I. INTRODUCTION

A. BACKGROUND OF THE STUDY

Electrophoresis is the separation of charged molecules in an applied electric field. The Principle of Electrophoresis states that if a mixture of electrically charged biomolecules is placed in an electric field, they will move towards the electrode of opposite charge [1].

One of the molecules that can be studied through Electrophoresis is the Deoxyribonucleic Acid or the DNA. The DNA is a nucleic acid molecule which contains the genetic instructions of living organisms. Generally, there are different types of dynamics models for DNA molecules. One of these models is the Bead- spring Model [2]. It incorporates molecular rigidity by means of springs between beads, which are second neighbors along the contour of the chain. These springs are equivalent to elastic forces having longitudinal and transversal contributions [3].

The most common method to separate DNA molecules according to their size is Gel Electrophoresis. In this method, a molecule is to pass through a gel which serves as a sieve to filter out the short strands from the long ones. Separation techniques like electrophoresis offer the analysis and structural identification of complex mixtures which may lead to drug discovery [4].

In addition to Electrophoresis, Entropic Trapping was introduced as another dynamic method to be used in the process. The idea in Entropic trapping includes the characterization of the movement of a long DNA polymer in an artificial channel with entropic traps [5]. Entropic trapping constitutes a different scheme for separating various length strands of DNA using chips etched to the channel. Deep portions of the chip, acting as entropic traps in a chain-length dependent manner, hinder DNA solute transport occurring under the influence of an externally-applied electric field [6].

There are currently existing computer simulations on Gel Electrophoresis which can help students in their experiments by providing a “virtual lab” and giving results faster than the mechanical way of doing it. There are no currently available computer simulations on Entropic trapping, but more studies are being made for the development of entropic trapping.

B. STATEMENT OF THE PROBLEM

The DNA is an important biomolecule being studied by many scientists. One way of studying the kinetics of the DNA molecule is through Electrophoresis. Since the movement of DNA molecules in a solution is independent of their length, a gel is usually introduced to the DNA. This gel serves as a random sieve. Gel electrophoresis can be used to separate DNA fragments. It uses an electric current to separate molecules of different sizes in a porous, sponge- like matrix. Unfortunately, the efficiency of gel electrophoresis decreases with the length of the DNA [7]. Therefore, much recent effort has been given and devoted to designing and creating well- defined microstructured devices for DNA separation.

A computer simulation that can provide numerical and graphical data for studying these DNA molecules is needed to provide a better understanding of the behavior of the molecule given a particular condition with different parameters during DNA separation.

C. OBJECTIVES

To develop a computer simulation of DNA motion in entropic trapping with the following functionalities:

1. allows users to enter the parameters needed for the simulation, particularly the temperature, electric field, and bead size
2. perform computations
    a. Force of interaction
        i. Force on the spring
        ii. Force on the bead
    b. Electric force
    c. Random force
    d. Integration of the Equation of Motion

3. outputs the results of the simulation
    a. numerical data containing the value of the position of the ith bead
    b. graphical data of the position of the ith bead with respect to time
4. outputs the graph resulting from the simulation in a PNG image file

D. SIGNIFICANCE

The computer simulation for the motion of the DNA in free- flow electrophoresis will be very helpful to microbiologists and chemists. Since they are the ones who are working in the laboratories most of the time, this can be an efficient tool in their study of the DNA. The simulation will show the movement of the DNA given an environment with entropic trappings, and as a result, they will be able to classify the large molecules from the small ones [8].

The concept of the separation of the DNA is not only useful to microbiologists and chemists, but also to pharmacists. This process can also help them know the effects of a medicine to a particular gene.

This proposed system can also be used by students studying chemistry or biochemistry. A normal experiment involving gel electrophoresis usually takes hours to execute, so with the help of this simulation, students can get results for their experiments at a much shorter time [9].

E. SCOPE AND LIMITATIONS

1. The simulation is concerned with electrophoresis using uniform transverse electricfield.
2. The simulation is concerned with the movement of the DNA in an Entropic trapping environment.
3. The simulation is limited to one- dimentional motion only.

4. The graphical output of the simulation can be saved as a PNG image file.

5. The chain size is set to three molecules.

6. The interbead distance is a fixed value.

F. ASSUMPTIONS

The following are assumed in this project:

1. The molecules are massless.
2. The molecules have constant net charge based on the base pairs.
3. The beads of the molecules are non- intersecting.
4. The molecule in the simulation is traversing in a free- flowing environment with random force.
5. The constants of the simulation are the following:
    - Spring constant, k = 100
    - Charge, qnet = 0.48 me^{-}
    - Transit time = 0.00051 sec.
    - Time step = 0.00000008925
    - Initial Electric Field, E_o = 65 V/m
    - Boltzmann constant = $1.38 \text{ x } 10^{-23}$ J/K

II. Review of Related Literature

Currently, many separation devices are used, or are being developed for the study of the DNA molecule. One example is gel electrophoresis which has existing computer simulations to demonstrate how it is done.

Randy Russell's Gel Electrophoresis Simulation is an interactive, "data- driven" application wherein the positions of the bands for each sample are controlled by simple numerical list variables. Its goal is to determine which of the proteins (hemoglobin, actin, and myosin) are present in liver and muscle tissue samples. It can also be modified to support different samples with different banding patterns [10].

Virtual Lab: Agarose Gel Electrophoresis of Restriction Fragments is a simulation of an agarose gel electrophoresis setup that allows you to understand how restriction enzyme digests are analyzed. It allows the user to set up and run a gel. One thing that can be observed in the simulation is if two fragments of DNA differ in size by only a small amount (say less than 100 bp for fragments larger than about 1 kb), they will run sufficiently close to one another to appear as a single band. [11].

SimGel is an interactive 2D Gel Electrophoresis Simulator that helps the user analyze ordinary electrophoresis quantitatively, interactively, and in details of single nucleotide/amino-acid of every spot with pin-point accuracy. SimGel allows the user to do the following: try many kind of restriction enzymes and examine optimal experimented condition in advance, pin-point the approximate location of the focused spots, view the location of spots without any smear and gel distortion, trace the location of spots(labeled/unlabeled) in arbitrary time-scale and physical scale, measure the spots in one nucleotide/amino-acid wise and review the nucleotide/amino-acid sequence interactively. [12].

The Physlets' BioGel is a simple applet of a gel electrophoresis simulation which allows the user to practice running gels without having to go through the long, messy process. The user can also add up to eight values for kilobase pairs, change the strength of the electric field, and alter the concentration of the gel agarose. [13].

Gel Explorer is a new Gel Documentation Imaging System provides for the analysis of many applications including DNA/RNA gels, protein gels, blots and arrays, TLC plates, and colony counting. The system is available with several camera options and includes many features offered on more expensive Gel Doc systems [14].

PeakMaster is a freeware program useful for people engaged in capillary zone electrophoresis. It predicts parameters of background electrolytes and analyte peaks. The model enables optimization of background electrolyte (BGE) systems for capillary zone electrophoresis. The model allows putting to use uni- or di- or trivalent electrolytes and allows also for modeling highly acidic or alkaline BGEs. It takes into account the dependence of ionic mobilities and dissociation of weak electrolytes on the ionic strength. The model calculates the effective mobility of analytes and predicts parameters of the system that are experimentally available, such as the transfer ratio, which is a measure of the sensitivity in the indirect UV detection or the molar conductivity detection response, which expresses the sensitivity of the conductivity detection. Further, the model enables evaluation of a tendency of the analyte to undergo electromigration dispersion or peak broadening. The suitability of the model is verified by comparison of the predicted results with experiments, even under conditions that are far from ideal (under extreme pH and a high ionic strength) [15].

The software package ElphoFit is a standalone computer application for the analysis of gel electrophoretic data. It is designed for Macintosh PCs and for the IBM XT, AT, PS/2 and compatibles. The program operates interactively with the user, who determines the course of evaluation. Data input is in the format of files providing values of gel electrophoretic migration distances or particle mobility (absolute or relative). Data processing involves a simultaneous least-square curve fitting algorithm (Newton-Gauss, Marquardt-Levenberg). Functions are fit to the database by adjusting their variables, representing physical parameters of the gel and the electrophoresed particle. The program output consists of tables and graphics accompanied by an explanatory text [16].

GelSite is a software that allows analysis of molecular biology/gel electrophoreris plates by increasing efficiency and accuracy in the data analysis. It increases efficiency by accurately and automatically analyzing band location. By placing lane information on the gel image, band RF values (locations) are automatically determined based on differential threshold levels of bands images. It also supports cut and paste from other applications. After an image of the gel is taken, the image can be pasted into the view allowing immediate analysis. In addition, it archives all work and saves all work information in an internal data base. This allows you to restore the work and use it for later analysis on the same gel, or use a past archive to get started in a new analysis [17].

Another type of 1D electrophoresis software us LabImage 1D. Some of its features are accurate automatic lane detection, RF Calibration, molecular weight calibration, quantification / normalization, build in analysis workflow, background subtraction, band detection, documentation and visualization and reporting**.** It has a step-by-step workflow, strong image analysis algorithms, flexible reporting and data export which makes it one of the most advanced software solutions in 1D analysis [18].

UN-SCAN-IT-GEL software turns your scanner into a high speed densitometer and allows you to automatically analyze gel images at Full Scanner Resolution. It improves accuracy and reproducibility of electrophoresis gel analysis, thus eliminating the need for guesswork; reduce gel analysis times and increase laboratory productivity; determine the relative abundance and position of each band or segment within the gel; save data in ASCII format, and export into spreadsheet, data analysis, and graphics programs; digitize (x,y) graphs; graph and analyze (x,y) data [19].

III. THEORETICAL FRAMEWORK

A computer simulation is a computer program that attempts to create an abstract or theoretical model of a particular system. Computer simulations try to find solutions to problems which enable the prediction of the behavior of the system from a set of parameters and initial conditions [20]. These computer models have been useful in different fields of science and mathematics.

Brownian Dynamics

Here, we consider a Brownian Dynamics Simulation of the movement of DNA molecules in an entropic trapped channel.

In a Brownian dynamics simulation, the components of a system are allowed to respond to the instantaneous forces present in a given configuration, which causes the system to take up a new configuration. The forces in this new configuration are then recalculated, the system is then allowed to take up yet another configuration, and a sequence of such steps simulates the evolution of the true system in time [21].

Entropic Trap Arrays

One way of separating molecules is through the use of entropic traps. Figure 1 shows the schematic drawing of entropic trapping.

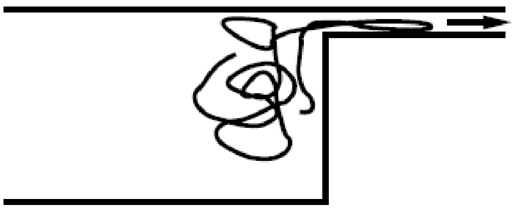


Figure 1. DNA in entropic trapping.

In this mechanism, the molecule gets stuck at the narrow areas of the device. This reduces the entropic free energy of the molecule. As the entropic energy loss increases with the chain length, long chains are expected to migrate slower than short ones [9].

Bead- Spring Model

The bead- spring model is adopted, representing the DNA molecule as *N* beads or monomers of diameter σ. The beads are connected by harmonic springs with constant k as follows:

$$\frac{V^{\mathrm{sp}}(r_{\mathrm{n}})}{k_{\mathrm{B}}T} = \frac{1}{2}k\left(\frac{r_{\mathrm{n}}}{\sigma}\right)^{2},$$

where $\boldsymbol{V^{sp}}$ is the potential energy of the spring, $\boldsymbol{r_n}$ is the distance between neighboring beads, $\boldsymbol{k_B}$ is the Boltzmann constant, and $\boldsymbol{T}$ is the temperature. All beads are assumed to interact with each other through the Weeks- Chandler- Andersen (WCA) potential,

$$\frac{V^{\mathrm{WCA}}(r)}{k_{\mathrm{B}}T} = \begin{cases} \left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} + \frac{1}{2}, & \frac{r}{\sigma} < 2^{1/6} \\ 0, & \text{otherwise}, \end{cases}$$

where $\boldsymbol{r}$ is the interbead distance [2].

Figure 2 is an illustration of a worm- like chain with vectors connecting positions on the chain [22].

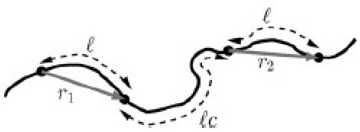


Figure 2. Worm- like chain

These vectors are crucial in knowing the position of the beads of the chain and how these positions affect their neighboring beads.

Runge- Kutta Algorithm

The Runge- Kutta Algorithm is an iterative method in numerical analysis used to approximate solutions of ordinary differential equations.

Consider the single variable problem

$$x' = f(t, x)$$

with initial condition $x(0) = x_0$. Suppose that x_n is the value of the variable at time t_n. The Runge-Kutta formula takes x_n and t_n and calculates an approximation for x_{n+1} at a brief time later, t_n+h. It uses a weighted average of approximated values of $f(t, x)$ at several times within the interval (t_n, t_n+h).

The formula is given by

$$x_{n+1} = x_n + {}^h\!/_6\,(a + 2\,b + 2\,c + d)$$

$$\text{where } a = f(t_n, x_n)$$

$$b = f(t_n + {}^h\!/_2, x_n + {}^h\!/_2\,a)$$

$$c = f(t_n + {}^h\!/_2, x_n + {}^h\!/_2\,b)$$

$$d = f(t_n + h, x_n + h\,c)$$

To solve the differential equation, we start with x_0 and find x_1 using the formula above. Then we plug in x_1 to find x_2 and so on [23].

Graphical User Interface

A **graphical user interface** (**GUI**) is a type of user interface which allows people to interact with a computer and computer-controlled devices which employ graphical icons, visual indicators or special graphical elements called "widgets", along with text, labels or text navigation to represent the information and actions available to a user. The actions are usually performed through direct manipulation of the graphical elements [24]. This system will make use of the GUI to accept input from the user, and equivalently, to show the desired outputs to the user.

IV. DESIGN AND IMPLEMENTATION

A. Context Flow Diagram

Figure 3 shows the basic structure of the system:

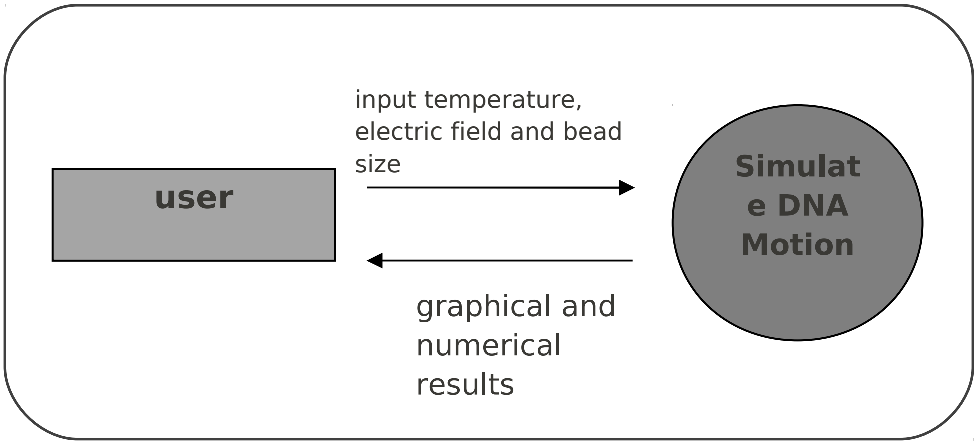

input temperature, electric field and bead size
user
Simulat e DNA Motion
graphical and numerical results

Figure 3. Context Flow Diagram of the system

The user of the system primarily inputs the parameters required for the simulation to run. Upon receiving the input parameters together with the other pre- defined parameter (constant) values, the system will then simulate the DNA motion. Afterwards, the simulation will output the results, both numerical and graphical results to the user.

B. Data Flow Diagram

Figure 4 shows the Top Level Data Flow Diagram of the system. The processes involved are inputting the parameters, computing the potential energies, computing the forces, integrating the main equation and displaying the results back to the user. The results will be saved in a database where the user can view previous simulation results.

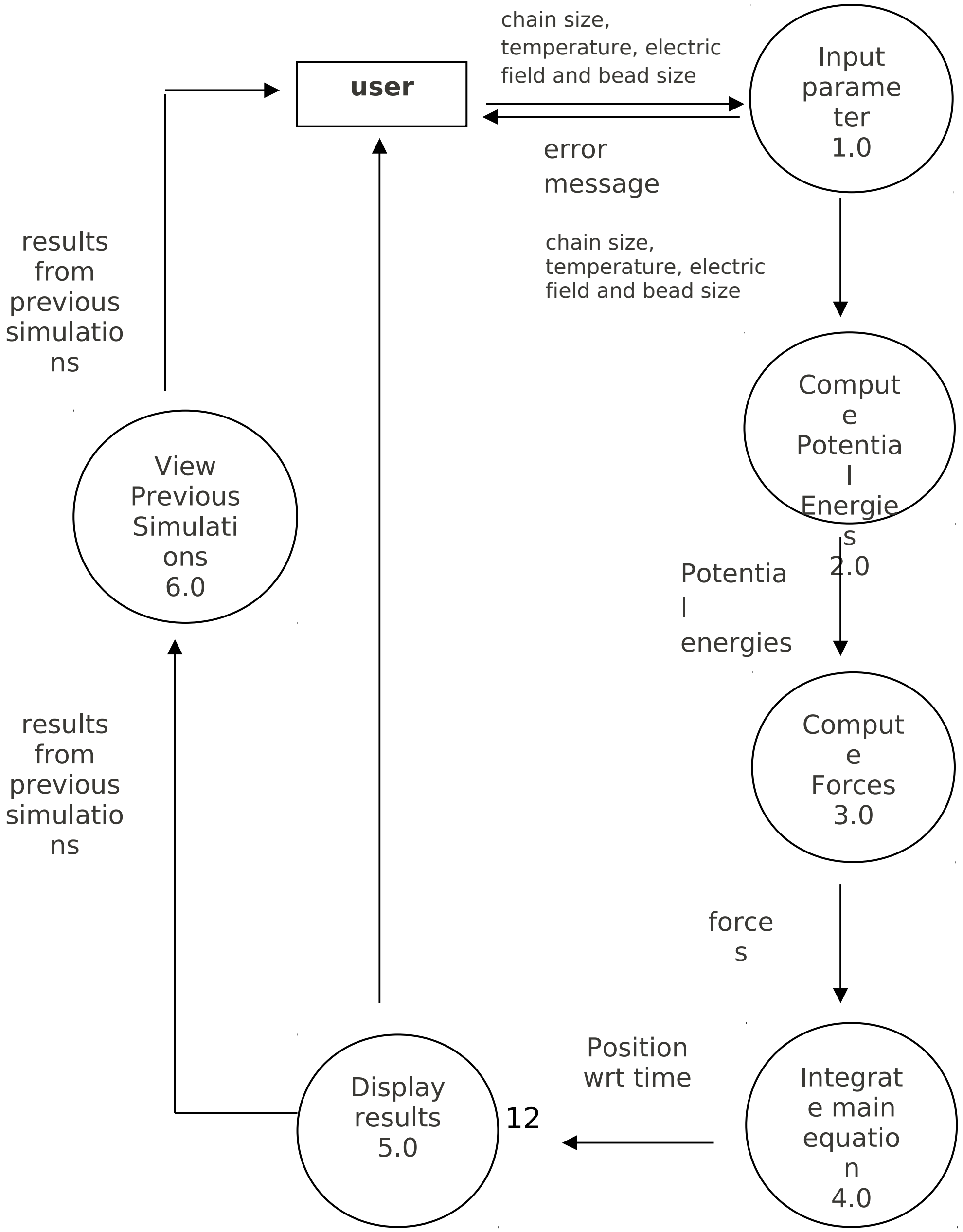

user
chain size, temperature, electric field and bead size
Input parameter 1.0
error message
chain size, temperature, electric field and bead size
Compute Potential Energies 2.0
Potential energies
Compute Forces 3.0
forces
Integrate main equation 4.0
Position wrt time
Display results 5.0
results from previous simulations
View Previous Simulations 6.0
results from previous simulations

Figure 4. Top Level DFD

Figure 5 shows the subexplosion of the first process, Input parameter. It includes the checking of the chain size, temperature, electric field, distance between beads and number of traps entered by the user and sends an error message if the input is incorrect.

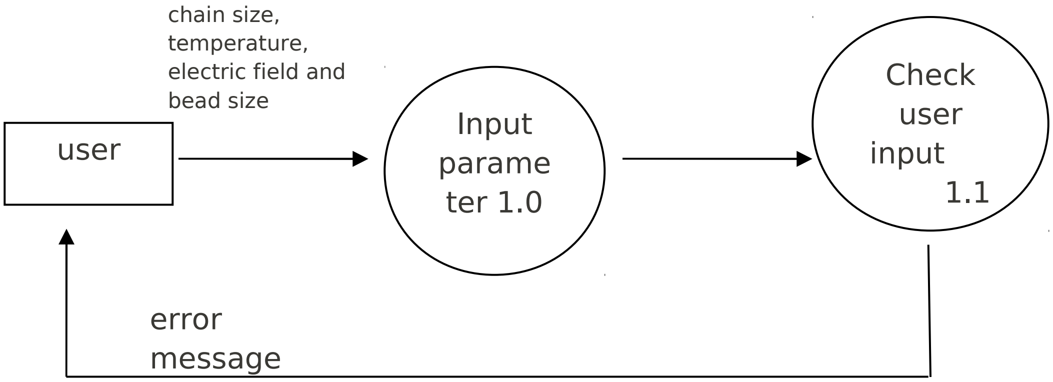


Figure 5. Subexplosion of the first process

Figure 6 shows the subexplosion of the second process, Compute Potential Energies. It includes the computation of the Potential Energy of the Spring and the Potential Energy of the Pair.

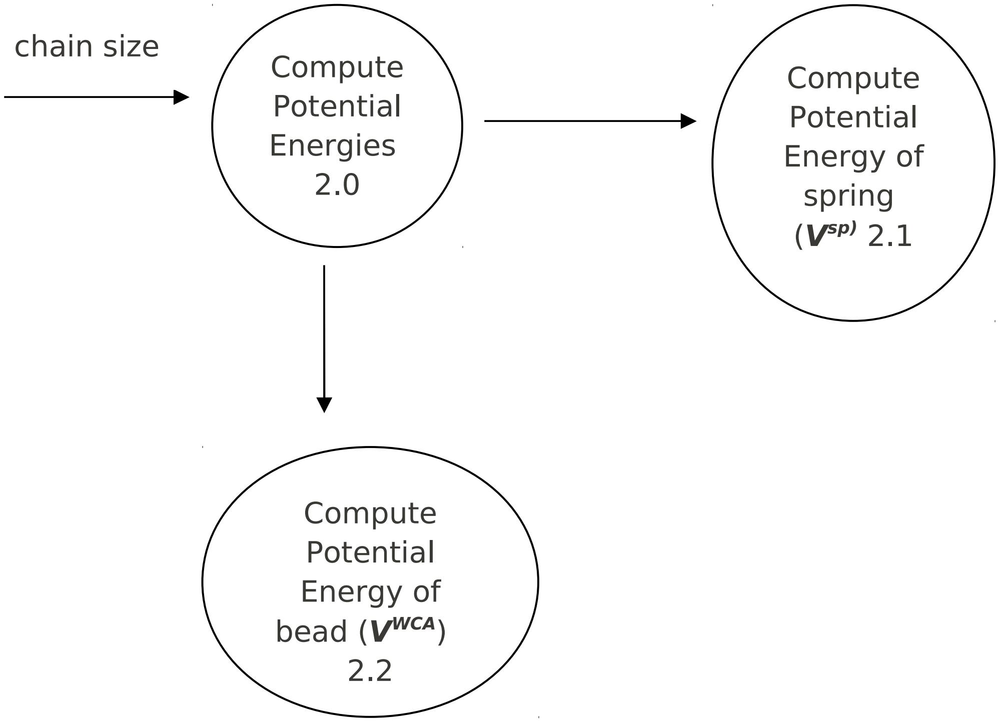

chain size
Compute Potential Energies 2.0
Compute Potential Energy of spring (V^{sp}) 2.1
Compute Potential Energy of bead (V^{WCA}) 2.2

Figure 6. Subexplosion of second process

Figure 7 shows the subexplosion of the third process, Compute Forces. It includes the computation of the Force of Interaction, the Electric Force and the Random Force.

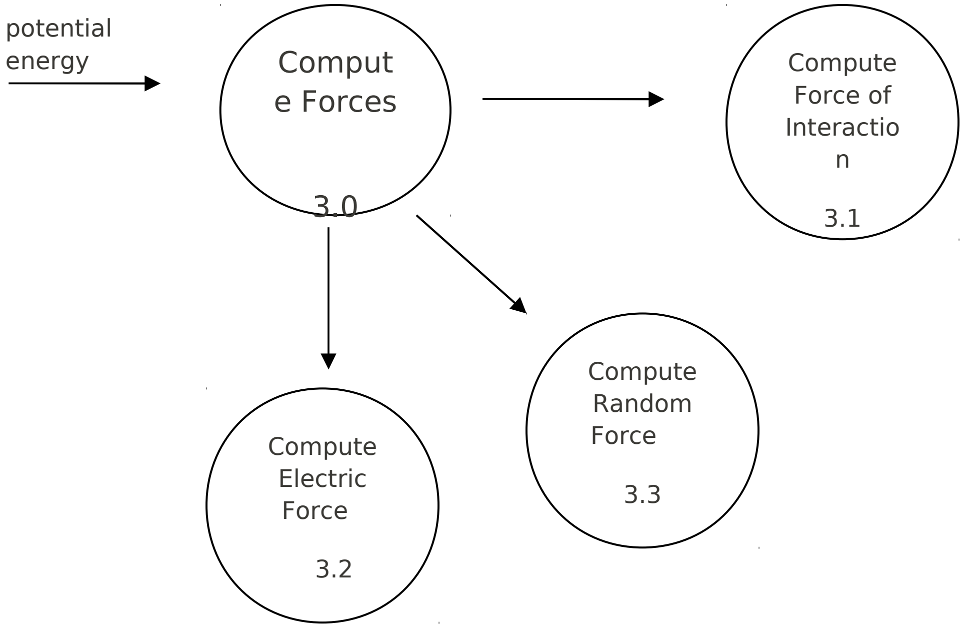


Figure 7. Subexplosion of third process

Figure 8 shows the subexplosion of the fourth process, Integrate main equation. It includes the integration of the Forces derived from the third process such as the Force of Interaction, the Electric Force and the Random Force.

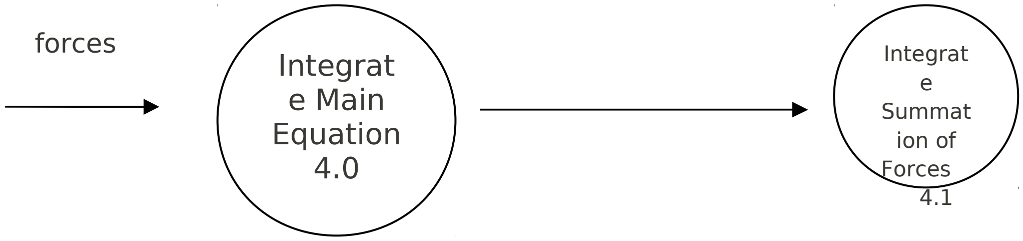


Figure 8. Subexplosion of fourth process

Figure 9 shows the subexplosion of the fifth process, Display Results. It includes displaying the numerical and graphical results. The graphs from previous simulations will then be saved to a database.

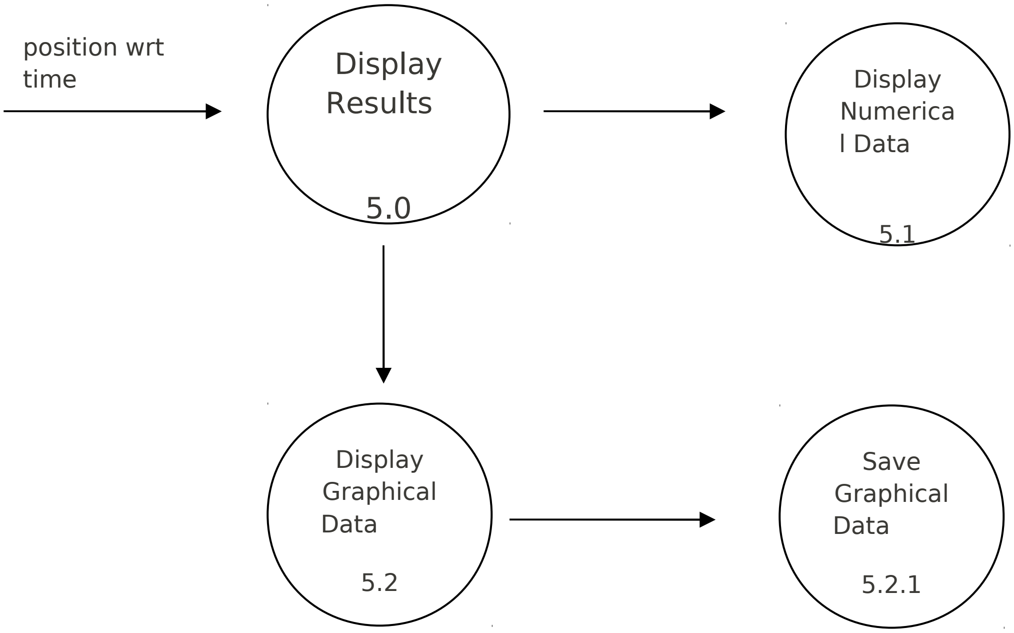


Figure 9. Subexplosion of the fifth process

C. Flowchart

Figure 10 shows the schematic diagram of the system.

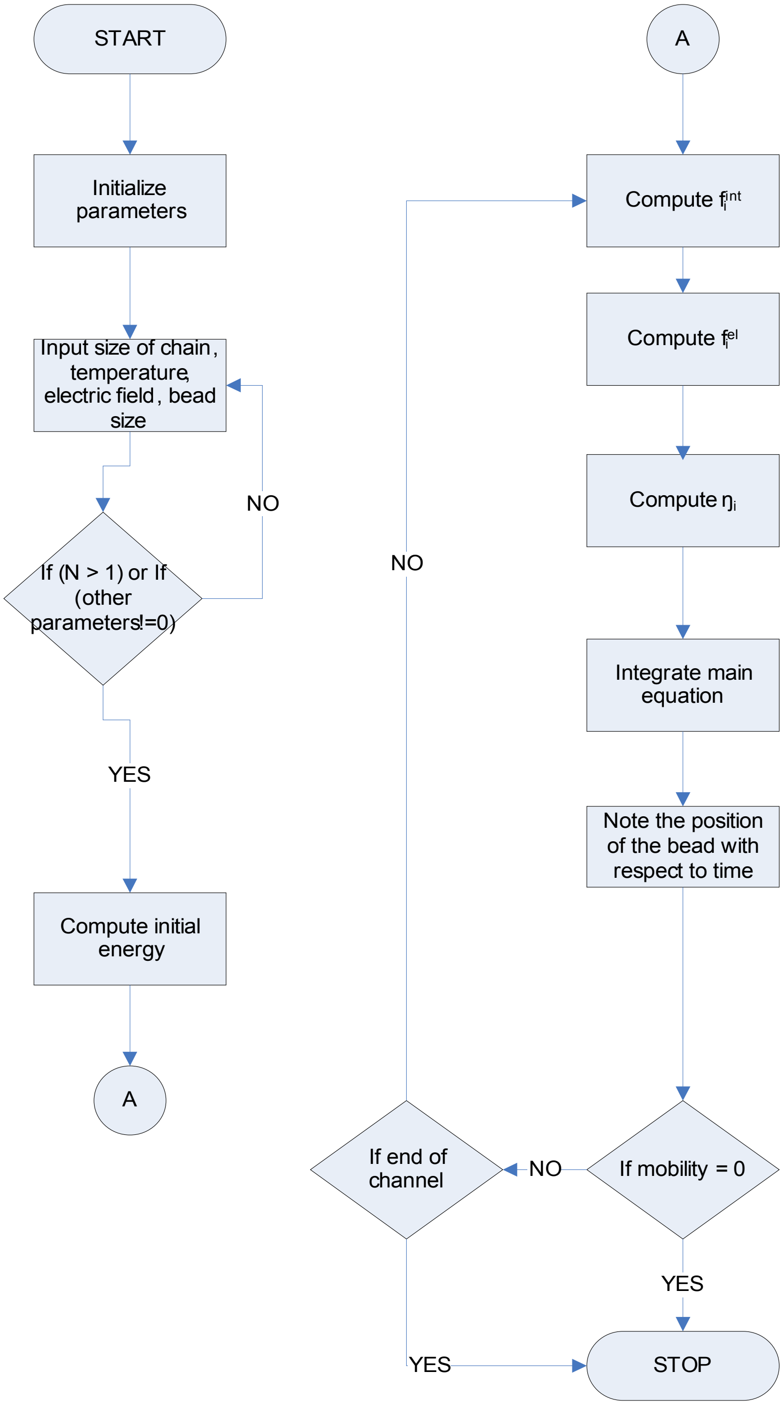


Figure 10. Flow chart

The main routines of the system:

1. Computing for the Force of Interaction, f_i^{int}

This includes the computation of the Potential Energies and the Forces of the spring and the bead.

The computation for the Force of the Spring is initiated by the following equation:

$$\frac{V_{sp}}{K_B T} = \frac{1}{2} k \left(\frac{r_n}{\sigma} \right)^2$$

where $\boldsymbol{V^{sp}}$ is the potential energy of the spring, $\boldsymbol{r_n}$ is the distance between neighboring beads, $\boldsymbol{\sigma}$ is the diameter of the bead, $\boldsymbol{k_B}$ is the Boltzmann constant, and $\boldsymbol{T}$ is the temperature.

Getting the negative derivative of this equation yields

$$f_{sp} = -\frac{kK_B T}{\sigma^2} r_n$$

which is the corresponding Force of the Spring.

The Potential Energy of the Bead-to-Bead Interaction is given by:

$$\frac{V_{wca}}{K_B T} = \left(\frac{\sigma}{r_n} \right)^{12} - \left(\frac{\sigma}{r_n} \right)^6 + \frac{1}{2} \quad for \quad \frac{r_n}{\sigma} < 2^{\frac{1}{6}}$$

where $\boldsymbol{V^{WCA}}$ is the Weeks-Chandler-Andersen Potential, $\boldsymbol{r}$ is the interbead distance, $\boldsymbol{\sigma}$ is the diameter of the bead, $\boldsymbol{k_B}$ is the Boltzmann constant, and $\boldsymbol{T}$ is the temperature.

Same as the spring, the derivative of this equation gives the Force of the Bead-to-Bead Interaction

$$f_{wca} = K_B T \left(\frac{12\sigma^{12}}{r_n^{13}} - \frac{6\sigma^6}{r_n^7} \right)$$

After computing for the Force of the Spring and the Force of the Bead-to-Bead interaction, the Force of Interaction is solved by getting the sum of the two forces

$$\mathbf{f_i^{int} = f^{spring} + f^{bead\text{-}bead}}$$

2. Computing for the Electric Force, f_i^{el}

The Electric Force is computed as

$$\boldsymbol{f_i^{el}} = \text{qnet}(E_o - E_T)$$

where $\boldsymbol{q_{net}}$ = 0.48*me*–, $\boldsymbol{E_o}$ is a constant electric field and E_T is the transverse electric field.

3. Computing for the Random Force, η_i

The Random Force is given by

$$\eta_i = \left(\frac{K_B T}{\sigma}\right)\sqrt{\frac{6\tau}{\Delta t}}\psi_i$$

where $\boldsymbol{k_B}$ is the Boltzmann constant, $\boldsymbol{T}$ is the temperature, $\boldsymbol{\sigma}$ is the diameter of the bead, $\boldsymbol{\tau}$ **=** ***5.1x10⁻⁴ sec.*** , $\boldsymbol{\Delta t = 1.75 \times 10^{-4} \tau}$ and $\boldsymbol{\psi_i}$ is is a random vector that has independent components uniformly distributed on [−1,1].

4. The Equation of Motion for the ith bead

The equation of motion for the ith bead is given by

$$\dot{r}_i = \frac{f_i^{\text{int}} + f_i^{el} + \eta_i}{\zeta}$$

It is integrated using the Fourth Order Runge Kutta to get the position of the ith bead with respect to time.

$$k_{1i} = \left(f_i^{\text{int}}(r_i) + f_i^{el}(r_i) + \eta_i\right) * \left(\frac{\Delta t}{\varsigma}\right)$$

$$k_{2i} = \left(f_i^{\text{int}}(r_i + \frac{k_{1i}}{2}) + f_i^{el}(r_i + \frac{k_{1i}}{2}) + \eta_i \right) * \left(\frac{\Delta t}{\varsigma}\right)$$

$$k_{3i} = \left(f_i^{\text{int}}(r_i + \frac{k_{2i}}{2}) + f_i^{el}(r_i + \frac{k_{2i}}{2}) + \eta_i \right) * \left(\frac{\Delta t}{\varsigma}\right)$$

$$k_{4i} = \left(f_i^{\text{int}}(r_i + k_{3i}) + f_i^{el}(r_i + k_{3i}) + \eta_i\right) * \left(\frac{\Delta t}{\varsigma}\right)$$

$$r_i^{next} = r_i + \frac{1}{6}(k_{1i} + 2k_{2i} + 2k_{3i} + k_{4i})$$

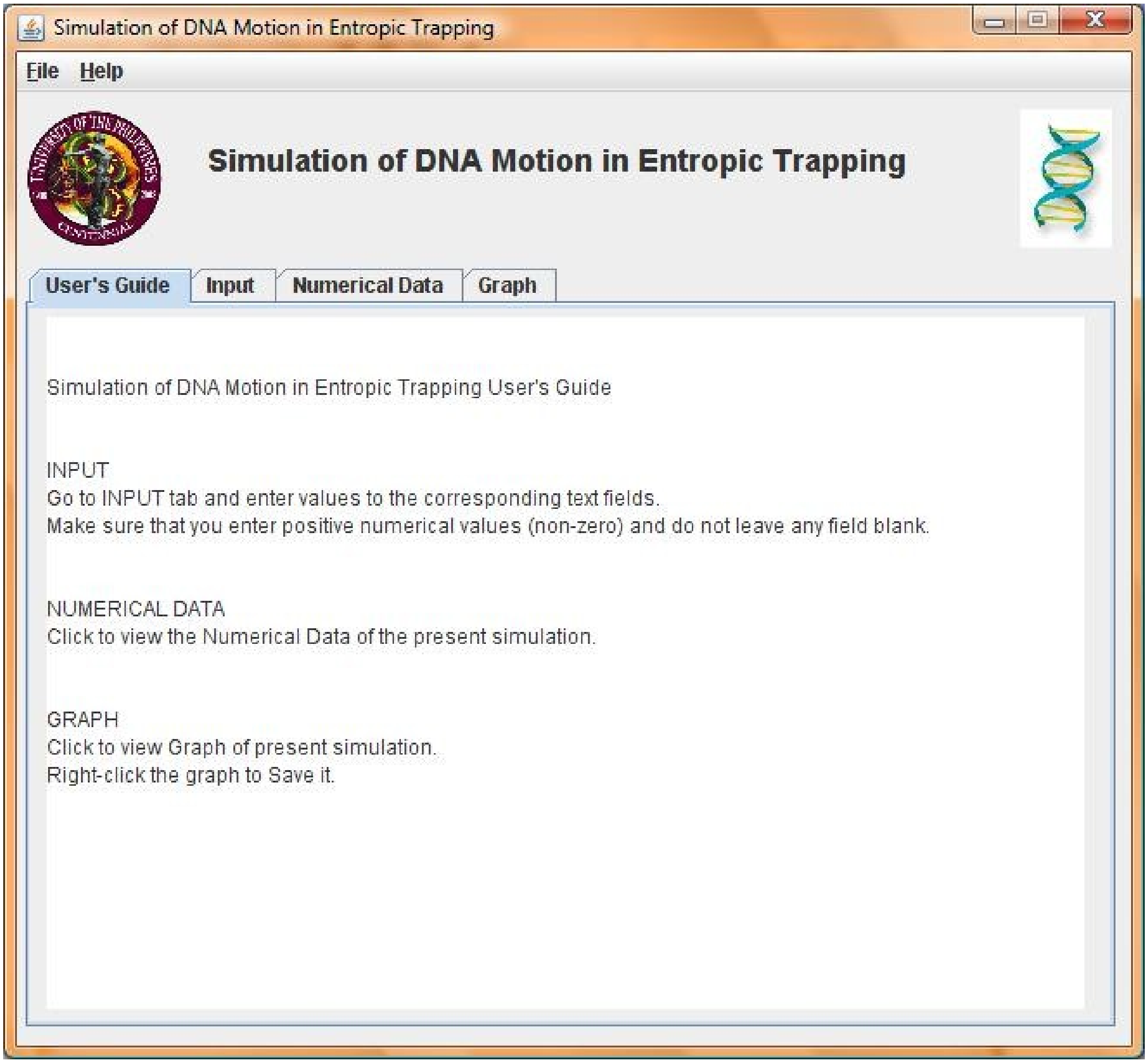
RK4 is used to get an approximation to the next position of the particle.

V. RESULTS

Screen Shots

The interface of the system contains four tabs: User's Guide, Input, Numerical Data and Graph.

Figure 11 shows the initial window that will appear when the system is run. It contains basic information on how to navigate the simulation.


Simulation of DNA Motion in Entropic Trapping
File
Help
Simulation of DNA Motion in Entropic Trapping
User's Guide
Input
Numerical Data
Graph

Simulation of DNA Motion in Entropic Trapping User's Guide

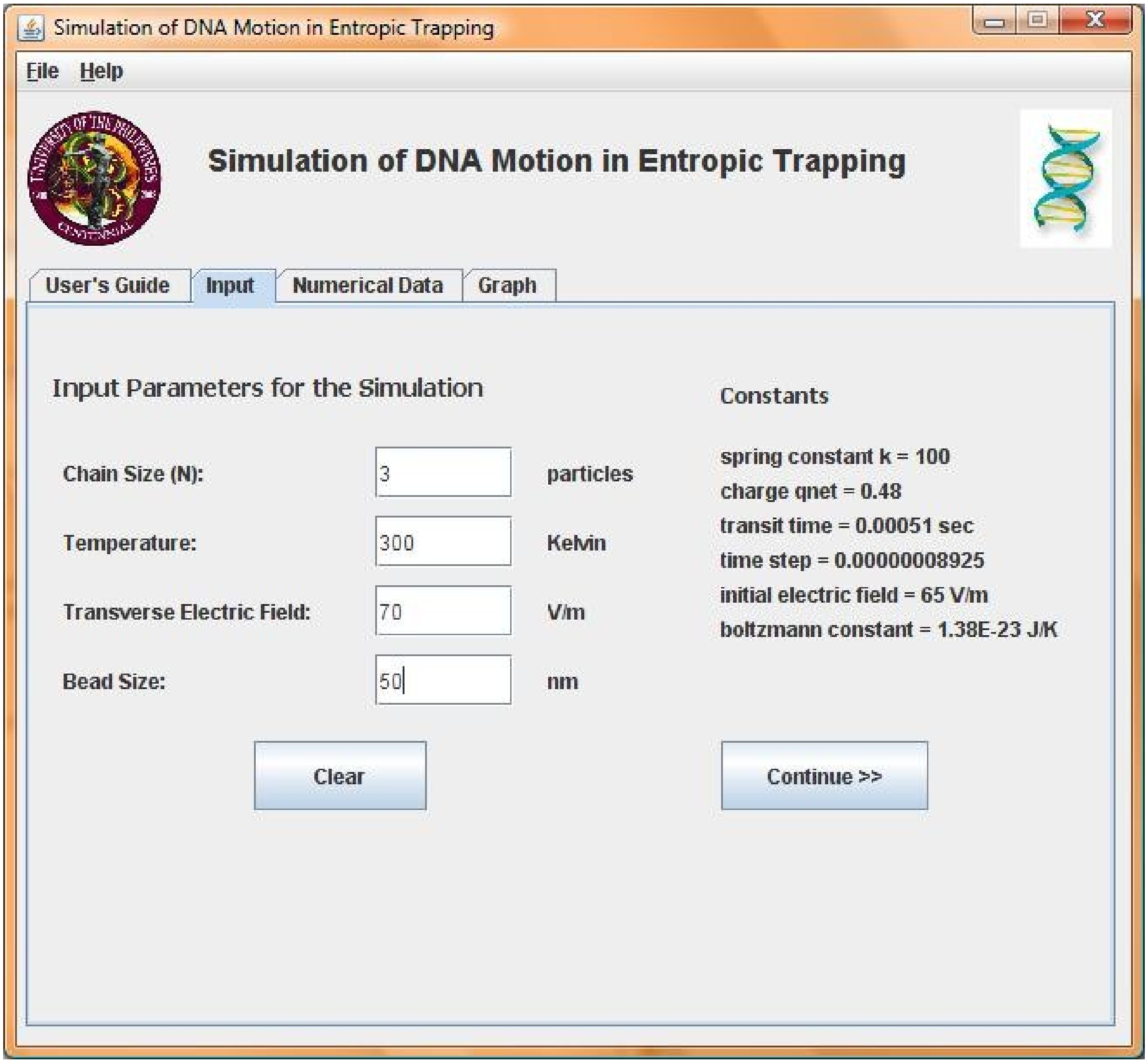
INPUT
Go to INPUT tab and enter values to the corresponding text fields.
Make sure that you enter positive numerical values (non-zero) and do not leave any field blank.

NUMERICAL DATA
Click to view the Numerical Data of the present simulation.

GRAPH
Click to view Graph of present simulation.
Right-click the graph to Save it.

Figure 11. Basic User's Guide

The second tab, Input as shown in Figure 12 allows the user to enter specific data or input values needed for the simulation.



Simulation of DNA Motion in Entropic Trapping

Input Parameters for the Simulation

Chain Size (N): 3 particles

Temperature: 300 Kelvin

Transverse Electric Field: 70 V/m

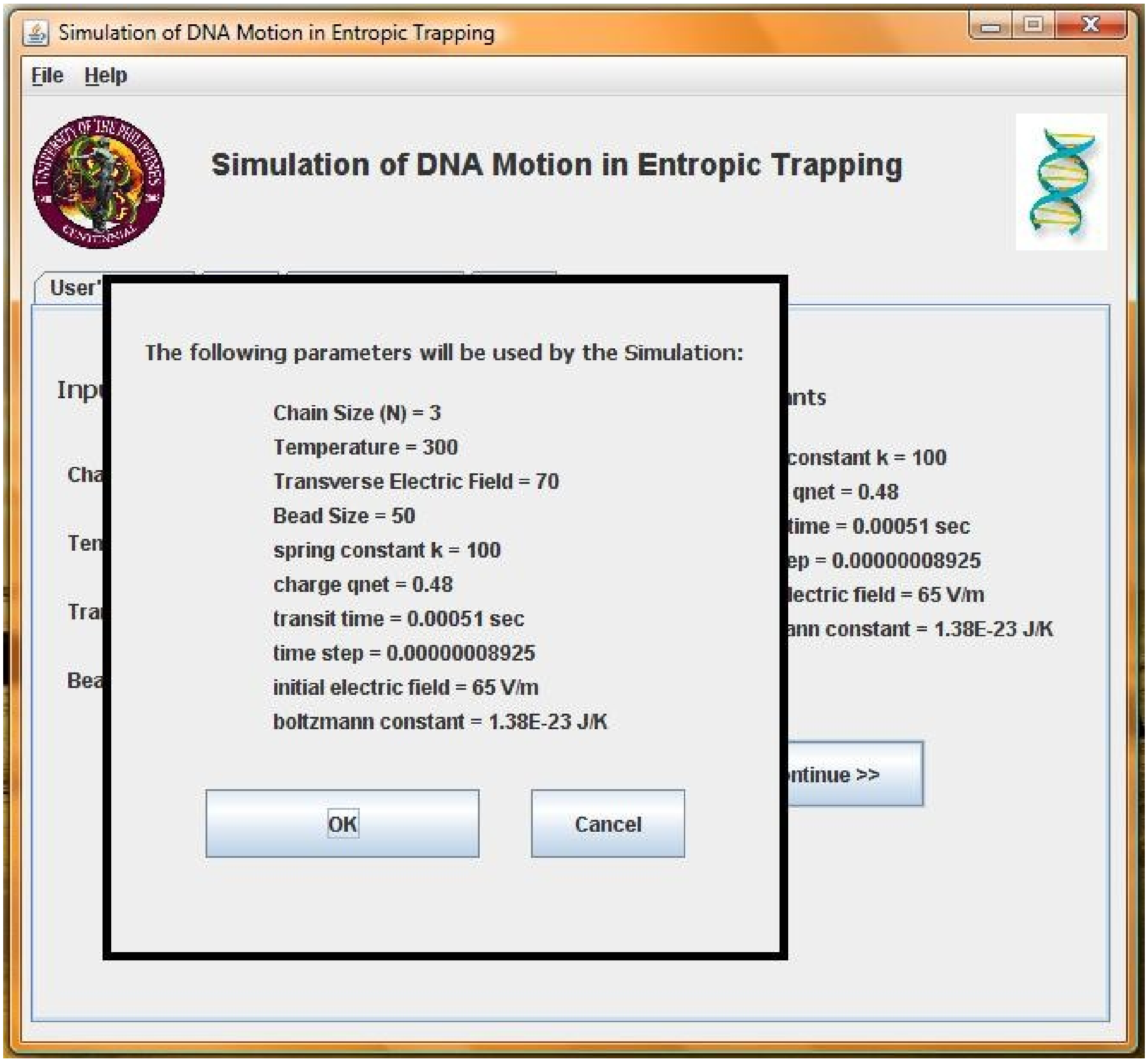
Bead Size: 50 nm

Constants

spring constant k = 100
charge qnet = 0.48
transit time = 0.00051 sec
time step = 0.00000008925
initial electric field = 65 V/m
boltzmann constant = 1.38E-23 J/K

Figure 12. Input Tab

Considering all values inputted are correct and after clicking the Continue Button, a summary of the parameters and constants is displayed to the user as shown in Figure 13.

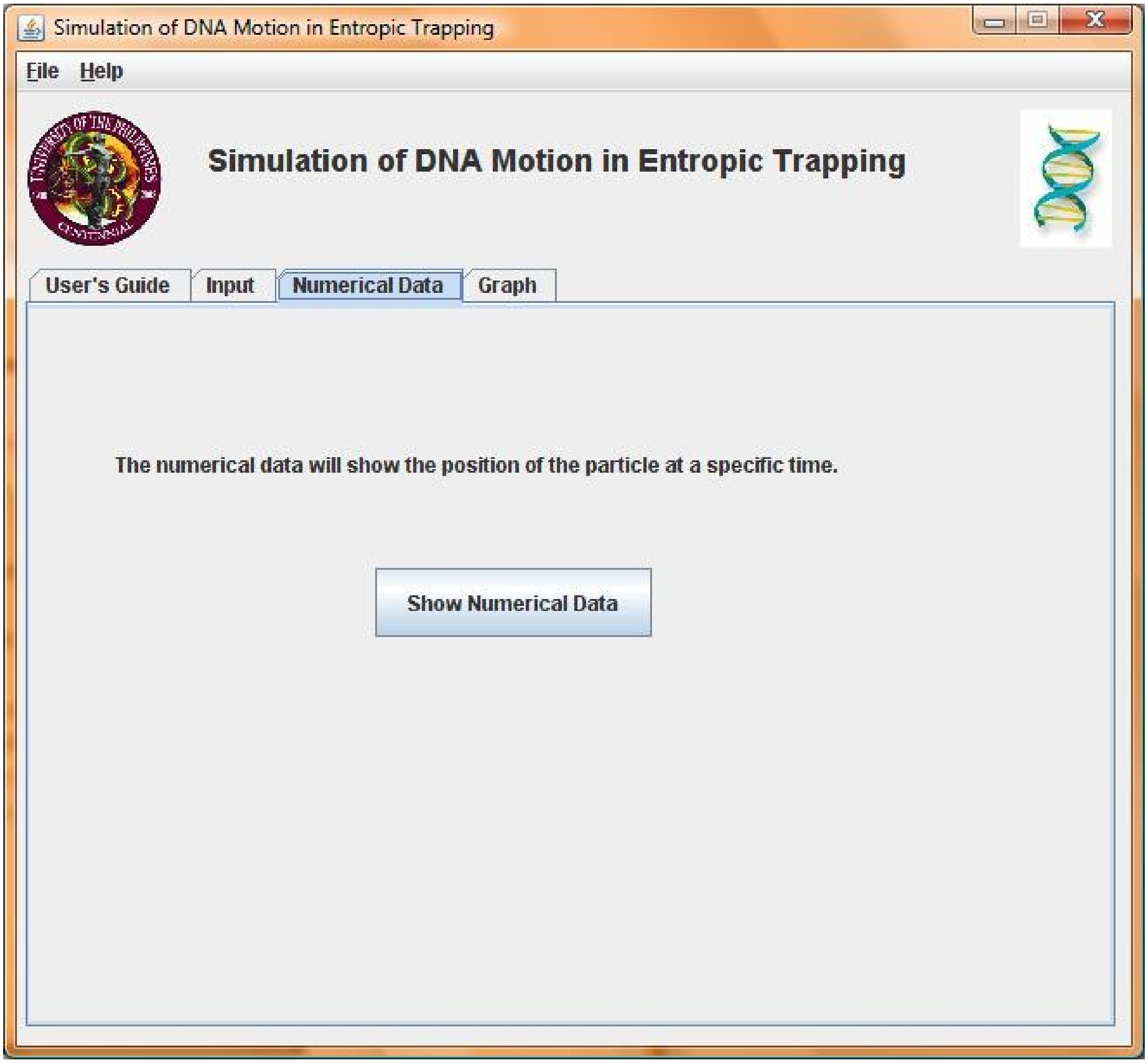


The following parameters will be used by the Simulation:

Chain Size (N) = 3
Temperature = 300
Transverse Electric Field = 70
Bead Size = 50
spring constant k = 100
charge qnet = 0.48
transit time = 0.00051 sec
time step = 0.00000008925
initial electric field = 65 V/m
boltzmann constant = 1.38E-23 J/K

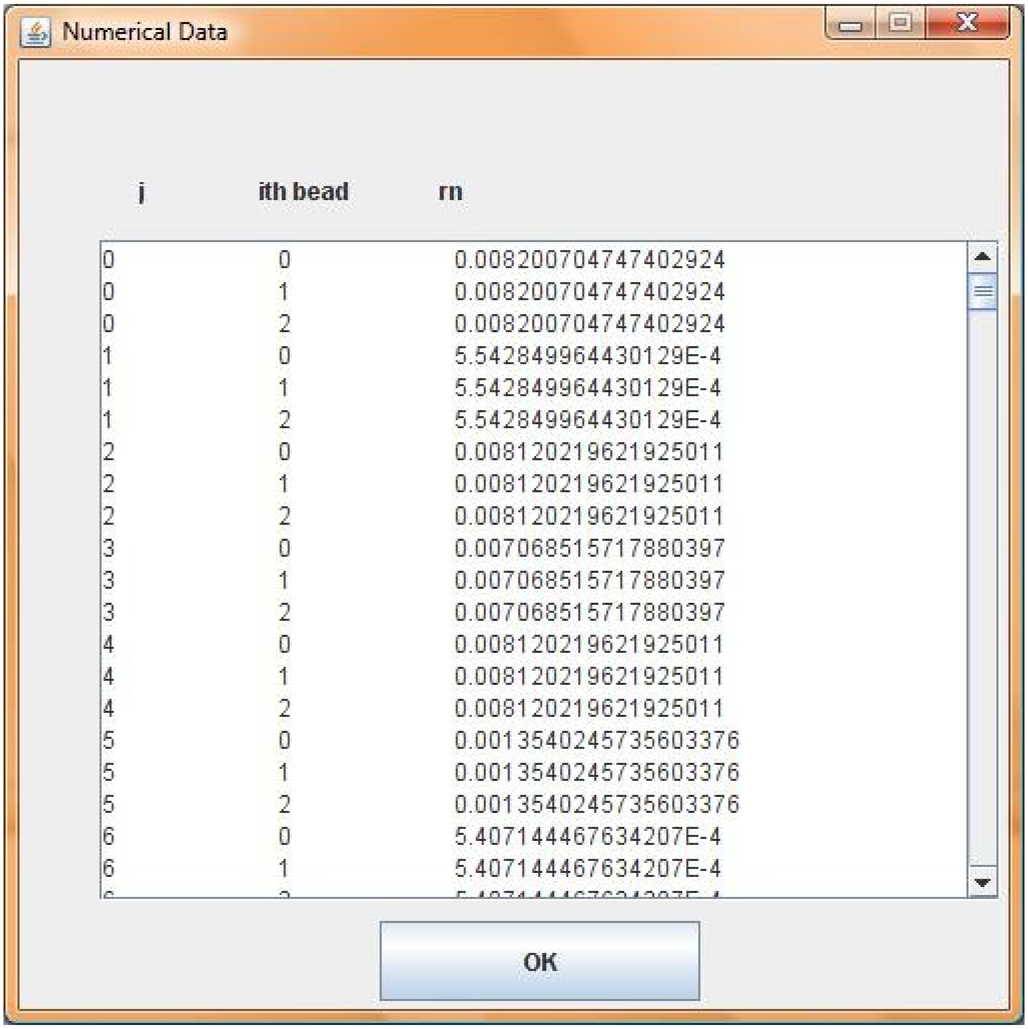
Figure 13. Summary of Parameters and Constants

The OK Button in Figure 13 leads to the Numerical Data tab where a button is found, shown in Figure 14. Clicking the Show Numerical Data Button starts the computations of the simulation and then displays the numerical data as shown in Figure 15, using the parameters—chain size =3, temperature = 100, electric field = 70 and bead size = 50.


Simulation of DNA Motion in Entropic Trapping
File
Help
Simulation of DNA Motion in Entropic Trapping
User's Guide
Input
Numerical Data
Graph
Show Numerical Data

The numerical data will show the position of the particle at a specific time.

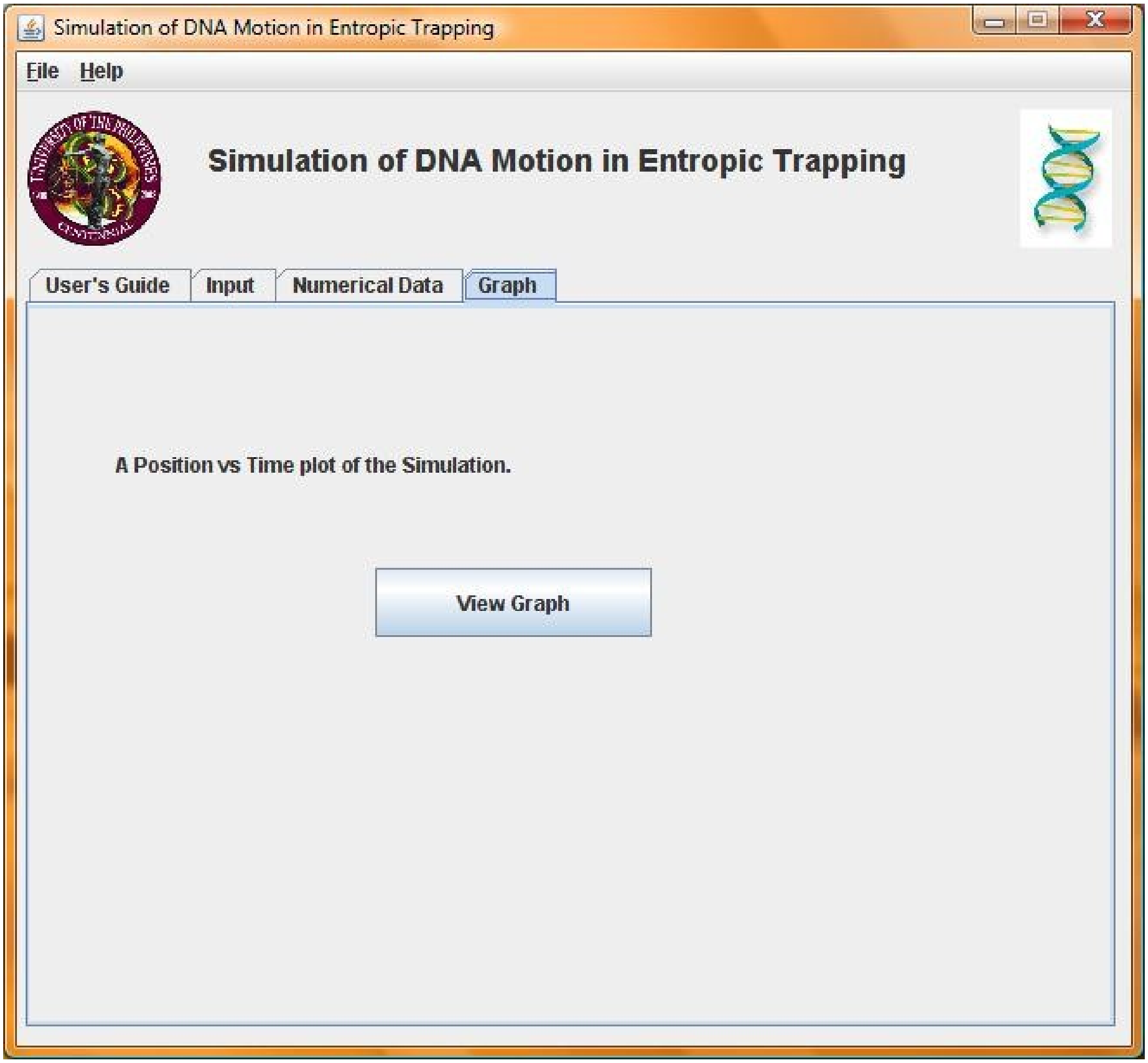
Figure 14. Numerical Data tab



j	ith bead	rn
0	0	0.008200704747402924
0	1	0.008200704747402924
0	2	0.008200704747402924
1	0	5.542849964430129E-4
1	1	5.542849964430129E-4
1	2	5.542849964430129E-4
2	0	0.008120219621925011
2	1	0.008120219621925011
2	2	0.008120219621925011
3	0	0.007068515717880397
3	1	0.007068515717880397
3	2	0.007068515717880397
4	0	0.008120219621925011
4	1	0.008120219621925011
4	2	0.008120219621925011
5	0	0.0013540245735603376
5	1	0.0013540245735603376
5	2	0.0013540245735603376
6	0	5.407144467634207E-4
6	1	5.407144467634207E-4

Figure 15. Numerical Data

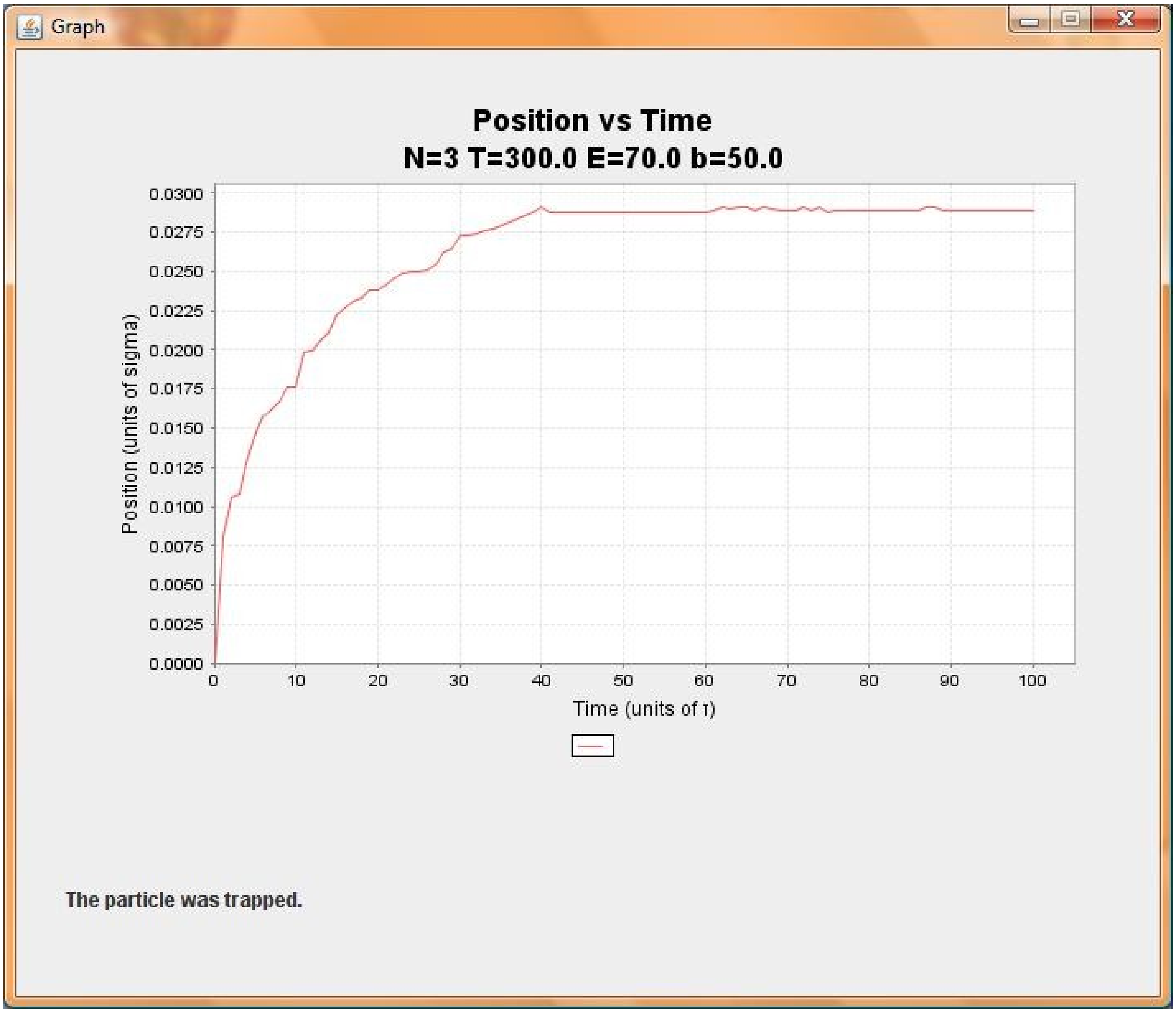
The Graph tab in Figure 16 contains a button which when clicked displays the graph resulting from the simulation..


Simulation of DNA Motion in Entropic Trapping
File
Help
Simulation of DNA Motion in Entropic Trapping
User's Guide
Input
Numerical Data
Graph
View Graph

A Position vs Time plot of the Simulation.

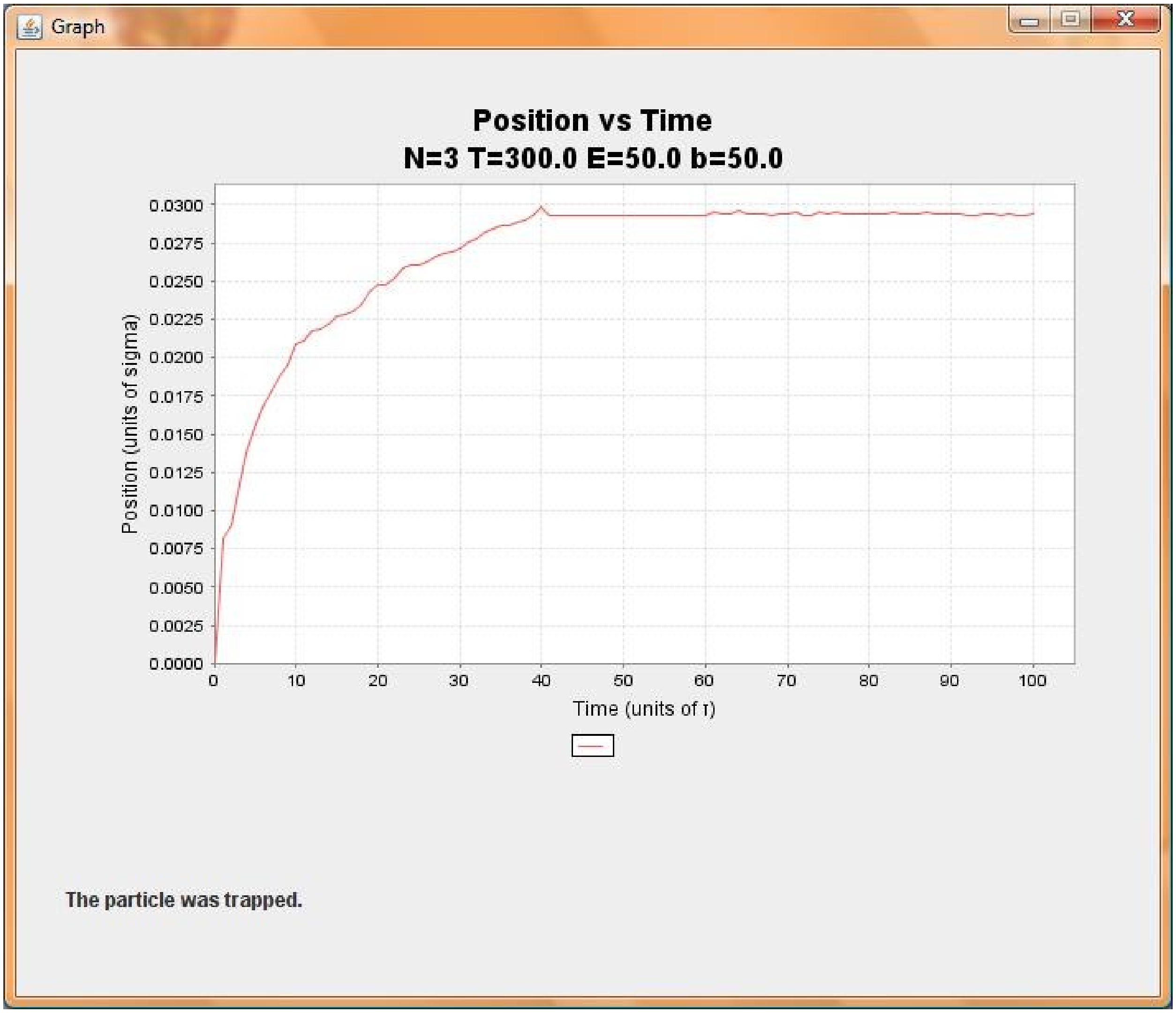
Figure 16. Graph tab

Figure 17 displays the graph when the electric field is 70 V/m. Figure 18 on the other hand displays the graph when the electric field is 50 V/m, and the last, Figure19 displays the graph when the electric field is 100V/cm



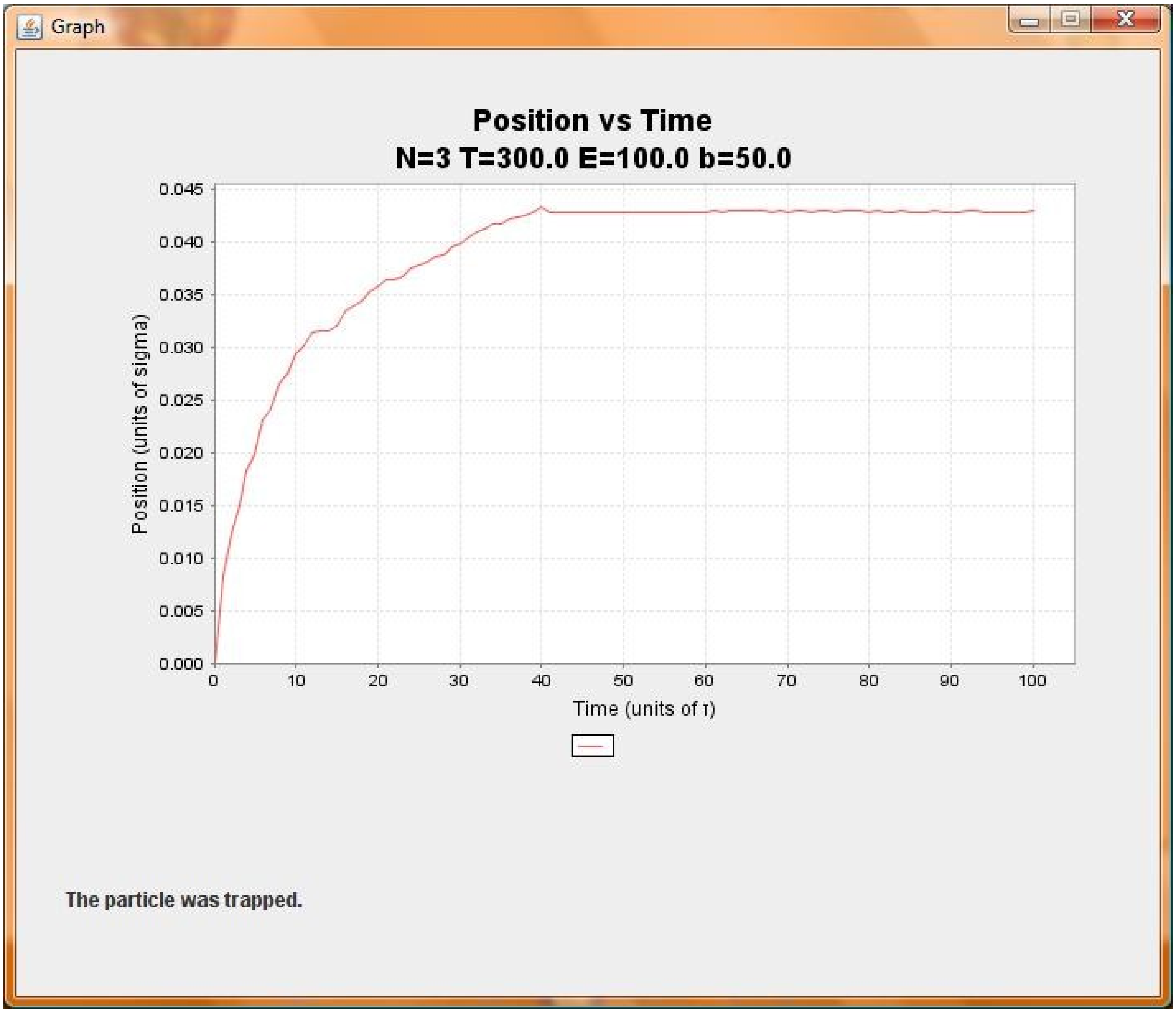
The particle was trapped.

Figure 17. Sample Graphical output, E = 70 V/m



The particle was trapped.

Figure 18. Graph, E = 50 V/cm

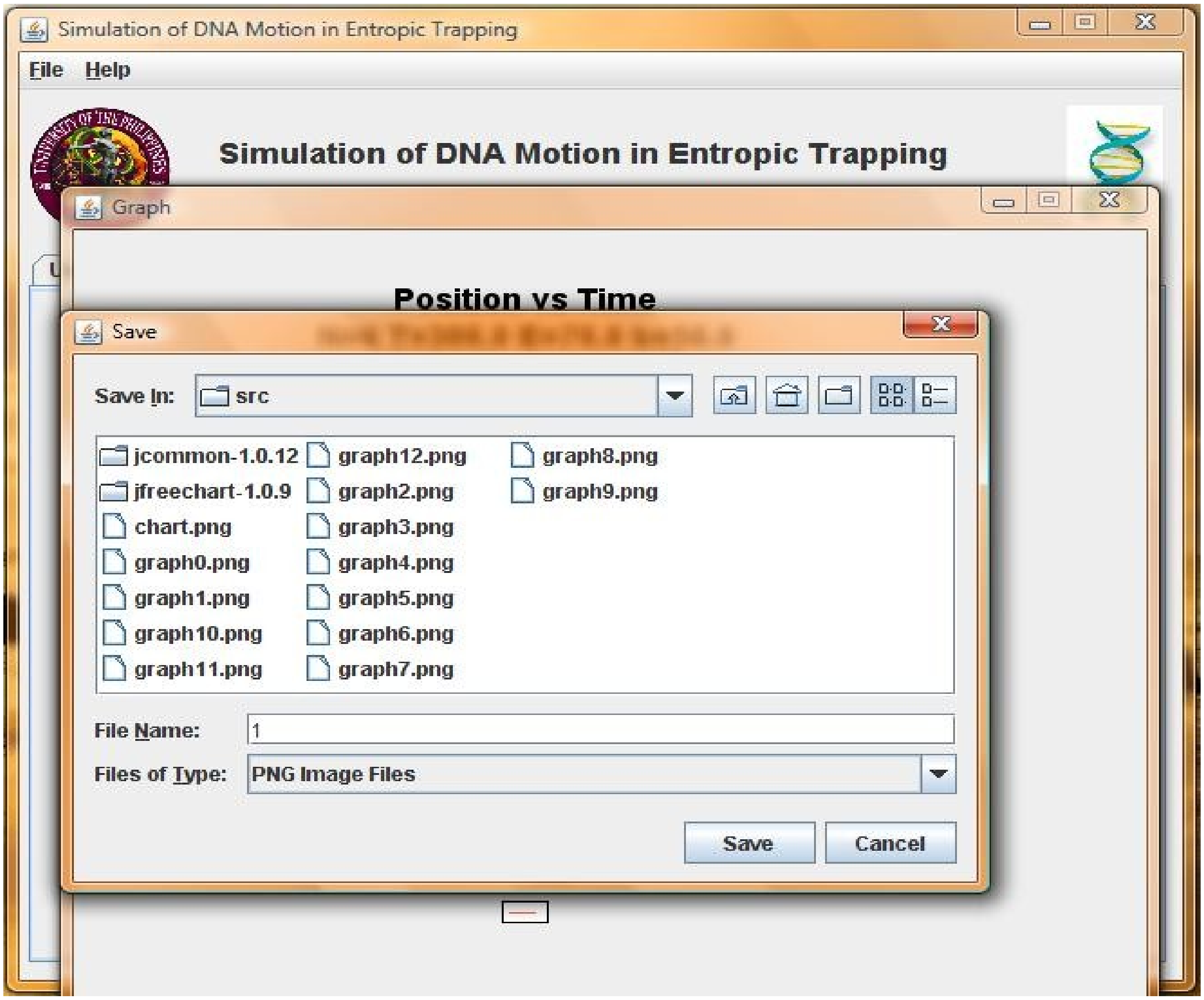


The particle was trapped.

Figure 19. Graph, E = 100 V/m

Based from the results, molecules are trapped faster when the electric field is increased. The plots of those with higher electric field have steeper slopes implying faster movement of the molecules, but were also trapped.

Right- clicking the graph gives you the option to save a graph in a particular directory, shown in Figure 20.



Save In: src

File Name: 1

Files of Type: PNG Image Files

Figure 20. Save Graph

Figures 19 and 20 show the Menu Items under File and Help respectively in the Menu Bar.

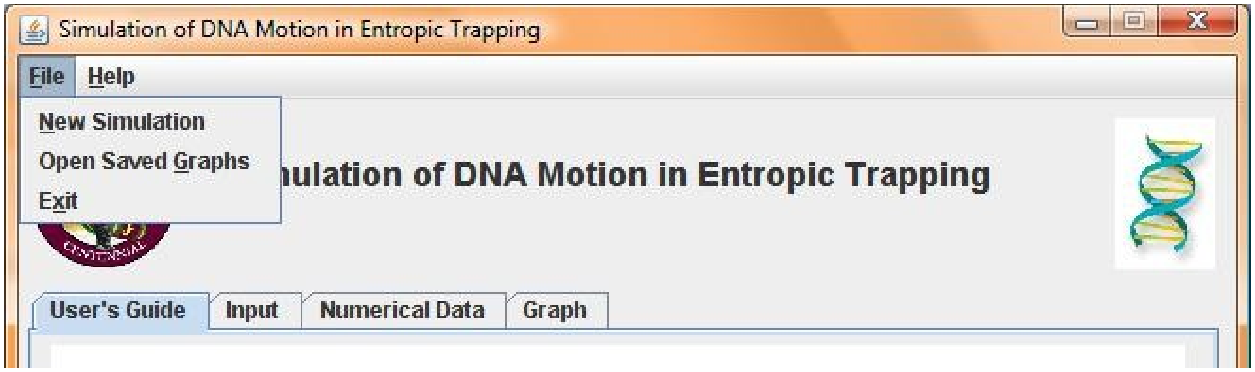

Simulation of DNA Motion in Entropic Trapping
File
Help
New Simulation
Open Saved Graphs
Exit
ulation of DNA Motion in Entropic Trapping
User's Guide
Input
Numerical Data
Graph

Figure 21. File Menu

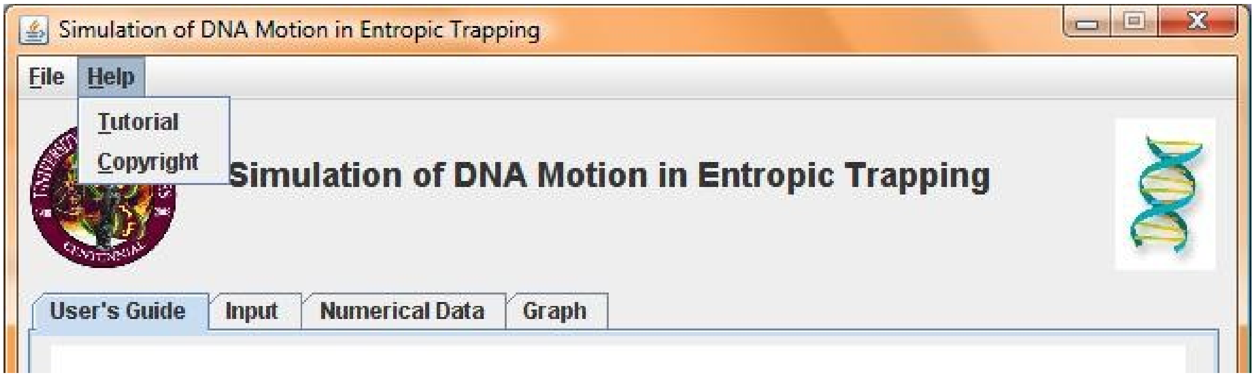
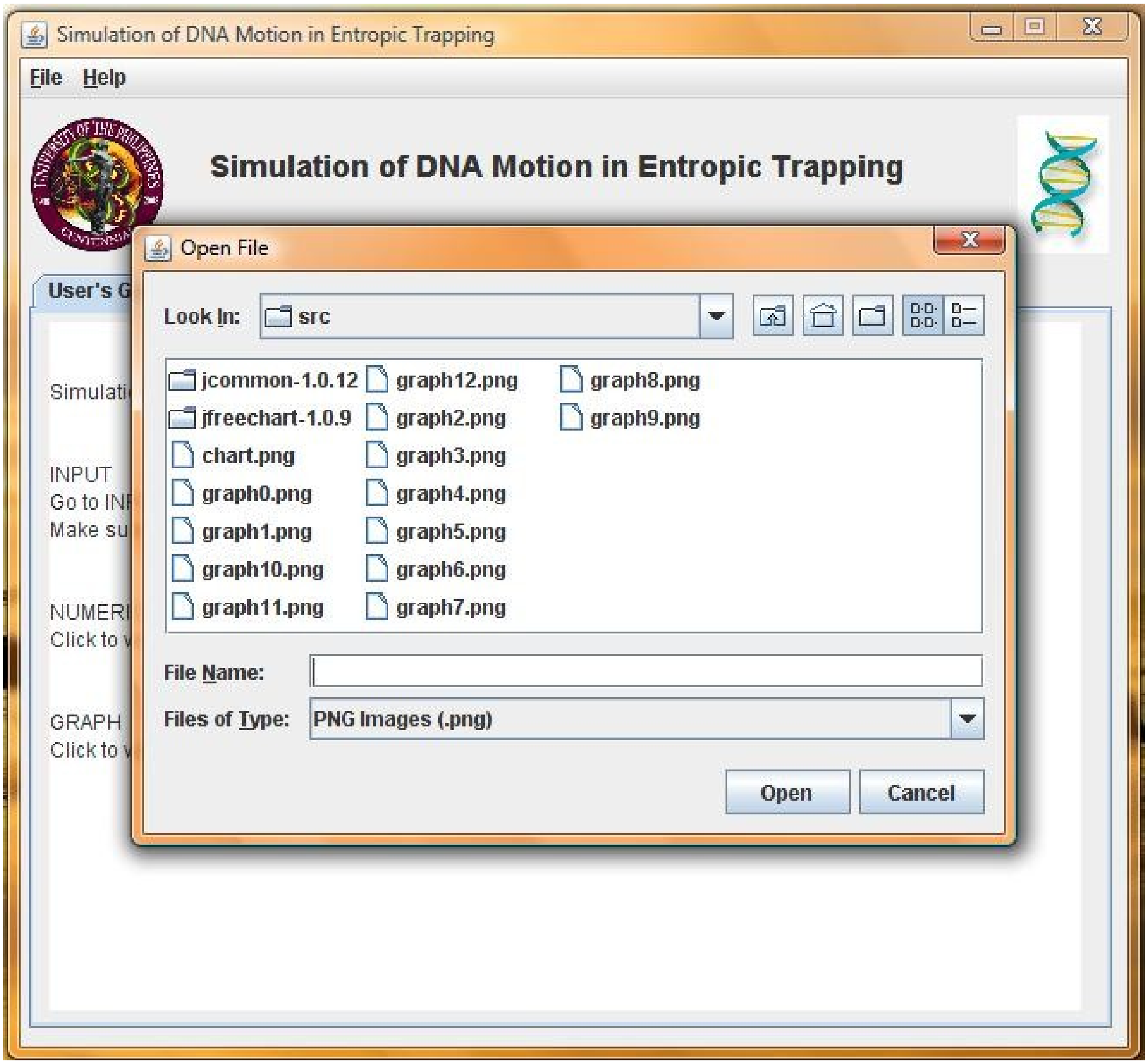

Simulation of DNA Motion in Entropic Trapping
File
Help
Tutorial
Copyright
Simulation of DNA Motion in Entropic Trapping
User's Guide
Input
Numerical Data
Graph

Figure 22. Help Menu

In the File Menu, the user can Open Saved Graphs as shown in Figure 21. The File Chooser allows the user to select and view graphs from previous simulations. The selected file will then be displayed in a separate frame.



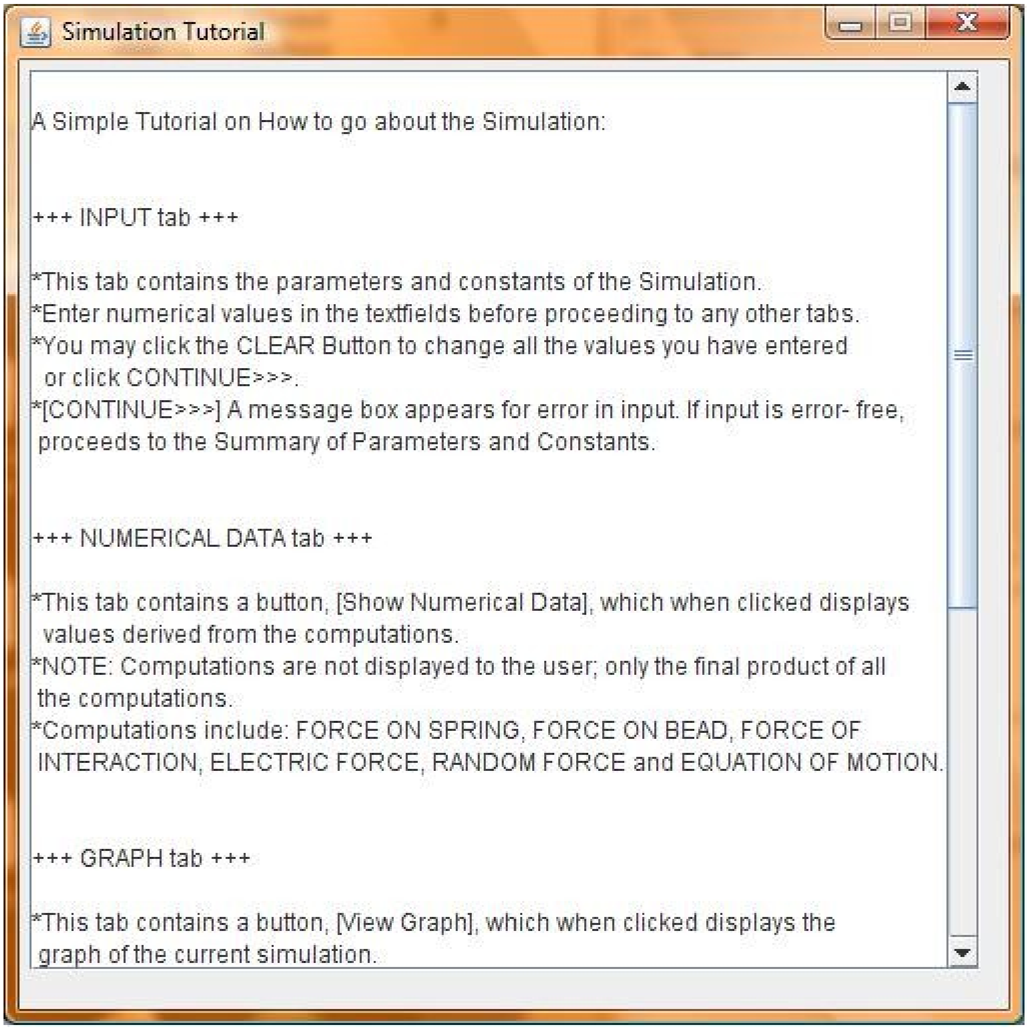
Look In: src

File Name:

Files of Type: PNG Images (.png)

Figure 23. Open Saved Graph

Under the Help Menu, a Simple Tutorial of the Simulation is available as shown in Figure 22. It contains basic information on what each tab in the simulation has and does.



A Simple Tutorial on How to go about the Simulation:

+++ INPUT tab +++

*This tab contains the parameters and constants of the Simulation.
*Enter numerical values in the textfields before proceeding to any other tabs.
*You may click the CLEAR Button to change all the values you have entered
 or click CONTINUE>>>.
*[CONTINUE>>>] A message box appears for error in input. If input is error- free,
 proceeds to the Summary of Parameters and Constants.

+++ NUMERICAL DATA tab +++

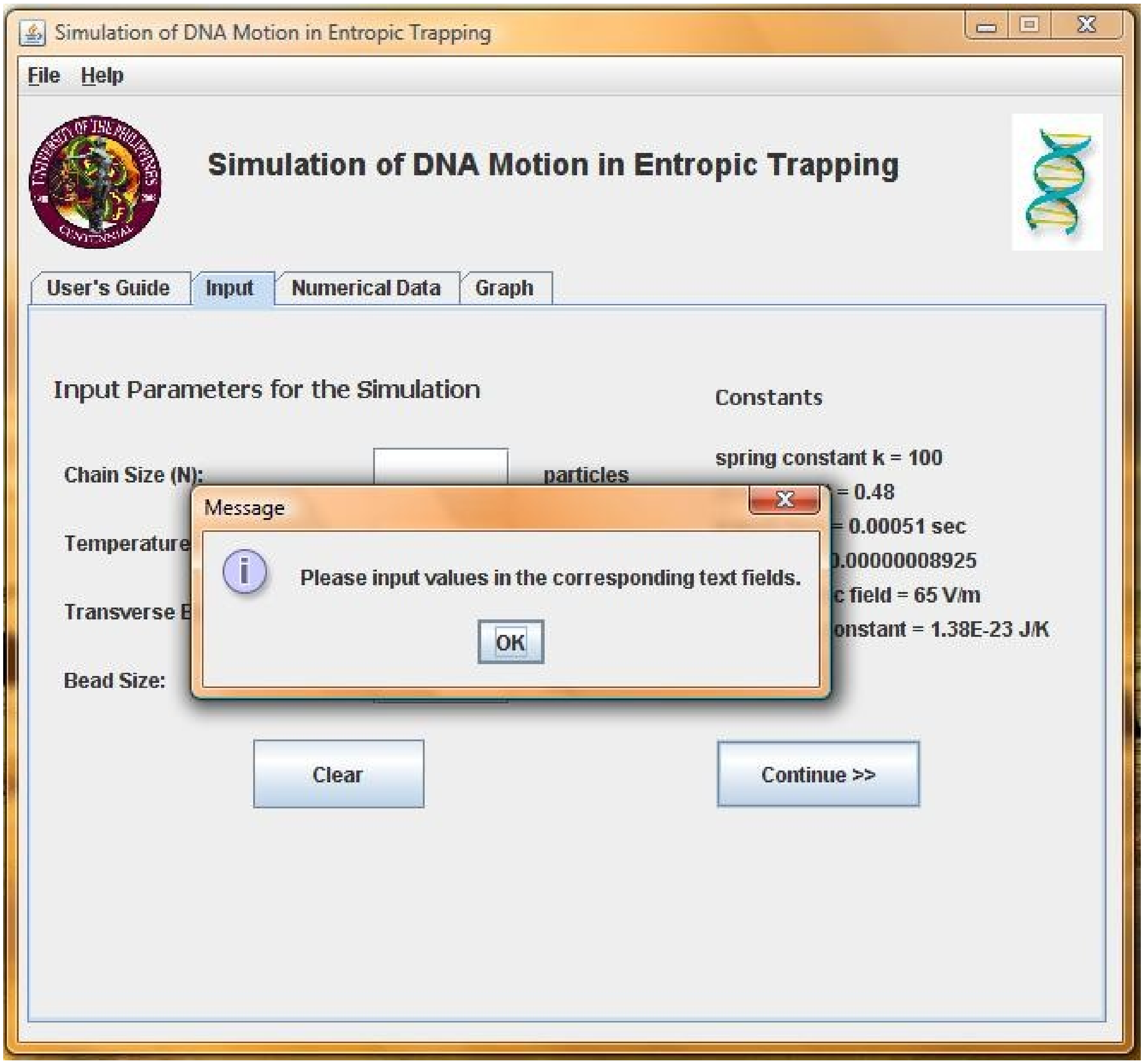
*This tab contains a button, [Show Numerical Data], which when clicked displays
 values derived from the computations.
*NOTE: Computations are not displayed to the user; only the final product of all
 the computations.
*Computations include: FORCE ON SPRING, FORCE ON BEAD, FORCE OF
 INTERACTION, ELECTRIC FORCE, RANDOM FORCE and EQUATION OF MOTION.

+++ GRAPH tab +++

*This tab contains a button, [View Graph], which when clicked displays the
 graph of the current simulation.

Figure 24. Simple Tutorial

There is also error- checking, so when an action is invalid, a message is prompted to the user, like in Figure 23, to report the incorrect input. Figure 24 displays a message that there is no input yet, so no numerical data can be displayed. Same as with Figure 25, since there is no input, there is nothing yet to graph. It is advised to input values first in the Input tab before clicking on the other tabs.



Simulation of DNA Motion in Entropic Trapping

Input Parameters for the Simulation

Chain Size (N): particles

Temperature

Transverse E

Bead Size:

Constants

spring constant k = 100

= 0.48

= 0.00051 sec

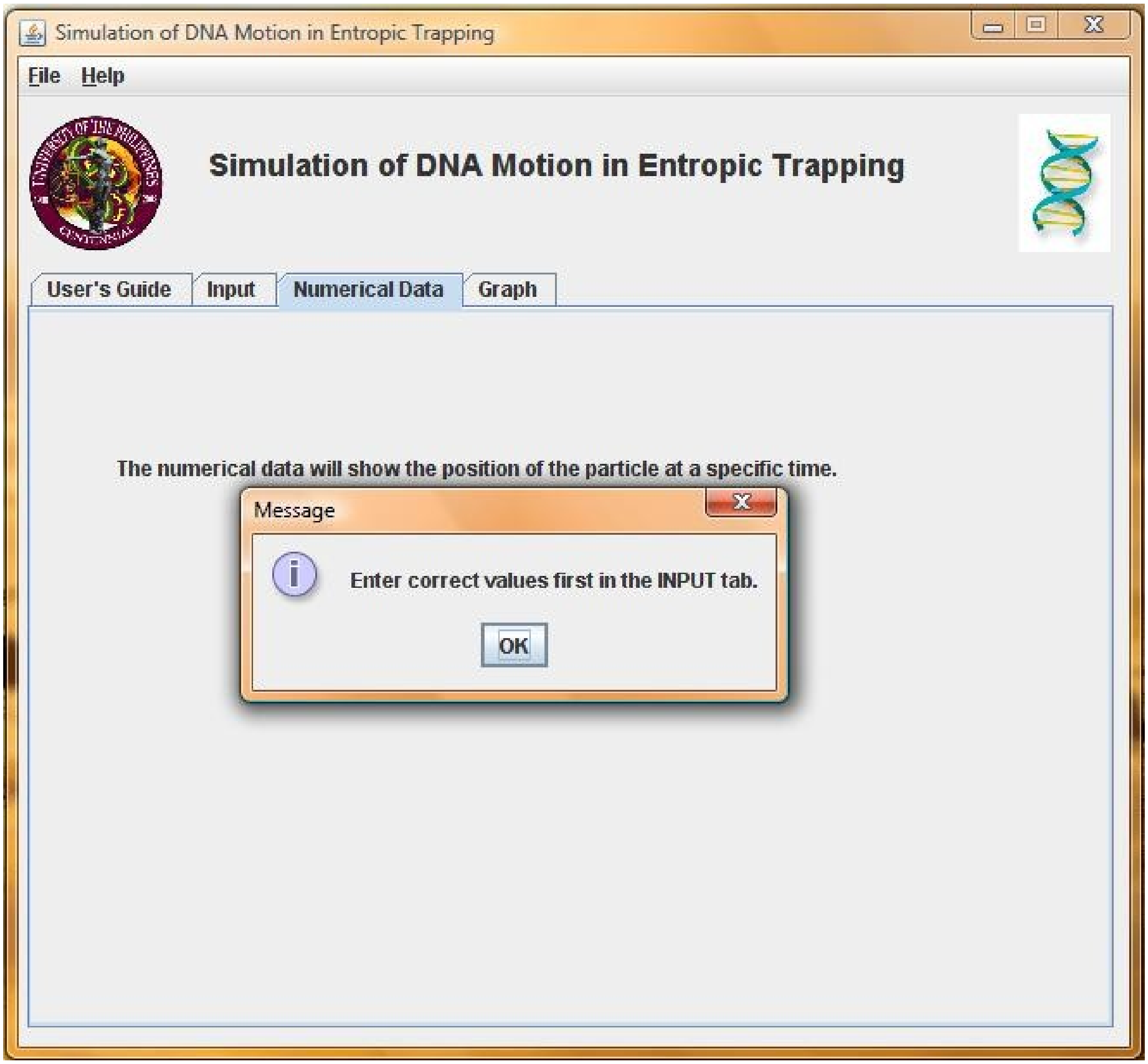
.00000008925

c field = 65 V/m

onstant = 1.38E-23 J/K

Please input values in the corresponding text fields.

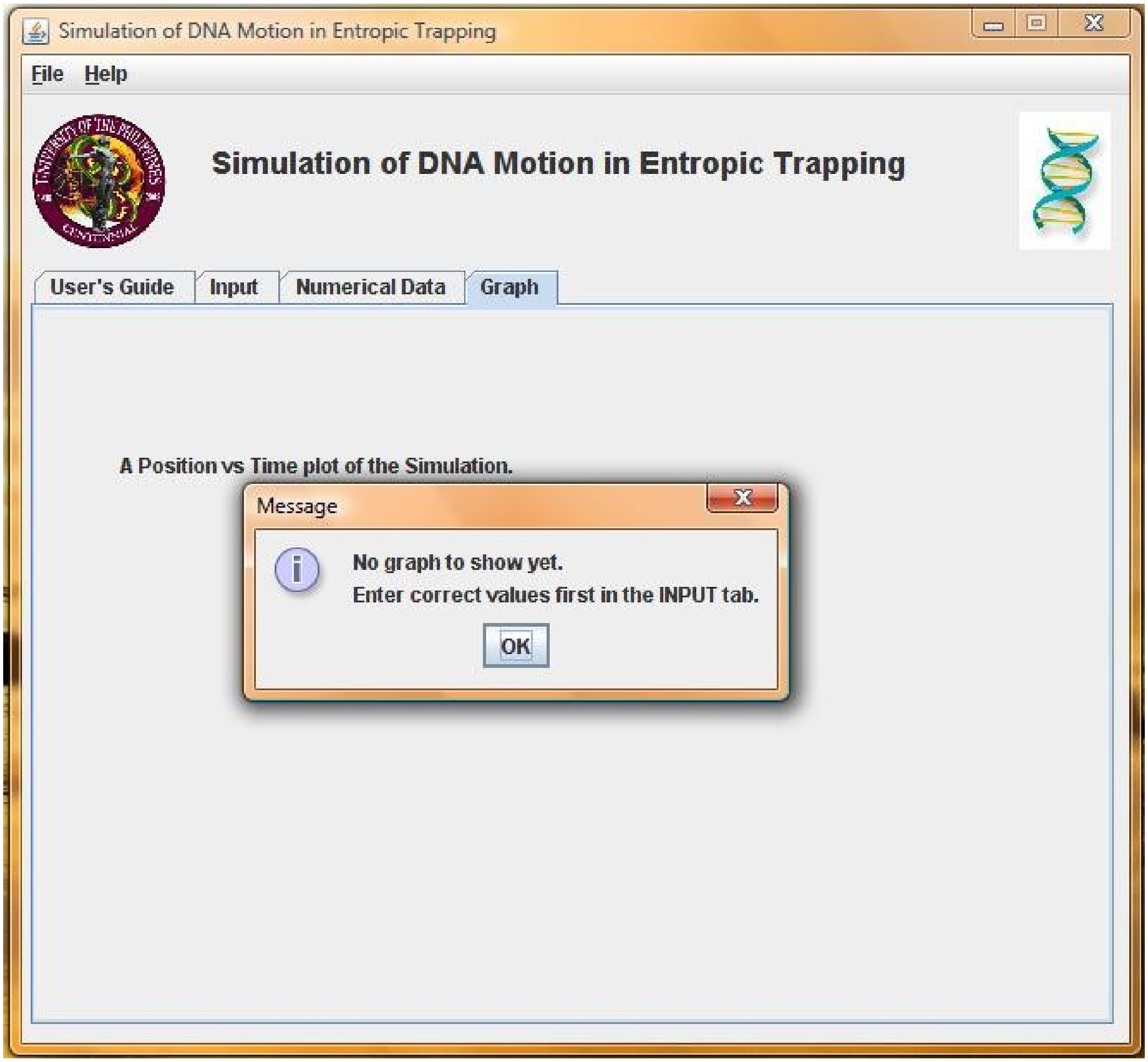
Figure 25. Error Message for input checking


Simulation of DNA Motion in Entropic Trapping
File
Help
Simulation of DNA Motion in Entropic Trapping
User's Guide
Input
Numerical Data
Graph
Message
OK

The numerical data will show the position of the particle at a specific time.

Enter correct values first in the INPUT tab.

Figure 24. Error message in Numerical Data tab when there is no input yet



Simulation of DNA Motion in Entropic Trapping

A Position vs Time plot of the Simulation.

No graph to show yet.
Enter correct values first in the INPUT tab.

Figure 26 Error message in Graph tab when there is no input yet

VI. DISCUSSION

The Simulation of DNA Motion in Entropic Trapping is a stand- alone application that computes for the Force of Interaction, Electric Force, Random Force, and the Equation of Motion. It also graphs the results from the computations of the Equation of Motion and allows the user to save these graphs for future references. This application is written using the Java TM Programming Language and uses JFreeChart for graphing purposes.

The user can easily predict the possible position of a particle with this application without actually doing it in the laboratories. In real- life experiments on entropic trapping, you need certain devices like the entropic traps and an electric field source, which may not be always available. Using this application virtually provides the user with devices needed for an experiment. In addition, the user can adjust the values of the electric field and temperature among others allowing the user to obtain various results and a basis for future comparisons.

All the computations are done by the system to provide the user with additional data just in case further analysis is desired. At the end of the computations, a position versus time graph is generated for the user. The graph provides the user with a better picture of what happened in the simulation. It summarizes the results of the computations in a graphical way. The flat lines in the graph indicate that there is no mobility, implying the trapping of the particle, while sloping lines indicate mobility.

The system uses the Fourth Order Runge Kutta Algorithm to integrate the Equation of Motion. This method approximates the next position of the particle within a particular time step. RK4 is an improved version of the Second Order Runge Kutta for it makes use of smaller time steps, giving a better approximation of the next position.

However, the system may take up a lot of resources because of the variety of computations involved. Since there are many computations, a lot of memory has to be allocated to accommodate all the needed functionalities.

Displaying the numerical results takes time due to the number of computations involved. The values entered at the start of the simulation go through the computations for the force of the spring, force of the bead, force of interaction, electric force, random force, then eventually to the integration of the equation of motion. Displaying the graph takes time as well, also due to the computations involved.

The result of the simulation depends on the range of the user's input. Entering very large values may cause problems in the computation especially in the handling of values by the data types. The user must input sensible values which can be applied in real experiments.

Most of the simulations available today are concerned with Gel Electrophoresis. On the other hand, this simulation is concerned with Electrophoresis in an Entropic Trapping environment. This will be useful to better understand other DNA separation techniques since it is an uncommon method.

VII. CONCLUSION

The Computer Simulation of DNA Motion in Entropic Trapping is a standalone application that allows users to enter the parameters needed for the simulation, particularly the temperature, electric field, and bead size.

The simulation performs computations based on the inputs of the user and some constant values. These computations include: Force of Interaction (Force on the Spring and Force on the Bead), Electric Force, Random Force and the integration of the Equation of Motion using the Fourth Order Runge Kutta.

It outputs the results of the simulation particularly the numerical data containing the value of the position of the ith bead and the graphical data of the position of the ith bead with respect to time. The graphical data resulting from the simulation is presented in a PNG image file.

VIII. RECOMMENDATION

The Simulation of DNA Motion in Entropic Trapping can be further improved by knowing the range of values that can be entered in the input text fields. This will give the assurance that the simulation results can really happen in real life. It is also recommended that the simulation may accept other chain size values aside from the current chain size of three. By allowing the user to enter different chain sizes, the user can try more combinations of the different parameters and observe for any trends, if possible.

Other Numerical Methods may also be considered in the integration of the Equation of Motion. Any algorithm may be used for the computation but it must be the most accurate and at the same time uses shorter time to implement. The data types used for the simulation may also be altered by finding more applicable data types to handle big numbers other than the Double.

Finally, this simulation may also be improved by adding animation or graphics showing the real-time traversal of the DNA in the channel.

IX. BIBLIOGRAPHY

[1] Sheehan, D. (1988). *Physical Biochemistry: Principles and Applications*.

[2] Nagahiro, S., Kawano, S., & Kotera, H. (2007 January). Separation of long DNA chains using a nonuniform electric field: A numerical study. *Physical Review, E 75*.

[3] García De La Torre, J., Freire, J. & Horta, A. (1975 July). *A bead and spring model for the stiffness of DNA.* Madrid, Spain from http://www3.interscience.wiley.com/cgi-bin/abstract/107587646/ABSTRACT

[4] Wan, H. & Thompson, R. (2005 June). *Capillary electrophoresis technologies for screening in drug discovery.* Molndal, Sweden. Science Direct Available online 22 June 2005 from http://www.sciencedirect.com/science?_ob=ArticleURL&_udi=B75D6-4GFV5S6-2&_user=10&_origUdi=B6T64-4F035WH-B&_fmt=high&_coverDate=08%2F31%2F2005&_rdoc=1&_orig=article&_acct=C000050221&_version=1&_urlVersion=0&_userid=10&md5=f7138da8dcbfc588763f7ede847e72ac

[5] Han, J., Turner, S. & Craighead, H. (1999). Entropic Trapping and Escape of Long DNA Molecules at Submicron Size Constriction. *Cornell University, School of Applied and Engineering Physics, Ithaca, New York 14853*
(Received 19 February 1999)

[6] Dorfman K.D.[1]; Brenner H.[2] (2002 July). *Modeling DNA Electrophoresis in Microfluidic Entropic Trapping Devices.* Biomedical Microdevices: Springer, Volume 4, Number 3, July 2002 , pp. 237-244(8)

[7] Streek, M., Schmid, F., Duong, T. & Ros, A. (2004 May). *Mechanisms of DNA separation in entropic trap arrays: A Brownian dynamics simulation.* Bielefeld, Germany.

[8] Streek, A. (2005). *Brownian Dynamics Simulation of Migration of DNA in structured Microchannels.* (Dissertation, University of Bielefeld).

[9] Han, J. & Craighead, H. (2000). *Separation of Long DNA Molecules in a Macrofabricated Entropic Trap Array.* Science.

[10] Russell, R. (2001 May). *Gel Electrophoresis* from

http://www.msu.edu/~russellr/portfolio/electrophoresis/electrophoresis.html

[11] Bowen, R. (2000). *Virtual Lab: Agarose Gel Electrophoresis of Restriction Fragments* from

http://www.vivo.colostate.edu/hbooks/genetics/biotech/gels/virgel.html

[12] Kim, C. & Konagaya, A. *The 2D Gel Electrophoresis Simulator: SimGel*. School of Knowledge Science, Japan Advanced Institute of Science and Technology, Japan from http://recomb2000.ims.u-tokyo.ac.jp/Posters/pdf/37.pdf

[13] Christian, W. *Physlets* from http://webphysics.davidson.edu/Applets/Applets.html

[14] Harvard Apparatus. (2004). *Gel Explorer* from http://www.harvardapparatus.com/webapp/wcs/stores/servlet/product_11051_10001_52355_-1_HAI_ProductDetail_37698__N#fulldescriptionitemlistingtab

[15] Gas, B., et. al., *PeakMaster* from http://natur.cuni.cz/~gas/

[16] Tietz, D. (1995). *ElphoFit*. National Institutes of Health, Maryland, USA from http://www.his.com/~djt/fergusonlit.html

[17] Nucleic Assays Corporation. (2003). *GelSite* Electrophoresis Analysis Software from http://www.nucleicassays.com/gelsite/

[18] Kapelan BioImaging Solutions. (2006). *LabImage1D* from http://www.labimage.net/gel_analysis/1d_analysis_software/features.html

[19] Silk Scientific Inc. *Un-Scan-It Gel Quantifying Software* from http://www.silkscientific.com/usiginfo.htm

[20] Wikipedia. *Computer Simulation* from http://en.wikipedia.org/wiki/Computer_simulation

[21] Underhill, P. & Doyle, P. (2005). Brownian Dynamics Simulations of Polymers and Soft Matter from http://www.springerlink.com/content/wr800x172485l505/

[22] Underhill, P. & Doyle, P. (2006 April). *Alternative spring force law for bead- spring chain models of the worm- like chain.* Department of Chemical Engineering, Massachusetts Institute of Technology, Cambridge, Massachusetts.

[23] My Physics Lab. *Runge Kutta Algorithm* from http://www.myphysicslab.com/runge_kutta.html

[24] Wikipedia. *Graphical User Interface* from http://en.wikipedia.org/wiki/Graphical_user_interface

X. APPENDIX

Source Code

```
/*SP.java*/

import java.awt.*;
import java.awt.event.*;
import java.awt.Window;
import java.awt.Component;
import javax.swing.*;
import javax.imageio.*;
import javax.swing.filechooser.*;
import java.util.*;
import java.util.Random.*;
import java.text.*;
import java.io.*;
import java.lang.*;
import java.lang.Math.*;
import java.awt.image.*;
import java.text.DecimalFormat;
import java.util.Hashtable;
import javax.swing.border.*;

import java.awt.BorderLayout;
import java.io.File;
import javax.swing.ImageIcon;

import org.jfree.chart.*;
import org.jfree.chart.plot.PlotOrientation;
import org.jfree.data.category.*;
import
org.jfree.data.general.DefaultPieDataset;
import org.jfree.data.xy.*;
import org.jfree.data.*;
import org.jfree.chart.ChartUtilities;


 public class SP extends JFrame{
        private JLabel inputParametersLabel;
        private JLabel chainSizeLabel;
        private JLabel temperatureLabel;
        private JLabel electricFieldLabel;
        private JLabel beadSizeLabel;
        private JLabel constantsLabel;
        private JLabel particlesLabel;
        private JLabel temperatureUnitLabel;
        private JLabel
electricFieldUnitLabel;
        private JLabel beadSizeUnitLabel;
        private JLabel springConstantLabel;
        private JLabel chargeLabel;
        private JLabel transitTimeLabel;
        private JLabel timeStepLabel;
        private JLabel
initialElectricFieldLabel;
        private JLabel boltzmannLabel;
        private JLabel UPlogo, DNAlogo;
        private JTextField
chainSizeTextField;
        private JTextField
temperatureTextField;
        private JTextField
electricFieldTextField;
        private JTextField beadSizeTextField;
        private JTextArea paramsTextArea;
        private JTextArea
computationsTextArea;
        private JScrollPane paramsScrollPane;
        private JButton startSimButton;
        private JButton openSavedGraphButton;
        private JButton continueButton;
        private JButton clearButton;
        private JButton
showNumericalDataButton;
        private JPanel contentPane,
contentPane2;
        private ButtonHandler handler;
        private JMenuBar bar;
        private JMenu fileMenu;
        private JMenu helpMenu;
        private JMenuItem newSimulationItem;
        private JMenuItem openSavedGraphItem;
        private JMenuItem exitItem;
        private JMenuItem aboutItem;
        private JMenuItem tutorialItem;
        JTextArea fTextArea;
        Hashtable hashtable;
        ArrayList Values;

        private JTabbedPane tabbedPane;
        private JPanel panel0;
        private JPanel panel1;
        private JPanel panel2;
        private JPanel panel3;
        int index0, index1, index2, index3;

        private JWindow w;

        File fFile = new File
("default.java");
```

```
        File outfile = new
File("simcount.txt");
        private JFrame frame1, frame2,
frame3, frame4;

        double k = 100;
        double qnet = (0.48*(1.6E-19)*167);
        double transit_time = 0.00051;
        double delta_t = 0.00000008925;
        double electricField_initial = 6500;
        double chainSize;
         double temperature;
        double electricField;
        double beadSize;
        double boltzmann_constant =
0.0000000000000000000000138;
        double bs,ef;
        static Random number = new Random();
        double [] fspring;
        double [] fbead;
        double [] delta_t2;
        boolean inputerr = false;
        public SP() {
                super();
                handler = new ButtonHandler();
                initializeComponent();
        }

        /*
         This method is called from within
the constructor to initialize the form.
         */
        private void initializeComponent() {

                JLabel titleLabel = new
JLabel("Simulation of DNA Motion in Entropic
Trapping");
                titleLabel.setFont(new
Font("Arial", Font.BOLD, 18));

                UPlogo = new JLabel();
                ImageIcon logo = new
ImageIcon("uplogo.gif");
                DNAlogo = new JLabel();
                ImageIcon logo2 = new
ImageIcon("dnalogo.gif");
                bar = new JMenuBar();
                fileMenu = new JMenu();
                helpMenu = new JMenu();

                newSimulationItem = new
JMenuItem();
                openSavedGraphItem = new
JMenuItem();
                exitItem = new JMenuItem();
                aboutItem = new JMenuItem();
                tutorialItem = new
JMenuItem();

                JPanel topPanel = new
JPanel();
                topPanel.setLayout( null );

        getContentPane().add( topPanel );

                UPlogo.setIcon(logo);
                DNAlogo.setIcon(logo2);
                fileMenu.setText("File");
                fileMenu.setMnemonic( 'F' );
        newSimulationItem.setText("New
Simulation");

        newSimulationItem.setMnemonic( 'N' );

        newSimulationItem.addActionListener(
                        new ActionListener() {
                                public void
actionPerformed( ActionEvent e ) {


newSimulationButton_actionPerformed(e);
                                }
                        }
                );

        fileMenu.add(newSimulationItem);


        openSavedGraphItem.setText("Open
Saved Graphs");


openSavedGraphItem.setMnemonic( 'G' );

        openSavedGraphItem.addActionListener(
                        new ActionListener() {
                                public void
actionPerformed( ActionEvent e ) {


openSavedGraphButton_actionPerformed(e);
                                }
                        }
                );

        fileMenu.add(openSavedGraphItem);

                exitItem.setText("Exit");
                exitItem.setMnemonic( 'x' );
                exitItem.addActionListener(
                        new ActionListener() {
                                public void
actionPerformed( ActionEvent e ) {

        System.exit( 0 );
                                }
                        }
                );
                fileMenu.add(exitItem);

                bar.add(fileMenu);

                helpMenu.setText("Help");
                helpMenu.setMnemonic( 'H' );



        tutorialItem.setText("Tutorial");
                tutorialItem.setMnemonic( 'T'
);

        tutorialItem.addActionListener(
                        new ActionListener() {
                                public void
actionPerformed( ActionEvent e ) {

        tutorial();
                                }
                        }
                );
```

```
                helpMenu.add(tutorialItem);


        aboutItem.setText("Copyright");
                aboutItem.setMnemonic( 'C' );
                aboutItem.addActionListener(
                        new ActionListener() {
                                public void
actionPerformed( ActionEvent e ) {

        about();
                                }
                        }
                );
                helpMenu.add(aboutItem);

                bar.add(helpMenu);
                setJMenuBar(bar);

                createHomePage();
                createPage1();
                createPage2();
                createPage3();

                tabbedPane = new
JTabbedPane();
                tabbedPane.addTab("User's
Guide", panel0);
                index0 =
tabbedPane.getTabCount()-1;
                tabbedPane.addTab( "Input",
panel1 );
                index1 =
tabbedPane.getTabCount()-1;
                tabbedPane.addTab( "Numerical
Data", panel2 );
                index2 =
tabbedPane.getTabCount()-1;
                tabbedPane.addTab( "Graph",
panel3 );
                index3 =
tabbedPane.getTabCount()-1;

                addComponent(topPanel,
UPlogo, 5, 0, 100, 100);
                addComponent(topPanel,
titleLabel, 110, 30, 450, 20);
                addComponent(topPanel,
DNAlogo, 580, 0, 80, 100);
                addComponent(topPanel,
tabbedPane, 5, 100, 630,440);

                this.setTitle("Simulation of
DNA Motion in Entropic Trapping");
                this.setLocation(new
Point(350, 50));
                this.setSize(new
Dimension(650, 600));


this.setDefaultCloseOperation(WindowConstant
s.EXIT_ON_CLOSE);
                this.setResizable(false);

        }//end of initializeComponent

        /** Add Component - Absolute
Positioning */
        private void addComponent(Container
container,Component c,int x,int y,int
width,int height) {
        c.setBounds(x,y,width,height);
                container.add(c);
        }

        /*Event Handling Methods*/

        private void 
chainSizeTextField_actionPerformed(ActionEve
nt e) {

        }

        private void 
temperatureTextField_actionPerformed(ActionE
vent e) {

        }

        private void 
electricFieldTextField_actionPerformed(Actio
nEvent e) {

        }

        private void 
beadSizeTextField_actionPerformed(ActionEven
t e) {

        }

        private void 
openSavedGraphButton_actionPerformed(ActionE
vent e) {

                boolean status = false;
                status = openFile ();
                if (!status){

        JOptionPane.showMessageDialog (null, 
"Error opening file!", "File Open Error", 
JOptionPane.ERROR_MESSAGE);
                }
        }

        boolean openFile(){

                JFileChooser chooser = new 
JFileChooser();
                chooser.setDialogTitle("Open 
File");


chooser.setFileSelectionMode(JFileChooser.FI
LES_ONLY);

        chooser.setCurrentDirectory(new 
File("."));

                ExampleFileFilter filter = 
new ExampleFileFilter();
                filter.addExtension("png");
                filter.setDescription("PNG 
Images");
                chooser.setFileFilter(filter);

                int returnVal = 
chooser.showOpenDialog(this);
                if(returnVal == 
JFileChooser.APPROVE_OPTION) {
                        fFile = 
chooser.getSelectedFile ();

                        Image image = null;
                        try {
                                File file = 
new File(fFile.getName());
                                image = 
ImageIO.read(file);
                        } catch (IOException 
ioe) {}

                        JFrame graphframe = 
new JFrame();
                        JLabel imglabel = new 
JLabel(new ImageIcon(image));


graphframe.getContentPane().add(imglabel, 
BorderLayout.CENTER);
                        graphframe.pack();

        graphframe.setVisible(true);
                }

                else if (returnVal == 
JFileChooser.CANCEL_OPTION) {
                        return true;
                }

                else {
                        return false;
                }

                return true;
         } // openFile

        private void 
continueButton_actionPerformed(ActionEvent 
e) {

                        summary();

        }

        private void 
clearButton_actionPerformed(ActionEvent e) {


        chainSizeTextField.setText("3");

        temperatureTextField.setText("");

        electricFieldTextField.setText("");
                beadSizeTextField.setText("");
        }

        private void 
backButton_actionPerformed(ActionEvent e) {

                back();

        }

        private void 
runButton_actionPerformed(ActionEvent e) {

                w.setVisible(false);
                setEnabled(true);
                tabbedPane.setEnabledAt(2, 
true);
```

```
                try{


tabbedPane.setSelectedComponent(panel2);

        tabbedPane.setEnabledAt(2, true);
                        createPage2();

                }catch(IllegalArgumentExcepti
on iae){}
        }

        private void
exitButton_actionPerformed(ActionEvent e) {

                System.exit(0);

        }

        void back(){

                w.setVisible(false);
                this.setVisible(true);
                setEnabled(true);
        }

        private void
viewGraphButton_actionPerformed(ActionEvent
e) {


        frame3.setVisible(false);
                        try{


tabbedPane.setSelectedComponent(panel3);
                        createPage3();
                        }catch(IllegalArgument
Exception iae){}

        }


  /** Use a BufferedReader wrapped around a
FileReader to read
    * the text data from the given file.
   **/
        public String readFile (File file) {

                StringBuffer fileBuffer;
                String fileString=null;
                String line;

                try {
                        FileReader in = new
FileReader (file);
                        BufferedReader dis =
new BufferedReader (in);
                        fileBuffer = new
StringBuffer () ;

                        while ((line =
dis.readLine ()) != null) {

        fileBuffer.append (line + "\n");
                        }

                        in.close ();
                        fileString =
fileBuffer.toString ();
                }catch  (IOException e ) {
                        return null;
                }
                return fileString;
        } // readFile

  /**
    * Use a PrintWriter wrapped around a
BufferedWriter, which in turn
    * is wrapped around a FileWriter, to
write the string data to the
    * given file.
   **/
        public static boolean writeFile (File
file, String dataString) {
                try {
                PrintWriter out = new
PrintWriter (new BufferedWriter (new
FileWriter (file)));
                out.print (dataString);
                out.flush ();
                out.close ();

                }catch (IOException e) {
                        return false;
                }
                return true;
        } // writeFile

        private void
newSimulationButton_actionPerformed(ActionEv
ent e) {

                try{

        this.setVisible(false);
                        new SP().show();

        frame4.setVisible(false);
                }catch(NullPointerException
npe){}
        }

        private void about() {


JOptionPane.showMessageDialog(null,"Simulati
on of DNA Motion in Entropic Trapping \n
Created by Ruth E. Gorospe \n 2004-01812 \n
BS Computer Science \n University of the
Philippines Manila \n (2008)");
        }

        private void tutorial(){
                JFrame tutorialFrame = new
JFrame();


tutorialFrame.getContentPane().setLayout(nul
l);


                JTextArea tutorialTextArea =
new JTextArea();
                tutorialTextArea.setText("\nA
Simple Tutorial on How to go about the
Simulation:\n\n\n"

                + "+++ INPUT tab +++\n\n"

                + "*This tab contains the
parameters and constants of the
Simulation.\n"

                + "*Enter numerical values in
the textfields before proceeding to any
other tabs.\n"

                + "*You may click the CLEAR
Button to change all the values you have
entered\n"

                + "  or click CONTINUE>>>.\n"

                + "*[CONTINUE>>>] A message
box appears for error in input. If input is
error- free,\n"

                + " proceeds to the Summary
of Parameters and Constants.\n\n\n"

                + "+++ NUMERICAL DATA tab ++
+\n\n"

                + "*This tab contains a
button, [Show Numerical Data], which when
clicked displays\n"

                + "  values derived from the
computations.\n"

                + "*NOTE: Computations are
not displayed to the user; only the final
product of all\n"

                + " the computations.\n"

                + "*Computations include:
FORCE ON SPRING, FORCE ON BEAD, FORCE OF\n"

                + " INTERACTION, ELECTRIC
FORCE, RANDOM FORCE and EQUATION OF
MOTION.\n\n\n"

                + "+++ GRAPH tab +++\n\n"

                + "*This tab contains a
button, [View Graph], which when clicked
displays the\n"

                + " graph of the current
simulation.\n"

                + "*Position (y) vs Time (x)
Graph : Position of the particle at a
particular time.\n\n"

                + "*NOTE: A flat line in the
plot represents trapping of particle,\n"

                + " while a sloping line in
the plot represents movement in the
particle.\n\n\n"

                + "+++ Saving of Graphs ++
+\n\n"

                + "*Right click the graph ->
Save As... -> Choose where to save file ->
Save\n\n\n"

                + "*+++ Open Saved Graphs ++
+\n\n"

                + "*Menu -> File -> Open
Saved Graphs -> Choose File to View ->
Open\n\n\n"

                + "You are now ready to use
the simulation! Try different
combinations/values for\n"

                + " temperature,
electricfield,... and observe their effects
on the DNA's motion.");

        tutorialTextArea.setEditable(false);

        tutorialTextArea.setCaretPosition(0);

                JScrollPane
tutorialScrollPane = new JScrollPane();

tutorialScrollPane.setViewportView(tutorialT
extArea);


addComponent(tutorialFrame,tutorialScrollPan
e , 5, 5, 475, 450);


        tutorialFrame.setTitle("Simulation
Tutorial");
                tutorialFrame.setLocation(new
Point(50, 60));
                tutorialFrame.setSize(new
Dimension(500, 500));

        tutorialFrame.setResizable(false);

        tutorialFrame.setVisible(true);
        }

        public void createHomePage(){
                panel0 = new JPanel();
                panel0.setLayout(null);

                JTextArea homeTextArea = new
JTextArea();

        homeTextArea.setText("\n\nSimulation
of DNA Motion in Entropic Trapping User's
Guide\n\n\nINPUT\nGo to INPUT tab and enter
values to the corresponding text
fields.\nMake sure that you enter positive
numerical values (non-zero) and do not leave
any field blank.\n\n\nNUMERICAL DATA\nClick
to view the Numerical Data of the present
simulation.\n\n\nGRAPH\nClick to view Graph
of present simulation.\nRight-click the
graph to Save it.");

        homeTextArea.setEditable(false);
                addComponent(panel0,
homeTextArea, 10, 5, 600, 400);

        }

        public void createPage1(){
                panel1 = new JPanel();
                panel1.setLayout( null );
```

```
               inputParametersLabel = new
JLabel("Input Parameters for the
Simulation");

       inputParametersLabel.setFont(new
Font("Tahoma", Font.BOLD, 14));

               chainSizeLabel = new
JLabel();
               chainSizeLabel.setText("Chain
Size (N):");
               particlesLabel = new
JLabel("particles");

               temperatureLabel = new
JLabel();

temperatureLabel.setText("Temperature:");
               temperatureUnitLabel = new
JLabel("Kelvin");

               electricFieldLabel = new
JLabel();

electricFieldLabel.setText("Transverse
Electric Field:");
               electricFieldUnitLabel = new
JLabel("V/m");

               beadSizeLabel = new JLabel();
               beadSizeLabel.setText("Bead
Size:");
               beadSizeUnitLabel = new
JLabel("nm");

               chainSizeTextField = new
JTextField("3");


chainSizeTextField.setEditable(false);
               temperatureTextField = new
JTextField();
               electricFieldTextField = new
JTextField();
               beadSizeTextField = new
JTextField();

               constantsLabel = new
JLabel("Constants");
               constantsLabel.setFont(new
Font("Tahoma", Font.BOLD, 12));

               springConstantLabel = new
JLabel("spring constant k = 100");
               chargeLabel = new
JLabel("charge qnet = 0.48");
               transitTimeLabel = new
JLabel("transit time = 0.00051 sec");
               timeStepLabel = new
JLabel("time step = 0.00000008925");
               initialElectricFieldLabel =
new JLabel("initial electric field = 65
V/m");
               boltzmannLabel = new
JLabel("boltzmann constant = 1.38E-23 J/K");

               continueButton = new
JButton();
```

```

continueButton.addActionListener(handler);
               clearButton = new JButton();


clearButton.addActionListener(handler);



chainSizeTextField.addActionListener(new
ActionListener() {
                      public void
actionPerformed(ActionEvent e) {


chainSizeTextField_actionPerformed(e);
                       }

               });



temperatureTextField.addActionListener(new
ActionListener() {
                      public void
actionPerformed(ActionEvent e) {


temperatureTextField_actionPerformed(e);
                       }

               });




electricFieldTextField.addActionListener(new
ActionListener() {
                      public void
actionPerformed(ActionEvent e) {


electricFieldTextField_actionPerformed(e);
                       }

               });



beadSizeTextField.addActionListener(new
ActionListener() {
                      public void
actionPerformed(ActionEvent e) {

       beadSizeTextField_actionPerformed(e);
                       }

               });


       continueButton.setText("Continue
>>");

       continueButton.addActionListener(new
ActionListener() {
                      public void
actionPerformed(ActionEvent e) {
                              try{

       chainSize =
Double.parseDouble(chainSizeTextField.getTex
t());

        temperature =
Double.parseDouble(temperatureTextField.getT
ext());

        electricField =
Double.parseDouble(electricFieldTextField.ge
tText());

        beadSize =
Double.parseDouble(beadSizeTextField.getText
());
                                }catch(NumberF
ormatException nfe){

                                }



if(temperatureTextField.getText().equals("")
){

        try{


        JOptionPane.showMessageDialog( null,
"Please input a numerical value in the
temperature text field.");


        temperatureTextField.requestFocus();

        }catch(NumberFormatException nfe){


        JOptionPane.showMessageDialog( null,
"Please input a numerical value in the
temperature text field." );


        temperatureTextField.requestFocus();

        }
                                        }

                                        else
if(electricFieldTextField.getText().equals("
")){

        try{


        JOptionPane.showMessageDialog( null,
"Please input a numerical value in the
electric field text field.");



electricFieldTextField.requestFocus();

        }catch(NumberFormatException nfe){


        JOptionPane.showMessageDialog( null,
"Please input a numerical value in the
electric field text field." );
```

```

electricFieldTextField.requestFocus();

        }
                                        }

                                        else
if(beadSizeTextField.getText().equals("")){

        try{


        JOptionPane.showMessageDialog( null,
"Please input a numerical value in the bead
size text field.");


        beadSizeTextField.requestFocus();

        }catch(NumberFormatException nfe){


        JOptionPane.showMessageDialog( null,
"Please input a numerical value in the bead
size text field." );


        beadSizeTextField.requestFocus();

        }
                                        }

                                        else
if(temperature == 0){

        JOptionPane.showMessageDialog( null,
"Temperature must not be a nonzero positive
number." );

        temperatureTextField.requestFocus();
                                        }

                                        else
if(electricField == 0){

        JOptionPane.showMessageDialog( null,
"Electric field must not be a nonzero
positive number." );


electricFieldTextField.requestFocus();
                                        }

                                        else
if(beadSize == 0){

        JOptionPane.showMessageDialog( null,
"Bead size must not be a nonzero positive
number." );

        beadSizeTextField.requestFocus();
                                        }

                                        else{

        continueButton_actionPerformed(e);
                                        }
                        }

                });
                clearButton.setText("Clear");

        clearButton.addActionListener(new
ActionListener() {
                        public void
actionPerformed(ActionEvent e) {

        clearButton_actionPerformed(e);
                        }

                });

                addComponent(panel1,
inputParametersLabel, 13,20,250,50);
                addComponent(panel1,
chainSizeLabel, 20,80,180,30);
                addComponent(panel1,
chainSizeTextField, 200,80,80,30);
                addComponent(panel1,
particlesLabel, 300, 80, 120, 30);
                addComponent(panel1,
temperatureLabel, 20,120,180,30);
                addComponent(panel1,
temperatureTextField, 200,120,80,30);
                addComponent(panel1,
temperatureUnitLabel, 300, 120, 120, 30);
                addComponent(panel1,
electricFieldLabel, 20,160,180,30);
                addComponent(panel1,
electricFieldTextField, 200,160,80,30);
                addComponent(panel1,
electricFieldUnitLabel, 300, 160, 120, 30);
                addComponent(panel1,
beadSizeLabel, 20,200,180,30);
                addComponent(panel1,
beadSizeTextField, 200,200,80,30);
                addComponent(panel1,
beadSizeUnitLabel, 300, 200, 120, 30);

                addComponent(panel1,
constantsLabel, 400,25,200,50);
                addComponent(panel1,
springConstantLabel, 400,75,200,20);
                addComponent(panel1,
chargeLabel, 400,95,200,20);
                addComponent(panel1,
transitTimeLabel, 400,115,200,20);
                addComponent(panel1,
timeStepLabel, 400,135,200,20);
                addComponent(panel1,
initialElectricFieldLabel, 400,155,200,20);
                addComponent(panel1,
boltzmannLabel, 400,175,200,20);

                addComponent(panel1,
clearButton, 130,250,100,40);
                addComponent(panel1,
continueButton, 400,250,120,40);

        }

        public void createPage2(){
                try{

                        panel2 = new JPanel();

        panel2.setLayout(null);

                        JLabel page2Label =
new JLabel("The numerical data will show the
position of the particle at a specific
time.");


        showNumericalDataButton = new
JButton();


showNumericalDataButton.addActionListener(ha
ndler);

        showNumericalDataButton.setText("Show
Numerical Data");


showNumericalDataButton.addActionListener(ne
w ActionListener() {
                                public void
actionPerformed(ActionEvent e) {

        compute();
                                }

                        });

                        addComponent(panel2,
page2Label, 50, 80, 450, 20);
                        addComponent(panel2,
showNumericalDataButton, 200,150,160,40);

                }catch(NumberFormatException
nfe){

                }catch(NullPointerException
npe){}

        }

        public void createPage3(){
                panel3 = new JPanel();
                panel3.setLayout(null);

                JLabel page3Label = new
JLabel("A Position vs Time plot of the
Simulation.");
                JButton viewGraphButton = new
JButton("View Graph");

        viewGraphButton.addActionListener(new
ActionListener() {
                        public void
actionPerformed(ActionEvent e) {

        showGraph(chainSize);
                        }

                });

                addComponent(panel3,
page3Label, 50, 80, 450, 20);
                addComponent(panel3,
viewGraphButton, 200,150,160,40);


        }

        private void summary(){

                w = new JWindow(this);
                JPanel pan = new JPanel();
                pan.setBorder(new
LineBorder(Color.black, 5));

        w.getContentPane().add(pan,"Center");
                pan.setLayout(null);
                w.addWindowListener(
                        new WindowAdapter() {
                                public void
windowClosed(WindowEvent we) {
                                        //
remove listener


we.getWindow().removeWindowListener(this);
                                        //
enable outer YourFrame instance

        setEnabled(true);
                                }
                        }
                );

                JLabel paramSummaryLabel =
new JLabel("The following parameters will be
used by the Simulation:");
                paramSummaryLabel.setFont(new
Font("Tahoma", Font.BOLD, 12));

                JLabel chainSizeValueLabel =
new JLabel("Chain Size (N) = " +
chainSizeTextField.getText());
                JLabel temperatureValueLabel
= new JLabel("Temperature = " +
temperatureTextField.getText());
                JLabel
electricFieldValueLabel = new
JLabel("Transverse Electric Field = " +
electricFieldTextField.getText());
                JLabel beadSizeValueLabel =
new JLabel("Bead Size = " +
beadSizeTextField.getText());
                springConstantLabel = new
JLabel("spring constant k = 100");
                chargeLabel = new
JLabel("charge qnet = 0.48");
                transitTimeLabel = new
JLabel("transit time = 0.00051 sec");
                timeStepLabel = new
JLabel("time step = 0.00000008925");
                initialElectricFieldLabel =
new JLabel("initial electric field = 65
V/m");
                boltzmannLabel = new
JLabel("boltzmann constant = 1.38E-23 J/K");

                chainSize =
Double.parseDouble(chainSizeTextField.getTex
t());
                temperature =
Double.parseDouble(temperatureTextField.getT
ext());
                electricField =
Double.parseDouble(electricFieldTextField.ge
tText());
                beadSize =
Double.parseDouble(beadSizeTextField.getText
());

                Values = new ArrayList();
                hashtable = new Hashtable();
```

```
                hashtable.put("Chain Size
(N)", new Double(chainSize));
                Values.add(new
Double(chainSize));
                hashtable.put("Temperature",
new Double(temperature));
                Values.add(new
Double(temperature));
                hashtable.put("Transverse
Electric Field", new Double(electricField));
                Values.add(new
Double(electricField));
                hashtable.put("Bead Size",
new Double(beadSize));
                Values.add(new
Double(beadSize));

                PrintStream MyOutput = null;
                try {
                        MyOutput = new
PrintStream(new
FileOutputStream("params.txt"));
                }catch (IOException e){

        System.out.println("params");
                }

                if (MyOutput != null) {
                       for (int i=0;
i<=Values.size()-1; i++) {
                                Double val =
((Double)(Values.get(i)));

MyOutput.println(val);


                        }
                        MyOutput.close();
                } else {
                        System.out.println("No
output file written");
                }


                JButton backButton = new
JButton("Cancel");

        backButton.addActionListener(new
ActionListener() {
                        public void
actionPerformed(ActionEvent e) {

        backButton_actionPerformed(e);
                        }

                });

                JButton runButton = new
JButton("OK");

        runButton.addActionListener(new
ActionListener() {
                        public void
actionPerformed(ActionEvent e) {

        runButton_actionPerformed(e);
                        }

                });
                addComponent(pan,
paramSummaryLabel, 25,30,350,30);
                addComponent(pan,
chainSizeValueLabel, 100, 70, 200, 20);
                addComponent(pan,
temperatureValueLabel, 100, 90, 200, 20);
                addComponent(pan,
electricFieldValueLabel, 100, 110, 200, 20);
                addComponent(pan,
beadSizeValueLabel, 100, 130, 200, 20);
                addComponent(pan,
springConstantLabel, 100,150,200,20);
                addComponent(pan,
chargeLabel, 100,170,200,20);
                addComponent(pan,
transitTimeLabel, 100,190,200,20);
                addComponent(pan,
timeStepLabel, 100,210,200,20);
                addComponent(pan,
initialElectricFieldLabel, 100,230,200,20);
                addComponent(pan,
boltzmannLabel, 100,250,200,20);

                addComponent(pan, runButton,
60,300,160,40);
                addComponent(pan, backButton,
250,300,90,40);

                setEnabled(false);
                w.setLocation(new Point(400,
200));
                w.setSize(new Dimension(400,
400));

                w.setVisible(true);

        }//end of summary

        private void compute(){
                try{

                        chainSize =
Double.parseDouble(chainSizeTextField.getTex
t());
                        temperature =
Double.parseDouble(temperatureTextField.getT
ext());
                        electricField =
Double.parseDouble(electricFieldTextField.ge
tText());
                        beadSize =
Double.parseDouble(beadSizeTextField.getText
());

                        if(( chainSize < 2 )
|| (chainSizeTextField.getText().equals(""))
||
(temperatureTextField.getText().equals(""))
||
(electricFieldTextField.getText().equals("")
) ||
(beadSizeTextField.getText().equals("")) ||
(chainSize == 0) || (temperature == 0) ||
(electricField == 0) || (beadSize == 0) ){

        JOptionPane.showMessageDialog( null,
"Enter correct values first in the INPUT
tab." );
                                try{

tabbedPane.setSelectedComponent(panel1);

                                }catch(Illegal
ArgumentException iae){}
                        }
                        else{
                        frame3 = new JFrame();


                        JLabel timeLabel = new
JLabel("j");
                        JLabel ithLabel = new
JLabel("ith bead");
                        JLabel iterLabel = new
JLabel("rn");
                        computationsTextArea =
new JTextArea();

                        chainSize =
Double.parseDouble(chainSizeTextField.getTex
t());
                        int csize = (int)
(chainSize);
                        beadSize =
(Double.parseDouble(beadSizeTextField.getTex
t()))*(0.000000001);
                        temperature =
Double.parseDouble(temperatureTextField.getT
ext());
                        electricField =
(Double.parseDouble(electricFieldTextField.g
etText()))*(100);

                        double stokes =
(6*Math.PI*beadSize*0.001);
                        int t = 1;
                        double [][] r = new
double[3][1000];
                        double [] fspring2 =
f_spring(chainSize, beadSize, temperature);
                        double [] fbead2 =
f_bead(chainSize, beadSize, temperature);
                        double [] fint2 =
f_int(chainSize, beadSize, temperature,
fspring2, fbead2);
                        double [] felectric2 =
f_electric(qnet, electricField);
                        double [] randforce2 =
random_force(temperature, beadSize);
                        DecimalFormat df1 =
new DecimalFormat("#,##0.0000000");
                        DecimalFormat df2 =
new DecimalFormat("#,##0.00000");
                        double [] delta_t2 =
new double[csize];
                        double [][] k1 = new
double[4][1000];
                        double [][] k2 = new
double[4][1000];
                        double [][] k3 = new
double[4][1000];
                        double [][] k4 = new
double[4][1000];
                        int h =1;
                        int i=0;
                        int j=1;
                        double [][] rn = new
double[4][1000];
                        double [] []fbead3 =
new double [4][1000];
```

```
                       double [] []fspring3 =
new double [4][1000];
                       double [][]fint3 = new
double [4][1000];
                       double [][]felectric3
= new double [4][1000];
                       double [][]randforce3
= new double [4][1000];
                       double [][]bd = new
double [4][1000];
                       rn[0][0] = 0;
                       rn[1][0] = 0;
                       rn[2][0] = 0;

                       try{

                                      for(j =
0; j <= 100; j++){

        bd[0][j] = (beadSize*3);

        bd[1][j] = (beadSize*2);

        bd[2][j] = (beadSize*1);



        fbead3[0][j] =
((boltzmann_constant*temperature)*(((12*(Mat
h.pow(beadSize, 12)))/
(Math.pow((Math.abs((bd[0][j])-(bd[1][j]))),
13))) - (((6*(Math.pow(beadSize, 6)))/
(Math.pow(Math.abs(((bd[0][j])-(bd[1][j]))),
7)))))) +
((boltzmann_constant*temperature)*(((12*(Mat
h.pow(beadSize, 12)))/
(Math.pow(Math.abs(((bd[0][j])-(bd[2][j]))),
13))) - (((6*(Math.pow(beadSize, 6)))/
(Math.pow(Math.abs(((bd[0][j])-(bd[2][j]))),
7))))));

        fbead3[1][j] =
((boltzmann_constant*temperature)*(((12*(Mat
h.pow(beadSize, 12)))/
(Math.pow((Math.abs((bd[1][j])-(bd[0][j]))),
13))) - (((6*(Math.pow(beadSize, 6)))/
(Math.pow(Math.abs(((bd[1][j])-(bd[0][j]))),
7)))))) +
((boltzmann_constant*temperature)*(((12*(Mat
h.pow(beadSize, 12)))/
(Math.pow(Math.abs(((bd[1][j])-(bd[2][j]))),
13))) - (((6*(Math.pow(beadSize, 6)))/
(Math.pow(Math.abs(((bd[1][j])-(bd[2][j]))),
7))))));

        fbead3[2][j] =
((boltzmann_constant*temperature)*(((12*(Mat
h.pow(beadSize, 12)))/
(Math.pow((Math.abs((bd[2][j])-(bd[0][j]))),
13))) - (((6*(Math.pow(beadSize, 6)))/
(Math.pow(Math.abs(((bd[2][j])-(bd[0][j]))),
7)))))) +
((boltzmann_constant*temperature)*(((12*(Mat
h.pow(beadSize, 12)))/
(Math.pow(Math.abs(((bd[2][j])-(bd[1][j]))),
13))) - (((6*(Math.pow(beadSize, 6)))/
(Math.pow(Math.abs(((bd[2][j])-(bd[1][j]))),
7))))));
```

```
        fspring3[0][j] = (((-
1)*boltzmann_constant*temperature*k)/
(beadSize*beadSize))*(Math.abs((bd[0][j])-
(bd[1][j])));

        fspring3[1][j] = ((((-
1)*boltzmann_constant*temperature*k)/
(beadSize*beadSize))*(Math.abs((bd[1][j])-
(bd[0][j])))) + ((((-
1)*boltzmann_constant*temperature*k)/
(beadSize*beadSize))*(Math.abs((bd[1][j])-
(bd[2][j]))));

        fspring3[2][j] = (((-
1)*boltzmann_constant*temperature*k)/
(beadSize*beadSize))*(Math.abs((bd[2][j])-
(bd[1][j])));



        fint3[0][j] = fbead3[0][j] +
fspring3[0][j];

        fint3[1][j] = fbead3[0][j] +
fspring3[0][j];

        fint3[2][j] = fbead3[0][j] +
fspring3[0][j];


        if((j < 41) || (j > 60)){

                felectric3[0][j] =
qnet*electricField_initial;

                felectric3[1][j] =
qnet*electricField_initial;

                felectric3[2][j] =
qnet*electricField_initial;

        }

        else{

                felectric3[0][j] =
qnet*(electricField_initial - electricField);

                felectric3[1][j] =
qnet*(electricField_initial - electricField);

                felectric3[2][j] =
qnet*(electricField_initial - electricField);

        }



        double randnum4 = 0;

        int high = 1;

        int low = -1;

        int randnum1 = number.nextInt( high -
low + 1) + low;

        double randnum2 = randnum1; //random
-1, 0, or 1
```

```
        double randnum3 = Math.random();
//random numbers between 0 and 1

        if((randnum2 == 0) && (randnum3 <=
0.5)){

                randnum4 = -1.0;

        }

        if((randnum2 == 0) && (randnum3 >=
0.51)){

                randnum4 = 1.0;

        }



        else{

                randnum4 =
(randnum2*randnum3); //random numbers (-1,1)
problem: -1 and 1 must also be included

        }



        randforce3[0][j] =
(boltzmann_constant*temperature/beadSize)*(M
ath.sqrt((6*transit_time)/delta_t))*(randnum
4);

        randforce3[1][j] =
(boltzmann_constant*temperature/beadSize)*(M
ath.sqrt((6*transit_time)/delta_t))*(randnum
4);

        randforce3[2][j] =
(boltzmann_constant*temperature/beadSize)*(M
ath.sqrt((6*transit_time)/delta_t))*(randnum
4);


        r[0][j] = (fint3[0][j] +
felectric3[0][j] + randforce3[0][j])/stokes;

        k1[0][j] = (fint3[0][j] +
felectric3[0][j] + randforce3[0]
[j])*(delta_t/stokes);

        k2[0][j] = ((fint3[0][j] +
felectric3[0][j] + randforce3[0][j]) +
((k1[0][j])/2))*(delta_t/stokes);

        k3[0][j] = ((fint3[0][j] +
felectric3[0][j] + randforce3[0][j]) +
((k2[0][j])/2))*(delta_t/stokes);

        k4[0][j] = ((fint3[0][j] +
felectric3[0][j] + randforce3[0][j]) +
(k3[0][j]))*(delta_t/stokes);

        rn[0][j] = Math.abs( (r[0][j]) +
((k1[0][j] + (2*k2[0][j]) + (2*k3[0][j]) +
k4[0][j] ) /6));


        r[1][j] = (fint3[1][j] +
felectric3[1][j] + randforce3[1][j])/stokes;

        k1[1][j] = (fint3[1][j] +
felectric3[1][j] + randforce3[1]
[j])*(delta_t/stokes);

        k2[1][j] = ((fint3[1][j] +
felectric3[1][j] + randforce3[1][j]) +
((k1[1][j])/2))*(delta_t/stokes);

        k3[1][j] = ((fint3[1][j] +
felectric3[1][j] + randforce3[1][j]) +
((k2[1][j])/2))*(delta_t/stokes);

        k4[1][j] = ((fint3[1][j] +
felectric3[1][j] + randforce3[1][j]) +
(k3[1][j]))*(delta_t/stokes);

        rn[1][j] = Math.abs( (r[1][j]) +
((k1[1][j] + (2*k2[1][j]) + (2*k3[1][j]) +
k4[1][j] ) /6));


        r[2][j] = (fint3[2][j] +
felectric3[2][j] + randforce3[2][j])/stokes;

        k1[2][j] = (fint3[2][j] +
felectric3[2][j] + randforce3[2]
[j])*(delta_t/stokes);

        k2[2][j] = ((fint3[2][j] +
felectric3[2][j] + randforce3[2][j]) +
((k1[2][j])/2))*(delta_t/stokes);

        k3[2][j] = ((fint3[2][j] +
felectric3[2][j] + randforce3[2][j]) +
((k2[2][j])/2))*(delta_t/stokes);

        k4[2][j] = ((fint3[2][j] +
felectric3[2][j] + randforce3[2][j]) +
(k3[2][j]))*(delta_t/stokes);

        rn[2][j] = Math.abs((r[2][j]) +
((k1[2][j] + (2*k2[2][j]) + (2*k3[2][j]) +
k4[2][j] ) /6));


        computationsTextArea.append(j +
"\t0\t" + (rn[0][j]) +"\n" );

        computationsTextArea.append(j +
"\t1\t" + (rn[1][j]) +"\n" );

        computationsTextArea.append(j +
"\t2\t" + (rn[2][j]) +"\n" );

        t = t+1;

        delta_t2[0] = 0.0;

        r[0][0] = 0.0;


        delta_t2[i+1] = (delta_t2[i]) +
0.000175;
                                        }//end
for j
                        }catch(ArrayIndexOutOf
BoundsException aiobe){}
```

```

computationsTextArea.setEditable(false);


computationsTextArea.setCaretPosition(0);

                      JScrollPane
computationsScrollPane = new JScrollPane();


computationsScrollPane.setViewportView(compu
tationsTextArea);

                      JButton okButton = new
JButton("OK");

       okButton.addActionListener(new
ActionListener() {
                              public void
actionPerformed(ActionEvent e) {

       viewGraphButton_actionPerformed(e);
                              }

                      });



frame3.getContentPane().setLayout(null);

                      addComponent(frame3,
timeLabel, 60, 50, 50, 30);
                      addComponent(frame3,
ithLabel, 120,50,50,30);
                      addComponent(frame3,
iterLabel, 210,50,50,30);
                      addComponent(frame3,
computationsScrollPane, 40,90,450,330);
                      addComponent(frame3,
okButton, 180,430,160,40);


       frame3.setTitle("Numerical Data");
                      frame3.setLocation(new
Point(450, 250));
                      frame3.setSize(new
Dimension(500, 500));

       frame3.setResizable(false);

       frame3.setVisible(true);
                      }
                      }catch(NumberFormatExc
eption nfe){

       JOptionPane.showMessageDialog( null,
"Enter correct values first in the INPUT
tab." );
                              try{



tabbedPane.setSelectedComponent(panel1);

                              }catch(Illegal
ArgumentException iae){}
                      }

       }
```

```
       private void showGraph(double
chainSize) {
               try{
                      chainSize =
Double.parseDouble(chainSizeTextField.getTex
t());
                      temperature =
Double.parseDouble(temperatureTextField.getT
ext());
                      electricField =
Double.parseDouble(electricFieldTextField.ge
tText());
                      beadSize =
Double.parseDouble(beadSizeTextField.getText
());


                      if(( chainSize < 2 )
|| (chainSizeTextField.getText().equals(""))
||
(temperatureTextField.getText().equals(""))
||
(electricFieldTextField.getText().equals("")
) ||
(beadSizeTextField.getText().equals("")) ||
(chainSize == 0) || (temperature == 0) ||
(electricField == 0) || (beadSize == 0) ){

       JOptionPane.showMessageDialog( null,
"No graph to show yet. \nEnter correct
values first in the INPUT tab." );
                              try{



tabbedPane.setSelectedComponent(panel1);

                              }catch(Illegal
ArgumentException iae){}
                      }

                      else{
                              frame4 = new
JFrame();
                              JLabel
trapLabel = new JLabel("The particle was
trapped.");
                              JButton
prevSimButton = new JButton("View Past
Simulations");

       prevSimButton.addActionListener(new
ActionListener() {
                                     public
void actionPerformed(ActionEvent e) {


openSavedGraphButton_actionPerformed(e);
                                     }

                              });



frame4.getContentPane().setLayout(null);

                              chainSize =
Double.parseDouble(chainSizeTextField.getTex
t());
                              int csize =
(int)(chainSize);
```

```
                              beadSize =
(Double.parseDouble(beadSizeTextField.getTex
t()))*(0.000000001);
                              bs =
Double.parseDouble(beadSizeTextField.getText
());
                              temperature =
Double.parseDouble(temperatureTextField.getT
ext());
                              electricField
=
(Double.parseDouble(electricFieldTextField.g
etText()))*(100);
                              ef =
Double.parseDouble(electricFieldTextField.ge
tText());
                              int numtrapctr
= 0;
                              double stokes
= (6*Math.PI*beadSize*0.001);
                              int t = 1;
                              double [][] r
= new double[csize][1100];
                              double []
fspring2 = f_spring(chainSize, beadSize,
temperature);
                              double []
fbead2 = f_bead(chainSize, beadSize,
temperature);
                              double []
fint2 = f_int(chainSize, beadSize,
temperature, fspring2, fbead2);
                              double []
felectric2 = f_electric(qnet, electricField);
                              double []
randforce2 = random_force(temperature,
beadSize);
                              DecimalFormat
df1 = new DecimalFormat("#,##0.000");
                              DecimalFormat
df2 = new DecimalFormat("#,##0.00000");

                              double [][] k1
= new double[3][1000];
                              double [][] k2
= new double[3][1000];
                              double [][] k3
= new double[3][1000];
                              double [][] k4
= new double[3][1000];

                              XYSeries
series = new XYSeries("");
                              int simctr = 1;
                              int simctr2 =
1;
                              int fctr = 0;
                              double []
delta_t2 = new double[csize];

                              XYDataset
dataset = null;
                              JFreeChart
chart = null;
                              ChartPanel
chartpanel = null;
                              int h =1;
                              int i=0;
                              int j=0;
                              double [][] rn
= new double[3][1000];
```

```
                              double []
[]fbead3 = new double [3][1000];
                              double []
[]fspring3 = new double [3][1000];
                              double []
[]fint3 = new double [3][1000];
                              double []
[]felectric3 = new double [3][1000];
                              double []
[]randforce3 = new double [3][1000];
                              double [][]bd
= new double [3][1000];
                              double [][]cm
= new double[3][1000];
                              cm[0][0] = 0;
                              cm[1][0] = 0;
                              cm[2][0] = 0;
                                      for(j =
1; j <= 100; j++){

       bd[0][j] = (beadSize*3);

       bd[1][j] = (beadSize*2);

       bd[2][j] = (beadSize*1);


       double randnum4 = 0;

       int high = 1;

       int low = -1;

       int randnum1 = number.nextInt( high -
low + 1) + low;

       double randnum2 = randnum1; //random
-1, 0, or 1

       double randnum3 = Math.random();
//random numbers between 0 and 1

       if((randnum2 == 0) && (randnum3 <=
0.5)){

               randnum4 = -1.0;

       }

       if((randnum2 == 0) && (randnum3 >=
0.51)){

               randnum4 = 1.0;

       }


       else{

               randnum4 =
(randnum2*randnum3); //random numbers (-1,1)
problem: -1 and 1 must also be included

       }


       series.add(0, cm[0][0] );

       series.add(0, cm[1][0] );

       series.add(0, cm[2][0] );
```

```
        fbead3[0][j] =
((boltzmann_constant*temperature)*(((12*(Mat
h.pow(beadSize, 12)))/
(Math.pow((Math.abs((bd[0][j])-(bd[1][j]))),
13))) - (((6*(Math.pow(beadSize, 6)))/
(Math.pow(Math.abs(((bd[0][j])-(bd[1][j]))),
7)))))) +
((boltzmann_constant*temperature)*(((12*(Mat
h.pow(beadSize, 12)))/
(Math.pow(Math.abs(((bd[0][j])-(bd[2][j]))),
13))) - (((6*(Math.pow(beadSize, 6)))/
(Math.pow(Math.abs(((bd[0][j])-(bd[2][j]))),
7)))))));

        fbead3[1][j] =
((boltzmann_constant*temperature)*(((12*(Mat
h.pow(beadSize, 12)))/
(Math.pow((Math.abs((bd[1][j])-(bd[0][j]))),
13))) - (((6*(Math.pow(beadSize, 6)))/
(Math.pow(Math.abs(((bd[1][j])-(bd[0][j]))),
7)))))) +
((boltzmann_constant*temperature)*(((12*(Mat
h.pow(beadSize, 12)))/
(Math.pow(Math.abs(((bd[1][j])-(bd[2][j]))),
13))) - (((6*(Math.pow(beadSize, 6)))/
(Math.pow(Math.abs(((bd[1][j])-(bd[2][j]))),
7)))))));

        fbead3[2][j] =
((boltzmann_constant*temperature)*(((12*(Mat
h.pow(beadSize, 12)))/
(Math.pow((Math.abs((bd[2][j])-(bd[0][j]))),
13))) - (((6*(Math.pow(beadSize, 6)))/
(Math.pow(Math.abs(((bd[2][j])-(bd[0][j]))),
7)))))) +
((boltzmann_constant*temperature)*(((12*(Mat
h.pow(beadSize, 12)))/
(Math.pow(Math.abs(((bd[2][j])-(bd[1][j]))),
13))) - (((6*(Math.pow(beadSize, 6)))/
(Math.pow(Math.abs(((bd[2][j])-(bd[1][j]))),
7)))))));




        fspring3[0][j] = (((-
1)*boltzmann_constant*temperature*k)/
(beadSize*beadSize))*(Math.abs((bd[0][j])-
(bd[1][j])));

        fspring3[1][j] = ((((-
1)*boltzmann_constant*temperature*k)/
(beadSize*beadSize))*(Math.abs((bd[1][j])-
(bd[0][j])))) + ((((-
1)*boltzmann_constant*temperature*k)/
(beadSize*beadSize))*(Math.abs((bd[1][j])-
(bd[2][j]))));

        fspring3[2][j] = (((-
1)*boltzmann_constant*temperature*k)/
(beadSize*beadSize))*(Math.abs((bd[2][j])-
(bd[1][j])));



        fint3[0][j] = fbead3[0][j] +
fspring3[0][j];
```

```
        fint3[1][j] = fbead3[0][j] +
fspring3[0][j];

        fint3[2][j] = fbead3[0][j] +
fspring3[0][j];


        randforce3[0][j] =
(boltzmann_constant*temperature/beadSize)*(M
ath.sqrt((6*transit_time)/delta_t))*(randnum
4);

        randforce3[1][j] =
(boltzmann_constant*temperature/beadSize)*(M
ath.sqrt((6*transit_time)/delta_t))*(randnum
4);

        randforce3[2][j] =
(boltzmann_constant*temperature/beadSize)*(M
ath.sqrt((6*transit_time)/delta_t))*(randnum
4);


        if(j < 41){

                felectric3[0][j] =
qnet*electricField_initial;

                felectric3[1][j] =
qnet*electricField_initial;

                felectric3[2][j] =
qnet*electricField_initial;



                r[0][j] = (fint3[0][j] +
felectric3[0][j] + randforce3[0][j])/stokes;

                k1[0][j] = (fint3[0][j] +
felectric3[0][j] + randforce3[0]
[j])*(delta_t/stokes);

                k2[0][j] = ((fint3[0][j] +
felectric3[0][j] + randforce3[0][j]) +
((k1[0][j])/2))*(delta_t/stokes);

                k3[0][j] = ((fint3[0][j] +
felectric3[0][j] + randforce3[0][j]) +
((k2[0][j])/2))*(delta_t/stokes);

                k4[0][j] = ((fint3[0][j] +
felectric3[0][j] + randforce3[0][j]) +
(k3[0][j]))*(delta_t/stokes);

                rn[0][j] = Math.abs((r[0][j])
+ ((k1[0][j] + (2*k2[0][j]) + (2*k3[0][j]) +
k4[0][j] ) /6));

                cm[0][j] = Math.abs(cm[0][(j-
1)] + (((cm[0][j]) + (rn[0][j]))/j));



                r[1][j] = (fint3[1][j] +
felectric3[1][j] + randforce3[1][j])/stokes;

                k1[1][j] = (fint3[1][j] +
felectric3[1][j] + randforce3[1]
[j])*(delta_t/stokes);
```

```
                k2[1][j] = ((fint3[1][j] +
felectric3[1][j] + randforce3[1][j]) +
((k1[1][j])/2))*(delta_t/stokes);

                k3[1][j] = ((fint3[1][j] +
felectric3[1][j] + randforce3[1][j]) +
((k2[1][j])/2))*(delta_t/stokes);

                k4[1][j] = ((fint3[1][j] +
felectric3[1][j] + randforce3[1][j]) +
(k3[1][j]))*(delta_t/stokes);

                rn[1][j] = Math.abs((r[1][j])
+ ((k1[1][j] + (2*k2[1][j]) + (2*k3[1][j]) +
k4[1][j] ) /6));

                cm[1][j] = Math.abs(cm[1][(j-
1)] + (((cm[1][j]) + (rn[1][j]))/j));



                r[2][j] = (fint3[2][j] +
felectric3[2][j] + randforce3[2][j])/stokes;

                k1[2][j] = (fint3[2][j] +
felectric3[2][j] + randforce3[2]
[j])*(delta_t/stokes);

                k2[2][j] = ((fint3[2][j] +
felectric3[2][j] + randforce3[2][j]) +
((k1[2][j])/2))*(delta_t/stokes);

                k3[2][j] = ((fint3[2][j] +
felectric3[2][j] + randforce3[2][j]) +
((k2[2][j])/2))*(delta_t/stokes);

                k4[2][j] = ((fint3[2][j] +
felectric3[2][j] + randforce3[2][j]) +
(k3[2][j]))*(delta_t/stokes);

                rn[2][j] = Math.abs((r[2][j])
+ ((k1[2][j] + (2*k2[2][j]) + (2*k3[2][j]) +
k4[2][j] ) /6));

                cm[2][j] = Math.abs(cm[2][(j-
1)] + (((cm[2][j]) + (rn[2][j]))/j));



                series.add(j, cm[0][j]);

                series.add(j, cm[1][j]);

                series.add(j, cm[2][j]);

        }


        else if(j > 60){

                felectric3[0][j] =
qnet*electricField_initial;

                felectric3[1][j] =
qnet*electricField_initial;

                felectric3[2][j] =
qnet*electricField_initial;
```

```
                r[0][j] = (fint3[0][j] +
felectric3[0][j] + randforce3[0][j])/stokes;

                k1[0][j] = (fint3[0][j] +
felectric3[0][j] + randforce3[0]
[j])*(delta_t/stokes);

                k2[0][j] = ((fint3[0][j] +
felectric3[0][j] + randforce3[0][j]) +
((k1[0][j])/2))*(delta_t/stokes);

                k3[0][j] = ((fint3[0][j] +
felectric3[0][j] + randforce3[0][j]) +
((k2[0][j])/2))*(delta_t/stokes);

                k4[0][j] = ((fint3[0][j] +
felectric3[0][j] + randforce3[0][j]) +
(k3[0][j]))*(delta_t/stokes);

                rn[0][j] = Math.abs((r[0][j])
+ ((k1[0][j] + (2*k2[0][j]) + (2*k3[0][j]) +
k4[0][j] ) /6));

                cm[0][j] = Math.abs(cm[0][39]
+ (((cm[0][j]) + (rn[0][j]))/j));



                r[1][j] = (fint3[1][j] +
felectric3[1][j] + randforce3[1][j])/stokes;

                k1[1][j] = (fint3[1][j] +
felectric3[1][j] + randforce3[1]
[j])*(delta_t/stokes);

                k2[1][j] = ((fint3[1][j] +
felectric3[1][j] + randforce3[1][j]) +
((k1[1][j])/2))*(delta_t/stokes);

                k3[1][j] = ((fint3[1][j] +
felectric3[1][j] + randforce3[1][j]) +
((k2[1][j])/2))*(delta_t/stokes);

                k4[1][j] = ((fint3[1][j] +
felectric3[1][j] + randforce3[1][j]) +
(k3[1][j]))*(delta_t/stokes);

                rn[1][j] = Math.abs((r[1][j])
+ ((k1[1][j] + (2*k2[1][j]) + (2*k3[1][j]) +
k4[1][j] ) /6));

                cm[1][j] = Math.abs(cm[1][39]
+ (((cm[1][j]) + (rn[1][j]))/j));



                r[2][j] = (fint3[2][j] +
felectric3[2][j] + randforce3[2][j])/stokes;

                k1[2][j] = (fint3[2][j] +
felectric3[2][j] + randforce3[2]
[j])*(delta_t/stokes);

                k2[2][j] = ((fint3[2][j] +
felectric3[2][j] + randforce3[2][j]) +
((k1[2][j])/2))*(delta_t/stokes);

                k3[2][j] = ((fint3[2][j] +
felectric3[2][j] + randforce3[2][j]) +
((k2[2][j])/2))*(delta_t/stokes);
```

```
                k4[2][j] = ((fint3[2][j] + 
felectric3[2][j] + randforce3[2][j]) + 
(k3[2][j]))*(delta_t/stokes);

                rn[2][j] = Math.abs((r[2][j]) 
+ ((k1[2][j] + (2*k2[2][j]) + (2*k3[2][j]) + 
k4[2][j] ) /6));

                cm[2][j] = Math.abs(cm[2][39] 
+ (((cm[2][j]) + (rn[2][j]))/j));



                series.add(j, cm[0][j]);

                series.add(j, cm[1][j]);

                series.add(j, cm[2][j]);



        }


        else{


                felectric3[0][j] = 
qnet*(electricField_initial - electricField);

                felectric3[1][j] = 
qnet*(electricField_initial - electricField);

                felectric3[2][j] = 
qnet*(electricField_initial - electricField);



                r[0][j] = (fint3[0][j] + 
felectric3[0][j] + randforce3[0][j])/stokes;

                k1[0][j] = (fint3[0][j] + 
felectric3[0][j] + randforce3[0]
[j])*(delta_t/stokes);

                k2[0][j] = ((fint3[0][j] + 
felectric3[0][j] + randforce3[0][j]) + 
((k1[0][j])/2))*(delta_t/stokes);

                k3[0][j] = ((fint3[0][j] + 
felectric3[0][j] + randforce3[0][j]) + 
((k2[0][j])/2))*(delta_t/stokes);

                k4[0][j] = ((fint3[0][j] + 
felectric3[0][j] + randforce3[0][j]) + 
(k3[0][j]))*(delta_t/stokes);

                rn[0][j] = Math.abs((r[0][j]) 
+ ((k1[0][j] + (2*k2[0][j]) + (2*k3[0][j]) + 
k4[0][j] ) /6));

                cm[0][j] = Math.abs(((cm[0]
[39]) + (rn[0][39]))/j);



                r[1][j] = (fint3[1][j] + 
felectric3[1][j] + randforce3[1][j])/stokes;

                k1[1][j] = (fint3[1][j] + 
felectric3[1][j] + randforce3[1]
[j])*(delta_t/stokes);

                k2[1][j] = ((fint3[1][j] + 
felectric3[1][j] + randforce3[1][j]) + 
((k1[1][j])/2))*(delta_t/stokes);

                k3[1][j] = ((fint3[1][j] + 
felectric3[1][j] + randforce3[1][j]) + 
((k2[1][j])/2))*(delta_t/stokes);

                k4[1][j] = ((fint3[1][j] + 
felectric3[1][j] + randforce3[1][j]) + 
(k3[1][j]))*(delta_t/stokes);

                rn[1][j] = Math.abs((r[1][j]) 
+ ((k1[1][j] + (2*k2[1][j]) + (2*k3[1][j]) + 
k4[1][j] ) /6));

                cm[1][j] = Math.abs(((cm[1]
[39]) + (rn[1][39]))/j);



                r[2][j] = (fint3[2][j] + 
felectric3[2][j] + randforce3[2][j])/stokes;

                k1[2][j] = (fint3[2][j] + 
felectric3[2][j] + randforce3[2]
[j])*(delta_t/stokes);

                k2[2][j] = ((fint3[2][j] + 
felectric3[2][j] + randforce3[2][j]) + 
((k1[2][j])/2))*(delta_t/stokes);

                k3[2][j] = ((fint3[2][j] + 
felectric3[2][j] + randforce3[2][j]) + 
((k2[2][j])/2))*(delta_t/stokes);

                k4[2][j] = ((fint3[2][j] + 
felectric3[2][j] + randforce3[2][j]) + 
(k3[2][j]))*(delta_t/stokes);

                rn[2][j] = Math.abs((r[2][j]) 
+ ((k1[2][j] + (2*k2[2][j]) + (2*k3[2][j]) + 
k4[2][j] ) /6));

                cm[2][j] = Math.abs(((cm[2]
[39]) + (rn[2][39]))/j);



                series.add(j, cm[0][39]);

                series.add(j, cm[1][39]);

                series.add(j, cm[2][39]);

        }

                                        }
                                        dataset 
= new XYSeriesCollection(series);
                                        chart = 
ChartFactory.createXYLineChart(("Position vs 
Time\n\nN=" + csize + " T=" + temperature + 
" E=" + ef + " b=" + bs), "Time (units of 
\u03C4)", "Position (units of 
sigma)",dataset, PlotOrientation.VERTICAL, 
true, true, false);
```

```
       chartpanel = new ChartPanel(chart);

       JScrollPane graphScrollPane = new
JScrollPane();


graphScrollPane.setViewportView(chartpanel);


       FileInputStream fin;

       FileOutputStream fout;
                                     try{

       int ctr =
Integer.parseInt(readFile(outfile));
                                     }catch(
NumberFormatException nfe){

       simctr = 1;
                                     }

                                     try{

       try{


       ChartUtilities.saveChartAsPNG(new
File(("graph" + simctr+".png")), chart,
600,300);



       fout = new FileOutputStream
(outfile);


       new
PrintStream(fout).println(simctr);

       simctr = simctr + 1;

       fout.close();



       }catch(IOException ioe){}
                                     }catch(
Exception ee){}

                      addComponent(frame4,
chartpanel, 50, 30, 600, 400 );
                      addComponent(frame4,
trapLabel, 30, 500, 200, 30);

       frame3.setVisible(false);

       frame4.setTitle("Graph");
                      frame4.setLocation(new
Point(300, 150));
                      frame4.setSize(new
Dimension(700, 600));

       frame4.setResizable(false);

       frame4.setVisible(true);
               }
               }catch(NumberFormatException
nfe){
```

```
       JOptionPane.showMessageDialog( null,
"No graph to show yet. \nEnter correct
values first in the INPUT tab." );
                              try{



tabbedPane.setSelectedComponent(panel1);

                              }catch(Illegal
ArgumentException iae){}
                       }
       }


       private double []  f_int(double
chainSize, double beadSize, double
temperature, double [] fspring, double []
fbead){

               chainSize =
Double.parseDouble(chainSizeTextField.getTex
t());
               beadSize =
(Double.parseDouble(beadSizeTextField.getTex
t()))*(0.000000001);
               temperature =
Double.parseDouble(temperatureTextField.getT
ext());
               int csize = (int)(chainSize);
               double r = 0.000000001;
               double fint [] = new
double[csize];
               double [] fspring2 =
f_spring(chainSize, beadSize, temperature);
               double [] fbead2 =
f_bead(chainSize, beadSize, temperature);
               DecimalFormat df1 = new
DecimalFormat("#,##0.000");
                              try{
                                     for(int
i = 0; i <= csize; i++){

       fint[i] = fspring2[i] + fbead2[i];

       r = r + 0.000000001;
                                     }

                              }catch(NullPoi
nterException npe){
                                     }catch(
ArrayIndexOutOfBoundsException aiobe){}

               return fint;
       }
       private double []  f_spring(double
chainSize, double beadSize, double
temperature) {

               chainSize =
Double.parseDouble(chainSizeTextField.getTex
t());
               int csize = (int)(chainSize);
               beadSize =
(Double.parseDouble(beadSizeTextField.getTex
t()))*(0.000000001);
               temperature =
Double.parseDouble(temperatureTextField.getT
ext());
```

```
               double r = 0.000000001;
               double vspring [] = new
double[csize];
               double fspring [] = new
double[csize];
               try{

                       for(int i = 0; i <=
1000; i++){

                              fspring[i] =
(((-1)*boltzmann_constant*temperature*k)/
(beadSize*beadSize))*(r);
                              r = r +
0.000000001;

                       }
               }catch(ArrayIndexOutOfBoundsE
xception aiobe){}
               return fspring;

        }

         private double []  f_bead(double
chainSize, double beadSize, double
temperature){

               chainSize =
Double.parseDouble(chainSizeTextField.getTex
t());
               int csize = (int)(chainSize);
                beadSize =
(Double.parseDouble(beadSizeTextField.getTex
t()))*(0.000000001);
               temperature =
Double.parseDouble(temperatureTextField.getT
ext());
                double r = 0.000000001;
               double vbead[] = new
double[csize];
               double fbead[] = new
double[csize];
               try{
                       for(int i =0; i <=
csize ; i++ ){
                              double cond []
= new double[csize];
                              cond[i] =
(r/beadSize);

                              if(cond[i] <
(Math.pow(2, (1/6)))){


       fbead[i] =
(boltzmann_constant*temperature)*(((12*(Math
.pow(beadSize, 12)))/(Math.pow(r, 13))) -
(((6*(Math.pow(beadSize, 6)))/(Math.pow(r,
7)))));

                              }

                              else{

       fbead[i] = 0;

                              }
                              r = r +
0.000000001;

                       }
```

```
               }catch(ArrayIndexOutOfBoundsE
xception aiobe){}
               return fbead;
        }

         private double []  f_electric(double
qnet, double electricField){

               chainSize =
Double.parseDouble(chainSizeTextField.getTex
t());
               int csize = (int)(chainSize);
               beadSize =
(Double.parseDouble(beadSizeTextField.getTex
t()))*(0.000000001);
               electricField =
(Double.parseDouble(electricFieldTextField.g
etText()))*(100);

               double felectric[] = new
double[csize];
               double [] potential_energy =
new double[csize];
               DecimalFormat df1 = new
DecimalFormat("#,##0.000");

               int channelSize = 100;

               try{
                       for(int i =0; i <=
csize ; i++ ){
                              for(int cs =
0; cs<=channelSize; cs++){
                                     if((cs
< 41) || (cs > 60)){

       felectric[i] =
qnet*electricField_initial;

                                     }

                                     else{

       felectric[i] =
qnet*(electricField_initial - electricField);

                                     }
                              }
                       }
               }catch(ArrayIndexOutOfBoundsE
xception aiobe){}
               return felectric;
       }

       private double []
random_force(double temperature, double
beadSize){

                       chainSize =
Double.parseDouble(chainSizeTextField.getTex
t());
                       int csize = (int)
(chainSize);
                       temperature =
Double.parseDouble(temperatureTextField.getT
ext());
                       beadSize =
(Double.parseDouble(beadSizeTextField.getTex
t()))*(0.000000001);
                       double randforce[] =
new double[csize];
```

```
                      DecimalFormat df1 =
new DecimalFormat("#,##0.000");


               try{
                      for(int i =0; i <=
csize ; i++ ){


                              double
randnum4 = 0;
                              int high = 1;
                              int low = -1;
                              int randnum1 =
number.nextInt( high - low + 1) + low;
                              double
randnum2 = randnum1; //random -1, 0, or 1
                              double
randnum3 = Math.random(); //random numbers
between 0 and 1
                              if((randnum2
== 0) && (randnum3 <= 0.5)){

       randnum4 = -1.0;
                              }
                              if((randnum2
== 0) && (randnum3 >= 0.51)){

       randnum4 = 1.0;
                              }

                              else{

       randnum4 = (randnum2*randnum3);
//random numbers (-1,1) problem: -1 and 1
must also be included
                              }

                              randforce[i] =
(boltzmann_constant*temperature/beadSize)*(M
ath.sqrt((6*transit_time)/delta_t))*(randnum
4);

                      }

               }catch(ArrayIndexOutOfBoundsE
xception aiobe){}
               return randforce;
       }


       public static void main(String[]
args) {
               SplashScreen splash = new
SplashScreen(10000);
               splash.showSplash();
               new SP().show();

       }//end of main

       public class ButtonHandler implements
ActionListener {
               public void
actionPerformed( ActionEvent e ) {

               }
       }//end of ButtonHandler

}//end of public class SP()

/*SplashScreen.java*/
```

```

import java.awt.*;
import javax.swing.*;

public class SplashScreen extends JWindow {

    private int duration;

    public SplashScreen(int d) {
        duration = d;
    }

    public void showSplash() {

        JPanel content =
(JPanel)getContentPane();
        content.setBackground(Color.white);

        int width = 280;
        int height =350;
        Dimension screen =
Toolkit.getDefaultToolkit().getScreenSize();
        int x = (screen.width-width)/2;
        int y = (screen.height-height)/2;
        setBounds(x,y,width,height);

        // Build the splash screen
        JLabel label = new JLabel(new
ImageIcon("dna2.gif"));
        JLabel maintitle = new JLabel
                ("Simulation of DNA Motion
in Entropic Trapping", JLabel.CENTER);
        maintitle.setFont(new Font("Sans-
Serif", Font.BOLD, 12));
       JLabel author = new JLabel
                ("Gorospe 2008",
JLabel.CENTER);
        author.setFont(new Font("Sans-
Serif", Font.BOLD, 10));
        content.add(label,
BorderLayout.CENTER);
        content.add(maintitle,
BorderLayout.NORTH);
       content.add(author,
BorderLayout.SOUTH);
       Color fontcol = new Color(20, 120,
150,  125);
        content.setBorder(BorderFactory.crea
teLineBorder(fontcol, 5));

        // Display it
        setVisible(true);

        // Wait while loading resources
        try { Thread.sleep(duration); }
catch (Exception e) {}

        setVisible(false);

    }

    public void showSplashAndExit() {

        showSplash();
        System.exit(0);

    }

    public static void main(String[] args) {
        SplashScreen splash = new
SplashScreen(10000);
        splash.showSplash();

    }
}

/*ExampleFileFilter.java*/

/*
 * @(#)ExampleFileFilter.java 1.16 04/07/26
 *
 * Copyright (c) 2004 Sun Microsystems, Inc.
All Rights Reserved.
 *
 * Redistribution and use in source and
binary forms, with or without
 * modification, are permitted provided that
the following conditions are met:
 *
 * -Redistribution of source code must
retain the above copyright notice, this
 *  list of conditions and the following
disclaimer.
 *
 * -Redistribution in binary form must
reproduce the above copyright notice,
 *  this list of conditions and the
following disclaimer in the documentation
 *  and/or other materials provided with the
distribution.
 *
 * Neither the name of Sun Microsystems,
Inc. or the names of contributors may
 * be used to endorse or promote products
derived from this software without
 * specific prior written permission.
 *
 * This software is provided "AS IS,"
without a warranty of any kind. ALL
 * EXPRESS OR IMPLIED CONDITIONS,
REPRESENTATIONS AND WARRANTIES, INCLUDING
 * ANY IMPLIED WARRANTY OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE
 * OR NON-INFRINGEMENT, ARE HEREBY EXCLUDED.
SUN MIDROSYSTEMS, INC. ("SUN")
 * AND ITS LICENSORS SHALL NOT BE LIABLE FOR
ANY DAMAGES SUFFERED BY LICENSEE
 * AS A RESULT OF USING, MODIFYING OR
DISTRIBUTING THIS SOFTWARE OR ITS
 * DERIVATIVES. IN NO EVENT WILL SUN OR ITS
LICENSORS BE LIABLE FOR ANY LOST
 * REVENUE, PROFIT OR DATA, OR FOR DIRECT,
INDIRECT, SPECIAL, CONSEQUENTIAL,
 * INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER
CAUSED AND REGARDLESS OF THE THEORY
 * OF LIABILITY, ARISING OUT OF THE USE OF
OR INABILITY TO USE THIS SOFTWARE,
 * EVEN IF SUN HAS BEEN ADVISED OF THE
POSSIBILITY OF SUCH DAMAGES.
 *
 * You acknowledge that this software is not
designed, licensed or intended
 * for use in the design, construction,
operation or maintenance of any
 * nuclear facility.
 */

/*
 * @(#)ExampleFileFilter.java 1.16 04/07/26
 */
```

```
import java.io.File;
import java.util.Hashtable;
import java.util.Enumeration;
import javax.swing.*;
import javax.swing.filechooser.*;

/**
 * A convenience implementation of
FileFilter that filters out
 * all files except for those type
extensions that it knows about.
 *
 * Extensions are of the type ".foo", which
is typically found on
 * Windows and Unix boxes, but not on
Macinthosh. Case is ignored.
 *
 * Example - create a new filter that
filerts out all files
 * but gif and jpg image files:
 *
 *     JFileChooser chooser = new
JFileChooser();
 *     ExampleFileFilter filter = new
ExampleFileFilter(
 *                   new String{"gif",
"jpg"}, "JPEG & GIF Images")
 *
chooser.addChoosableFileFilter(filter);
 *     chooser.showOpenDialog(this);
 *
 * @version 1.16 07/26/04
 * @author Jeff Dinkins
 */
public class ExampleFileFilter extends
FileFilter {

    private static String TYPE_UNKNOWN =
"Type Unknown";
    private static String HIDDEN_FILE =
"Hidden File";

    private Hashtable filters = null;
    private String description = null;
    private String fullDescription = null;
    private boolean
useExtensionsInDescription = true;

    /**
     * Creates a file filter. If no filters
are added, then all
     * files are accepted.
     *
     * @see #addExtension
     */
    public ExampleFileFilter() {
        this.filters = new Hashtable();
    }

    /**
     * Creates a file filter that accepts
files with the given extension.
     * Example: new ExampleFileFilter("jpg");
     *
     * @see #addExtension
     */
    public ExampleFileFilter(String
extension) {
        this(extension,null);
    }

    /**
     * Creates a file filter that accepts
the given file type.
     * Example: new ExampleFileFilter("jpg",
"JPEG Image Images");
     *
     * Note that the "." before the
extension is not needed. If
     * provided, it will be ignored.
     *
     * @see #addExtension
     */
    public ExampleFileFilter(String
extension, String description) {
        this();
        if(extension!=null)
addExtension(extension);
        if(description!=null)
setDescription(description);
    }

    /**
     * Creates a file filter from the given
string array.
     * Example: new ExampleFileFilter(String
{"gif", "jpg"});
     *
     * Note that the "." before the
extension is not needed adn
     * will be ignored.
     *
     * @see #addExtension
     */
    public ExampleFileFilter(String[]
filters) {
        this(filters, null);
    }

    /**
     * Creates a file filter from the given
string array and description.
     * Example: new ExampleFileFilter(String
{"gif", "jpg"}, "Gif and JPG Images");
     *
     * Note that the "." before the
extension is not needed and will be ignored.
     *
     * @see #addExtension
     */
    public ExampleFileFilter(String[]
filters, String description) {
        this();
        for (int i = 0; i < filters.length;
i++) {
            // add filters one by one
            addExtension(filters[i]);
        }
        if(description!=null)
setDescription(description);
    }

    /**
     * Return true if this file should be
shown in the directory pane,
     * false if it shouldn't.
     *
     * Files that begin with "." are ignored.
     *
     * @see #getExtension
     * @see FileFilter#accepts
     */
    public boolean accept(File f) {
       if(f != null) {
           if(f.isDirectory()) {
              return true;
           }
           String extension =
getExtension(f);
           if(extension != null &&
filters.get(getExtension(f)) != null) {
              return true;
           };
       }
       return false;
    }

    /**
     * Return the extension portion of the
file's name .
     *
     * @see #getExtension
     * @see FileFilter#accept
     */
     public String getExtension(File f) {
       if(f != null) {
           String filename = f.getName();
           int i =
filename.lastIndexOf('.');
           if(i>0 && i<filename.length()-1)
{
              return
filename.substring(i+1).toLowerCase();
           };
       }
       return null;
    }

    /**
     * Adds a filetype "dot" extension to
filter against.
     *
     * For example: the following code will
create a filter that filters
     * out all files except those that end
in ".jpg" and ".tif":
     *
     *   ExampleFileFilter filter = new
ExampleFileFilter();
     *   filter.addExtension("jpg");
     *   filter.addExtension("tif");
     *
     * Note that the "." before the
extension is not needed and will be ignored.
     */
    public void addExtension(String
extension) {
       if(filters == null) {
           filters = new Hashtable(5);
       }
       filters.put(extension.toLowerCase(),
this);
       fullDescription = null;
    }


    /**
     * Returns the human readable
description of this filter. For
     * example: "JPEG and GIF Image Files
(*.jpg, *.gif)"
```

```
     *
     * @see setDescription
     * @see setExtensionListInDescription
     * @see isExtensionListInDescription
     * @see FileFilter#getDescription
     */
    public String getDescription() {
        if(fullDescription == null) {
            if(description == null ||
isExtensionListInDescription()) {
                fullDescription =
description==null ? "(" : description + " (";
                // build the description from
the extension list
                Enumeration extensions =
filters.keys();
                if(extensions != null) {
                    fullDescription += "." +
(String) extensions.nextElement();
                    while
(extensions.hasMoreElements()) {
                        fullDescription +=
", ." + (String) extensions.nextElement();
                    }
                }
                fullDescription += ")";
            } else {
                fullDescription = description;
            }
        }
        return fullDescription;
    }

    /**
     * Sets the human readable description
of this filter. For
     * example: filter.setDescription("Gif
and JPG Images");
     *
     * @see setDescription
     * @see setExtensionListInDescription
     * @see isExtensionListInDescription
     */
    public void setDescription(String
description) {
        this.description = description;
        fullDescription = null;
    }

    /**
     * Determines whether the extension list
(.jpg, .gif, etc) should
     * show up in the human readable
description.
     *
     * Only relevent if a description was
provided in the constructor
     * or using setDescription();
     *
     * @see getDescription
     * @see setDescription
 * @see isExtensionListInDescription
     */
    public void
setExtensionListInDescription(boolean b) {
        useExtensionsInDescription = b;
        fullDescription = null;
    }

    /**
     * Returns whether the extension list
(.jpg, .gif, etc) should
     * show up in the human readable
description.
     *
     * Only relevent if a description was
provided in the constructor
     * or using setDescription();
     *
     * @see getDescription
     * @see setDescription
     * @see setExtensionListInDescription
     */
    public boolean
isExtensionListInDescription() {
        return useExtensionsInDescription;
    }
}
```

XI. ACKNOWLEDGMENT

I would like to dedicate this part of my thesis to everyone who has given me inspiration and strength to accomplish all these.

All praises and thanks to our **Lord God**, He is the source of everything. Thank You Lord for all the blessings You continue to shower us in every waking day. From the start, You have been a witness to all the sacrifices encountered while doing this thesis, and in return I would like to extend my gratitude for Your guidance up to the end.

Family. The persons who have been eye witnesses to all my "*puyatan* sessions" and tantrums. Thank you **Daddy** for all the support you've given me. I appreciate all the help especially when you accompanied me during my thesis defense. Thank you also for all the reminders you constantly give me. **Mommy**, thank you also for all the help you have given me, for waking up during my "*puyatan* sessions" to prepare hot chocolate and soup for me. Thank you for being concerned with my academics, because in that way I am reminded of my responsibilities in school. **Dad and Mama**, thank you for providing us what we need. I appreciate all your sacrifices and the values you have instilled in us. **Ate Roan**, eventhough you're far from us, you never forget to be our ATE. Thank you for all the encouragement and thank you for all the advice you've given me. Thank you for the laptop, it was really a big help to me. I was also surprised and touched when you called during my thesis defense. *Ayan Ate, natupad ko na pag-uwi mo sa June graduate nako.* **Ate Ia**, thank you for all the pressure you have given me (*haha*), for pushing me to finish this thesis. Though we had a lot of misunderstandings, you still find ways to make our sisterhood better and better. I strived to finish my studies and you served as one of my inspirations. **Isay**, super thank you for listening to all my rants and stories about everything! I know it's hard to be my roommate, but still you have been nice to me. Thanks for understanding my craziness and for supporting me in whatever I do. You're the first person to know everything (well, mostly) I am planning to do, thanks for all the advice and for being one big ear (haha). **Yan-Yan**, thanks for allowing me to annoy you because you're so cute when you're irritated by me (haha). Thanks for all the *kakulitan* and for always reminding me to finish my thesis so that we can play outside and ride your bike. Thanks for always making our home lively and fun, Baby Maya. Thanks also to all my relatives for giving me their full support—the Gorospe and Eugenio clans. I am very thankful for being part of this family, I love you all!

Friends. College would not be fun without these people who have sacrificed with me especially during hell weeks and katoxican. **Angge**, thank you for being there from the start and for sharing with me your stories and problems in life. I am very proud of you because after everything that has happened, you're still strong enough to face all of them. Thank you for all the encouragement when I'm feeling hopeless and for striving hard to finish our requirements in school because in that way, we are also motivated to accomplish them. I know you'll be successful in your career because I can see the determination in you. You are one of the reasons why college life was fun and meaningful. Thanks Imps! **Farrah**, thank you for all your help and friendship. Like Angge, I know that you'll also go far because you have the skills to show to everyone. Thank you for sharing your knowledge on certain things, for being patient in answering my queries, and for listening to all my crazy stories. To my **Bubbles and Blossom**, thank you both for all the memories we've shared in our four years in this university. Both of you have been with me since our first year and I'm very fortunate to have you guys along with me. I have learned a lot from you and I will always be thankful for that. **Ginel**, for all the laughter you've shared with me and for being nice to Yan-Yan. Thanks for the friendship and you are also one of the persons that have been with me for these past years. Thank you also for sharing with me your stories and for being a constant ym buddy during late nights/early mornings. I may not be always available to accompany you due to my "addiction", but I'm just here if ever you need a friend. **Therese**, thank you for being an inspiration to the whole block because of your hardwork and dedication. Your independent and responsible attitude makes you an even better person. Thank you's also to my co-centennial graduates/blockmates: **Jackie** for being a good friend and a good company, thanks for all the laughter!; **April** for all the stories we've shared, **Donnel** for the kakulitan, **Mark** for the fishballs and pandesal (nasan na? haha), **Roy** the comsci god for answering my unending Java questions, **JC** for also helping me in Java and **Cromwell** for your company and stories. Thank you also to my other **Cooltoh** friends **Cathie**, **Christine**, **Jill, Pearl** and **Karen** (Happy Birthday!) for being wonderful and supportive friends. And to all my blockmates and classmates thank you for all the moments shared during our four years in college, may it be cramming moments, fun, sad or toxic. Thank you also to all my **highschool friends** from St. Paul for being good and supportive friends.

Mentors. Thank you **Sir Solano** for being an adviser who can give encouragement to an advisee. Your calmness had helped me to successfully finish this thesis. Thank you for all the help and inspirational words you have shared with me. To **Ma'am Sarah**, super thank you for helping me have a topic and for all the help in dealing with this thesis. I know I have been *makulit* with all my questions, but you were still patient with me. Thank you for your time and effort, I appreciate it a lot. This thesis would not be possible without Sir Solano and especially you, Mam Sarah. Thank you also to **Sir Patiño** for being my second-adviser, and for all the comments and suggestions for the improvement of my thesis. Thank you also to all my other mentors who have been with us since our first years, **Doc Magboo**, **Ma'am Sheila**, **Sir Co**, **Sir Chua**, **Sir Baes**, among others. And thank you to **Ate Eden** for allowing me to borrow thesis and for being accommodating to the students eventhough you always ask me about…

Others. Thank you to **Bokbok**, **Pabling** and others especially the **Hanky Panky group and Company** for being part of my college life. My four years in UP Manila would not be the same without you guys (haha). You have been important personalities in my college life. Thanks for being our inspiration and source of happiness. You may not know it, but I am thankful that you have been part of my journey here in UP. Of course, thank you to the **San Miguel Beermen/ Magnolia Beverage Masters** for being my inspiration because I really feel better everytime I see you guys win or just even play basketball. Your passion in your work has inspired me to do well also in my studies (haha drama). But seriously, you guys have also been a part of my college life because watching you makes me happy and lightens my toxic school life. I would also like to include the SLR and the other teams for putting up great games which I enjoy and lessens all the stress I am feeling in school. Hahaha! *At talagang sinama ko sila dito, di ba?* Thank you.. thank you.. thank you.. family, friends, enemies, everyone! You were all my inspiration and I will be forever thankful for that. Thank You, Lord!

And whatsoever ye do in word or deed, do all in the name of
the Lord Jesus, giving thanks to God and the Father by him.
— Colossians 3.17