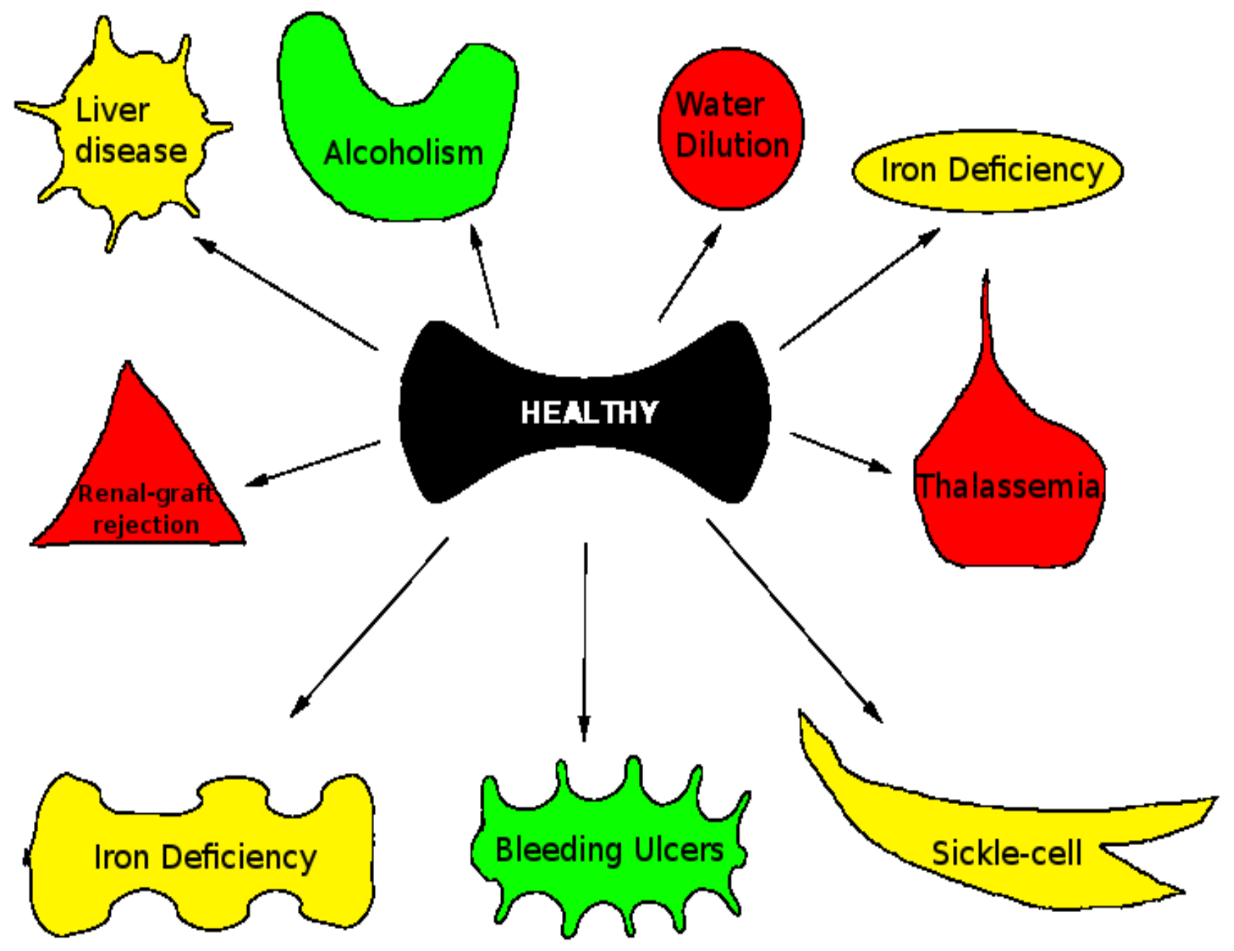
Modeling and simulation of red blood cell light scattering

Andrew Gearhart, Tarek Zohdi, Jim Demmel, Frans Kuypers

Research supported by Microsoft (Award #024263) and Intel (Award #024894) funding and by matching funding by U.C. Discovery (Award #DIG07-10227)

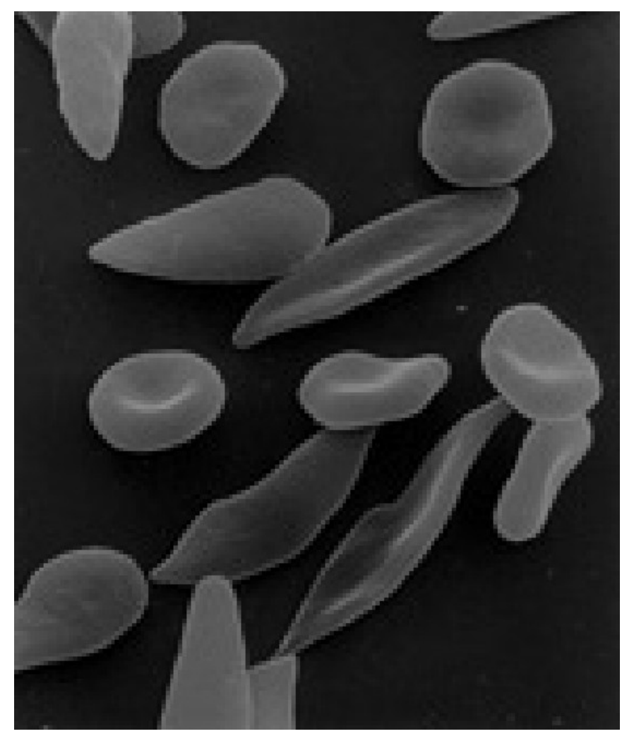
Red blood cell malformations



Due to various diseases and acute conditions, the shape and composition of erythrocytes (red blood cells, or RBCs) can vary widely within human samples.

The figure to the left illustrates a few of the geometric distortions observed in RBCs.

Traditional observation of RBCs requires a microscope and qualified observer. This is time-consuming and expensive.



As a case in point, sickle-cell anemia is a genetic disorder characterized by RBCs that have a "sickle" appearance (see figure to left).

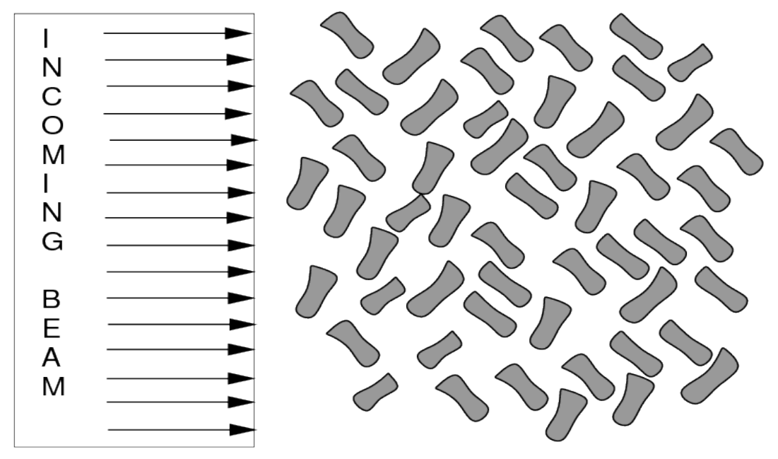
This malformed geometry prevents cells from traveling through small vessels, and significantly shortens average lifespan.

It is a disorder that is prevalent in central Africa (see figure below), and thus represents a significant health and economic burden upon developing countries.

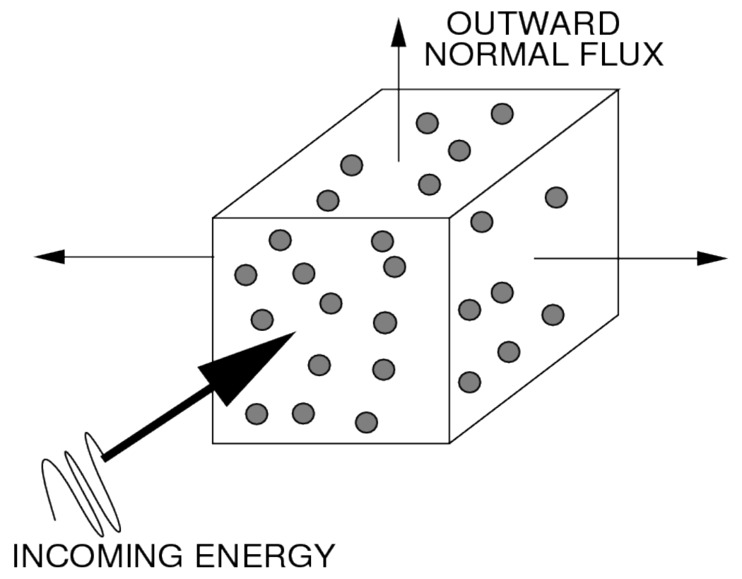
Noting the prevalence of RBC malformations and lack of skilled personnel in the Third world, a faster and cheaper approach to RBC analysis may possibly have significant global impact.

In the future, a program may be able to analyze the flux from a sample, and make a strong inference into any possible RBC disorders. This would significantly reduce the expense of actual human analysis.

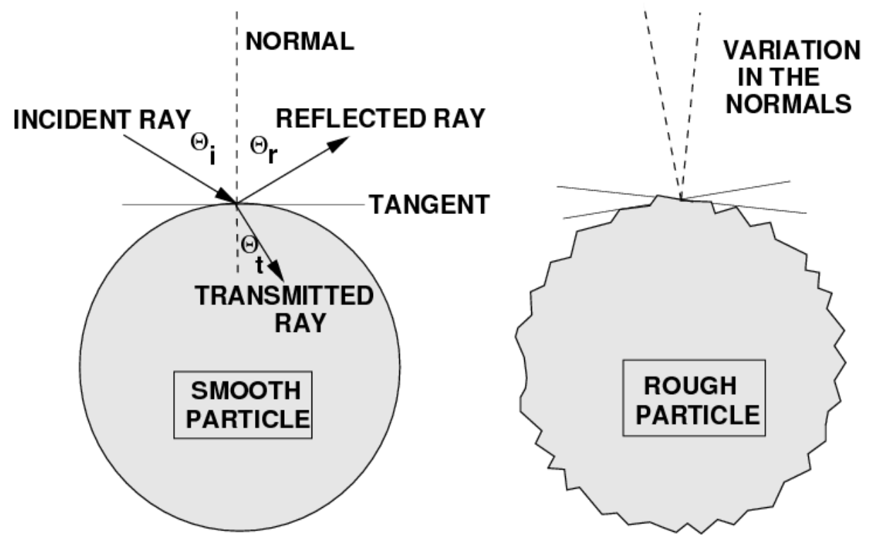
Modeling



To model this problem, we simulate a wavefront collision with a volume that is composed of randomly distributed and oriented objects.



A model of light scattering should accurately reproduce the flux out of a blood sample when under the influence of an outside energy source (a light beam, in this case).



In a simplified version of the simulation, we assume that the incoming energy can be represented as a set of ray vectors that impact and reflect from the objects in the sample. For each impact event, both the angle of incidence and energy of the transmitted ray must be calculated.

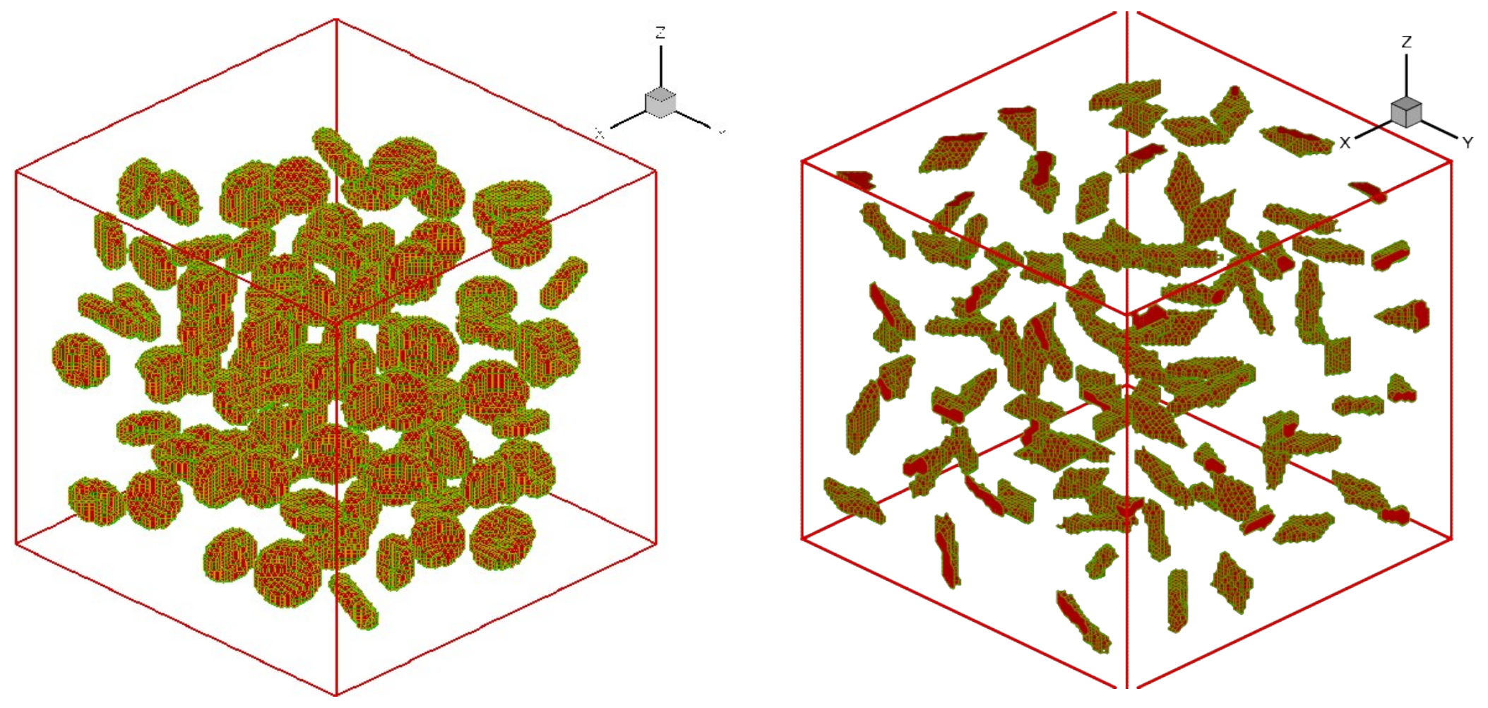
Modeling

RBC-type objects can be created via a generalized ellipsoid equation (see equation below). The experimentally-determined parameters c1,c2,c3 can be determined via experimental data, or modified to represent a malformed geometry.

$$F:=\left(\frac{2(z-z_0)}{b}\right)^2-\left(1-\frac{(x-x_0)^2+(y-y_0)^2}{b^2}\right)\left(c_0+c_1\left(\frac{(x-x_0)^2+(y-y_0)^2}{b^2}\right)+c_2\left(\frac{(x-x_0)^2+(y-y_0)^2}{b^2}\right)^2\right)^2=0$$

In the above equation, (x0,y0,z0) are the center of the object and b is the radius.

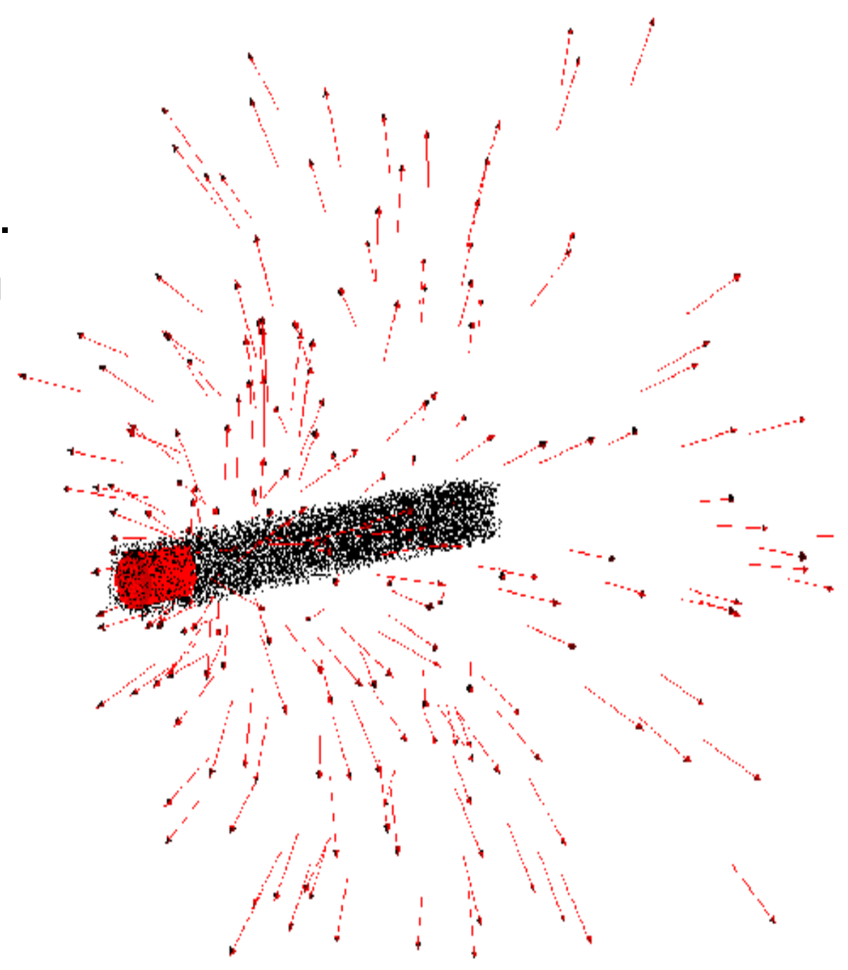
Objects are propagated within the simulated sample space using the classical random sequential addition algorithm. This algorithm places objects randomly within the space while avoiding overlaps by assuming that we are placing a sphere with a radius equal to the largest part of the actual experimental object. The figure to the lower left shows a space propagated with normal RBCs, while the figure on the lower right is filled with "sickle-like" objects.

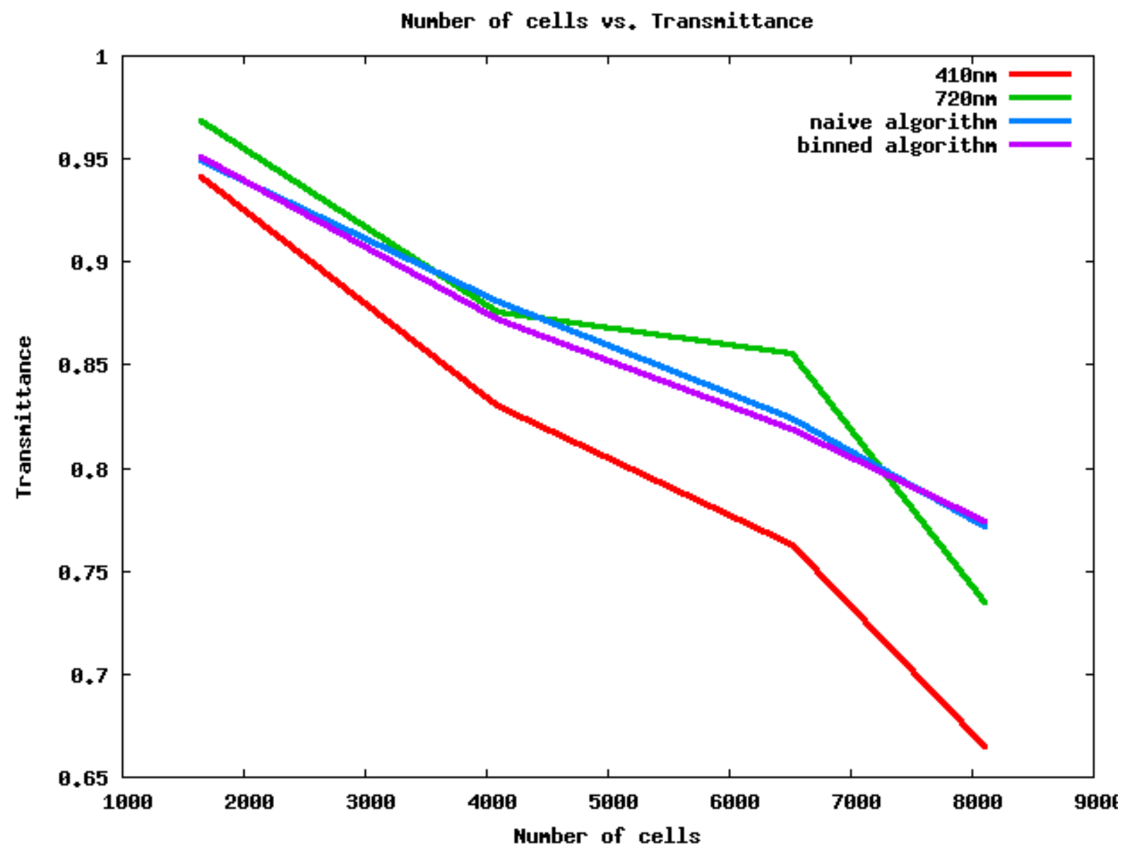


Ray simulation

In a thermodynamically-coupled version of the simulation, the energy of the transmitted ray is assumed to be directly translated into heat energy. This raised the temperature of the cell, resulting in a variable refractive index.

To the right is a figure depicting the initial stages of a beam traversal through a sample of blood cells. In the lower image, one can see the "flux" vectors that are generated by the simulation.





As can be seen in the figure at left, the simulations reproduce the results of Zohdi and Kuypers (2006) by relating quite closely to laboratory measures of laser transmittance through samples. The simulation models 720nm light more accurately due to the ratio of refractive indices used in the collision calculations.

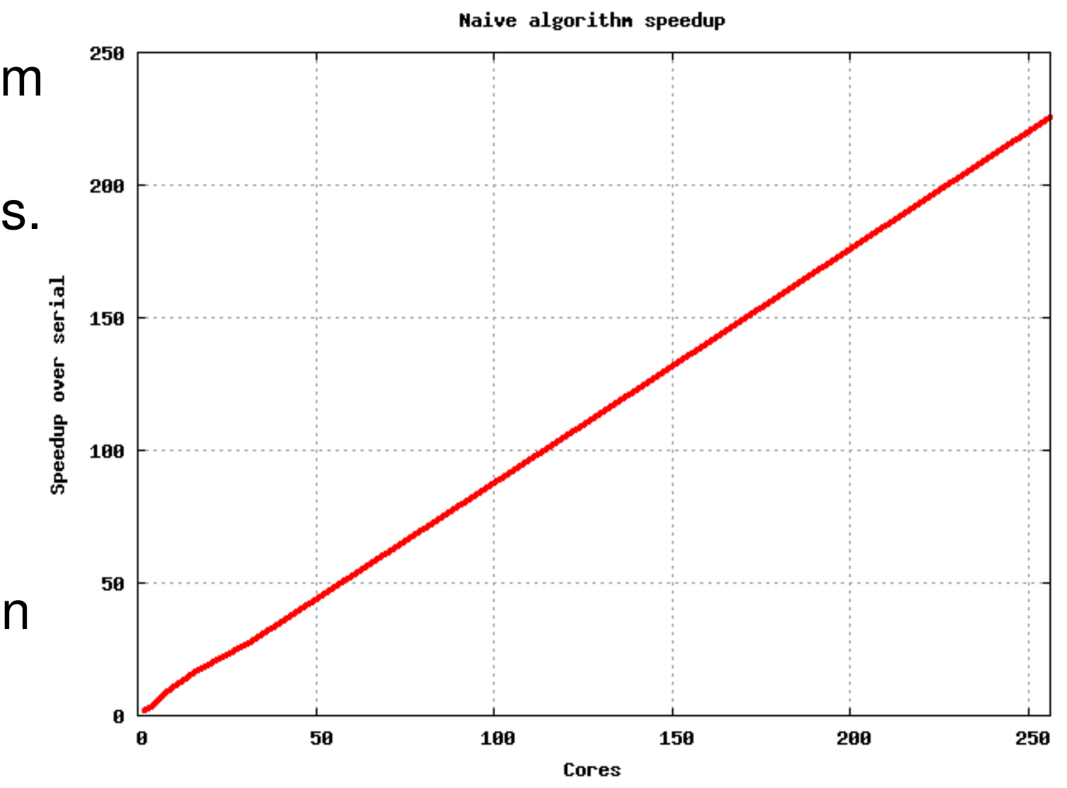
Algorithmic Issues

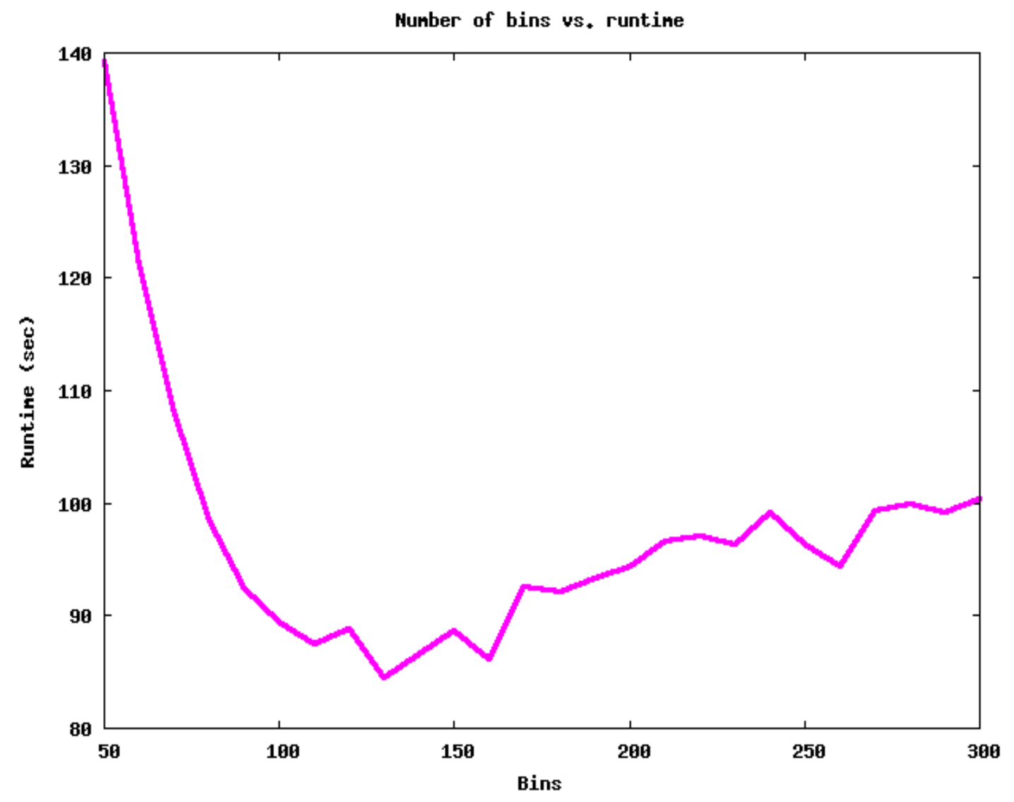
The original publication regarding ray-traced RBC scatter (Zohdi and Kuypers (2006)) utilized a serial algorithm based upon the classical approach presented by Spencer and Murty (1961).

As ray-tracing is an embarrassingly parallel task, the classical (naive) algorithm lends itself nicely to being parallelized by passing groups of rays to different cores. If cell data is regarded as constant, processors can proceed independently until completion. Thus, perfect speedup can be obtained (see Figure at lower right).

While the speedup plot for the naive algorithm is optimistic, the initial serial runtime for 1000 rays and 8000 cells is about 28 minutes. In many environments, it is not practical to keep "throwing cores" at the problem until runtimes are acceptable. Clearly, a more efficient algorithm is required.

By subdividing the domain into "bins", we can significantly reduce the number of cells that must be checked for collisions with rays.

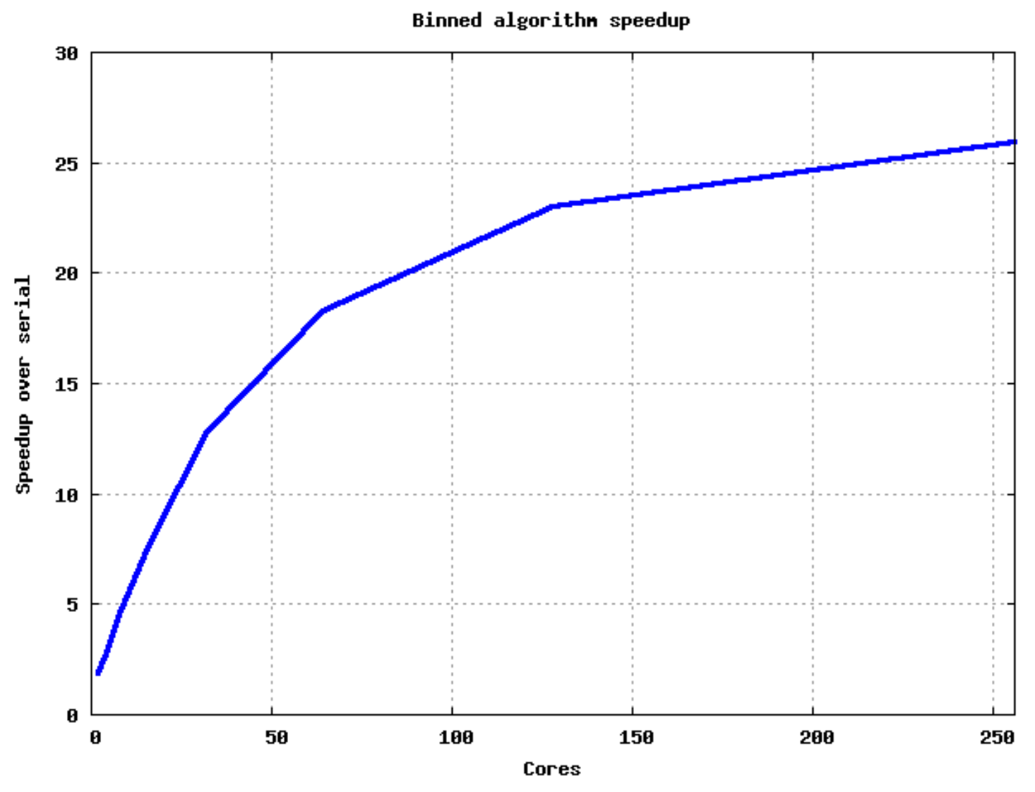




Once rays and cells have been binned, we can update the rays by iterating over bins. Thus, we check each ray against all cells within the current bin and all neighboring bins. In the worst situation, we must check 27 bins in 3D space. A simple geometric decomposition works well as the cells and rays are uniformly distributed.

The figure to the left explores a portion of the tuning space for bins. The plot shows various values for the number of bins in the X,Y direction (with only one bin in the Z direction, for simplicity).

Unfortunately, the current binning algorithm does not scale linearly like the naive approach. However, the serial runtime for 1000 rays and 8000 cells has been reduced to just under 12 seconds. This rapid runtime allows for much larger numbers of rays to be simulated in reasonable amounts of time, increasing the sensitivity of the model.



Future Work

- Currently, this simulation models healthy RBCs. These shapes are quite regular, and have been studied extensively. Unfortunately, no mathematical models currently exist for the many types of deformed shapes. These models need to be constructed, with the caveats of efficiency and accuracy.

- The ultimate goal of the simulation is to approach real time speeds on multicore chips for reasonable numbers of rays and scatterers. Currently, 10000 rays and 8000 cells requires a runtime of ~30 seconds with 4 cores. Optimally, this value should be an order of magnitude smaller.

- It is currently unclear as to the resolution (number of rays) required to accurately differentiate between various cell shapes. Once accurate malformed models have been constructed, this question needs to be explored for possible algorithmic implications.