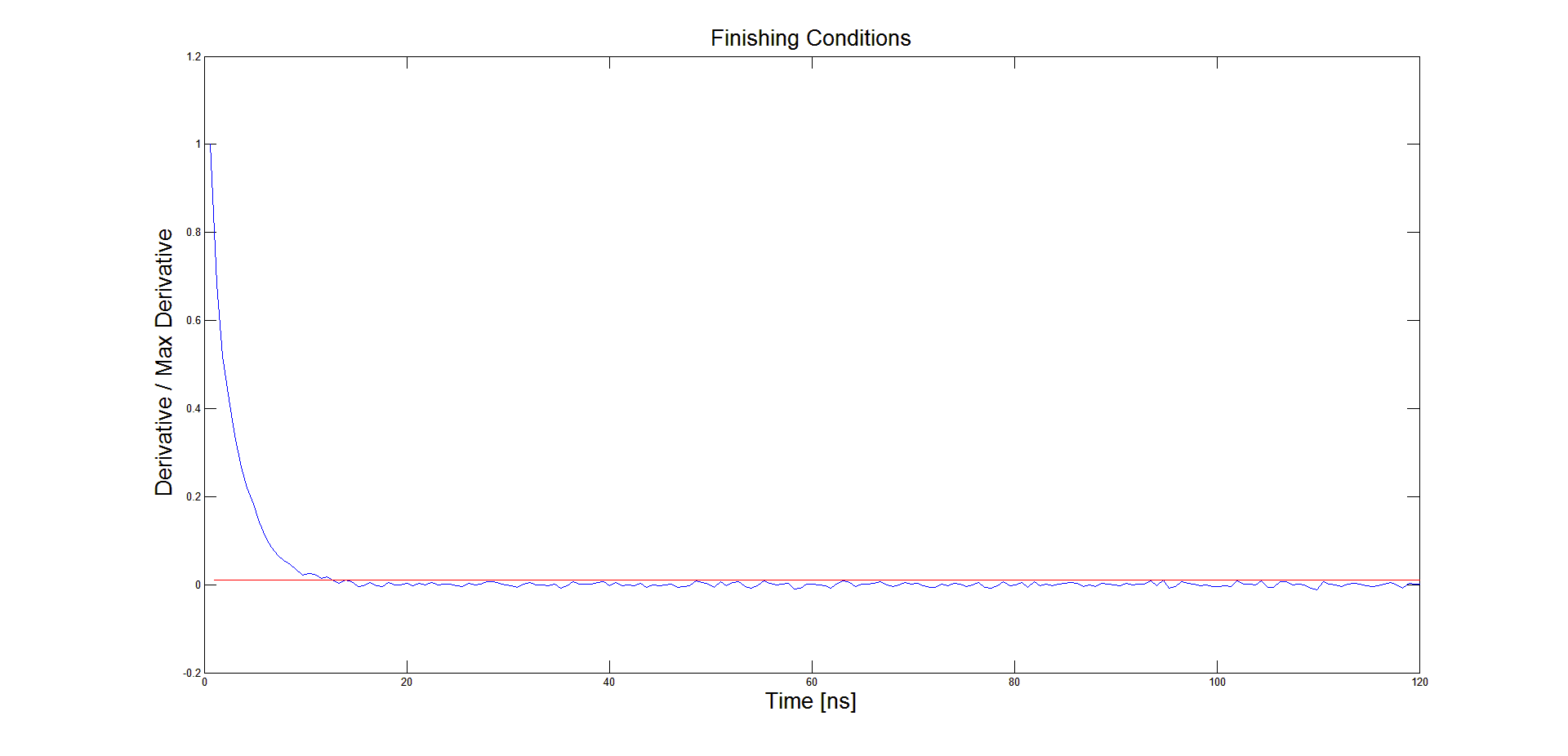
## Exit

The simulation can end in three different ways: the specified number of time steps have been simulated, the user manually quit, or automatically, after meeting the ending conditions specified in the configuration XML file. The first two are self-descriptive, but the auto completion method requires some explanation. The simulation keeps a constant number of excitons in the contact. As the simulation runs, more excitons leave the contact requiring that more excitons get added to the simulation. Eventually, the number of excitons leaving the device and entering the device becomes equal. This means that the total number of excitons in the simulation should become close to constant. An illustration of this can be seen in **Fig. 2**.



**Fig. 2** Plot of the total exciton count vs. time

While the plot in **Fig. 2** shows the total number of excitons reach some steady state, on a point to point basis there is still some slight change in the total number of excitons. The larger the number of starting excitons in the simulation, the larger these fluctuations in the total number of excitons will be. This means that, while the overall trend shows the total number of excitons reaching some constant value, the individual numerical differentiation shows changes proportional to the number of excitons that were initialized in the simulation. To scale each derivative to a range that can be consistently compared across simulations, the numerical differentiation of **Fig. 2** is divided by the maximum of the numerical differentiation up until that point in time. This puts the changes of the total exciton count in perspective with the size of the simulation. **Fig.** 3 illustrates the results of the scaling. No matter what the size of the simulation, the plot will have values less than |1| which allows the user to choose common threshold values.



**Fig. 3** Derivative of Fig.2 divided by the maximum derivative. The red line shows a threshold line of .01.

# References

[1] M. S. Arnold, J. L. Blackburn, J. J. Crochet, S. K. Doorn, J. G. Duque, A. Mohite, and H. Telg,

“Recent developments in the photophysics of single-walled carbon nanotubes for their use as active and passive material elements in thin film photovoltaics.,” *Phys. Chem. Chem. Phys.*, vol. 15, no. 36, pp. 14896–918, 2013.

[2] A. Davoody and I. Knezevic, “Theoretical study of exciton dynamics in carbon nanotube

composites,” Poster presented at ICPS, Zurich, Switzerland, 2013 .