A Single-Agent Model of Axon Guidance

Callon, Kate[a], Cussen, Hannah[b]
[a]Computer Science, Stanford University, kcallon@cs.stanford.edu
[b]Computer Science, Stanford University, hcussen@cs.stanford.edu

December 9, 2022

1 Introduction

Repository Location https://github.com/kcallon/Axon-Guidance-Simulation

The ability of our brain to receive, process, and respond to information is largely determined by a complex network of neurons. As such, the proper wiring of this network is essential during embryogenesis. One process involved in this development is axon guidance, where neuronal connections are formed when a differentiating neuron sends out a growing axon into the embryonic environment. This axon migrates to a synaptic target, laying down its final shaft positioning that will remain at the end of development as it goes. The process of axon guidance is mediated by several signaling molecules and receptors. In our project, we are modeling axon guidance in silico as a search problem where the agent (representing the axon) seeks to lay a path that maximizes reward (provided by signaling molecules) in the environment. Biological modeling in general is important as it can be used as a metric to determine whether or not we have a complete understanding of how a biological process works. Furthermore, it can be used to derive hypotheses on how an unknown process may work. While axon guidance has been thoroughly researched, some exact mechanisms remain unclear or unknown, and a simulation such as this project could be beneficial in deriving and testing hypotheses for these mechanisms which could later be confirmed by biological results.

2 Literature Review

Since our simulation models a biological phenomenon, it is important to note that the basis of our design is from the results of biological research literature. In particular, "The Molecular Biology of Axon Guidance" [2] and the "Long-Range Guidance of Spinal Commissural Axons by Netrin1 and Sonic Hedgehog from Midline Floor Plate Cells" [4] provide statistically significant evidence that the molecules and receptors we have incorporated in our design behave the way they do in an actual dorsal spinal cord environment. Furthermore, the paper "Mutations Affecting Growth Cone Guidance in Drosophila" provided the biological evidence for the slit, roundabout and commissureless axon mutation phenotypes which we will also simulate with our model [3]. One key difference between these research works and our model is that while they use imaging to represent the biological phenotype, we are using a matrix to represent the axon growth pattern. Finally, in the research dissertation "Computational Model of Axon Guidance", Rui Costa similarly simulates axon guidance in silico [1]. Like our model, he bases axon movement off of attraction and repulsion to netrin, sonic hedgehog, and a target molecule, and incorporates the DCC, Comm, and Robo receptors in order to guide axon movement. Another similarity was the invention of a target ligand which diffuses from the synaptic target in the spinal cord environment. This target ligand aided the axon in identifying the location of the synaptic target and incentivizing it to move towards this target in both of our models. However, in terms of differences of our models, Costa uses more of a systems biology approach to calculate the concentrations of the molecules in the dorsal spinal cord environment, and also models his axon as 3-D unlike our 2-D approach.

3 Dataset

Mutant Name	Netrin	SHH	Slit	Robo	DCC	Comm	Phenotype	Matrix Output
Wildtype	1	1	1	1	1	1		*wildtype.txt*
Slit	1	1	0	1	1	1		*slit.txt*
Roundabout	1	1	1	0	1	1		*roundabout.txt*
Commissureless	1	1	1	1	1	0		*comm.txt*

Figure 1: Data Input with Images.

For the dataset, the input is a set of gene configurations. The value 0 represents the gene is a mutant, and the value 1 represents the gene is functioning normally. The output given a set of gene configurations is the figure found in the phenotype column. We also have a representation of this phenotype as a matrix output in .txt files within our repository.

4 Baseline

The baseline we have constructed is a set of hand-crafted rules which dictates the axon behavior given a set of conditions. They are also follows:

1. If the Comm gene is on, then the Robo gene is off.
2. If the DCC gene is on, then the axon will be attracted to Netrin and SHH.
3. If the Robo gene is on and Slit is present, then the DCC gene is off and the axon will no longer be attracted to Netrin and SHH.
4. If the Robo gene is on, then the axon will be repelled by Slit.
5. If the axon crosses into the floor plate, then the Comm gene will be turned off.

5 Main Approach

We modeled this as a search problem where the axon is the agent, choosing how to grow in the environment to maximize the reward.

Input The input is two gene configurations. The first controls the functionality of the COMM, DCC, and ROBO genes in the axon. The second controls the functionality of the genes that produce the netrin, slit, and SHH proteins in the environment. All genes are functional by default, and can be made nonfunctional (representing the "knockout" mutation).

Output The output is an 11x10 matrix showing where the axon has grown. Figure 2 shows an example output of the matrix with no axon growth (the edges of the midline are delineated by the — character), and an example matrix showing the predicted growth path for a wildtype simulation. An `x` represents the axon shaft, and the `T` represents the synaptic target.

Simulation Environment The simulation environment is a 2d matrix where each element in the matrix has a concentration of the ligands. The concentration is calculated following exponential decay, $C_x = C_0 * (decay\_rate)^{x-x_0}$. (x, C_x) are the location and concentration at a given grid cell, and (x_0, C_0) are the location and initial concentration of the location from which the ligand diffuses. For Netrin, SHH, and Slit, these ligands diffuse from the middle column. The target ligand diffuses radially from the synaptic target, which is on the right side of the midline. See Figure 3 for a visualization of the placement and diffusion of these ligands.

Figure 2: Environment setup, predicted wildtype growth

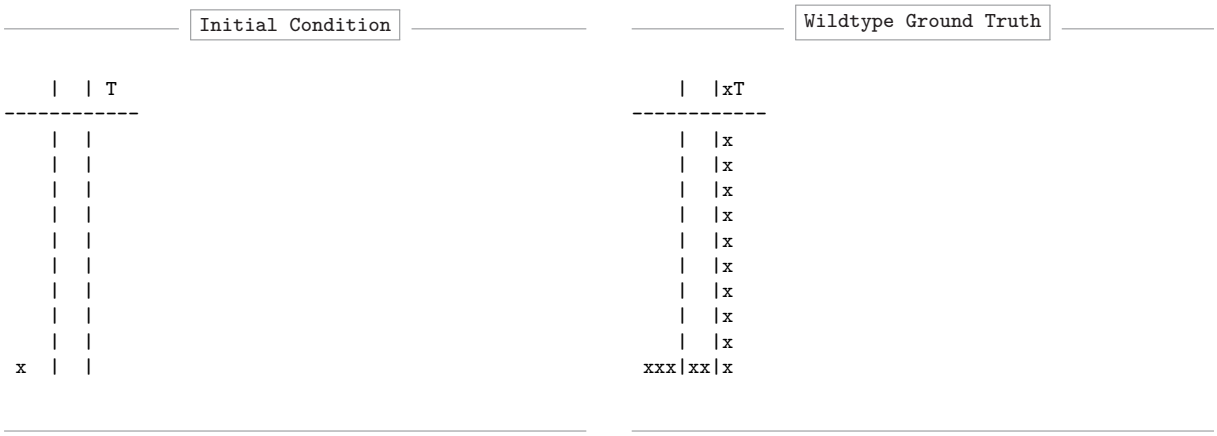
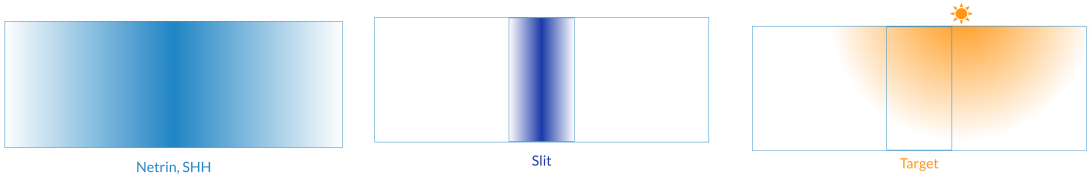


Figure 3: Visualization of diffusion of ligands



States, Actions, Rewards The state will consist of the axon length, the activated genes in the axon, the current location of the axon's head, and the cells that the axon's head has occupied previously. We are not counting the ligand concentrations as part of the state, since ligand concentrations are calculated once at the beginning of the simulation and then remain constant.

Start and end states The start state will be the axon head in the lower left-hand corner of the matrix. There are two end states: either the axon crosses into the 0-th row of the matrix, or it grows to a maximum length. Crossing into the 0th row of the matrix represents movement outside of the simulated area, presumably toward the synaptic target.

Actions The actions that the agent can take are moving up, down, left, or right once per simulation step. The action must be to move into a square that is within the simulation environment and is not already occupied by previous axon growth.

Rewards The rewards are a linear combination of the ligand concentrations in the square,

$$\sum_i w_i l_i * [f(genes)]$$

$f : genes \to \{0, 1\}$ is a function of the genes that are active at that time. It returns a 1 or a 0, signifying if the axon is sensitive or insensitive to the ligand at that time.

Algorithm The axon agent uses greedy search. It senses the directly adjacent squares, then selects the action that maximizes its immediate reward. The greedy search algorithm, while in general is not an effective algorithm, mimics the biological reality that a cell can only sense its local environment. An algorithm like A* would provide a faster, guaranteed correct path, but it's not accurate to the biological reality we are trying to simulate.

6 Evaluation Metric

Since our simulation is outputting a matrix showing the axon's path, we created ground truth text files that describe the paths shown in the dataset. These ground truth files are located in the `examples` directory of the repository.

Qualitatively, we can visually compare our simulation output to the text file encoding. Quantitatively, we wrote a utility to find the Levenshtein distance between the ground truth file and the simulation output.

The Levenshtein distance, also known as the edit distance, is how many edit operations must be made to turn one string into another. We chose this metric because it lets us compare the 2d output as a 1d diff operation.

The error rate is the Levenshtein distance divided by 156, the number of characters in a simulation output file.

7 Results and Analysis

Table 1: Simulation Similarity to Ground Truth

Mutant	Total Reward	Edit Distance	Error
wildtype	16.36	0	0%
roundabout	15.35	13	8.33%
slit	17.46	4	2.54%
comm	2.07	0	0%

See Appendix 1 to view side-by-side runs of our predicted ground truth and the simulation output.

Notably, the `wildtype` and `comm` simulations matched the ground truth precisely.

The `roundabout` mutant is the least similar to the desired output.

We also arrived at the following initial concentrations, decay rates, and weighting for the ligands.

Table 2: Ligand weights

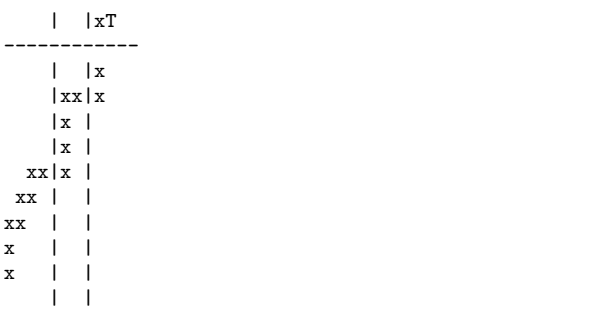
Ligand	Initial Concentration	Decay Rate	Weight
netrin	1	0.3	1
slit	1	0.3	1
shh	1	0.3	1
target	1	0.1	100

8 Error Analysis

For the majority of the project, we kept the midline homogenous: concentrations for netrin, slit, and shh were all uniform within the midline. However, this resulted in a behavior where the axon would never fully cross the midline until it got quite close to the synaptic target. We thought of two ways to fix this: either penalize the axon for turning, or make the midline non-homogenous.

`Wildtype Behavior, Homogenous Midline`

`Homogenous Midline, Target Ligand Decay Rate = 0.9`



Penalizing the axon for turning would reflect the biological reality that axons prefer to grow straight. In experimentation, finding the appropriate penalty weight proved difficult, and created strange and overly complicated growth patterns.

Making the midline non-homogenous created the desired behavior of crossing the midline.

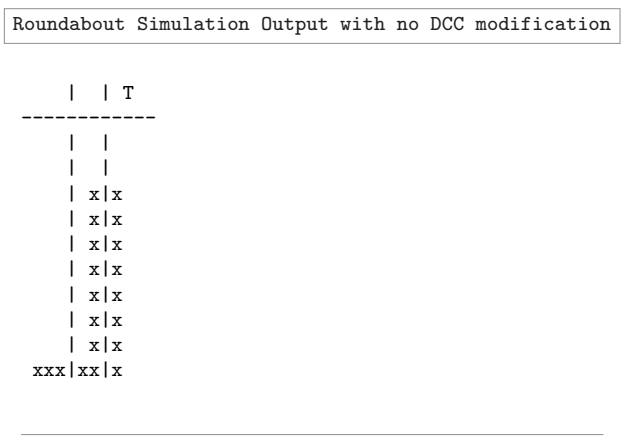
8.1 Strengths

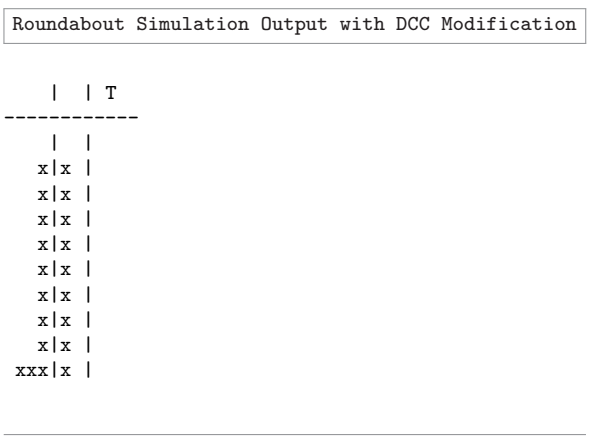
Explainable, Editable, Extensible, Efficient. Since the axon makes decisions based on a linear combination of weights, it is easy to decompose a decision into its constituent parts. Then, the weights can be adjusted to effect the way a decision is made. The implementation is therefore explainable and editable.

Moreover, adding new genes and gene interactions is relatively straightforward. One must add to the `geneConfig` and `activatedGene` dictionaries in `axon.py`. Adding new gene interactions happens in the `modulateGenes` method in `axon.py`. The relative ease of adding and modifying gene interactions leads to an exciting prospect: programming as mechanism proposal.

The greedy algorithm is also efficient: running 4 simulations on the 4 provided configuration inputs took an average of 0.2921 seconds with a standard deviation of 0.0317, as measured by the python time module (see Appendix 2).

Programming as Mechanism Proposal In attempting to extend axon agent in a way that reproduced the expected mutant behavior, this process proposed a mechanism that would produce this behavior. Since we restricted ourselves to programming in a biologically accurate way (for example, the axon only knew the local concentrations of ligands, never its absolute position in the environment), the act of extending the simulation is equivalent to proposing a hypothesis for the biological mechanism. From our biological understanding, the roundabout mutation results because there are actually multiple robo genes, and thus a mutation only in robo1 has an effect on slit repulsion in the midline, but does not result in the same phenotype as the slit mutation because there are still robo receptors present that are repelled by slit. This results in the roundabout mutant phenotype of the axon crossing and recrossing the midline, as it is repelled by slit but to a much lesser extent and still attracted to netrin and SHH in the midline. To represent a roundabout mutation, we created a robo2 gene. When this gene was mutated, our hypothesis was that the axon will be attracted to the midline (DCC activated and Comm deactivated) when it is not in the midline (represented by a slit concentration of 0). However, if it is in the presence of slit, it will be repelled since some robo receptors are still functioning, and therefore will leave the midline. This initial hypothesis led to the behavior depicted on the left of the figure below, with no modification to DCC. As pictured, the axon knows to cross into the midline when it's outside the midline, and vice versa. However, it does not recross the midline fully as it does in the biological phenotype. The key error in this behavior was that when the axon recrossed in the midline it moved up instead of left. Our first thought was that this could be due to the fact that it was the reward for netrin and SHH was too high, as there were higher concentrations of these molecules above than to the left. However, when we tried to turn DCC off (and thus the axon is no longer attracted to these ligands), it resulted in the behavior pictured to the right in the figure below, where the axon just crossed and recrossed on the other side of the midline. This led us to realize that currently our model had no incentive for the axon to fully recross the midline, making the behavior impossible. As such, in order to achieve this, we must come up with a hypothesis as to why the axon would be incentivized to move this way. This was a really interesting aspect of this project, as it shows that programming simulations may be used to hypothesize about unknown biological mechanisms. If the hypothesis works in the simulation, it may indicate that this is also how the mechanism works biologically, and can be further tested in a wet lab for confirmation. In other words, simulations can be used to discover things about biological mechanisms just like wet lab experiments can, and using a simulation may be more efficient.

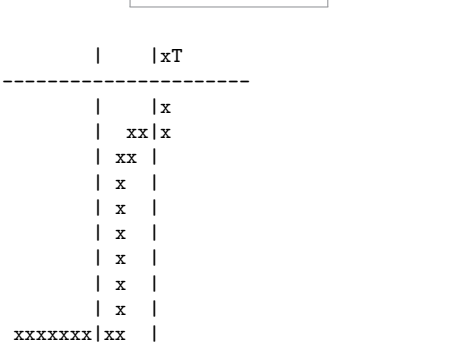




8.2 Weaknesses

Not robust to environment size The system is not robust with respect to changing environment size, which was a disappointing discovery. When the environment size was changed from having 10 columns to 20 columns, the following behavior was observed for the wildtype mutant:

Wildtype 20 Cols



This fragility likely has to do with the ligand concentration calculations. The column in the middle is chosen to have maximum concentrations, and all other columns are calculated to have some fraction of the maximum concentration depending on the distance. Once the axon reaches that maximum concentration column, it has no incentive to exit it. This fragility and susceptibility to local maxima is inherent in a greedy algorithm.

Edit Distance is not the best metric A shortcoming of the edit distance as an error metric is that it does not capture high-level structural similarity, only point differences. Since we really care about the high-level structure of the growth pattern, finding a different metric is an avenue for further research.

9 Future Work

The future work to consider for this project is as follows:

Lower Error for the Roundabout Mutation As discussed, the roundabout mutation in our project was the least accurate because the exact mechanism in which the axon fully recrosses the midline is still unclear. In future work, we would hope to derive a hypothesis that suggests why the axon has incentive to fully recross, and see if that leads to a more accurate simulation output.

Larger Simulation Environments Currently our simulation environment is an 11x10 matrix. As this is a relatively small environment, it would be interesting to increase the size of the matrix and play with factors such as increasing midline size to see what the outcome of axon behavior would be. Ideally, it would exhibit similar behavior, and if not we would want to look into why this is the case.

Extend System to Include Other Ligands and Genes While in our simulation we worked with the ligands and genes we believed to have the most significant effect on axon guidance, this process is incredibly complex. As such, future iterations of this project could include more ligands and genes present in this process in order to more accurately represent the environment axon guidance occurs in.

10 Code

https://github.com/kcallon/Axon-Guidance-Simulation

References

Rui Ponte Costa. Computational model of axon guidance. *arXiv preprint arXiv:1508.01537*, 2015.

CS Goodman and M Tessier-Lavigne. The molecular biology of axon guidance. *Science*, 274(5290): 1123–1133, 1996.

Mark Seeger, Guy Tear, Dolors Ferres-Marco, and Corey S Goodman. Mutations affecting growth cone guidance in drosophila: genes necessary for guidance toward or away from the midline. *Neuron*, 10 (3):409–426, 1993.

Zhuhao Wu, Shirin Makihara, Patricia T Yam, Shaun Teo, Nicolas Renier, Nursen Balekoglu, Juan Antonio Moreno-Bravo, Olav Olsen, Alain Chédotal, Frédéric Charron, et al. Long-range guidance of spinal commissural axons by netrin1 and sonic hedgehog from midline floor plate cells. *Neuron*, 101 (4):635–647, 2019.

Appendix 1

Wildtype Ground Truth

Wildtype Simulation Output

Roundabout Ground Truth

Roundabout Simulation Output

Slit Ground Truth

Slit Simulation Output

Comm Ground Truth

```
   x|  | T
------------
   x|  |
   x|  |
   x|  |
   x|  |
   x|  |
   x|  |
   x|  |
   x|  |
   x|  |
  xxx|  |
```

Comm Simulation Output

```
   x|  | T
------------
   x|  |
   x|  |
   x|  |
   x|  |
   x|  |
   x|  |
   x|  |
   x|  |
   x|  |
  xxx|  |
```

Appendix 2

Time Trial data:
10 runs of `python code/run_sims_compare_output.py` on a 2019 Macbook Pro, 1.4 Ghz Quad-core Intel Core i5, 16GB RAM.

Table 3: Time Trials

Trial	Elapsed Time (s)
1	0.3501
2	0.3271
3	0.3269
4	0.2719
5	0.3009
6	0.2643
7	0.2663
8	0.2786
9	0.2674
10	0.2683