e - publication, Wolfram Research, Wolfram Information Center,
http://library.wolfram.com/infocenter/MathSource/9281/ (2015)

Pi from agent border crossings by NetLogo package

Calculate Pi from the shapes of the forests by ABM (Agent Based Modeling)

Ervin Wirth

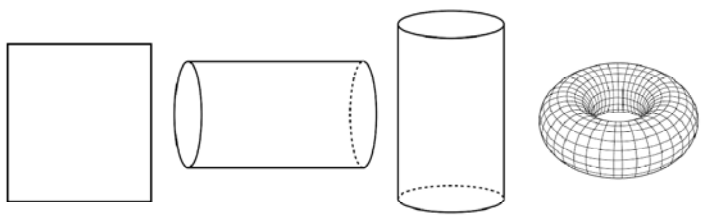
In this tutorial I will guide you through a novel way of calculating the perimeter (or even the surface) of formations with agent based modelling. The agents are wandering (as random walking) in the raster world, and they register the land cover changes under their feet into a common sum. The pi can be approximated with the simulation result of a round shaped world compared to a quadratic. In addition this document serves as a tutorial for NetLogo-Mathematica Link containing data (spatial) manipulation and visualisation.

Agent-based modeling

The term agent-based modeling (ABM) refers to the use of computational methods to investigate processes and problems viewed as dynamic systems of interacting agents. An example might be attempting to model crowd behavior in a football stadium using computational agents to represent individuals in the crowd. [1]
Agents are beings that can follow instructions. Each agent can carry out its own activity, all simultaneously.

About NetLogo

The NetLogo is a multi-agent programmable modeling environment. The NetLogo world is made up of agents. The worlds topology has four potential values: torus, box, vertical cylinder, or horizontal cylinder. The topology is controlled by enabling or disabling wrapping in the x or y directions. A torus wraps in both directions, meaning that the top and bottom edges of the world are connected and the left and right edges are connected. So if a turtle - the basic agent - moves beyond the right edge of the world it appears again on the left and the same for the top and bottom. [2]



▲ **Figure 1.** The world types of NetLogo: box, horizontal cylinder, vertical cylinder, torus

About *Mathematica* Link

The primary developer of the NetLogo-Mathematica link was Eytan Bakshy. This link provides modelers with an easy to use, real-time link between NetLogo and Mathematica. Together, these tools can provide users with a highly interactive, self-documenting work flow that neither can provide alone. With the NetLogo-Mathematica link, you can run all of these tools side-by-side with NetLogo. [3]

Here are a few examples of what you can do with the Mathematica-NetLogo link:

- Analyze your model in real-time with seamless two-way data conversion
- Develop high quality, custom visualizations of model data
- Collect detailed simulation data across large multi-dimensional parameter spaces
- Rapidly develop interactive interfaces for exploring model behavior
- Have direct access to patches and network data with built-in functions

Start the NetLogo package and model from *Mathematica* with the code cell:

```
Clear["Global`*"]
<<NetLogo`
NLStart["c:/Program Files (x86)/NetLogo 5.2.1/"];
NLLoadModel["d:/Works/MathLib/Introduction.nlogo"];
```

Generally the agent-based modellers' (AgentSheets, Breve, Mason, NetLogo, StarLogo, Swarm, Repast) visuality is far from the expected, thus the connection between the NetLogo and *Mathematica* and this document fill this gap.

Setup worlds

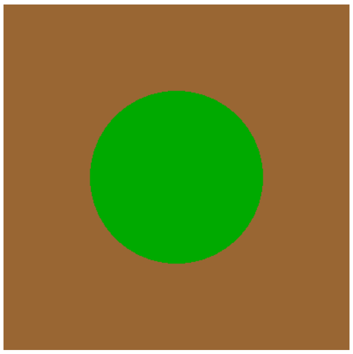
The 'setup' and the 'world' functions are defined in the model or script of NetLogo written in Logo, a simple artificial intelligence language. You can look them up in the attached .nlogo file (references).

```
NLCommand["clear-all", "setup", "world", 1]
```

Round Forest World surrounded by arable land

In the Round Forest (#1) raster model you can see two types of land cover: the forest; and the arable land around it. It was artificially created by the QGIS, which is a free and open source Geographic Information System (GIS). Inside the GIS software Python codes were written in order to reach the proper extension, resolution, and raster format as .asc (Esri ASCII Grid). In the NetLogo environment you have to use the 'gis' extension for import the raster.

```
ArrayPlot[NLGetPatches["covername"],
 ColorRules → {"arable_land" → Brown,
   "forests" → Darker[Green]}, Frame → False,
 ImageSize → 150]
```

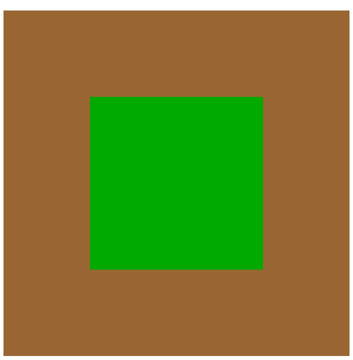


▲ **Figure 2.** I am using a 400 x 400 box world as base of the simulation. In case of using rougher resolution you cannot approximate well the value of Pi.

□ Quadratic Forest World

In the Quadratic Forest (#2) raster model you can see two types of land cover: the forest in square form; and the arable land around it.

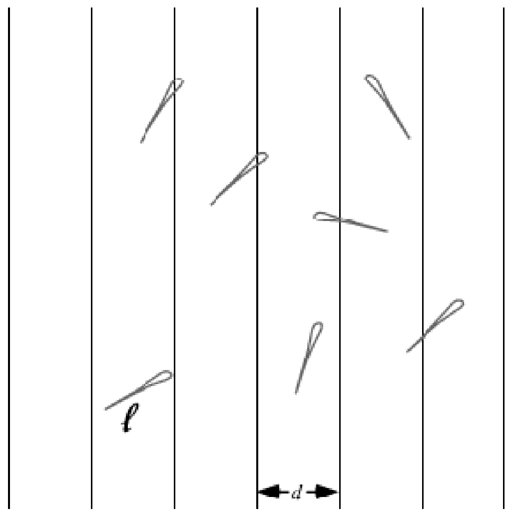
```
NLCommand["clear-all", "setup", "world", 2]
ArrayPlot[NLGetPatches["covername"],
 ColorRules → {"arable_land" → Brown,
   "forests" → Darker[Green]}, Frame → False,
 ImageSize → 150]
```



▲ **Figure 3.** A square shaped forest in the same extent and resolution.

■ Back in time

Buffon’s needle was the earliest problem in geometric probability to be solved; it can be solved using integral geometry. The solution, in the case where the needle length is not greater than the width of the strips, can be used to design a Monte Carlo method for approximating the number π, although that was not the original motivation for de Buffon’s question. [4]



▲ **Figure 4.** Illustration of the needle problem.

■ The Problem of the Random Walk

A random walk is a mathematical formalization of a path that consists of a succession of random steps. The term “random walk” was originally proposed by Karl Pearson in a letter to Nature (1905), which described a mosquito infenstation in a forest. At each time a single mosquito moves a fixed length at a randomly chosen angle. Peason wanted to know the distribution of the mosquitos after a long time. [5] Lord Rayleigh gave an answer to the problem with the corresponding probability density function.

Random walks are able to sample from a state space which is unknown. When this approach is used in computer science it is known as Markov Chain Monte Carlo or MCMC for short. Often, sampling from some complicated state space also allows one to get a probabilistic estimate of the space's size. The estimate of the permanent of a large matrix of zeros and ones was the first major problem tackled using this approach.

Setup agents

After the world setup - static part - the dynamic part follows, the agents. I distribute an amount of agents randomly on the world. They are elementary, only have two property, the ‘memory’ and the ‘path’. It first means that they store their actual landcover type - where they stand - into their ‘cache’. So every time there is just one string value in their brain (‘forest’ or ‘arable land’). The latter property is an increasing list, which consist the coordinates of their routes.

```
NLCommand["clear-all", "setup", "world", 1]
StartShot = ArrayPlot[NLGetPatches["covername"],
   ColorRules → {"arable_land" → Brown,
     "forests" → Darker[Green]}, Frame → False,
   DataRange → {{-400, 0}, {0, 400}}];
```

First I set up three agents randomly distributed in the round world. They act as scouts by tracking changes under their feet.

```
NLCommand["setup-agents", 3]
```

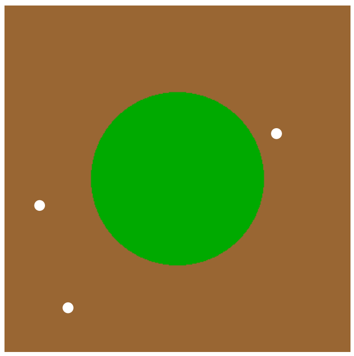
Get the origins of the agents into a list:

```
startingpoints = NLReport["map last [path] of persons"]

{{-85.013, 252.616}, {-359.853, 169.306}, {-326.84, 50.7662}}
```

Represent them by points:

```
agents = ListPlot[startingpoints,
   PlotStyle → Directive[PointSize[Medium], White],
   AspectRatio → 1, Axes → None, Frame → False,
   DataRange → {{-400, 0}, {0, 400}}];
Show[StartShot, agents, ImageSize → 150]
```



Agents' random walk

This chapter shows the movements of the agents. Their rotations follows a mentioned uniform distribution included in the NetLogo code (rt random 360). The stepsize and the simulation length are parameterized. In this part I simulate with a fixed stepsize, which is 20 pixel length.

```
SimLength = 5;
StepSize = 20;
NLCommand["go-agents", SimLength, StepSize]
```

First I read out the agents' movements from their 'path' property with a report.

```
paths = NLReport["[path] of persons"]

{{{-359.853, 169.306}, {-340.727, 163.458},
  {-357.307, 152.274}, {-338.286, 158.455},
  {-318.298, 157.757}, {-323.137, 177.162}},
 {{-85.013, 252.616}, {-104.903, 254.707},
  {-124.709, 257.49}, {-119.532, 238.172},
  {-101.406, 246.624}, {-114.789, 231.761}},
 {{-326.84, 50.7662}, {-329.623, 30.9608},
  {-309.772, 28.5234}, {-328.043, 20.3887},
  {-338.043, 3.06821}, {-339.118, 0.267466}}}
```

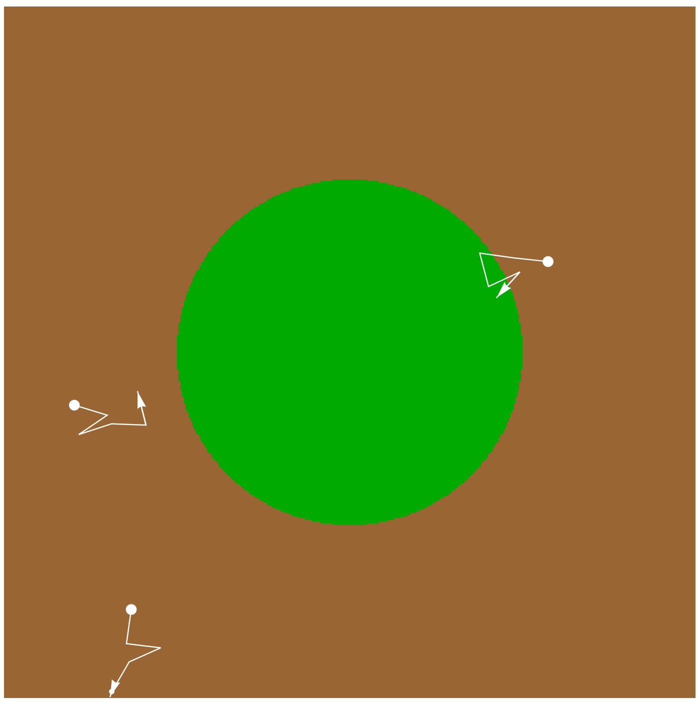
Then I want to visualise their last step with an arrow, therefore I put the components of array vectors after each other by some mapping.

```
arrows =
 Transpose[{#[[SimLength]] & /@ paths,
   #[[SimLength + 1]] & /@ paths}]

{{{-318.298, 157.757}, {-323.137, 177.162}},
 {{-101.406, 246.624}, {-114.789, 231.761}},
 {{-338.043, 3.06821}, {-339.118, 0.267466}}}
```

At last the startplot and the agent vectors are shown together:

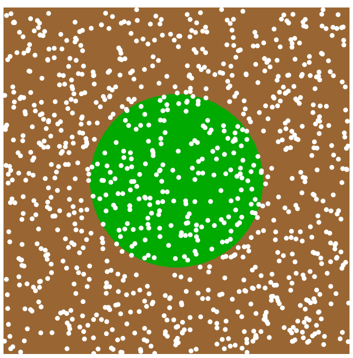
```
lineplot = ListLinePlot[paths, AspectRatio → 1,
   Axes → None, Frame → False,
   DataRange → {{-400, 0}, {0, 400}}, PlotStyle → White];
Show[StartShot, lineplot, agents,
 Epilog → {White, Arrowheads[Small], Arrow /@ arrows},
 ImageSize → 300]
```



■ Simulation on the worlds

I simulate the agents' wandering through a time interval and count the land cover changes under their feet. If we use higher agent number our Pi approximation in the end will be better. So at this part I put down 1000 agents and visualise their movements:

```
NLCommand["clear-all", "setup", "world", 1]
NLCommand["setup-agents", 1000]
startingpoints = NLReport["map last [path] of persons"];
agents = ListPlot[startingpoints,
   PlotStyle → Directive[PointSize[Small], White],
   AspectRatio → 1, Axes → None, Frame → False,
   DataRange → {{-400, 0}, {0, 400}}];
SimStart = Show[StartShot, agents, ImageSize → 150]
```



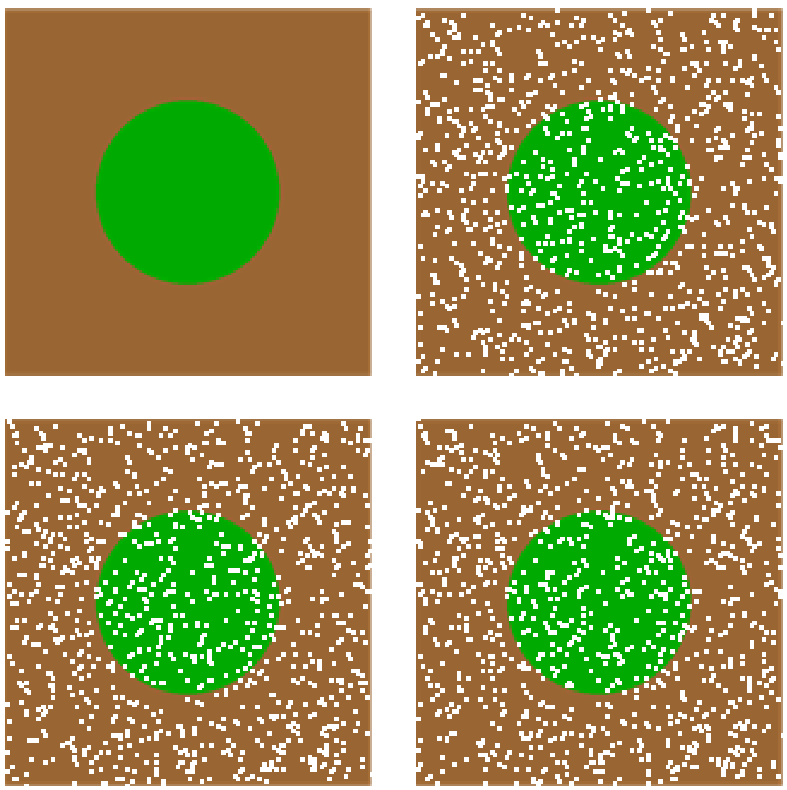
▲ **Figure 5.** The agents appear on the world as white points. All the agents have a unique identifier (id).

The modules are the most obvious way to handle NetLogo (NL) and Mathematica together. Here I write a module (CaptureWalking) to simulating the agents' wandering. The module simulates 10 years of wandering, but I call it two times to create a simulation series. By this the simulation will be 20 years long. During the procedure each agent moves one step (with constant 1 pixel stepsize length in radius) in a random direction per years.

```
CaptureWalking[] := Module[{},
   NLCommand["go-agents", 10, 1];
   positions = NLReport["map last [path] of persons"]
  ];
```

Then the simulation on a graphicsgrid:

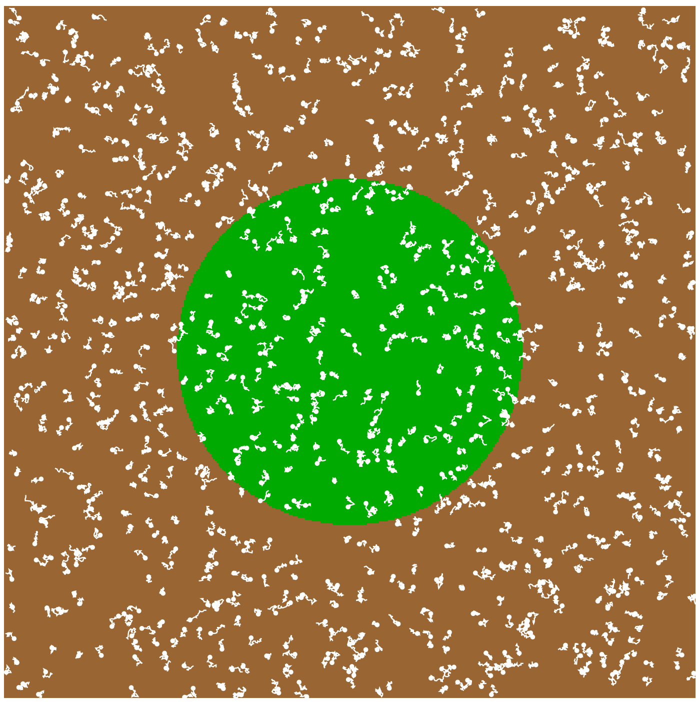
```
patchShots = Table[CaptureWalking[], {2}];
renderedShots =
  Map[
   Show[StartShot,
     ListPlot[#,
      PlotStyle → Directive[PointSize[Small], White],
      AspectRatio → 1, Axes → None, Frame → False,
      DataRange → {{-400, 0}, {0, 400}}]] &, patchShots];
Rasterize[
 GraphicsGrid[
  Partition[Join[{StartShot, SimStart}, renderedShots],
   {2}, {2}]]]
```



▲ **Figure 6.** Simulation of 20 years in graphicsgrid: 0 year without and with agents; after 10 years of wandering; after 20 years

The simulation can be shown in one image as well. In that the border crossing can be seen better.

```
paths = NLReport["[path] of persons"];
lineplot = ListLinePlot[paths, AspectRatio → 1,
   Axes → None, Frame → False,
   DataRange → {{-400, 0}, {0, 400}}, PlotStyle → White];
Show[StartShot, lineplot, agents, ImageSize → 300]
```



▲ **Figure 7.** Simulation result in one picture by paths.

During the wandering I registrate the landcover border crossings of each agent.
Each agent (marked as ‘i’) has its position by coordinate pairs at each epoch (t):

$$p(i, t) = \{x, y\} \quad p : \mathbb{Z}^2 \to \mathbb{R}^2 \tag{1}$$

There can be a finite land cover type in the investigated worlds:

$$L = \{\text{arable land}, \ \text{forest}\} \tag{2}$$

The cluster function refers to a particular world (theta). Thereby all the coordinates of the torus world has an attribute value - a cluster - what can be ‘arable land’ or ‘forest’.

$$c(\theta, \{x, y\}) = l \quad c : \mathbb{R}^3 \to L \tag{3}$$

The following ‘s’ function words the changes:

$$s(\theta,\ i,\ t) = \begin{cases} 1, & \text{if } c(\theta,\ p(i,\ t+1)) \neq c(\theta,\ p(i,\ t)) \\ 0, & \text{if } c\,(\theta,\ p\,(i,\ t+1)) = c(\theta,\ p\,(i,\ t)) \end{cases} \qquad (4)$$

$$s : \mathbb{Z}^3 \to \{0,\ 1\}$$

Then I cumulate the Monte Carlo integral with the ‘s’ function, which collects the total changes as potential:

$$P(\theta,\ a,\ d) = \sum_{i=1}^{a} \sum_{t=0}^{d} s(\theta,\ i,\ t) \qquad (5)$$

$$P:\ \mathbb{Z}^3 \to \mathbb{Z}$$

The RandomWalk module is responsible for the simulation, it collects the potentials. The parameters of the model are the world identifier, agent count, simulation period in years, stepsize, simulation repetition (sample length).

```
RandomWalk[w_, a_, d_, l_, z_] :=
 Module[{worldid = w, agentnum = a, simlength = d,
   stepsize = l, samplen = z},
  NLCommand["clear-all"];
  Clear[L];
  L = {};
For[i = 0, i ≤ (samplen - 1), i++,
    NLCommand["setup", "world", worldid]
    NLCommand["setup-agents", agentnum, "go-agents",
     simlength, stepsize]
    AppendTo[L, NLReport["crossi"]]
  ]
 ]
```

□ Result on Round Forest World

I calculate the potential of equation (5) by running the simulation of wandering - registering with 1000 agents for 100 years on Round Forest World:

$$P(1,\ 1000,\ 100) = \sum_{i=1}^{1000} \sum_{t=0}^{100} s(1,\ i,\ t) \qquad (6)$$

By *Mathematica* code:

```
RandomWalk[1, 1000, 100, 10, 50]
```

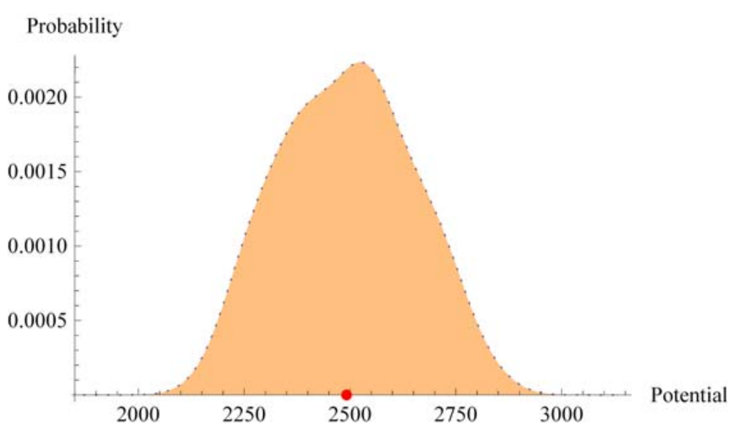
Look at the result list:

```
L
```

```
{2368., 2572., 2498., 2401., 2404., 2741., 2364.,
 2188., 2682., 2525., 2518., 2737., 2398., 2594.,
 2606., 2472., 2820., 2428., 2639., 2635., 2281.,
 2699., 2340., 2692., 2321., 2635., 2527., 2260.,
 2380., 2247., 2463., 2527., 2492., 2574., 2511.,
 2565., 2407., 2281., 2534., 2370., 2444., 2735.,
 2268., 2691., 2594., 2353., 2270., 2547., 2543., 2446.}
```

▲ **Figure 8.** The list of landcover border crosses (MC integral).

```
hist = SmoothHistogram[L, PlotStyle → Dotted,
  Filling → Bottom,
  FillingStyle → Directive[Opacity[0.5], Orange],
  AxesLabel → {Potential, Probability},
  Epilog → {Red, PointSize[Medium], Point[{Mean[L], 0}]},
  ImageSize → 300]
```



▲ **Figure 9.** The histogram of results, which looks alike normal distribution.

Finally I determine the statistical expectation as mean to characterize the Round Forest World

```
NRound = Mean[L]
```

```
2491.74
```

Result on Quadratic Forest World

Run the simulation on Quadratic Forest World.

```
RandomWalk[2, 1000, 100, 10, 50]
```

Calculate the statistical expectation as mean to characterize the Quadratic Forest World

```
NQuadratic = Mean[L]
```

```
3166.18
```

Summary

If the side of the square is two radius long, and the radio of the circle is one radius long. Then the perimeter of the square must be eight radius, and the perimeter of circle shall be two radius multiplied by Pi .
I presume the integral counts of landcover border crosses in the Quadratic and Round Forest Worlds analogical with perimeters, therefore:

$$\frac{\text{NRound}}{\text{NSquare}} = \frac{2\,r\,\pi}{8\,r} \tag{7}$$

$$\pi \approx \frac{\text{NRound}}{\text{NSquare}} \cdot 4 \tag{8}$$

```
NumPi = NRound / NQuadratic * 4
```

```
3.14794
```

Conclusion

I reached a rough approximation of Pi, which depends on raster resolution and parameter setups. Higher parameter setups or finer resolution will result in better approximations. This model proof that’s logical agents are able to calculate perimeters, border sums or surface amounts.

Outlook

In the article the rotations followed a uniform distribution which can be changed to any kind of distribution e. g. normal distribution, also the step-size was fixed which can also follow a distribution. So the system can be decomposed to many elements (rotation, step-size, agent headcount, world resolution, simulation length) which the results or the approximations are based. The modifications of these elements could be also interesting, for example, which setup gives the better approximation with minimum simulation length and agent count.

References

[1] Michael J de Smith, Michael F Goodchild, Paul A Longley, Geospatial Analysis, Fifth Edition, The Winchelsea Press, Winchelsea, UK, 2015

[2] Wilensky, U. (1999). NetLogo. http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.

[3] Bakshy, E., Wilensky, U. (2007). NetLogo-Mathematica Link. http://ccl.northwestern.edu/netlogo/mathematica.html. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.

[4] Buffon's Needle Problem -- from Wolfram *MathWorld* http://mathworld.wolfram.com/BuffonsNeedleProblem.html.

[5] Martin Z. Bazant. "Lecture 1: Introduction to Random Walks and Diffusion." Department of Mathematics, MIT (2005): n. pag. Print.

About the Authors

Ervin Wirth is a PhD student at the university mentioned below and his research topics is artificial intelligence in geographical information systems.

Ervin Wirth
Department of Photogrammetry and Geoinformatics
Budapest University of Technology and Economy, H-1111, Hungary
wirth.ervin@epito.bme.hu