Proceedings of The National Conference
On Undergraduate Research (NCUR) 2017
University of Memphis, TN
Memphis, Tennessee
April 6-8, 2017

Optimal Path for Finding Randomly Distributed Targets

Samuel Norman
Computer Engineering
San Jacinto College
Pasadena, TX 77504 USA

Faculty Advisor: Prof. Nathanial Wiggins

Abstract

The NASA Swarmathon is a competition focused the development and optimization of search algorithms. The objective of this competition is to design an algorithm for rover-robots in the effort of contributing to research toward planetary exploration; these rovers must be able to find randomly distributed targets (or resources) and bring them to a predesignated location. Further, the project uses NASA's Swarmathon Robotic Operating System's (ROS) simulation environment for testing a variety of students designed and modified search patterns. The more basic/outdated Gaussian Distribution algorithm, used in the competition, employs a random number generator to determine the direction the rover travels to in its search. A drastic improvement to the Gaussian Distribution algorithm uses a more chaotic approach - created by Edward Lorenz – the Lorenz attractor system models' hydrodynamic fluid flow which has a chaotic structure. The specific nature of the Lorenz attractor does not repeat values and maintains a tight format, unlike the Gaussian Distribution method. This results in the added ability to cover an area more thoroughly and with higher efficiency. This means that, if the Lorenz method for target acquisition is used, the rovers will be more effective and efficient at collection than if the Gaussian distribution method is used. During testing, eighteen successful simulations were executed in total, each lasting thirty recorded minutes. These tests used three distinct target distribution methods, which showed that the average number of targets collected was greater using the Lorenz algorithm. These results prove the conclusion that collecting randomly distributed targets is more efficient and consistent with the Lorenz attractor path.

Keywords: Robotic Operating System, Lorenz Attractor, Chaos, Path Optimization

1. Introduction

Swarming technology is a field of robotics that is primarily inspired by nature. In fact, applications include package delivery[1], farming[2], cleaning[3], oceanic exploration[4], exploration and chemical analysis on other planets[5], and tritium mining on the moon[6]. NASA plans to use swarming technology by implementing it into rovers to send to other planets. Moreover, instead of sending a single multi-million-dollar robot and counting on a successful mission, NASA intends to send a swarm of several thousand-dollar robots that communicate and collaborate to achieve tasks. In other words, a swarm of robots that run autonomously to complete tasks are more productive while also lowering monetary risks.

A related study of swarm robotics is the NASA Swarmathon: a robotics competition based on the development of swarming technology. This competition focuses on the rovers' collection and return of randomly distributed targets. University of New Mexico participates as the admin team, which oversees the organization and writes the base code that is implemented into the robots. Moreover, all the rover code is running in the Robotic Operating System (ROS) distribution Indigo and is written in C++.

The original search algorithm for the robot is called "random walk," and it is based on an ant's path for finding resources[7]. In this search algorithm, the rover picks a random heading and drives in that direction for 50 centimeters and then picks a new random heading. The heading is generated in radians from a combination of a random number generator and a Gaussian distribution.

$$\boldsymbol{f(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}} \tag{1}$$

The Gaussian distribution is modeled in equation (1), where $\mu = Current\ Heading$, $\sigma^2 = 0.25$, $x = Random\ Number$, and $f(x|\mu, \sigma^2) = New\ Heading$.

The new proposed search algorithm is a projection of the Lorenz attractor, equation (2) and Figure 1, onto a 2D plane.

$$\boldsymbol{\begin{cases} \dot{x} = \sigma(y - x) \\ \dot{y} = x(\rho - z) - y \\ \dot{z} = xy - \beta z \end{cases} \qquad \sigma = 10, \ \beta = \frac{8}{3}, \quad \rho = 28.} \tag{2}$$

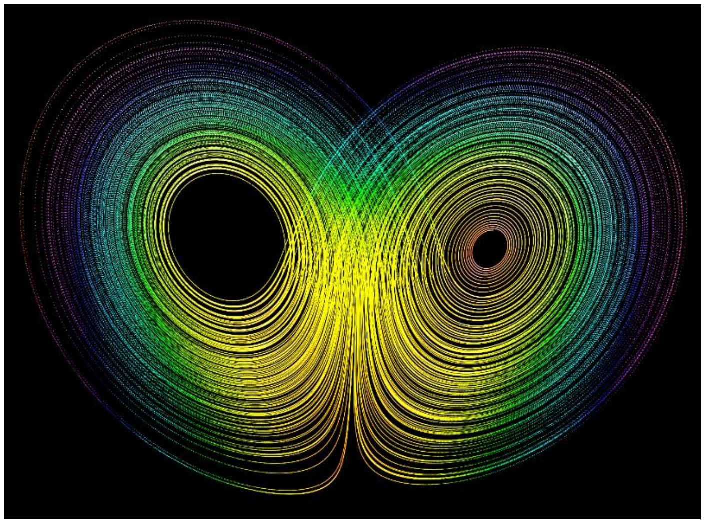

Figure 1: Lorenz attractor generated in Chaoscope

The Lorenz attractor is a non-linear system of differential equations and was introduced by Edward Lorenz in 1963. This system models hydrodynamic fluid flow without using a forcing method that would produce both periodic, and non-periodic solutions in experiments[8]. However, independent solutions cannot be derived by this system from its output[8]. The only way to be certain of the next number generated by the system is to know the system's initial conditions and the current iteration number[8]. In fact, the system of equations never repeats or intersects with itself. This is because the original system is three-dimensional: the two-dimensional projection will have intersection points but will not follow the same path. Essentially, this means the rover following the Lorenz path will not search the same area twice.

In the past, chaotic search patterns have been studied in the form of direct path optimization, yet the most successful have been those of chaotic bee path optimization[9], or chaotic paths to avoid threats[10]. These path optimizations are based on running multiple iterations of chaotic decision-making to find a path toward a defined location. This differs from finding the optimal path for discovery of randomly distributed targets. Current studies seem to show that, for the detection of random targets, ant foraging patterns are the most efficient known model[7].

2. Methodology and Results

The two operating systems that were used were Windows 10 and Linux Ubuntu 14.04. In Windows 10, Scilab was used to generate the original function by graphing the numerical approximation of the Lorenz Attractor. This process was completed with an ODE solver function which is internal to the program; the next step was to apply a two-dimensional projection of the graph to generate a path for a rover to follow (on a two-dimensional plane). Specifically, the Lorenz attractor time started at 10 time units, has a step size of 0.5, and ran until 20 time units.
After the projection, the points were centered at zero and two scale variables were created for the 'x' and 'y' direction, respectively. This is because the simulation environment for the NASA Swarmathon is an 8m by 8m square; the scale variables in Scilab were both set to 6m to allow for rover drift.

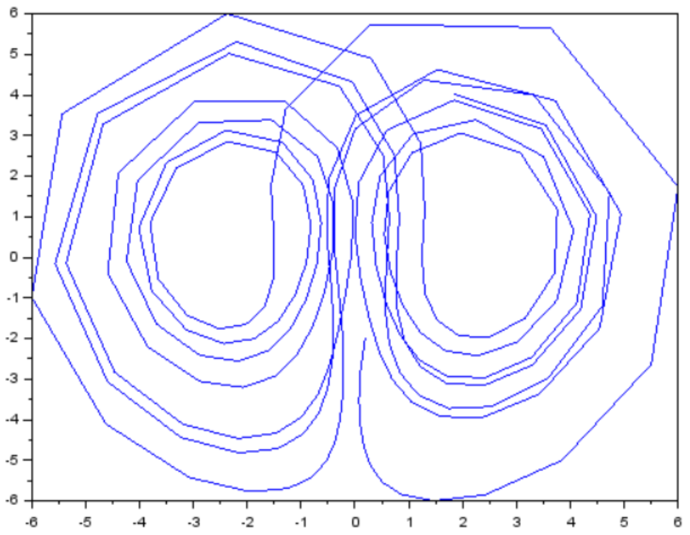

Figure 2: Robotic path generated in Scilab

The Scilab code's output is in the form of a .csv: a Java program inside of Windows 10 that was then written in NetBeans to translate the .csv into something that ROS could use. The Java program then read the .csv file, and created two vectors internally from the points listed in the file. The same code counted the size of the vectors then generated a file in C++ with the x and y coordinate arrays and an integer array (to function as multiple Booleans). This made it possible to keep track of which points had already been reached by the rover and an integer representative of the size of the arrays. Once the points were imported into ROS, the search controller code was modified to use the points instead of random headings. When both C++ files were in the correct folder, the code was then complied. In ROS, using catkin tools, the code then recompiled – but only if the "date modified" was more recent than the previously compiled code. This compiling method is used to optimize compilation speed.

Sometimes trying different algorithms meant switching back and forth between files frequently. To combat this issue, each file had a new save version created so that the ROS would recognize the file that needed to be compiled. This was done before compiling the code. After this, the code was compiled two times: to ensure that catkin implemented the changed code into ROS. In the following step, ROS wan run inside Ubuntu 14.04 on a Dell Inspiron 13 with an Intel core i7-7700u: running Intel integrated graphics with 8GB of DDR4 memory. The complete Lorenz path driven by the rover can be seen in Figure 3.

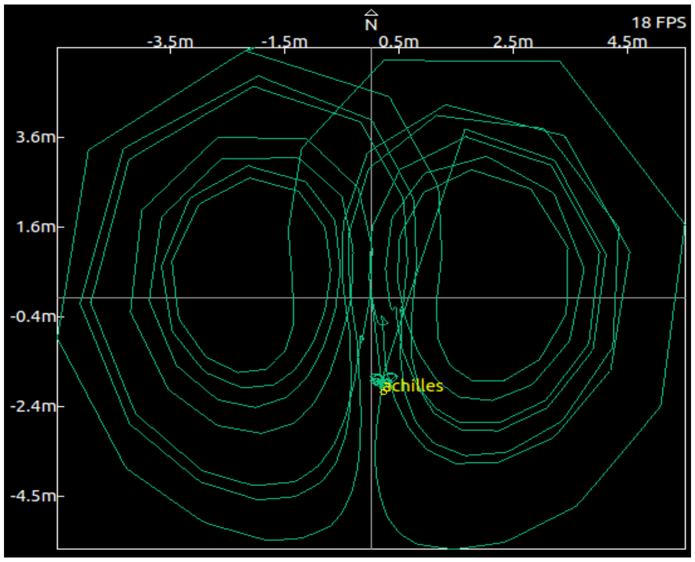

Figure 3: Lorenz attractor driven by rover in the ROS simulation environment

Three different target distribution methods with 256 targets, each, were used to test the applicability of the search methods in various situations. The Clustered distribution used 64 targets per cluster - 4 clusters were randomly placed around the simulation environment. The Power Law distribution placed clusters randomly around the simulation environment where the number of targets per cluster is always in perfect squares. The Uniform distribution placed individual targets randomly in the simulation environment. Additionally, for each target distribution method, both the Lorenz path and Random Walk search methods were run a total of three times, which resulted in eighteen complete simulations being run.

Each simulation was set to last an entirety of thirty minutes. In fact, there were cases where errors in the simulation environment or errors in the rover code would cause them to malfunction. For example, if the rover became trapped, a target got wedged in the rover's gripper, or the rover got stuck inside the home target area and pushed all the targets out. In these cases of rover error, the simulations were shut down and restarted. This is because the data to be tested was in relation to the search method, not the functionality of the gripper or other similar systems. The generated world was consistent, so the targets were placed unpredictably around the simulation; this provided a better sample of data to pull from.

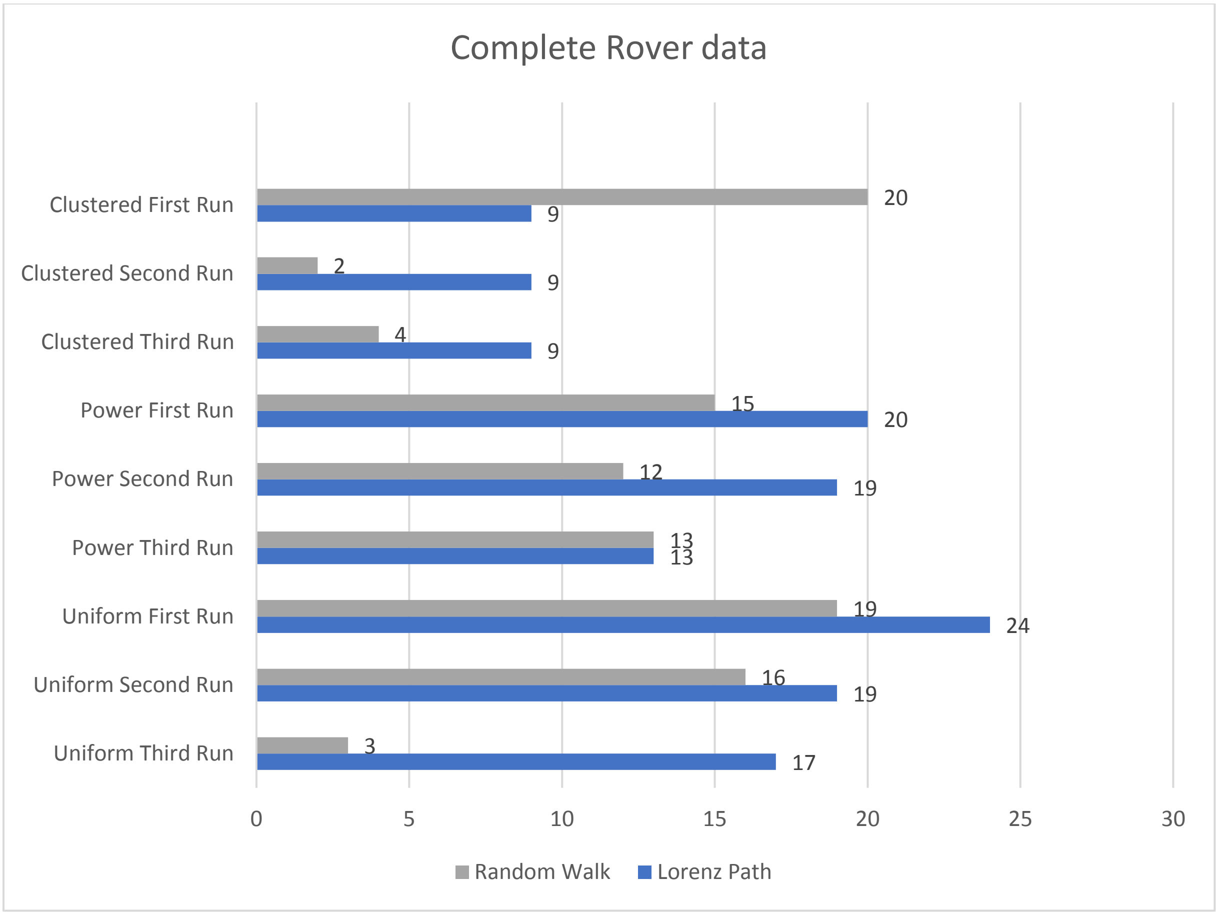

Figure 4: This graph illustrates the number of targets collected on each run of every target distribution and search method

3. Conclusion

While finding random targets, the Lorenz attractor path is more consistent than the Random Walk path. This is supported by the data because the Random Walk path is a combination of a random number generator and the Gaussian distribution method. The Lorenz attractor path followed a specific path that had very little space between the searched areas, which created a very thorough search algorithm. While the Random Walk had no defined shape, which caused it to sometimes search the same area again. Due to the described structure of the Lorenz attractor, it outperformed the Random Walk path in all target distribution arrangements, as seen in Figure 5.

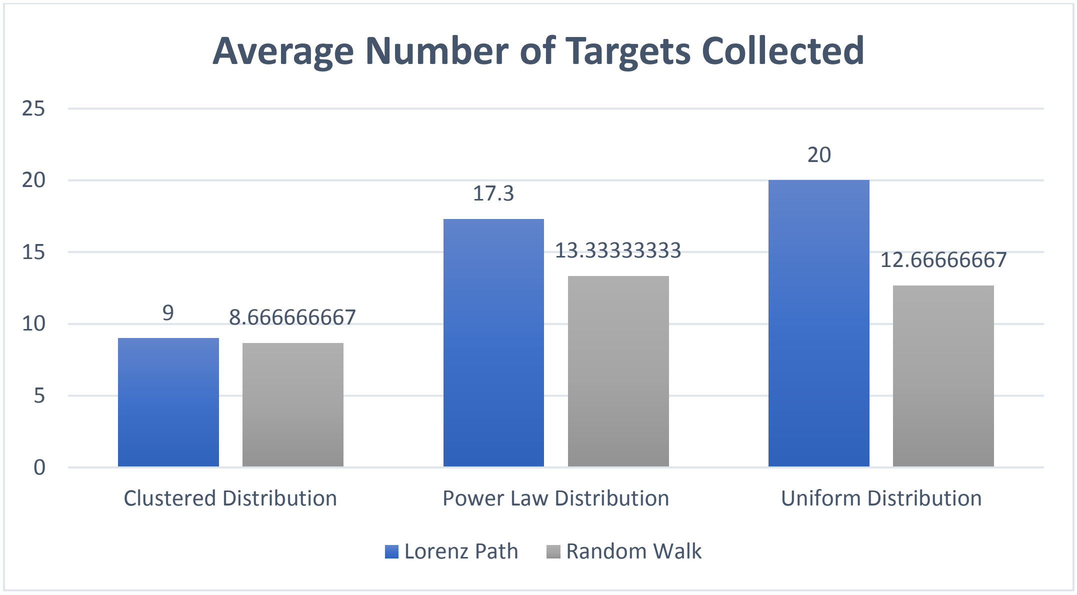

Figure 5: Graph of average number of targets collected in each target distribution by each search algorithm

Some possibilities to expand upon this study would be to select and change a few of the environment's elements. The search area in this project is a pre-defined area and the Lorenz Path was scaled to fit this specification. Further, it is possible that in an undefined or irregularly shaped search area, the Lorenz path would not perform as well because of the areas that the rover would not encounter (due to the shape of the attractor). Moreover, the localization algorithm used is based on distance traveled on a two-dimensional plane, therefore, changing the terrain of the search area would affect the structure of the attractor and the path driven by the rover.

4. References

1. Claes, Rutger, Tom Holvoet, and Jelle Van Gompel. 2010. "Coordination in hierarchical pickup and delivery problems using delegate multi-agent systems." *Proceedings of the 4th Workshop on Artificial Transportation Systems and Simulation* 1-7.
2. Anil, H., K. S. Nikhil, V. Chaitra, and BS Guru Sharan. 2015. "Revolutionizing farming using swarm robotics." *Intelligent Systems, Modelling and Simulation (ISMS), 2015 6th International Conference* 141-147.
3. Altshuler, Yaniv, Alfred M. Bruckstein, and Israel A. Wagner. 2005. "Swarm robotics for a dynamic cleaning problem." *Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE* 209-216.
4. Schmickl, Thomas, Ronald Thenius, Christoph Moslinger, Jon Timmis, Andy Tyrrell, Mark Read, James Hilder et al. 2011. "CoCoRo--The Self-Aware Underwater Swarm." *Self-Adaptive and Self-Organizing Systems Workshops (SASOW), 2011 Fifth IEEE Conference* 120-126.
5. Sand, Stephan, Siwei Zhang, Maximilian Mühlegg, Guillermo Falconi, Chen Zhu, Thomas Krüger, and Stefan Nowak. 2013. "Swarm exploration and navigation on mars." *Localization and GNSS (ICL-GNSS), 2013 International Conference* 1-6.
6. Schmitt, Harrison H. 2004. "Mining the Moon." *Popular Mechanics 12.*
7. Forrest, Stephanie, Melanie E. Hecker, and Drew Levin. 2015. "Volatility and spatial distribution of resources determine ant foraging strategies." *European Conference on Artificial Life (ECAL).*
8. Lorenz, Edward N. 1963. "Deterministic nonperiodic flow." *Journal of the atmospheric sciences, 20(2)* 130-141.
9. Huang, Li-Ren, and Jiann-Horng Lin. 2009. "Chaotic Bee Swarn Optimization Algorithm for Path Planning of Mobile Robots." *WSEAS Internation Conference on EVOLUTIONARY COMPUTING.*
10. Volos, Ch. K., I. M. Kyprianidis, and I. N. Stoudoulos. 2012. "A chaotic path planning generator for autonomous mobile robots." *Robotics and Autonomous Systems 60, no. 4* 651-656.