

Running head: COMPUTATIONAL MODELING OF PARASITE INTERACTION

Computational Modeling of the Interaction of the
T. cruzi Parasite and its Environment

Cailin K. Andruss

Westmont College

MS#1019

955 La Paz Rd.

Santa Barbara, CA 93108

Abstract

The Virtual Parasite Project (VPP) aims to design an *in silico* laboratory that provides insight into the parasite-host dynamics. It focuses on the behavior of the *T. cruzi* parasite, the causative agent of Chagas' disease, using host-parasite models of biophysical interactions. The goal of project was to design a prototype environment that could be used in future development of the project. Development of this component of the project began in the programming language Scheme and used a VPP developed visualizer called SimRender. Following numerous programming problems with Scheme, the this project was completed using NetLogo instead. This turned out to be an ideal programming environment including an intuitive and well documented programming language, a customizable interface, and built-in visualization component.

Keywords: Modeling, *Trypanosoma cruzi*, Parasite, Three Dimentional, Scheme, SimRender, NetLogo, Virtual Parasite Project, Chagas' Disease

Computational Modeling of the Interaction of the *T. cruzi* Parasite and its Environment

Chagas' disease, caused by infection with the *Trypanosoma cruzi* parasite, is deadly and wide spread, with 16-18 million people infected and another 120 million at risk for infection [1]. Unfortunately, very little is known about the parasite that could be used in treatment and prevention. The goal of the Virtual Parasite Project (VPP) is to design an *in silico* laboratory that provides insight into the parasite-host dynamics of the *T. cruzi* parasite with its host by initially modeling the host-parasite biophysical interactions [2]. To that end, the goal of this Bioinformatics and Bioengineering Summer Institute (BBSI) project was to implement a prototype model that could be used in the future development of the project.

Method

During the first summer of the project, I wrote a program in Scheme [3] (a dialect of Lisp) that would produce a binary output file readable by the three dimensional graphical simulator SimRender [4]. The Scheme program included functions for updating the position of each cell and a basic function that models the elastic collisions of the parasites with the walls of the world. Over the subsequent academic year, I added several aspects of physics to the basic movement functions, including gravity and buoyancy. The force of gravity on a single parasite is given by the equation:

$$\vec{F}_{grav} = -mg\hat{k}$$

where m is the mass of the parasite, calculated by multiplying the volume of the parasite by the density of the parasite, and g is the acceleration of gravity. Similarly, the force of buoyancy on a single parasite is given by the equation:

$$\vec{F}_{buoy} = \frac{4\pi a^3 \rho_{fluid} g}{3} \hat{k}$$

where a is the radius of the parasite, ρ is the density of the fluid, and g is the acceleration of gravity. In essence this calculates the mass of fluid displaced by the parasite by finding the volume of the parasite and multiplying that by the density of the fluid, and then multiplies the result by the acceleration of gravity. Since gravity and buoyancy are both forces, I multiplied them by the mass of the parasite to get the acceleration of the parasite. Since acceleration is the amount a velocity changes each timestep, multiplying acceleration by the length of the timestep and then adding the result to the z-direction of the parasite's velocity vector changes the parasite's velocity to reflect how it is affected by the forces of gravity and buoyancy.

I also rewrote the program to increase efficiency by streamlining computations and eliminating errors, making the code significantly neater and more concise. At the beginning of the final summer I started adding functions so that the parasites would detect host cells within a van der Waal's radius of themselves, 'stick' to the host by finding a location for the parasite along its velocity vector such that the surfaces of the cells were touching, and then invade based on the cell cycle state of the host cell.

For example, to choose a host, a parasite calls the following function:

```
(define (choose-host p s)
  (local ((define (close-host? h)
            (and (symbol=? (cell-type h) 'Host-cell)
                 (< (distance-s p h) VDW-DIST))))
          (define possible-hosts (list->vector (filter close-host? s)))
          (define random-ref (random (add1 (vector-length possible-hosts))))
          (define random-host
            (cond
             [(or (= (vector-length possible-hosts) 0)
```

```

      (= random-ref (vector-length possible-hosts))) null]
    [else (cell-indx (vector-ref possible-hosts random-ref))]))))

(make-cell (cell-indx p) (cell-pos p) (cell-vel p) (cell-rad p)
           (cell-chrg p) (cell-type p) random-host))

```

The function takes a parasite and a list of all the cells in the world as arguments. It filters the list to so that only hosts within a van der Waals radius (VDW-DIST) remain, and then chooses a random host, or no host. It then returns the original parasite with the cell-state chaged to either the index of the chosen host or null if no host was chosen.

Part of the way through the summer, I was introduced to the program NetLogo 3D Preview 3 [5]. This program incorporates an intuitive language, customizable interface, and built-in visulization system. I contacted Dr. Uri Wilensky, the author of one of the sample models included with the 2D version of NetLogo, and asked him if he was planning to port it to NetLogo 3D. He gave me contact information of Michelle Wilkerson who was working on converting several of the 2D models to 3D. She sent me the draft of the program GasLab Gas in a Box [6]. Using this as a starting point, I re-wrote the VPP simulation in NetLogo, focusing on sticking, invasion, and replication of parasites.

For example, the function for choosing a random host that I showed earlier in Scheme would be the following in NetLogo:

```

to choose-host
  let myhostindex -1
  if ( random 1 = 0 )
  [
    ask particles with [ kind = "host" ]
    [ set myhostindex ( who-of ( one-of particles in-radius vdw-distance ) ) ]
    set state myhostindex
  ]
end

```

The fuction, which is called by a parasite, chooses a host within the van der Waals radius, and sets the state of the parasite to the index of that host. To allow for the possibility that no host is chosen, the host is only chosen if the result of asking for a random number between zero and one

is zero. As you can see many functions like “one-of” and “in-radius” are already defined in the NetLogo language. Also, NetLogo uses a language that is object oriented rather than purely function and recursion based like Scheme, making it easier to manage collections of different kinds of parasites, and to execute commands one by one in order.

Results

In both the Scheme/SimRender and the NetLogo models, parasites are represented by red spheres, and host cells by yellow spheres, as shown in Figure 1.

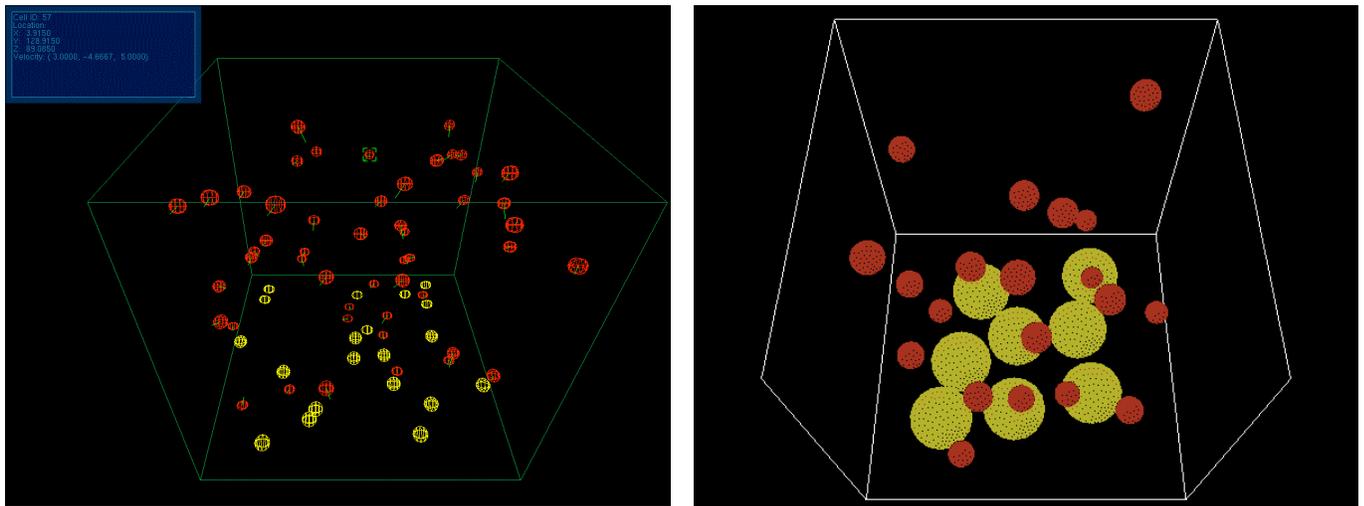


Figure 1: A view of the simulation in SimRender is on the left, and a view of the simulation in NetLogo is on the right.

Parasites collide elastically with the walls of the world, each other, and host cells. If a parasite collides with a host cell, it chooses randomly whether to stick or to bounce off, and implements the appropriate function. If it sticks, then it chooses randomly whether or not to invade, and if it decides to invade, the cell turns purple and begins entering the host cell, as seen in Figure 2.

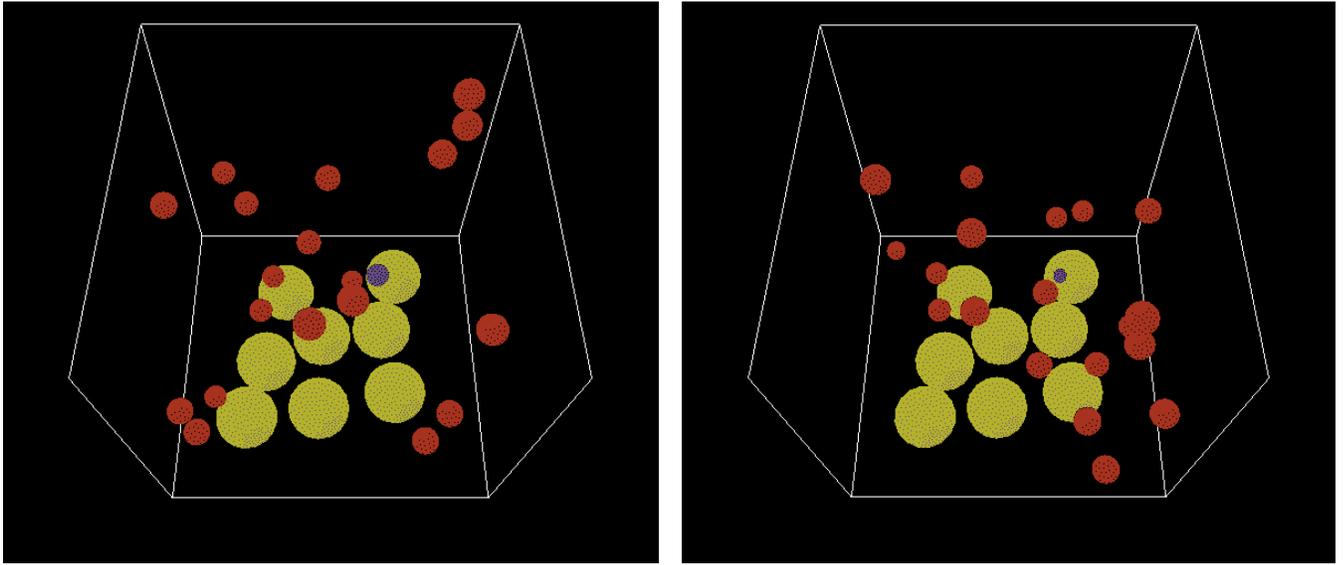


Figure 2: on the left a parasite has just decided to invade (purple sphere), and on the right the same parasite is almost through invading the host cell.

Once inside, the cell replicates by creating a random number of new cells between two and ten, and then lyses the cell, spilling the new cells into the world, seen in Figure 3.

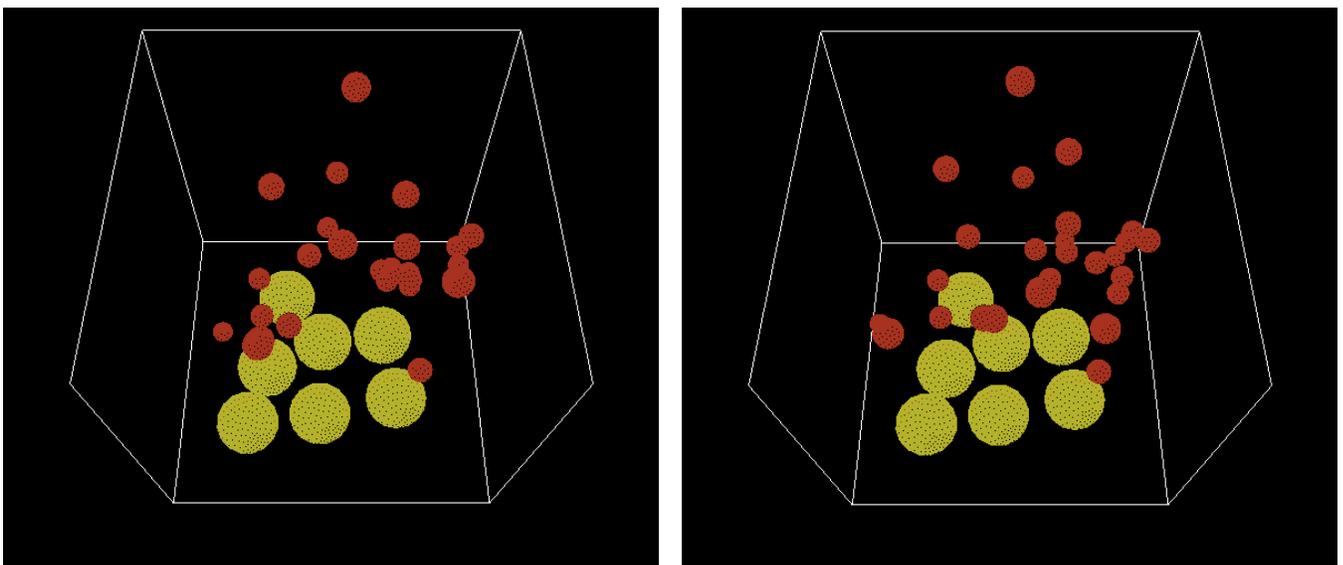


Figure 3: on the left, a cluster of parasites remain where there was once a host cell (see Figure 2). On the right, the cluster has expanded as the parasites begin to move into the world.

Since only one parasite can invade a given host, any other parasites that are stuck to the host cell when it lyses are told to swim away in a random direction.

The interface (see Figure 4) includes sliders for setting several variables including the number of parasites, the number of hosts, the initial speed of the parasites, and the color theme. Netlogo allows the user to plot various dynamically changing graphs. My program plots the number of parasites as the simulation progresses. A switch controls whether a movie is recorded of the simulation as it runs [7].

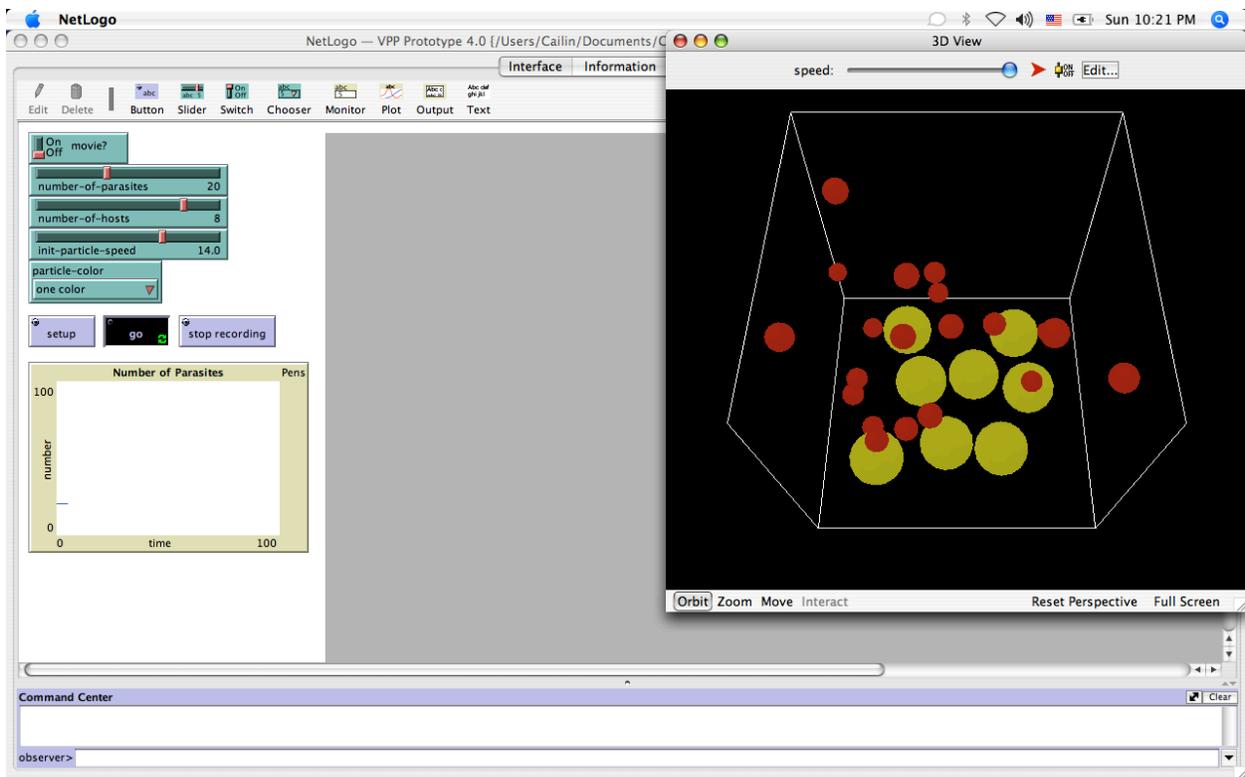


Figure 4: the NetLogo interface for the VPP simulation (program is currently running).

Discussion

Although Scheme is awkward to use in some cases and has very few modeling specific functions built in, it is SimRender that has the most serious limitations. It does not allow for the creation of new cells in the middle of the simulation, and is difficult to use in that compiling it is tricky and it does not work on Macintosh. The NetLogo environment, on the other hand, is ideal for this type of simulation. The programming language is intuitive and very well documented, with a great deal of built-in functionality. The interface is easily customizable and allows for adjustment of the program by people who are not familiar with the language. It even allows for recording videos of the simulation and exporting plots, and facilitates creation of new cells mid-simulation without issue. Other things, like randomizing the list of cells before updating at each time step so that no cell has the advantage of going first, are also done automatically by NetLogo. It is easier to control the relative sizes of the world and cells in SimRender, as well as the lengths of time. However, it should be possible to operationally define relative units of size and time in NetLogo with some research and testing.

The prototype itself is a good start in that parasites move and react to each other, stay contained within the world, invade host cells, and replicate. However, there are several problems that need to be resolved. First, the new parasites that are created after the invasion of a host cell do not react in the same way as the original parasites. They do not stick to host cells, and sometimes they bang repeatedly into a wall until they break through. Also, while a great deal of progress was made on adding gravity and buoyancy calculations to the simulation, these

functions conflict with the functions controlling collisions, leading to odd behaviors of the parasites. Other functions controlling the host cell cycle and replication, van der Waals interactions with host cells and charge-charge interactions with parasites, as well as randomized velocity changes still need to be added to the model.

The NetLogo environment will allow the model to be refined and expanded easily in ways that may not have been possible with Scheme and SimRender. For example, NetLogo allows the user to define ways of interacting with the simulation, like moving parasites and hosts around using click and drag.

In summary, although NetLogo was found late in the project, it made the BBSI project more successful than expected. Not only did the project result in a prototype model that could be used in future development, but it also led to the discovery of a programming environment which could be used for the main Virtual Parasite Project.

Acknowledgements

This work was supported by grant EEC0234104 from the NSF/NIH Bioinformatics and Bioengineering Summer Institute program. The author would like to thank Dr. Tarynn M. Witten for her guidance throughout the BBSI program especially during the portions at Virginia Commonwealth University, as well as Dr. Wayne Iba for his help, especially during the academic year at Westmont College. Many thanks also to Dr. Gary An who suggested using NetLogo, and to Dr. Uri Wilensky & Michelle Wilkerson, who provided the draft of the 3D version of GasLab Gas in a Box. Without all of these people this project would not have been possible. Any questions about this paper should be addressed to the author at MS#1019, 955 La Paz Rd., Santa Barbara, CA 93108; or e-mailed to candruss@westmont.edu. The author's

project website can be found at: <http://ramsites.net/~ckandruss/> . The Virtual Parasite Project website is located at <http://www.vcu.edu/csbc/vpp> .

Figures

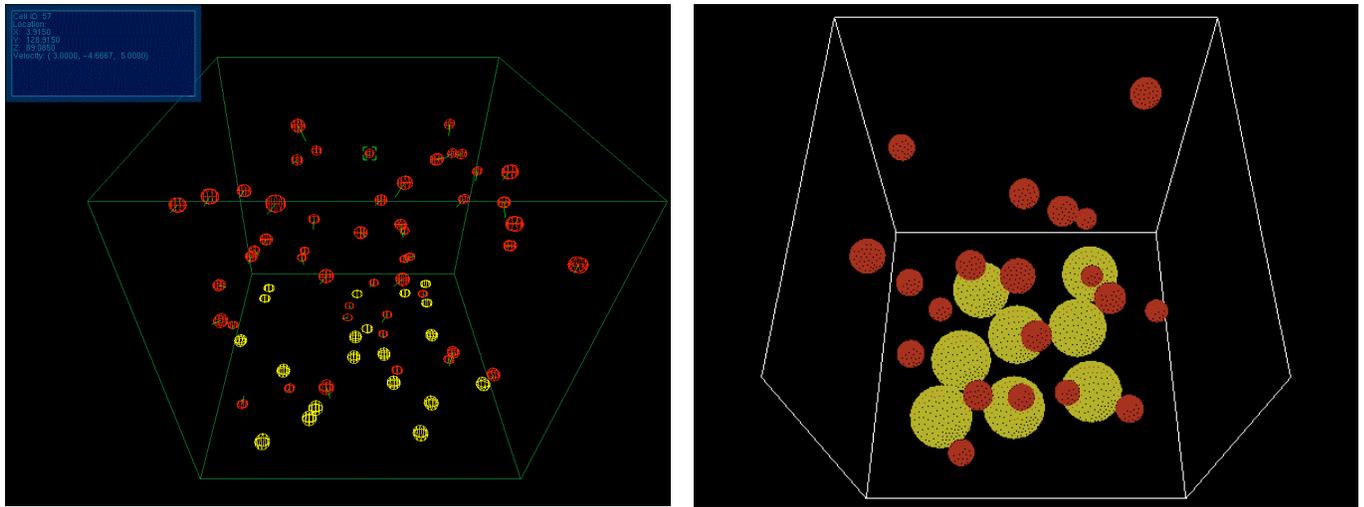


Figure 1: A view of the simulation in SimRender is on the left, and a view of the simulation in NetLogo is on the right.

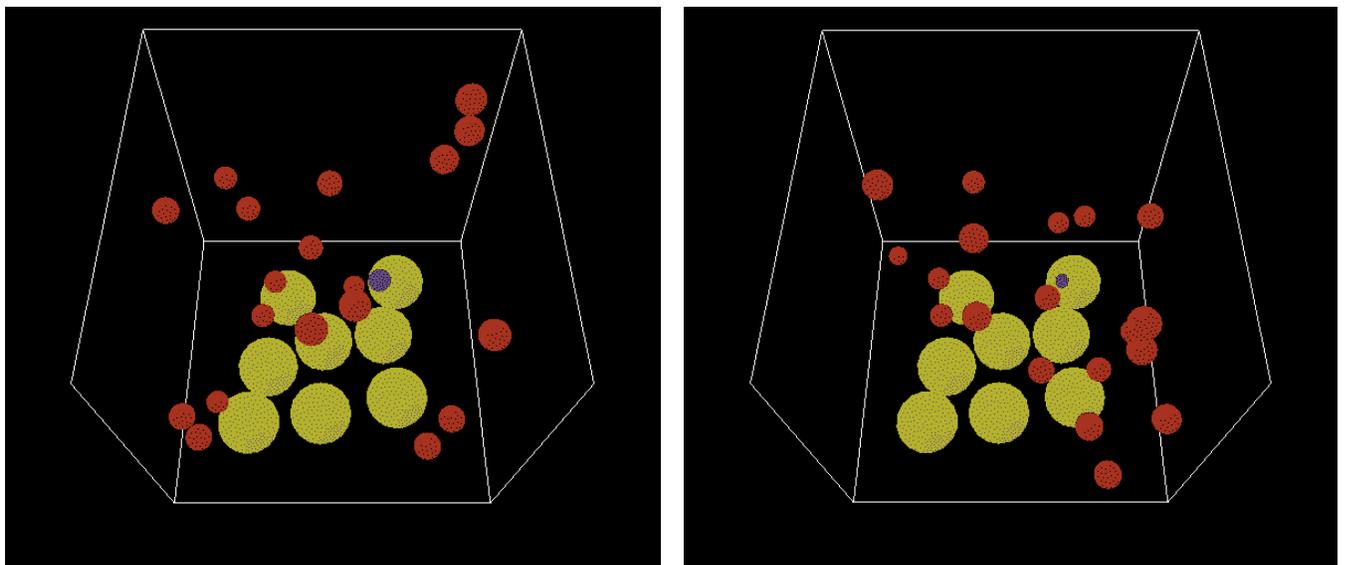


Figure 2: on the left a parasite has just decided to invade (purple sphere), and on the right the same parasite is almost through invading the host cell.

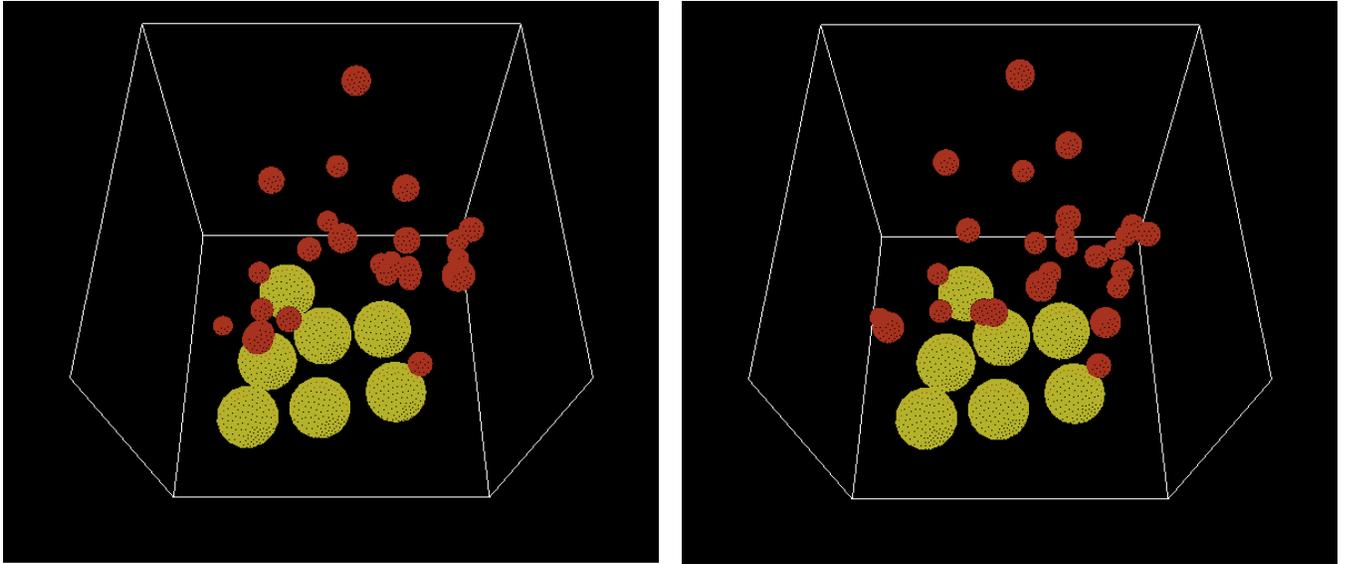


Figure 3: on the left, a cluster of parasites remain where there was once a host cell (see Figure 2). On the right, the cluster has expanded as the parasites begin to move into the world.

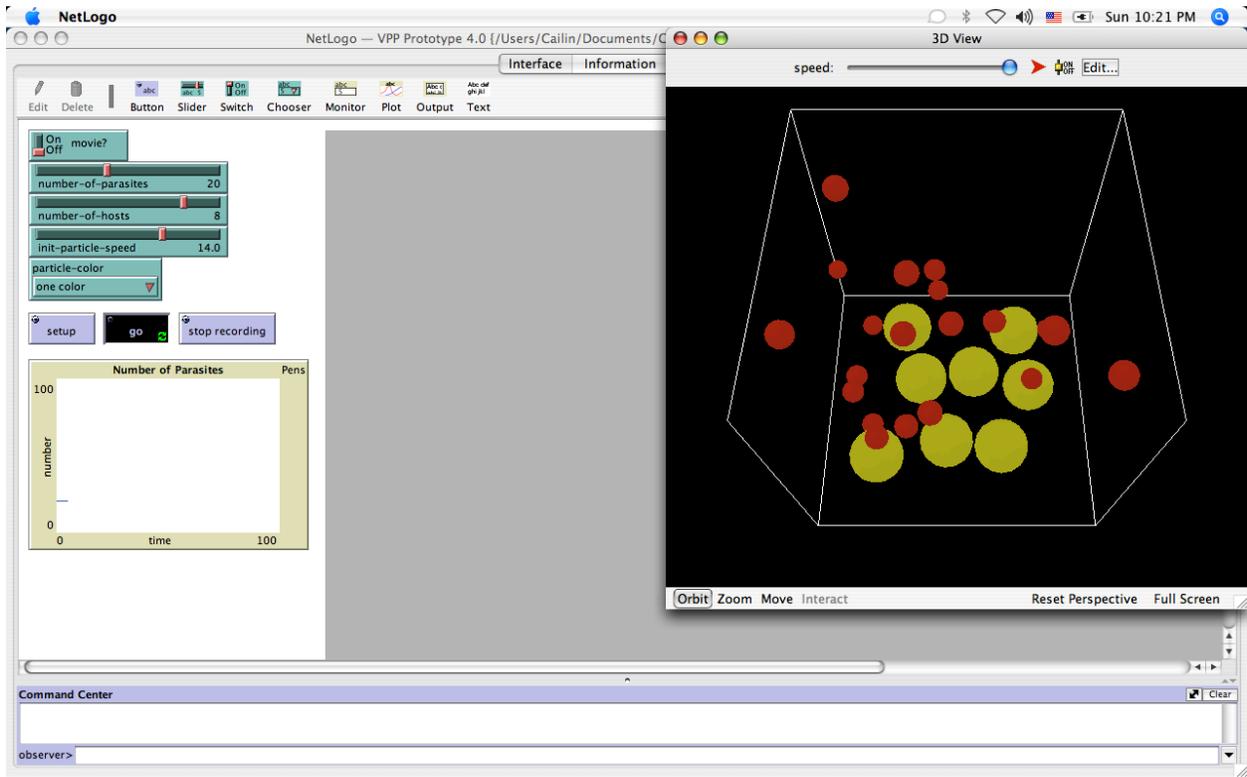


Figure 4: the NetLogo interface for the VPP simulation (program is currently running).

References

1. Special Programme for Research and Training in Tropical Diseases (TDR) (2002). Strategic Direction for Research: Chagas Disease, UNDP/World Bank/WHO, <http://www.who.int/tdr/diseases/chagas/files/direction.pdf>
2. Witten, T. M., et al. (2005). The Virtual Parasite Project – Towards a Biologically Sound Simulation Model of Parasite Dynamics: T-cruzi as a Prototype, Internal operations document (<http://www.vcu.edu/csbc/vpp>).
3. Programmed in the DrScheme programming environment, <http://www.plt-scheme.org/software/drscheme/> . For more information on the Scheme programming language visit <http://www.schemers.org/> .
4. SimRender (2004). In house CSBC software package for visualization of Virtual Parasite Program simulation data.
5. NetLogo 3D Preview 3 (2006). <http://ccl.northwestern.edu/netlogo/>
6. The model incorporates functions from the draft of the 3D version of the 2D model:
Wilensky, U. (2006). GasLab Gas in a Box. <http://ccl.northwestern.edu/>
Since it is not published yet, there is no official way to refer to it as of July 2006.
7. A sample video is available on the author's project website: <http://ramsites.net/~ckandruss/> ,
and the Virtual Parasite Project website: <http://www.vcu.edu/csbc/vpp/> .