How Intelligence Works - Neural Network Agent in Flatworld

Sheng Zha, Yiming Yang

December 14, 2010

Abstract

In order to help the agent in a 2-dimensional world to live longer by finding and eating nutrient objects, we implemented a series of decision models based on neural networks - first eye, then ears, and eventually memory. We discuss the design and inner architecture of these models and then analyze their performance during experiments.

Keywords

Flatworld **Neural Networks** **Memory**

1 Introduction

Neural network is a mathematical model and computational model that is inspired by the structure and functional aspects of biological neural networks, which turned out to be a powerful tool for complex tasks. In fact, inspiration from the nature's design often gives us hints for useful models and methods, for which genetic algorithm is another good example.

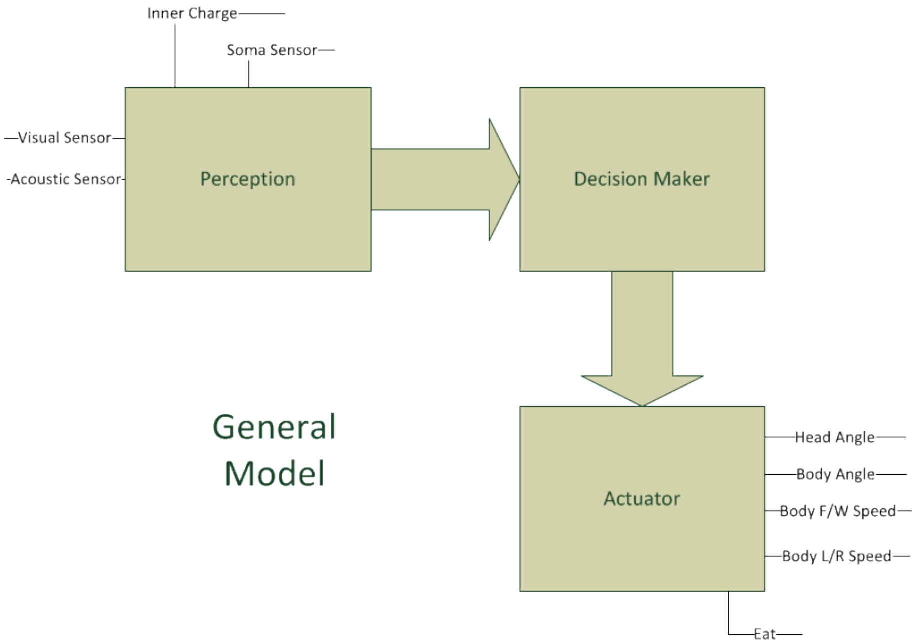
The task of this project is to design a computational module (a brain) for the agent to let it live as long as possible in the two-dimentional virtual world – FlatworldII. Flatworld is a simulated two-dimensional space that contains various kinds of 2D objects in it. These 2D objects consists of the agent, and edible objects. The agent has sensors, both internal and external, and actuators. The sensors include: three kinds of visual sensors dedicated respectively for R/G/B colors, acoustic sensors dedicated for ten different frequencies, eight soma sensors that give a binary code of whether certain edge intersects an object or not, and metabolic charge sensor. The agent is capable of various kinds of acts such as moving and eating by stimulating certain actuators. A reasonable goal for the agent is to find nutritious food objects and eat them and to live as long as possible.

We develop the brain model in a step by step way, through which we seek for the solution to the problems in previous models.

2 Approach

Our design of the agent's brain is biomimetic, which consists of three main functioning parts: perception part, strategic decision part and execution part (or driving part). Each of those parts do different job as their name indicate, and by observing the behavior of the agent we find the problems in each parts and try to improve them step by step.

Generally, we divide the development course of the brain model into five milestons including the milestone 0 which doesn't do anything, each of which solves a significant problem of the models in previous milestones.



A.II.1 General Model

Milestone0 does nothing. The models in it are just base values for later models. Milestone1 tries to make the agent eat objects in a greedy way. The agent of this series of models are to certain level capable of finding the nearest object and eats them. Milestone2 addresses the problem of the agent eating poisonous object and eating food object when it is not even hungry. Milestone3 exploits the effective usage of acoustic information and improvements in strategic decision making. Milestone4 introduces the transient memory and gains further improvement in the decision making part.

Model and module diagrams are available in the Appendix.II and Appendix.IV.

2.1 Milestone 0

Milestone0 consists of two models.

2.1.1 Model 0.0

The first model 0.1 just sits there waiting to die. This model lives 2000 timesteps and it would be the base value for all other models.

2.1.2 Model 0.1

The second model 0.2 still doesn't move its body, but it turns its head around. This behavior turns out to be useful for us to train the visual perception part.

2.2 Milestone 1

In this milestone we tries to achieve the greedy search for objects of the agent, which certainly adress the problem of previous models, which is that the agent doesn't move. Although the brightness is not the only criterion for distance, we use it for the sake of simplicity.

2.2.1 Developed Modules

Brightness calculator:
This module calculates the brightness that an eyelet sees. It sums the squared value of R/G/B and takes the square root of that sum.
Brightest Direction Finder:
This module gives the direction of the hightest brightness, and the angle output could be used in the execution part. The second layer in this module uses lateral inhibition, which allows the brightest node to remain positive and the rest returns to 0. No learning is involved in this network. The output node is a canonical model, whose weight is from the angle setting of the eyes.
With this module, the agent can find the direction of the brightest object in the field of vision. But what if the agent sees nothing?
See-Nothing Indicator:
This module simply expresses the idea that, if the sum of the brightness of all eyes is zero, then the agent is seeing nothing. Now the agent is able to tell whether it is heading towards the dark abyss. If so, we let the agent turns towards

another direction.

2.2.2 Model 1.0

This model turns both head and body towards the brightest object the agent sees, and eat everything it touches. If it sees nothing, it turns right by 30 degrees.

2.3 Milestone 2

The models within this milestone tries to solve the problem of object classification and improving the decision of whether to eat an object based merely on visual information.

2.3.1 Developed Modules

Color Classifier:

This is a competitive learning (winnner take all) network, which is capable of tell the type of certain colors. Still, there is a problem of different brightness. A model with only the color classifier gets a poor classification result without normalization. This module is trained in the model 0.1 by connecting it to the visual information interface.

Vision Normalizer:

Since we already got the brightness calculator in the previous model, it is easy to produce a vision normalizer which makes the agent able to grasp the invarient of color information. It divide the values of all channels whether After normalization, the color classifier works much better.

Type Decoder:

The function of type decoder is just splitting the type value 1/2/3 into three binary codes for the sake of easiness of training associator.

Type Associator:

The type associator tells the expected charge increment/decrement of certain type. It is trained using least mean square rule in model 1.0 by interfacing the inner charge information and visual information.

Nutrient-Food Object Classifier:

Food object classifier tells whether certain type of objects are food objects. Only the type of objects whose expected charge increment is positive is food type.

Food Object Visual Sifter:

This visual sifter only allows the visual information of food type pass through.

Hungry Indicator:

The hungry indicator tells whether the agent is hungry by thresholding. The threshold is set maually

2.3.2 Model 2.0

This model is made only able to see food object by filtering visual information. We only modified the perception part to make it able to tell whether an object is edible by visual information, and the rest parts are the same as the model 1.0.

2.3.3 Model 2.1

The actuator is the same as the model 1.0. This model makes use of the original visual information when it sees no food object. The idea is that food objects are possibly located around other objects. If it sees nothing at all, its behavior is the same as the model 1.0.

2.3.4 Model 2.2

The improvement in this model is that the decision of eating is no longer greedy. Instead, the agent only decides to eat when it is not hungry. In this way, the agent won't waste the extra charge that it is not able to store.

2.4 Milestone 3

2.4.1 Developed Modules

Sound/Noise Template Collector:
This neuron is used to record the sound template for later use. It moves its weight towards the wave pattern it perceives at certain rate.
Sound Template Denoiser:
It denoises the sound templates by subtracting certain times of the input buttom noise. The parameter here is picked through experiments and behavior observation.
Sound Intensity Calculator:
This neuron is similar to the brightness calculator, the only difference of which is

different number of dimensions.

Sound Difference-To-Template Calculator:

This module calculates the difference of the input patter to three recorded templates. It gives an analog quantity of difference which is quite useful.

Sound Classifier:

This classifier is almost the same as the color classifier. The three inputs are the three difference distances.

Sound Averager:

This module averages the acoustic information from the left and right ear and gives an average sound of what the agent is perceiving.

Valid-Touch Indicator:

In this model, we consider the 1 2 8 edges to be valid in order to avoid the accidental touch of poisonous object and eat by accident. These edges are always visually covered, so that the agent could always tell what kind of object is it touching.

2.4.2 Model 3.0

The actuator remains the same as in model 2.2. This model is the first attempt of the advanced usage of acoustic information. Since the sound the agent perceives is mainly affected by the nearest object, the sound information is capable of to certain level telling the type of the nearest object. This function of acoustic information is really efficient for judging the object type of the accidentally touches. The three original templates are obtained in the modified model in model 1.0 greedy eater. They are obtained when the agent is eating, and they are sorted according to the charge difference into three templates. These original templates are noisy and the classification based on those templates are not very effective. Hence, we come up with the idea of denoising. The noise templates are obtained by four anonymous agents in the milestone 0. They go straight up/down/left/right respectively and the they took three snapshots of the sound record at three equal distance. The mean of these twelve records forms the template for bottom noises. With these templates, we found through experiments a comparatively good set of templates, which could be used for nearby object classification. The sound food object classifier is trained in a similar way to the visual classifier. To handle the problem of accidentally eating poison, we use two standars in deciding whether certain touch is valid, which are the nearest object's sound information and the visual information. The front three edges are covered by the eye which could tell the type. The rear and side edges are covered by the two ears.

2.4.3 Model 3.1

This model tries to reduce the dependency on visual information when trying to find food objects. Since the agent is not able to look through other objects, and it has very limited angle of visual field, the visual information is quite limited. Sound information, on the other hand, is not blocked by anything. Hence, we reduced the part of logic of looking for non-food object.

2.4.4 Model 3.2

This model makes use of acoustic information as navigation when it sees no food object. Since there is a difference between the sounds two ears are perceiving, the difference of their similarity to the food object sound templates could be made use of by turning the agent to the direction where there are possibly food objects. The L/R comparator does a comparison of two difference values, and gives positive 1 when right side wins, and negative 1 otherwise. The rest part is the same as model 3.1.

2.5 Milestone 4

2.5.1 Developed Modules

Maintained Memory Cell:
The memory cell is an ordinary module which returns the stored brightness value and the stored angle. If the SET bit is 1 then the return value becomes the input, otherwise the memory is reset to origin and turns into unused state. It is maintained by the memory updater module through reasoning.
Memory Updater:
This module reasons the possible current state of certain memory entry. It reasons in the way of triangularization. The whole calculation uses the inverse of brightness as the unit for distance. The IBDRate, or Inverse Brightness - Distance Rate is trained in one of the anonymous model in milestone 2. It is obtained by dividing the difference of inverse brightness by the size of a step, and

only the rate for food objects are obtained. Since it would be very noisy because of the different comparative angle of the object edge to the agent, the obtaining process is under careful supervision.

Single Memory Cell Manager:

This module does the following job: if the input pattern is similar to this memory entry, then this memory entry is modified to the current input patter. If it's not that similar, then leave this entry alone.

Similar-Memory Eliminator:

This module is capable of eliminating similar patterns by reseting one of the two input entries. In the memory network, this module is added pairwise to each two of the memory cells.

The operation of insertion is performed by an inserter. It is a combination of a memory cell and all single memory cell managers. If update happens at the scope of single memory cell manager, then the insertion is complete. Otherwise, the ON and SET bit of a new cell is set to one, and the pattern is stored in this new cell.

2.5.2 Model 4.0

In this model, the memory system is introduced. What's stored in the memory is the observation of brightness and relative angle. The memory entries are maintained through reasoning, say, triangularization. The rest decision are the same as previous model 3.2, the only difference is that if the agent's current observation contains no food object, then it tries to recall the memory entries that might be around. The criterion for the distance of memory is the brightness, and if the reasoned memory brightness hits a threshold, then the corresponding entry is considered to be around.

2.5.3 Model 4.1

This model has an improvement in behavior control. If the agent sees nothing in front of it, and no other heuristic such as memory and ear navigation could help this, then instead of heading towards the abyss, the agent stops body movement and tries to turn its head to decide a good heading.

2.5.4 Model 4.2

The improvement in this model is also in the aspect of behavior control. It makes use of the waiting time before eating a food object when it is not hungry, by turing its head around and probing the surroundings and memorizing the favorable targets.

2.5.5 Model 4.3

This is our final model. In this model, we tries a human-like strategy, which makes use of two criteria for deciding whether to eat an object the agent is touching. If the agent is not very hungry and it has plenty of energy, then it uses another timestep to check these objects with eyes. If it's very hungry, then it becomes eager to eat objects, and it uses its ears to decide whether to eat within the same timestep. This model is robust enough, that if there is a food object within the agent's easy reach, it is very likely to find it. And because of the series of strategic decisions, it is able to maintain its energy at comparatively high level. Then if it's of low inner charge, it's likely that the agent is not gonna be able to find a food object, and the rest probing is a waste of time. Hence, in order to save that part of energy, we set another inner charge threshold, below which the agent becomes hopeless, and quietly sits there waiting for the apocalypse. In the meanwhile, it turns its head around very slowly. If it sees an object, then it can be relieved out of the despair and go for the food object.

3 Results

We've run 100 experiments for each model with a random start point. All figures are available in Appendix.III.

3.1 Model 1.0

Statistic Data: (100 Random Startpoint Experiments) Maximum timesteps lived: 511

Mean timesteps lived: 175.45

Variance of life span: 6109.8

Maximum number of objects obtained: 114

Mean number of objects obtained: 36.23

Variance of number of objects obtained: 285.2294

Percentage of food objects in obtained objects: 34.25%

Percentage of neutral objects in obtained objects: 33.43%

Percentage of poisonous objects in obtained objects: 32.32%

Average mileage gain before getting an object: 9.7107

Average mileage gain before getting a food object: 28.3496

Behavior:

The agent eats objects in a greedy way.

3.2 Model 2.0

Statistic Data: (100 Random Startpoint Experiments) Maximum timesteps lived: 1688

Mean timesteps lived: 1196

Variance of life span: 54967

Maximum number of objects obtained: 265

Mean number of objects obtained: 180.42

Variance of number of objects obtained: 1093.6

Percentage of food objects in obtained objects: 60.97%

Percentage of neutral objects in obtained objects: 20.39%

Percentage of poisonous objects in obtained objects: 18.63%

Average mileage gain before getting an object: 13.2604

Average mileage gain before getting a food object: 21.7475

Behavior:

It is capable of judging visually the type of objects. It accidentally eats other kinds of objects through accidental touch when passing by. Sometimes it keeps spining at the same point because it sees no food object.

3.3 Model 2.1

Statistic Data: (100 Random Startpoint Experiments) Maximum timesteps lived: 2169

Mean timesteps lived: 1679.3

Variance of life span: 79840

Maximum number of objects obtained: 268

Mean number of objects obtained: 210.89

Variance of number of objects obtained: 1136.9

Percentage of food objects in obtained objects: 64.97%

Percentage of neutral objects in obtained objects: 23.56%

Percentage of poisonous objects in obtained objects: 11.47%

Average mileage gain before getting an object: 15.9276

Average mileage gain before getting a food object: 24.5162

Behavior:

This model no longer spins at the same point. it turns to non-poisonous objects when it doesn't see a nutrient object.

3.4 Model 2.2

Statistic Data: (100 Random Startpoint Experiments) Maximum timesteps lived: 2837

Mean timesteps lived: 2202

Variance of life span: 87249

Maximum number of objects obtained: 266

Mean number of objects obtained: 209.28

Variance of number of objects obtained: 919.1127

Percentage of food objects in obtained objects: 65.16%

Percentage of neutral objects in obtained objects: 23.48%

Percentage of poisonous objects in obtained objects: 11.37%

Average mileage gain before getting an object: 15.9495

Average mileage gain before getting a food object: 24.4786

Behavior:

The agents of this model wait by a nutrient object when it is not hungry. If it finds an object and it's not hungry, it just waits there and do nothing.

3.5 Model 3.0

Statistic Data: (100 Random Startpoint Experiments) Maximum timesteps lived: 3436

Mean timesteps lived: 2901.3

Variance of life span: 128830

Maximum number of objects obtained: 267

Mean number of objects obtained: 216.48

Variance of number of objects obtained: 1098.6

Percentage of food objects in obtained objects: 70.4%

Percentage of neutral objects in obtained objects: 28.67%

Percentage of poisonous objects in obtained objects: 0.93%

Average mileage gain before getting an object: 18.6378

Average mileage gain before getting a food object: 26.4745

Behavior:

It rarely eats poisonous objects even when it's passing by one.

3.6 Model 3.1

Statistic Data: (100 Random Startpoint Experiments) Maximum timesteps lived: 3797

Mean timesteps lived: 3255.1

Variance of life span: 278940

Maximum number of objects obtained: 240

Mean number of objects obtained: 201.86

Variance of number of objects obtained: 1188.4

Percentage of food objects in obtained objects: 83.20%

Percentage of neutral objects in obtained objects: 16.28%

Percentage of poisonous objects in obtained objects: 0.53%

Average mileage gain before getting an object: 17.7655

Average mileage gain before getting a food object: 21.3538

Behavior:

This model has no apparent behavioral difference to the previous model.

3.7 Model 3.2

Statistic Data: (100 Random Startpoint Experiments) Maximum timesteps lived: 3740

Mean timesteps lived: 2776.2

Variance of life span: 464520

Maximum number of objects obtained: 248

Mean number of objects obtained: 175.3

Variance of number of objects obtained: 2229.4

Percentage of food objects in obtained objects: 80.48%

Percentage of neutral objects in obtained objects: 19.09%

Percentage of poisonous objects in obtained objects: 0.43%

Average mileage gain before getting an object: 16.5088

Average mileage gain before getting a food object: 20.5132

Behavior:

This model turns to certain direction because of the ear navigation when it sees no nutrient food in sight from time to time.

3.8 Model 4.0

Statistic Data: (100 Random Startpoint Experiments)

Maximum timesteps lived: 3804

Mean timesteps lived: 3259.3

Variance of life span: 191000

Maximum number of objects obtained: 247

Mean number of objects obtained: 201.18

Variance of number of objects obtained: 872.8764

Percentage of food objects in obtained objects: 83.58%

Percentage of neutral objects in obtained objects: 15.87%

Percentage of poisonous objects in obtained objects: 0.55%

Average mileage gain before getting an object: 17.8895

Average mileage gain before getting a food object: 21.4048

Behavior:

This model recalls its memory and tries to find the object according to memory when it sees no nutrient object in sight.

3.9 Model 4.1

Statistic Data: (100 Random Startpoint Experiments)

Maximum timesteps lived: 3421

Mean timesteps lived: 2882.9

Variance of life span: 145300

Maximum number of objects obtained: 215

Mean number of objects obtained: 172.1

Variance of number of objects obtained: 606.2551

Percentage of food objects in obtained objects: 84.47%

Percentage of neutral objects in obtained objects: 14.97%

Percentage of poisonous objects in obtained objects: 0.56%

Average mileage gain before getting an object: 15.0464

Average mileage gain before getting a food object: 17.8119

Behavior:

This model turns its head and scans around memorizing the location of nutrient objects when it is waiting to get hungry by a food object.

3.10 Model 4.2

Statistic Data: (100 Random Startpoint Experiments)

Maximum timesteps lived: 3786

Mean timesteps lived: 2129.0

Variance of life span: 784810

Maximum number of objects obtained: 237

Mean number of objects obtained: 126.68

Variance of number of objects obtained: 3253.7

Percentage of food objects in obtained objects: 83.34%

Percentage of neutral objects in obtained objects: 16.04%

Percentage of poisonous objects in obtained objects: 0.62%

Average mileage gain before getting an object: 17.0087

Average mileage gain before getting a food object: 20.4079

Behavior:

This model doesn't move its body when it sees nothing. Instead, it turns its head and tries to find the direction by turning its head.

4 Discussion

We carry out 100 experiments for each model with different start positions of the agent.

First, let's consider the mileage. From the results above, we found that the mileage is strongly related to its surviving time. So the longer the agent can survive, the longer it can run.

Then consider the surviving time and the objects the agent eats. In Model 1 series, agent can live for a very short time - its longest time is around 500 timesteps and its shortest time is less than 100 timesteps. This is because the agent in this model just eats greedily without judging which kind the touched object is. Since the objects are distributed uniformly around the world, the agent is quite easy to die by eating too many poisonous object.

In Model 2.0, we make the agent choose the position of the brightest nutrient object as its heading direction. So the agent can live longer than in Model 1.0 in average. And it can eat more nutrient objects than the other two kinds. However, from the life span figure, we can see that the difference of the longest and the shortest live time can be over 1000 timesteps. It tells that the model is far from robust.

In Model 2.1, we add the touch receptor to help judge whether the agent should eat the touched objects. As the figures show, the surviving time is promoted a bit and the longest time can reach 2000 timesteps. And the performance of eating nutrient objects is also improved a bit.

In Model 2.2, as we add a threshold function on the agent so that it can eat the nutrient object in a suitable time and do not waste the increasing of the charge. Then the agent can live much longer than the former models. But the performance of eating nutrient objects remains similar as that in Model 2.1.

As a whole, in Model 2 series, the performance of the agent is greatly improved. But they are not robust enough.

In Model 3 series, we add ears to help the agent. In Model 3.0, the ears help the agent decide whether to eat the touched objects or not. Then it can live up to 3400 timesteps, much better than Model 2 series. And as we classify both the nutrient and the normal objects as eatable ones, the number of eaten poisonous objects can be definitely reduced.

In Model 3.1, we just change the direction choosing function a bit, so it doesn't not turn to food or poisonous objects. And its surviving time can reach up to 3700 timesteps. And it can eat more nutrient objects.

In Model 3.2, we make the ears help choose the heading direction. As a result, the average surviving time is longer than than in Model 3.0 and 3.1.

As a whole, after adding ears to the agent, Model 3 series acts better than Model 2 series, but the agent can still miss some nearby nutrient objects because it doesn't see them and the ears can be affected by noise.

In Model 4 series, we add the memory to the agent. Model 4.0 is the original one with just memory contained. As memory can help the agent besides the eye and ears, the surviving time still remains rather high and the variance can be reduced. So the agent acts a statistically better performance within this model.

In Model 4.1, we add a function that when the agent waits for the right time to eat the touched nutrient object, its head can scan the other directions to remember nutrient objects within its memory. The surviving time and the eating performance can still be very high. But since the direction of the head is not always the same as that of the body, the agent can step out of the world and never come back. So for some experiments, the surviving time is rather low.

In Model 4.2, we make the agent search for normal food when it cannot scan any nutrient objects by eye, ears or memory. So in this model, the surviving time can be longer than that in Model 4.1. However, the average performance is not as good as that of Model 4.0.

Because of the time limit, we failed to complete the implementation of the final model. But we are already on the half way.

To sum up, by adding eye, ears and memory, the agent becomes more and more skilled in finding nutrient objects and survive longer. Especially the memory, can definitely improve the surviving time of the agent, which is what we expect.

5 Acknowledgements

Special thanks to Professor Thomas P. Caudell, who provided us with the opportunity to have one of the greatest courses we've ever had, and also who wrote this simulated world flatworld II. This is a very interesting course project and we found ourselves sort of indulged into it. Seeing the agent becomes more capable and our original ideas realized, our happiness and feeling of fulfillment

are beyond words.

6 References

Simon Haykin, Neural Networks - A Comprehensive Foundation, Second Edition 2008.

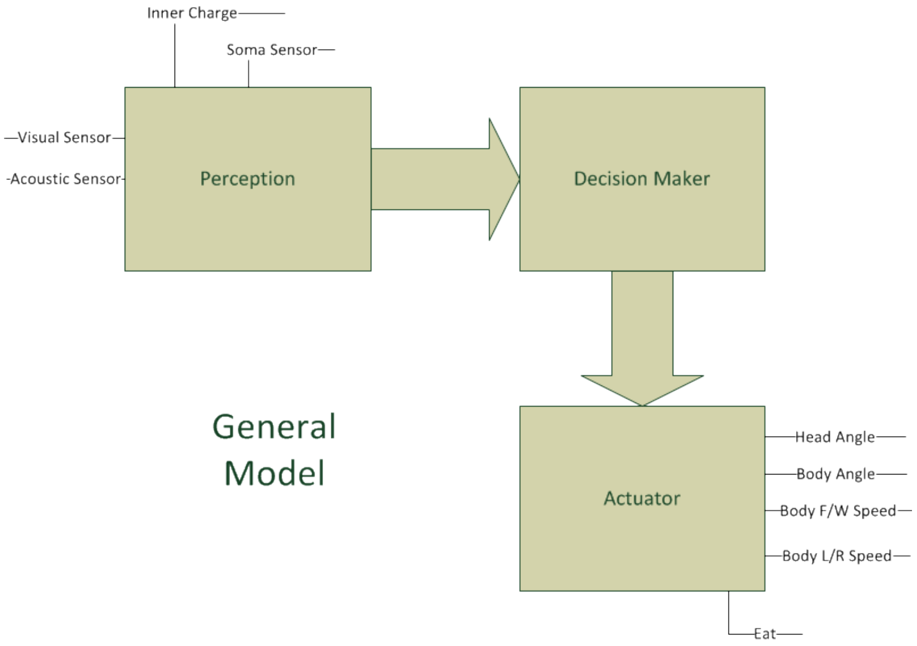
FlatWorld II V1.0 Class Version.

Appendix

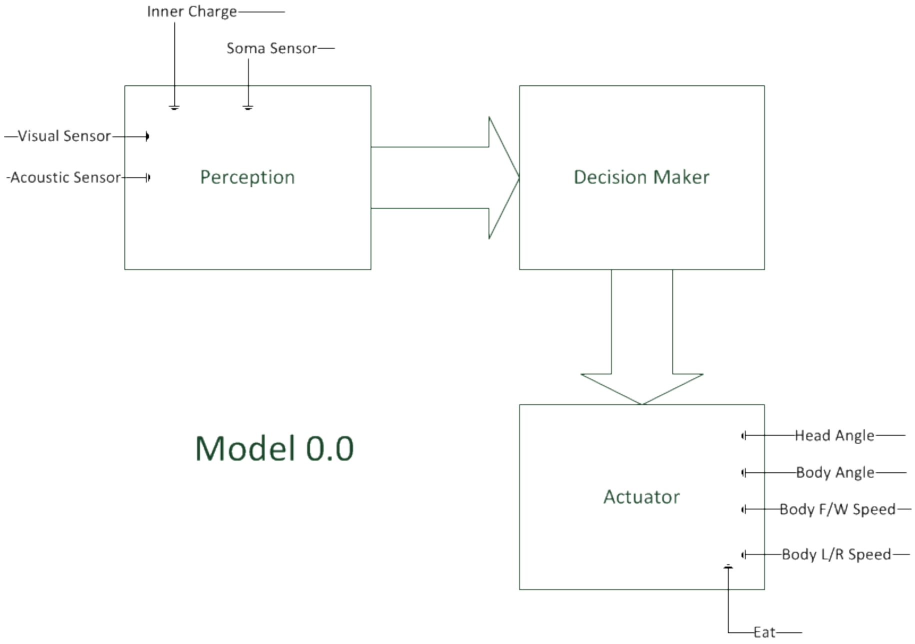
I. Source code We divide the source code files into two packages. One is for the OpenGL version, the other is for the non-graphics version. The first category runs slowly but you can see the behaviors of the agent directly. The latter one can run quite fast, so we use it to run experiments. They are both in the attachment.

Besides, there is a Bash file named *t.sh*. It is for running programs for a specific times and recording the results in another file. Therefore, we just let all the models only output the related statistics information when the agent dies. And in *t.sh*, in each iteration, it will call the program once.

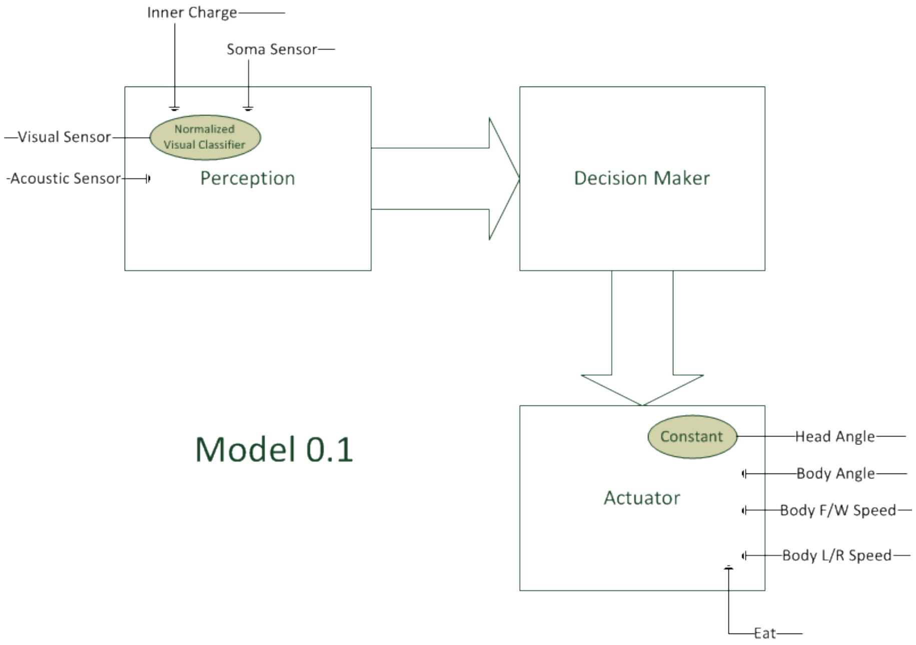
II. Model Framework Diagrams



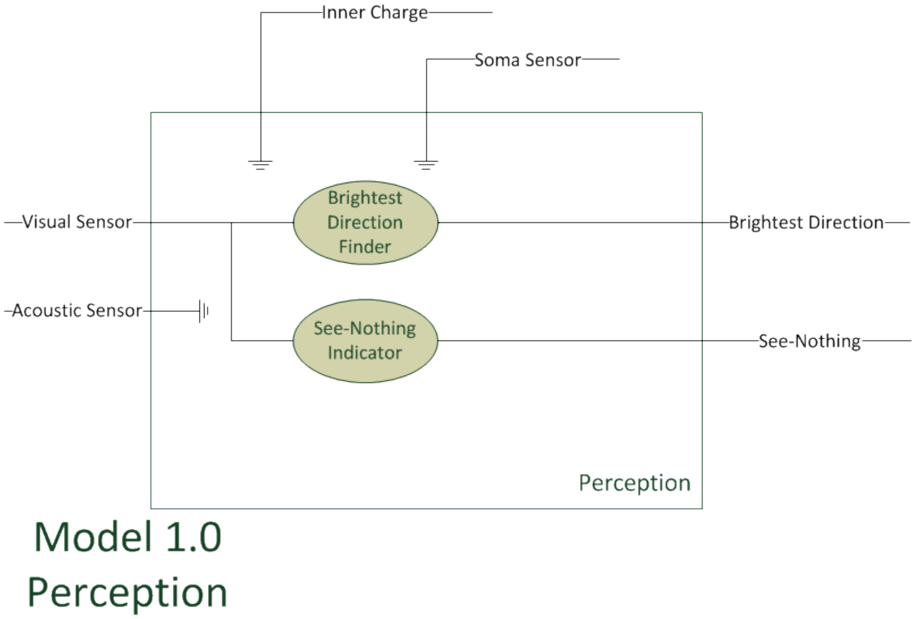
A.II.1 General Model



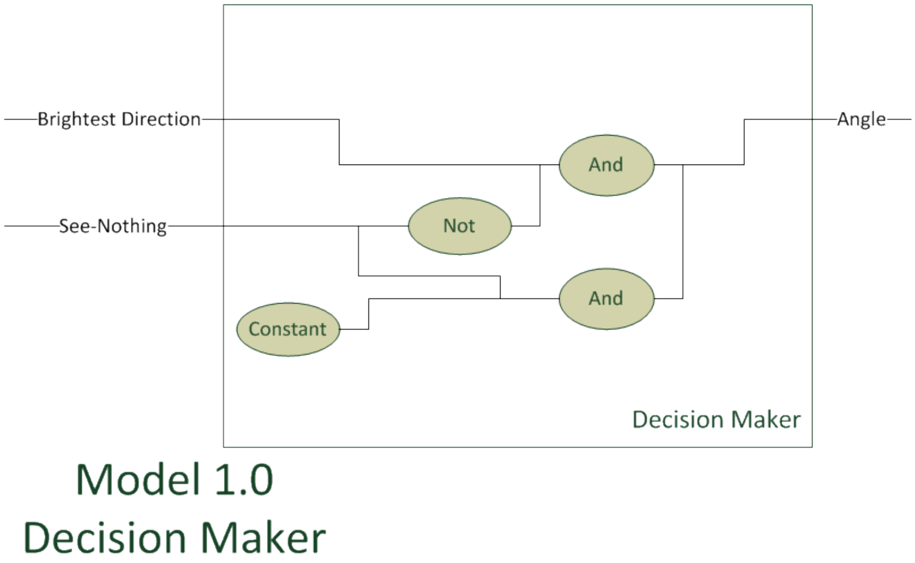
A.II.2 Model 0.0



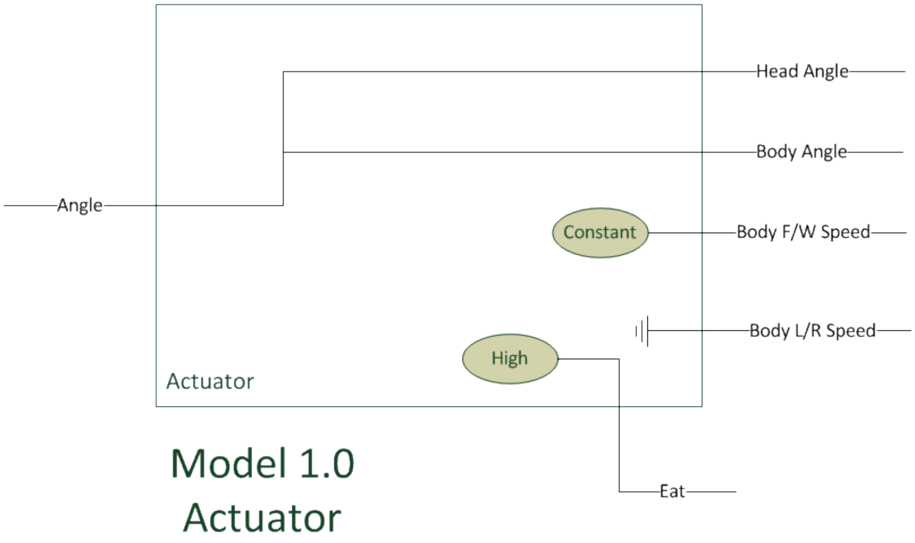
A.II.3 Model 0.1



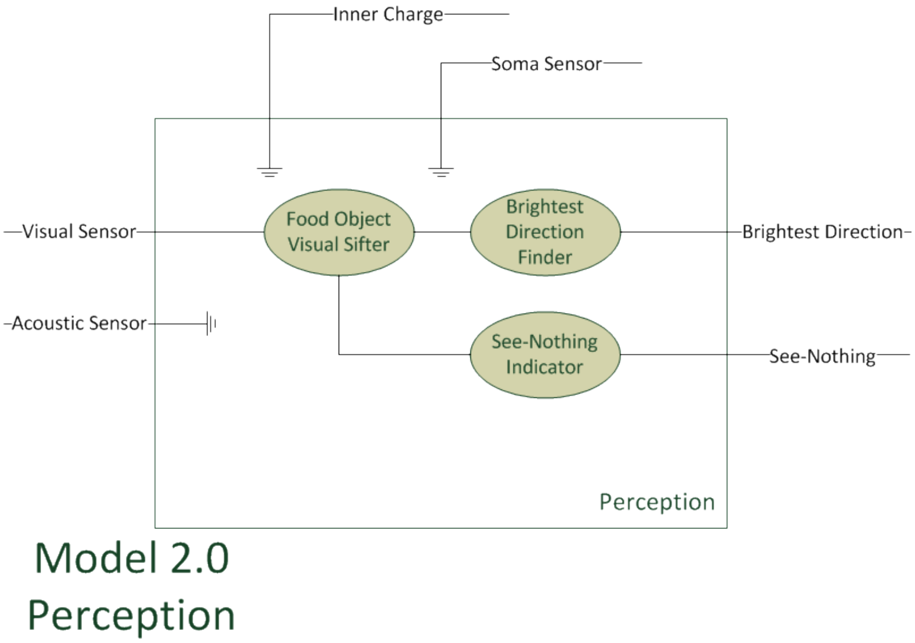
A.II.4a Model 1.0 Perception Part



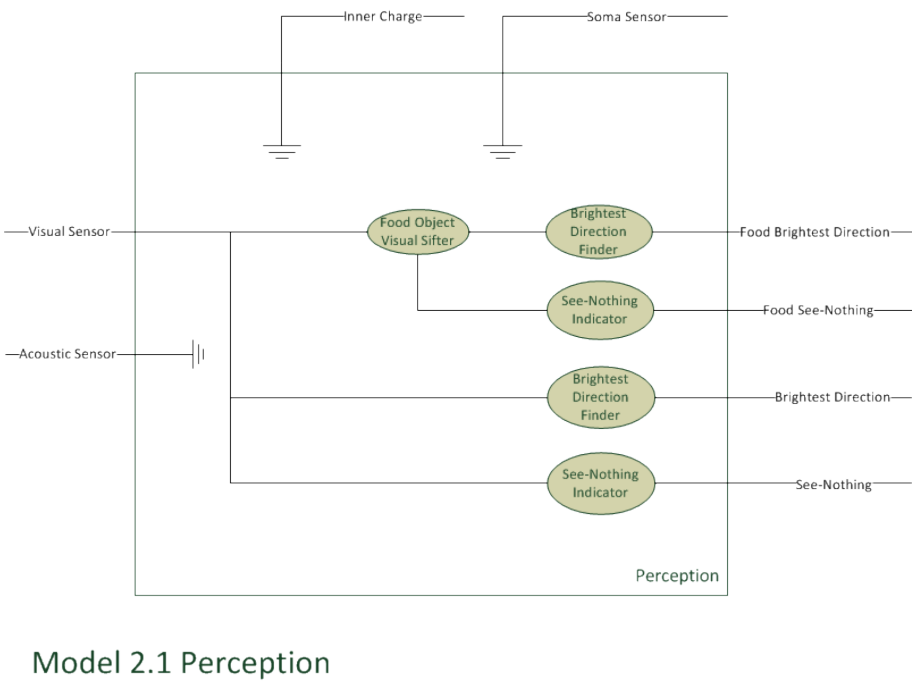
A.II.4b Model 1.0 Decision Maker



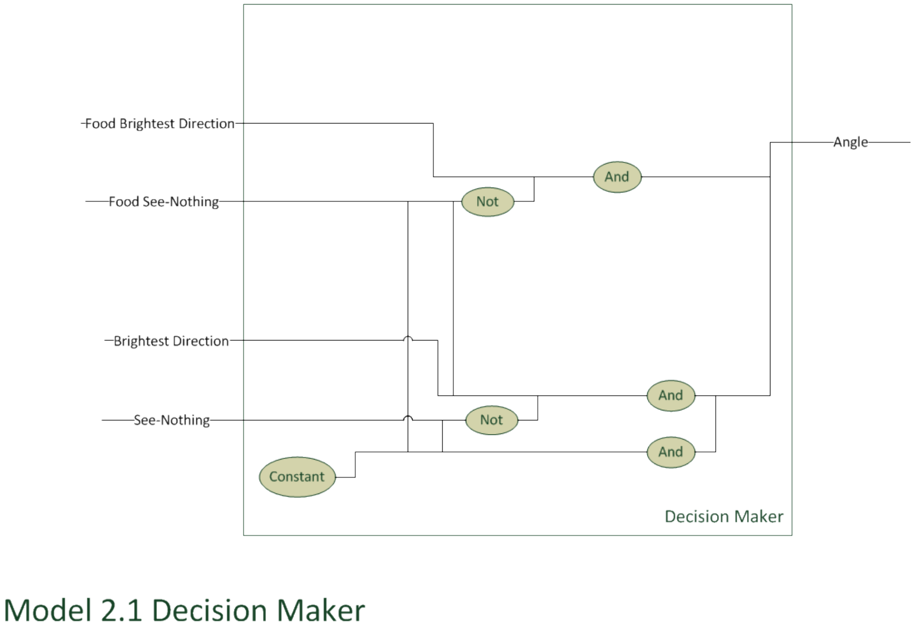
A.II.4c Model 1.0 Actuator



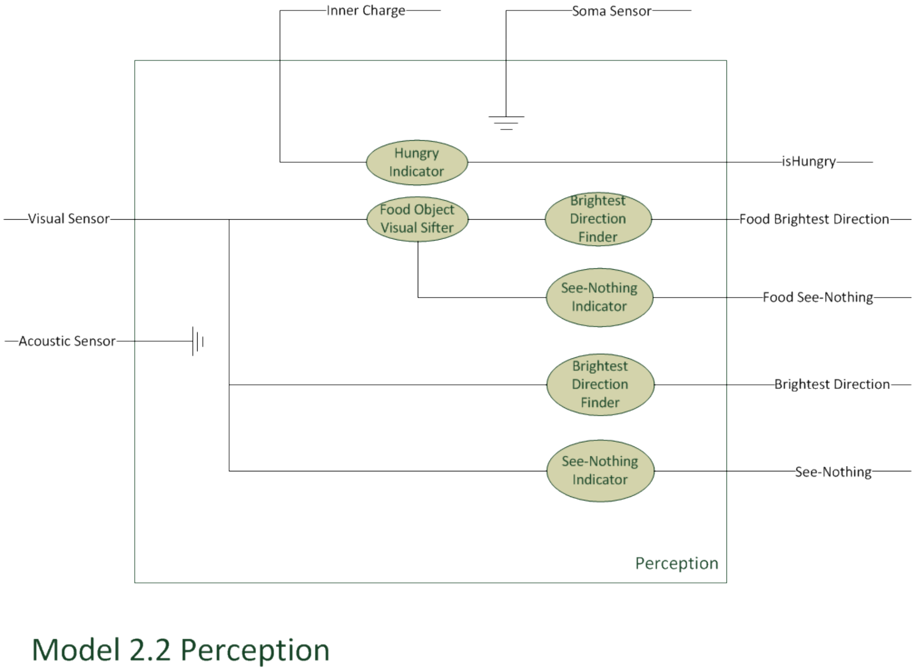
A.II.5 Model 2.0 Perception Part



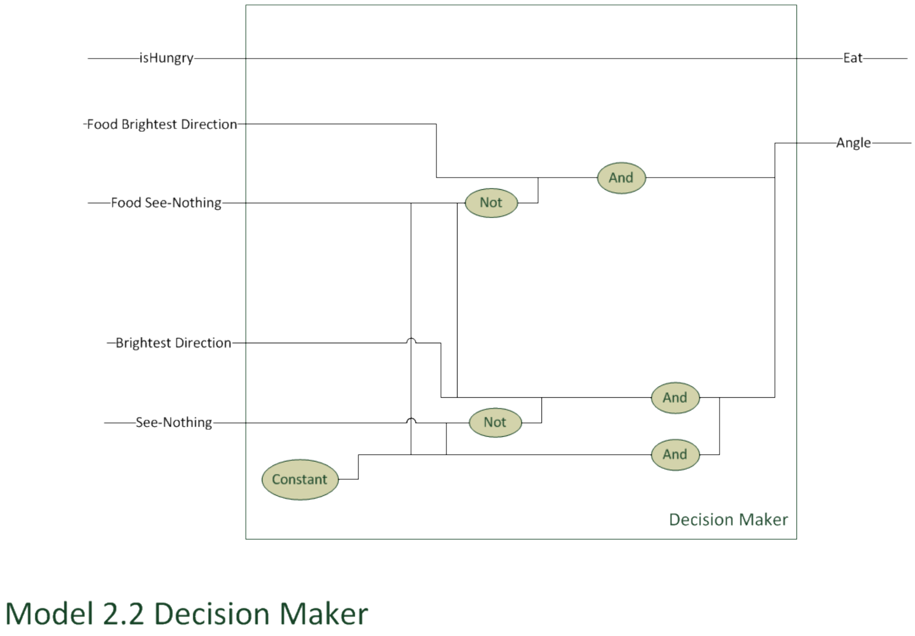
A.II.6a Model 2.1 Perception Part



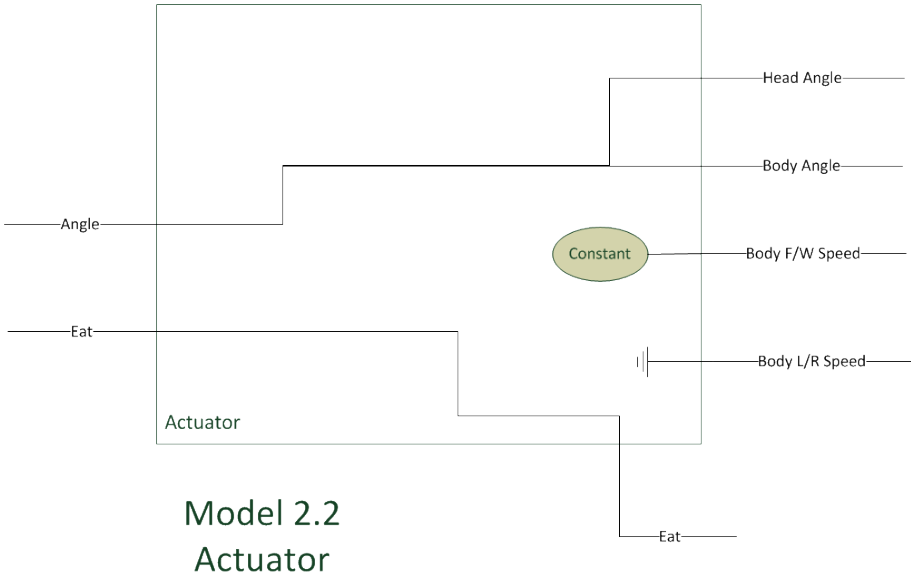
A.II.6b Model 2.1 Decision Maker



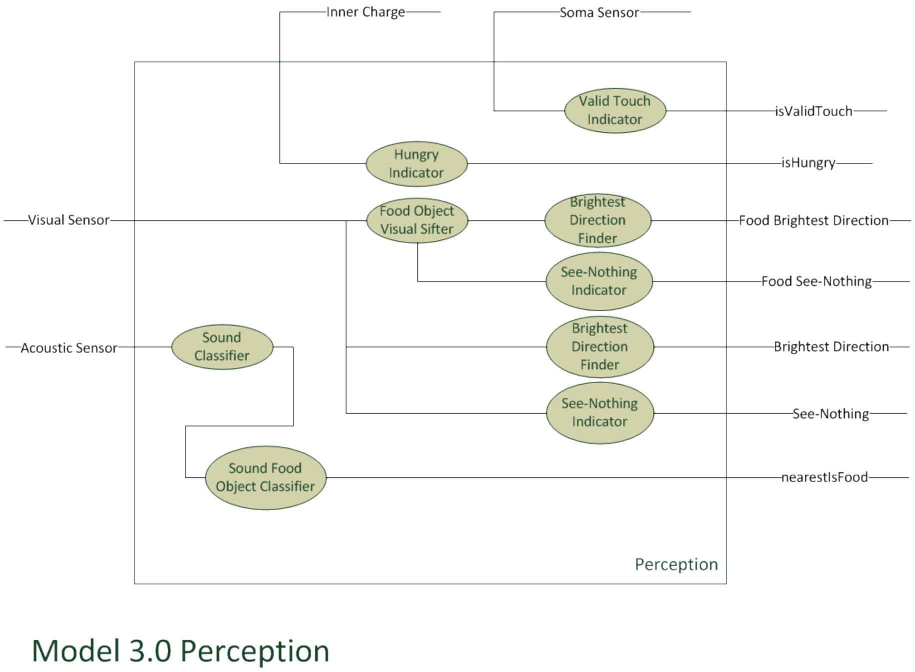
A.II.7a Model 2.2 Perception Part



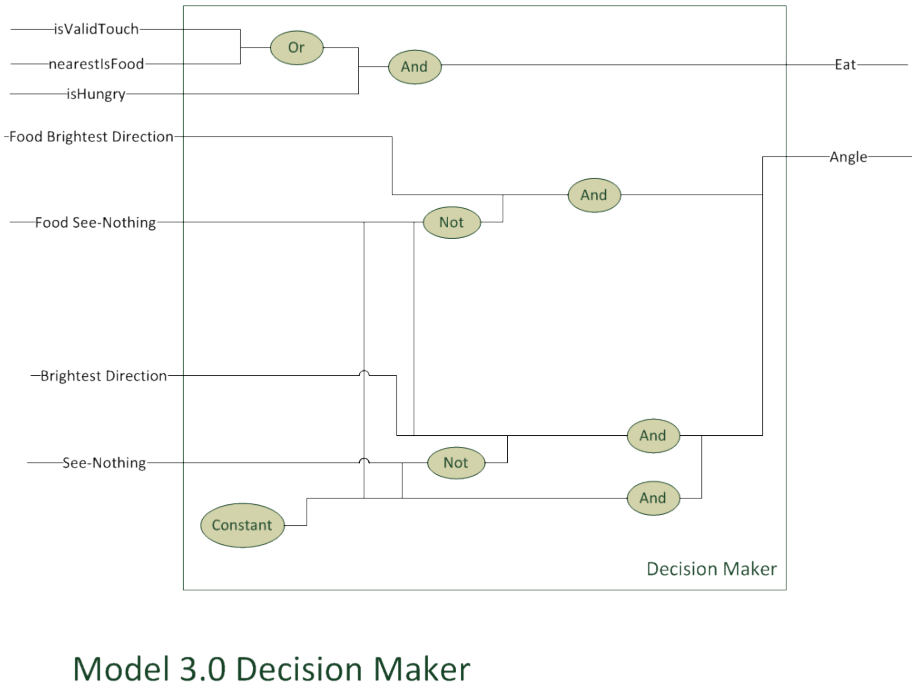
A.II.7b Model 2.2 Decision Maker



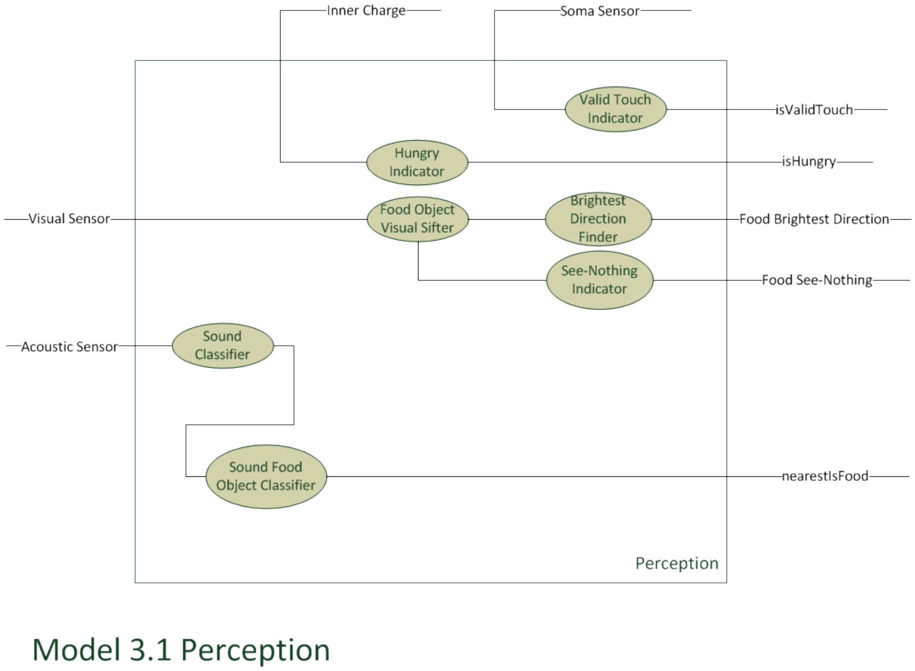
A.II.7c Model 2.2 Actuator



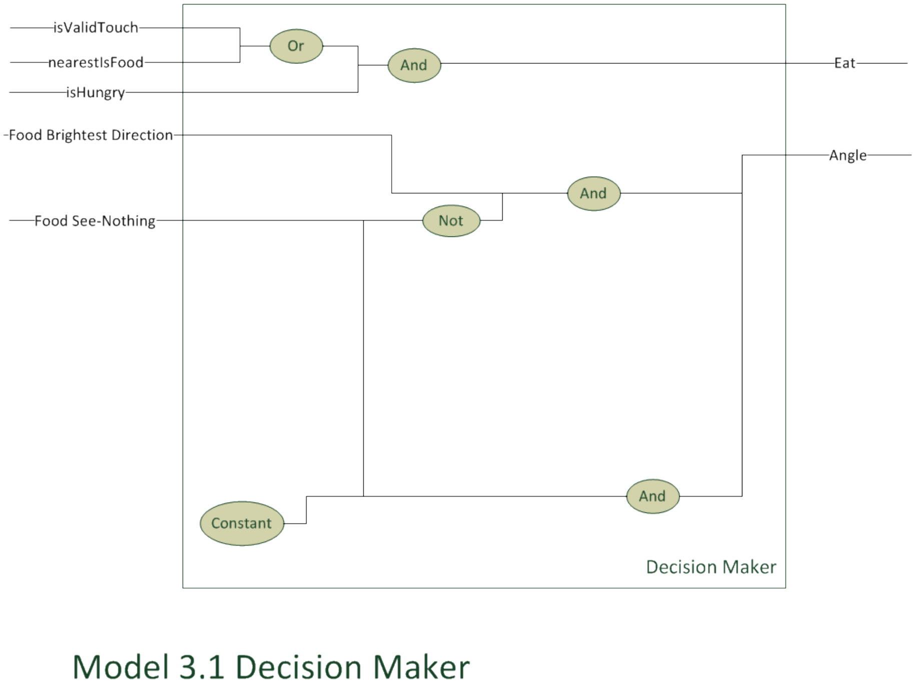
A.II.8a Model 3.0 Perception Part



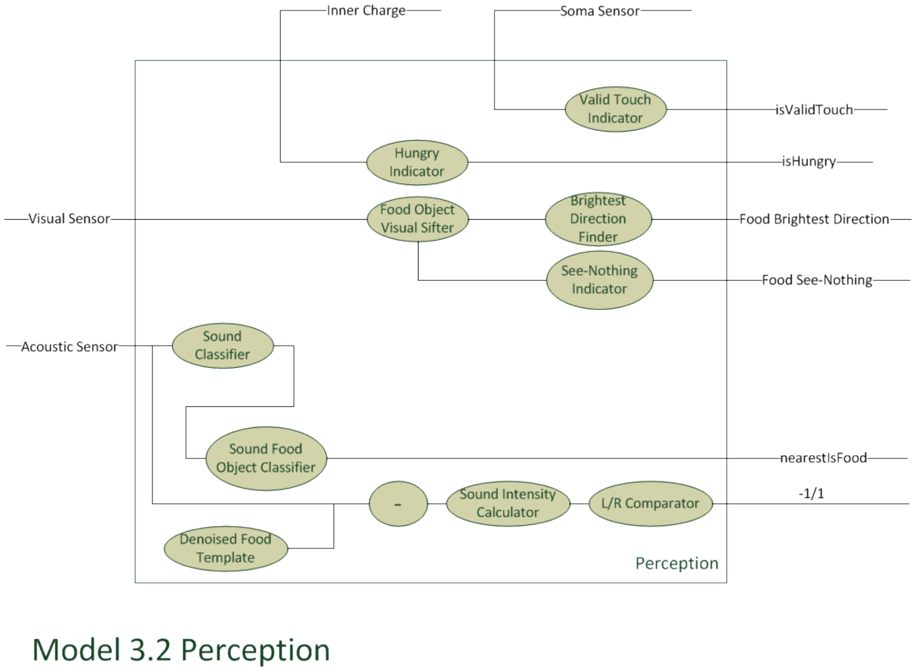
A.II.8b Model 3.0 Decision Maker



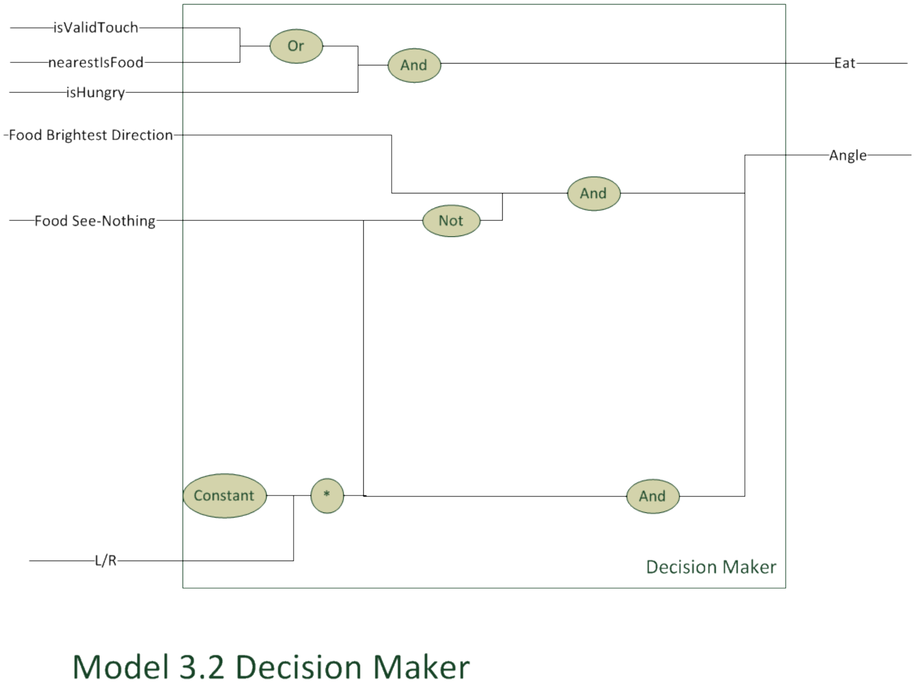
A.II.9a Model 3.1 Perception Part



A.II.9b Model 3.1 Decision Maker

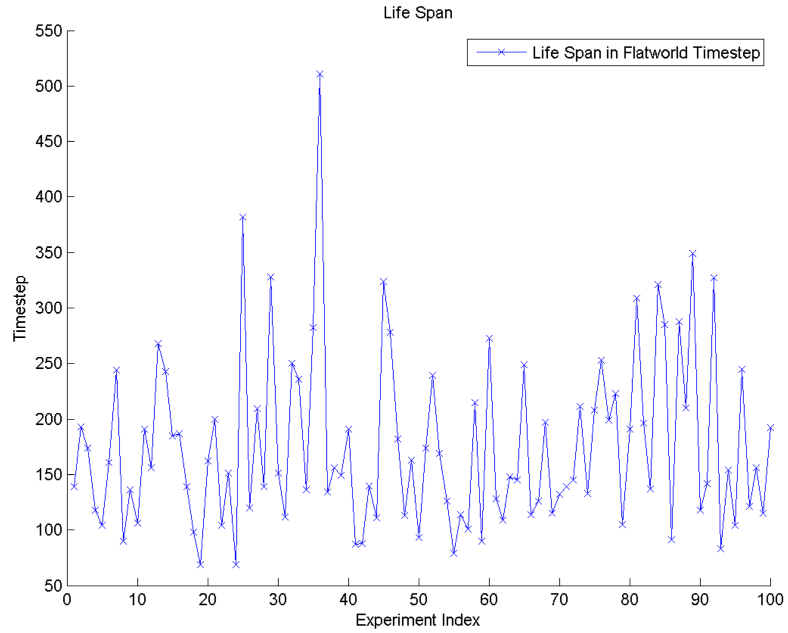


A.II.10a Model 3.2 Perception Part

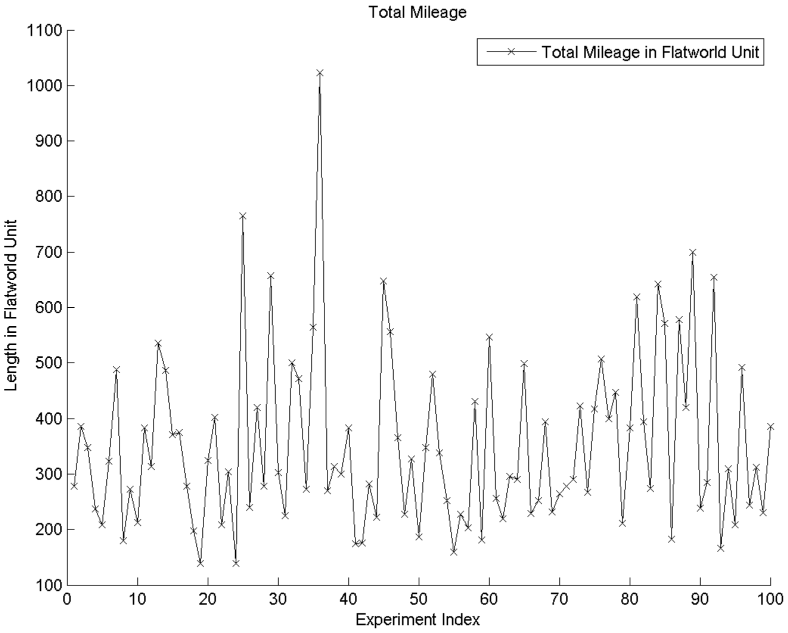


A.II.10b Model 3.2 Decision Maker

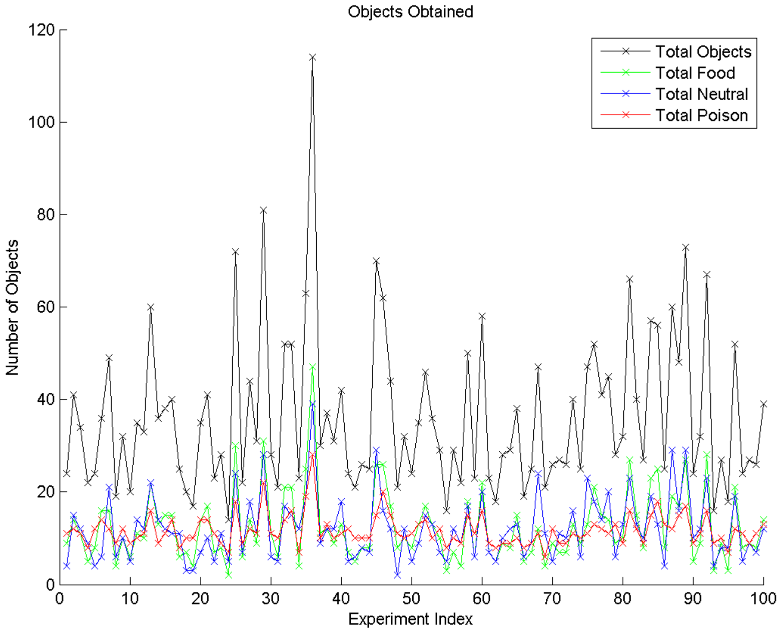
III. Results Statistic Figures



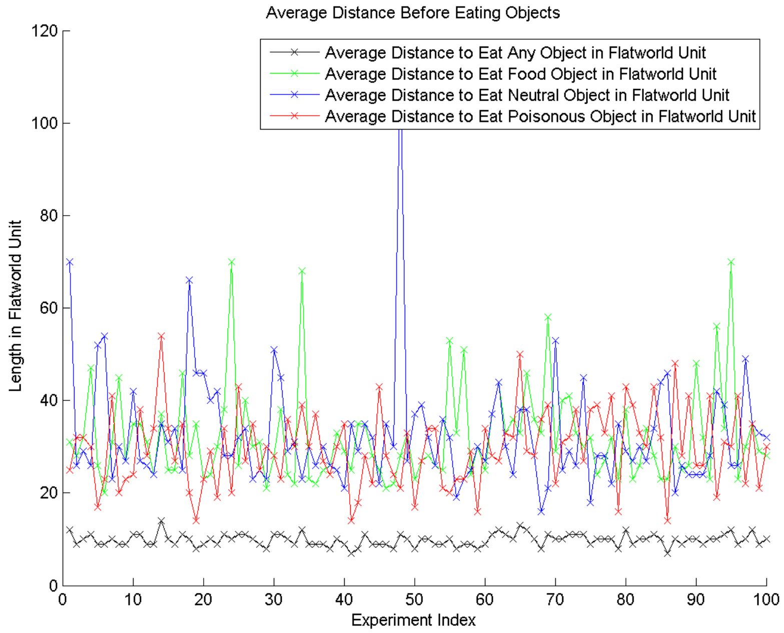
A.III.1a Model 1.0 Life Span



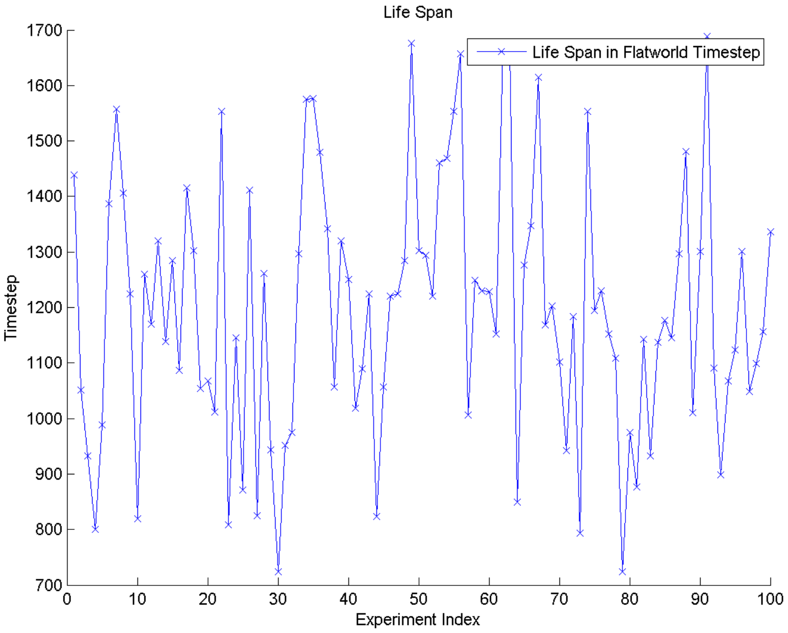
A.III.1b Model 1.0 Mileage



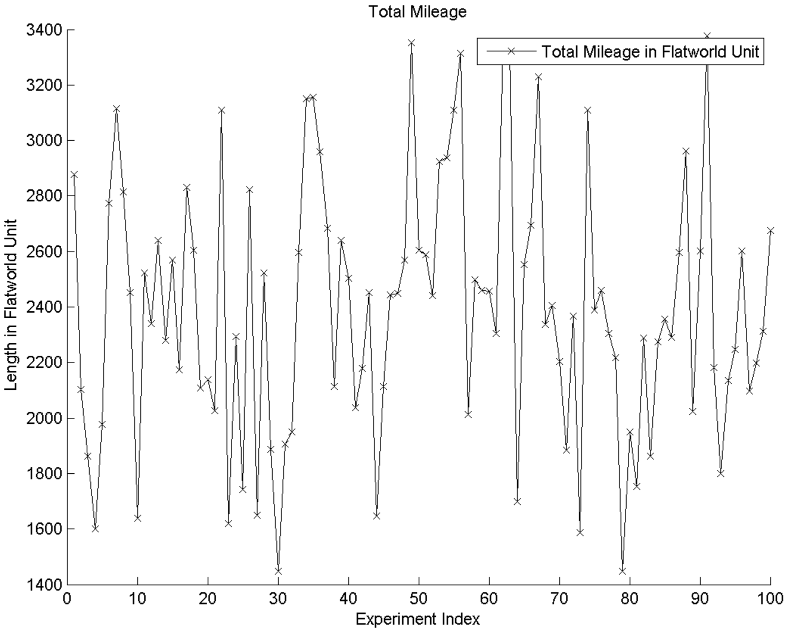
A.III.1c Model 1.0 Obtained Objects



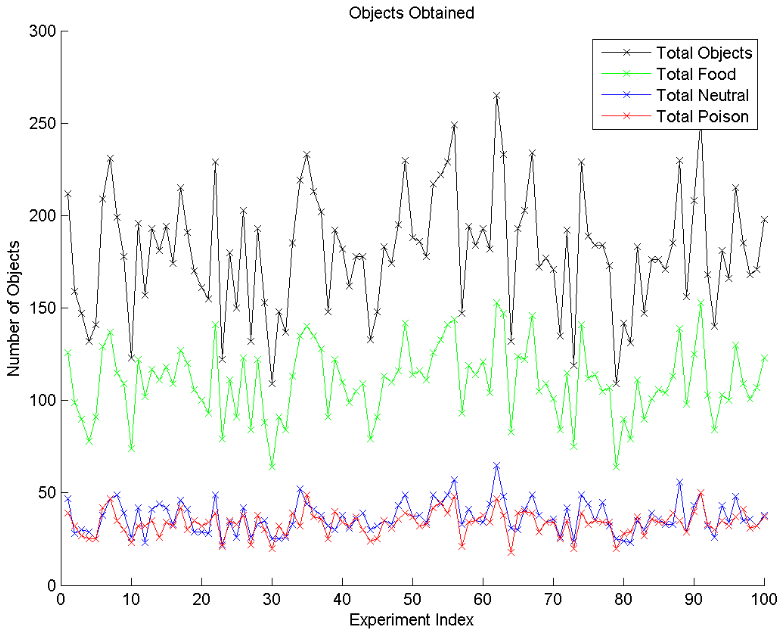
A.III.1d Model 1.0 Average Distance Before Eating Objects



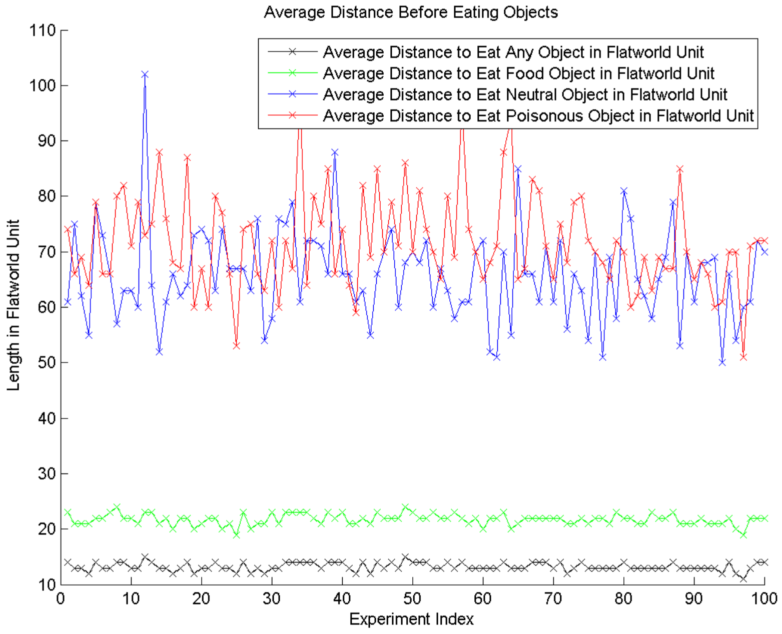
A.III.2a Model 2.0 Life Span



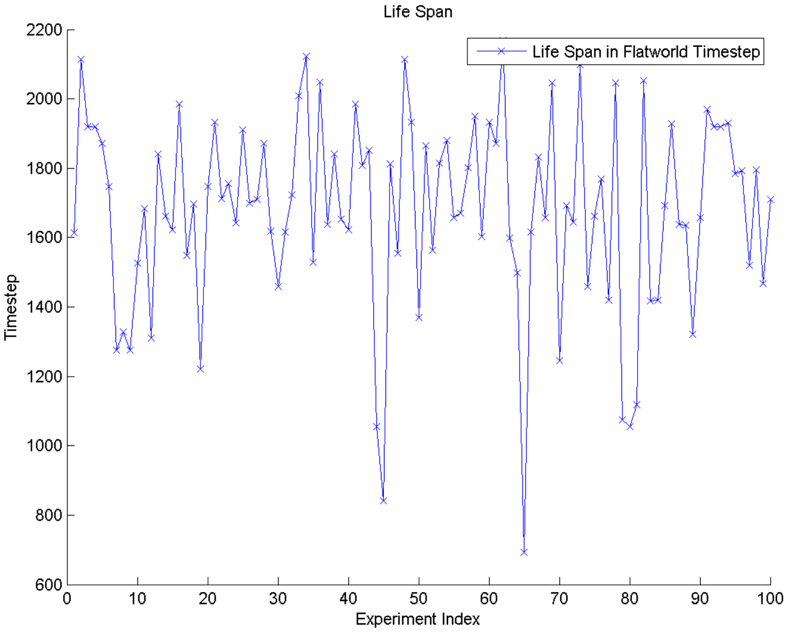
A.III.2b Model 2.0 Mileage



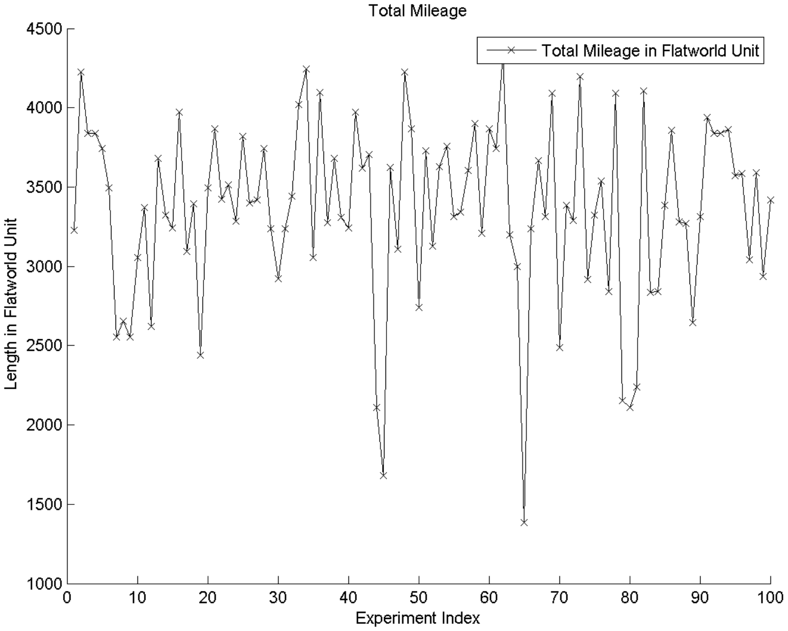
A.III.2c Model 2.0 Obtained Objects



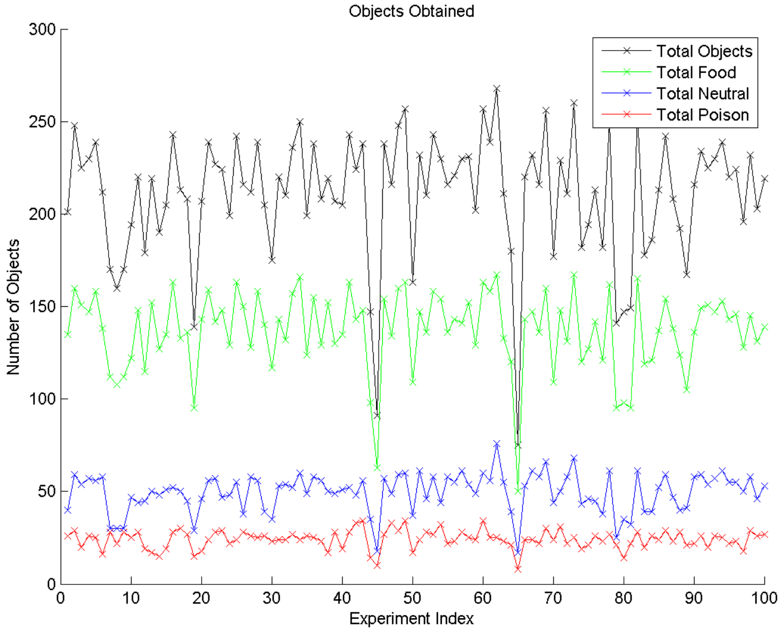
A.III.2d Model 2.0 Average Distance Before Eating Objects



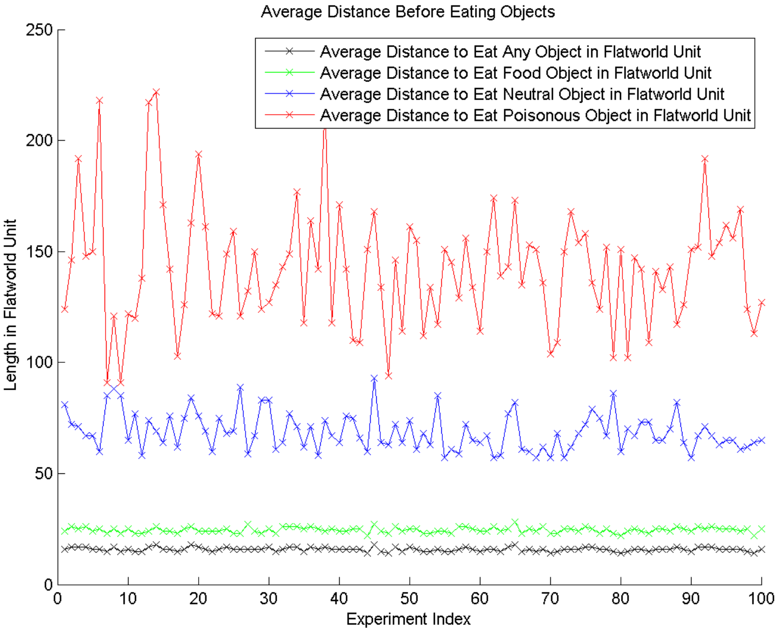
A.III.3a Model 2.1 Life Span



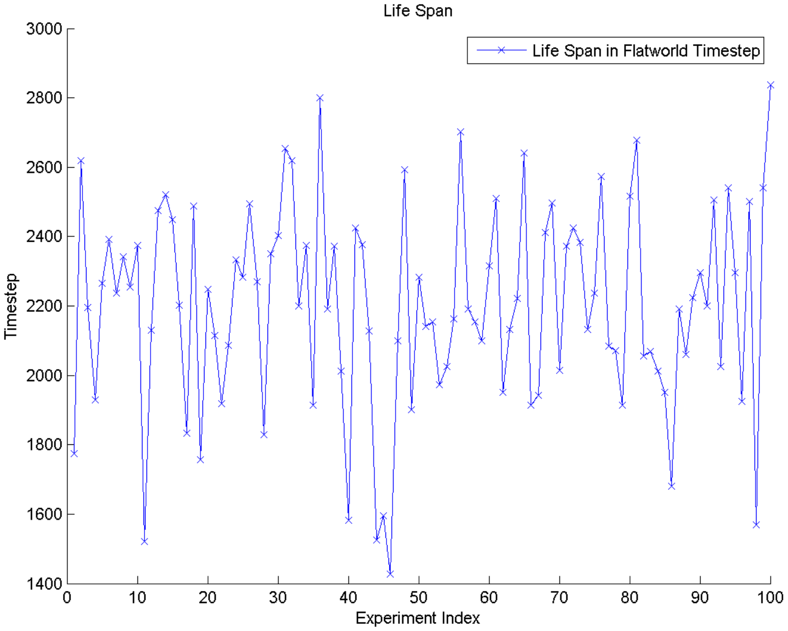
A.III.3b Model 2.1 Mileage



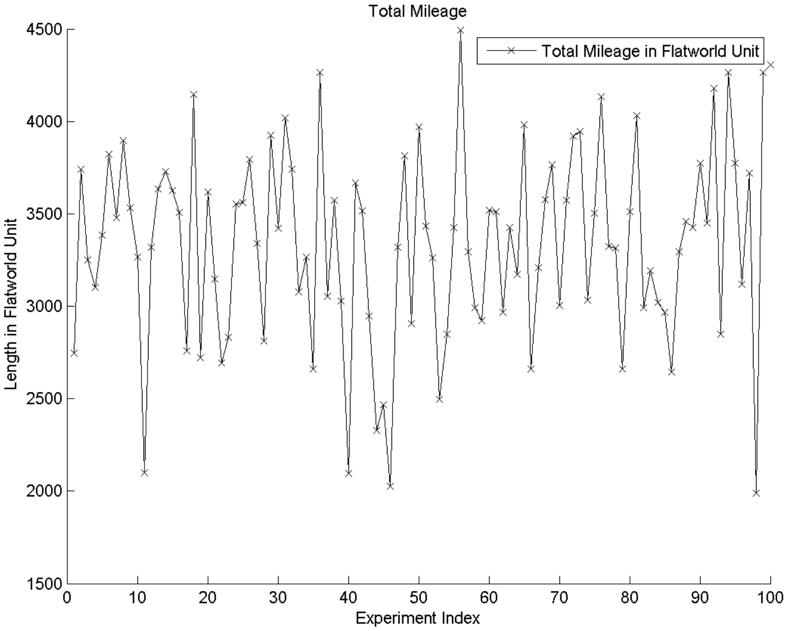
A.III.3c Model 2.1 Obtained Objects



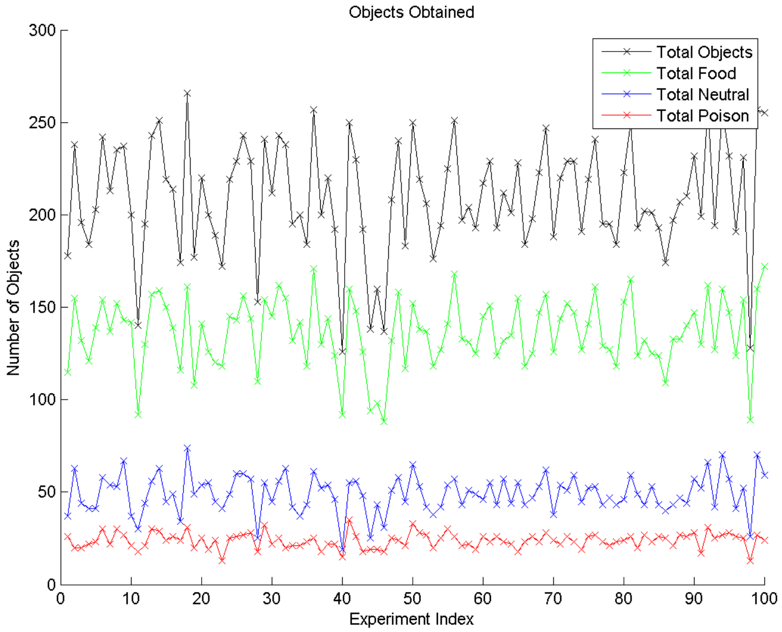
A.III.3d Model 2.1 Average Distance Before Eating Objects



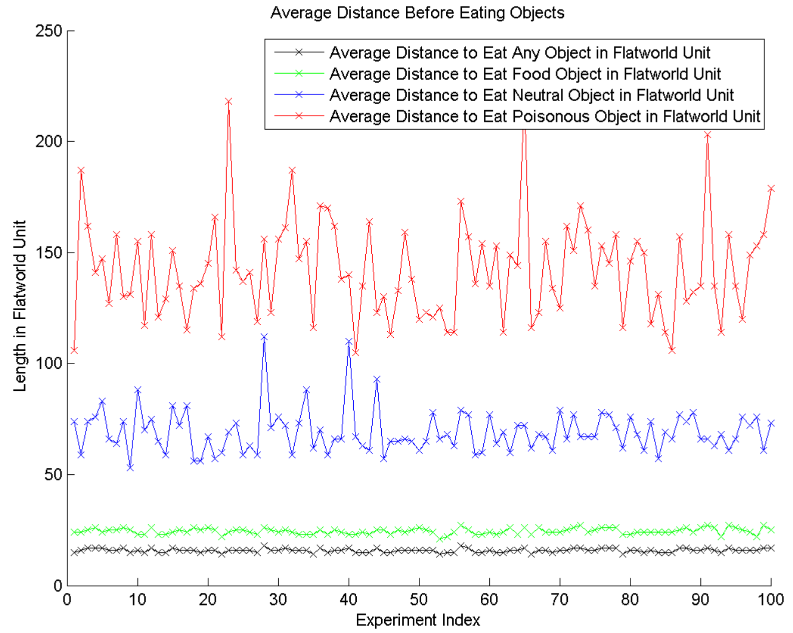
A.III.4a Model 2.2 Life Span



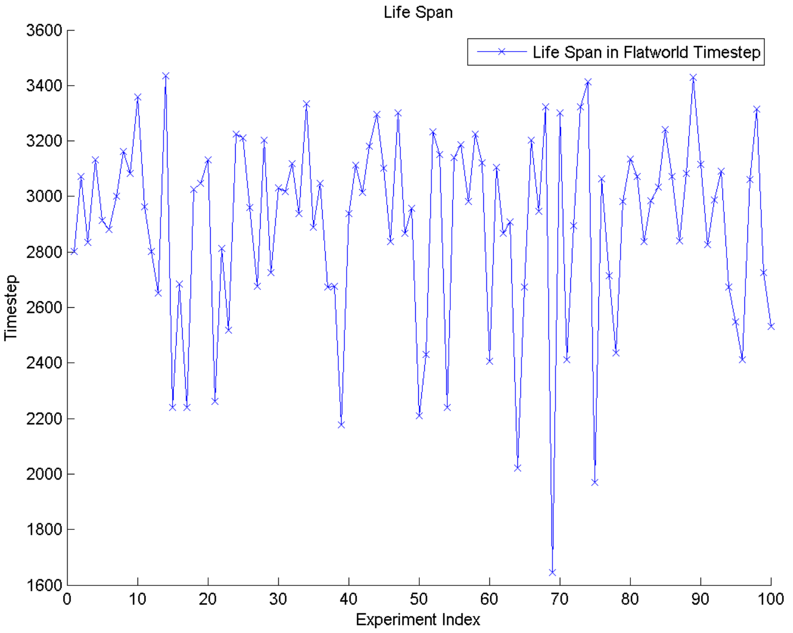
A.III.4b Model 2.2 Mileage



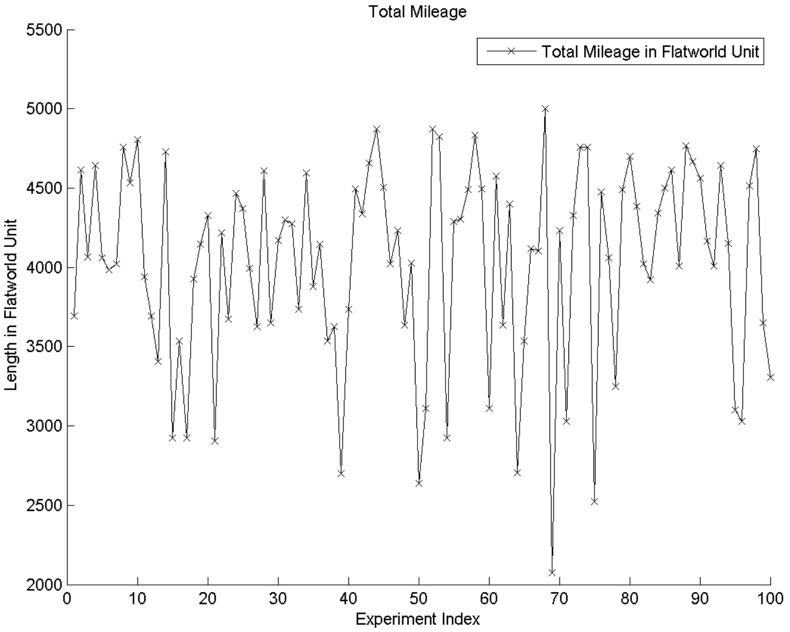
A.III.4c Model 2.2 Obtained Objects



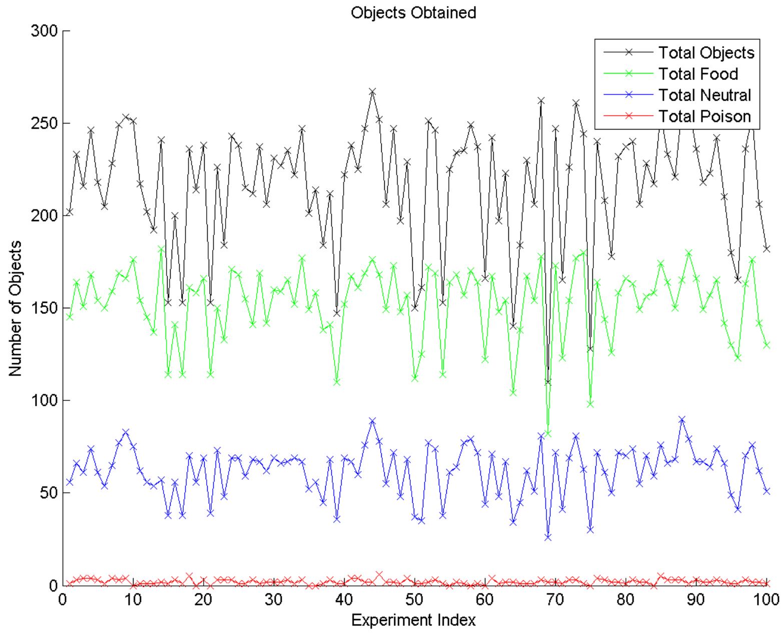
A.III.4d Model 2.2 Average Distance Before Eating Objects



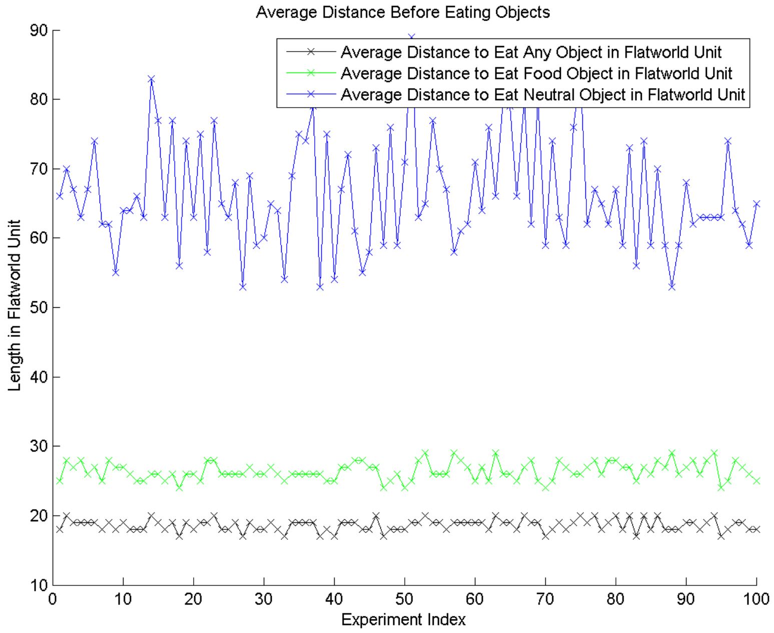
A.III.5a Model 3.0 Life Span



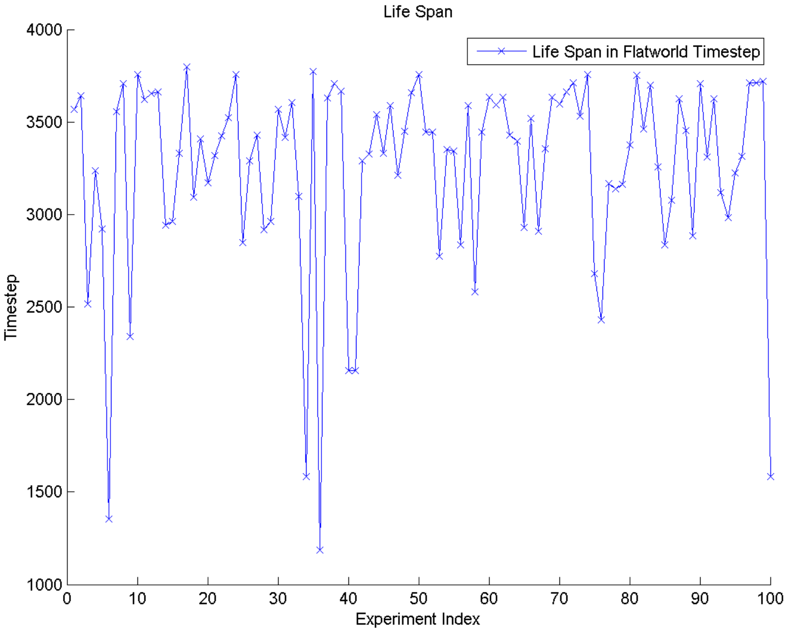
A.III.5b Model 3.0 Mileage



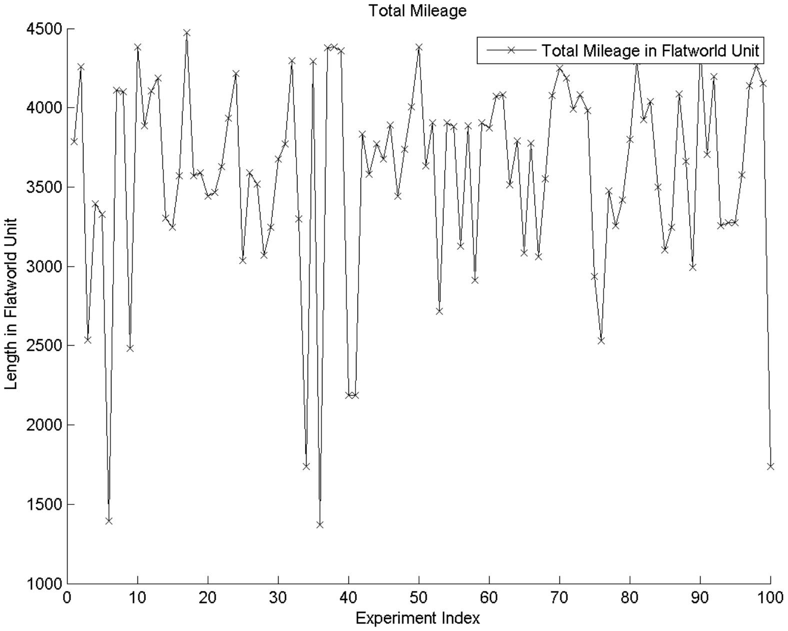
A.III.5c Model 3.0 Obtained Objects



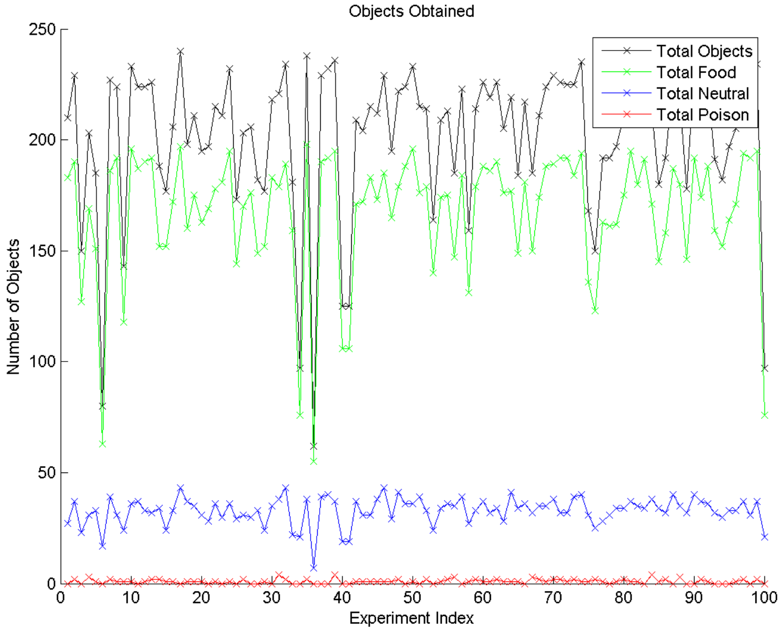
A.III.5d Model 3.0 Average Distance Before Eating Objects



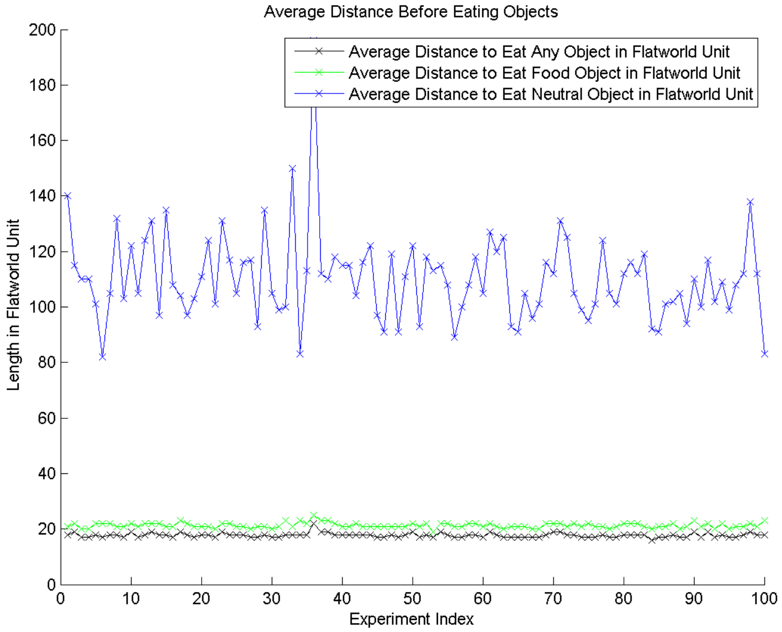
A.III.6a Model 3.1 Life Span



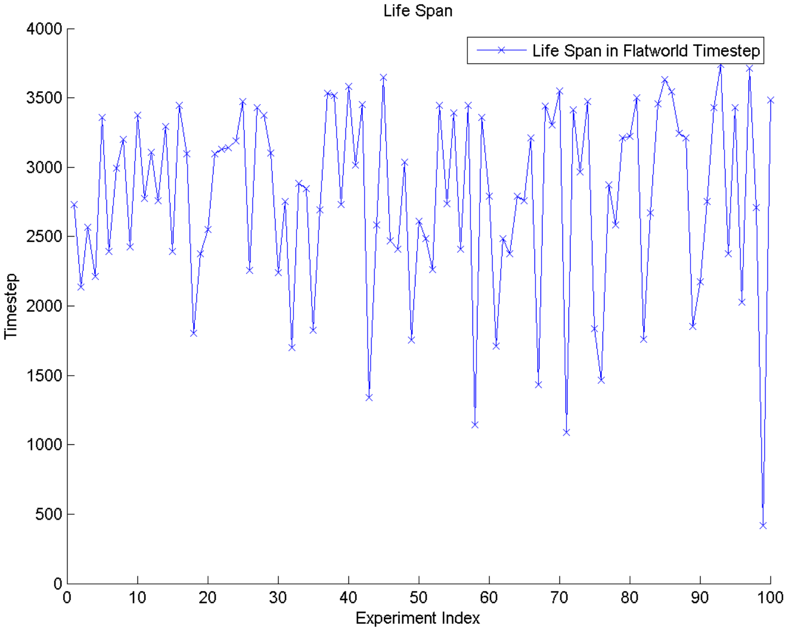
A.III.6b Model 3.1 Mileage



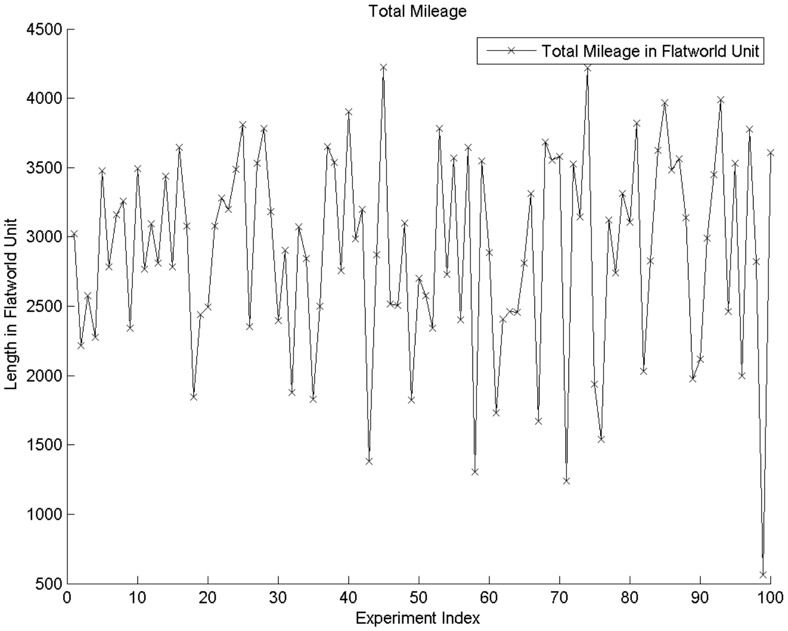
A.III.6c Model 3.1 Obtained Objects



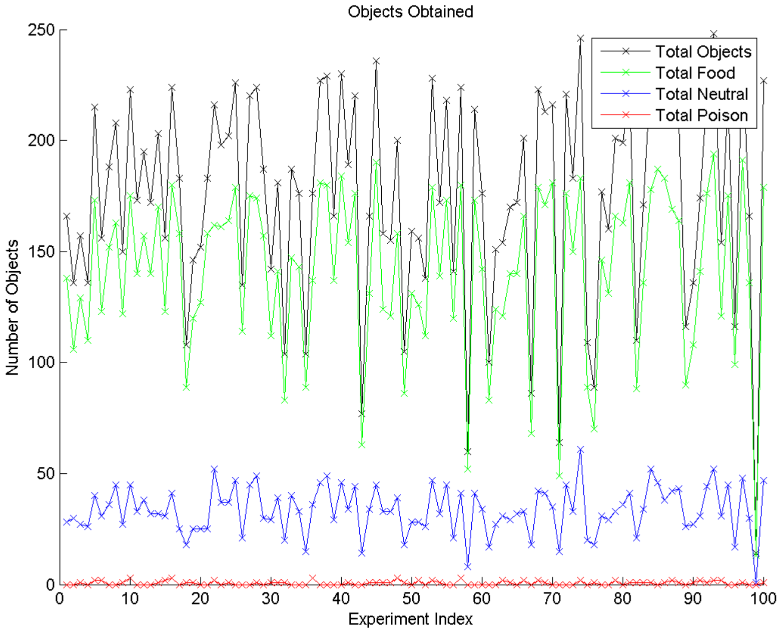
A.III.6d Model 3.1 Average Distance Before Eating Objects



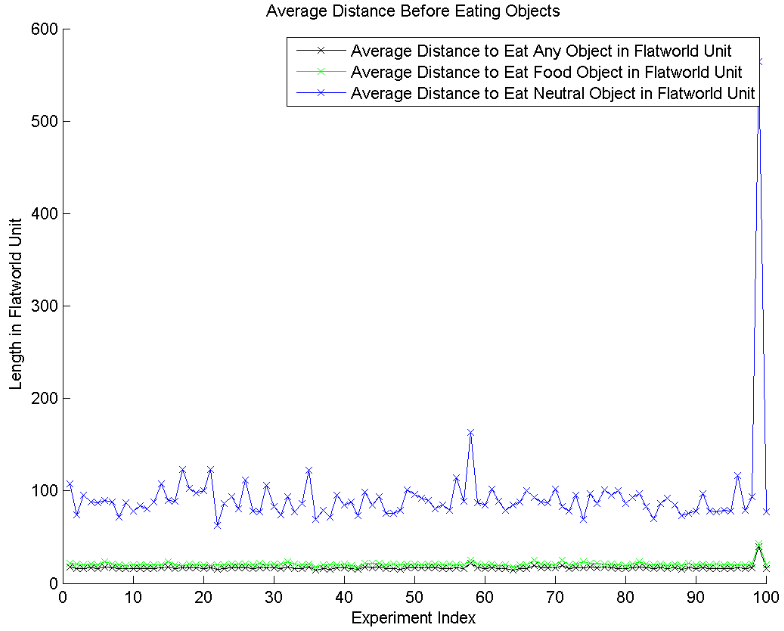
A.III.7a Model 3.2 Life Span



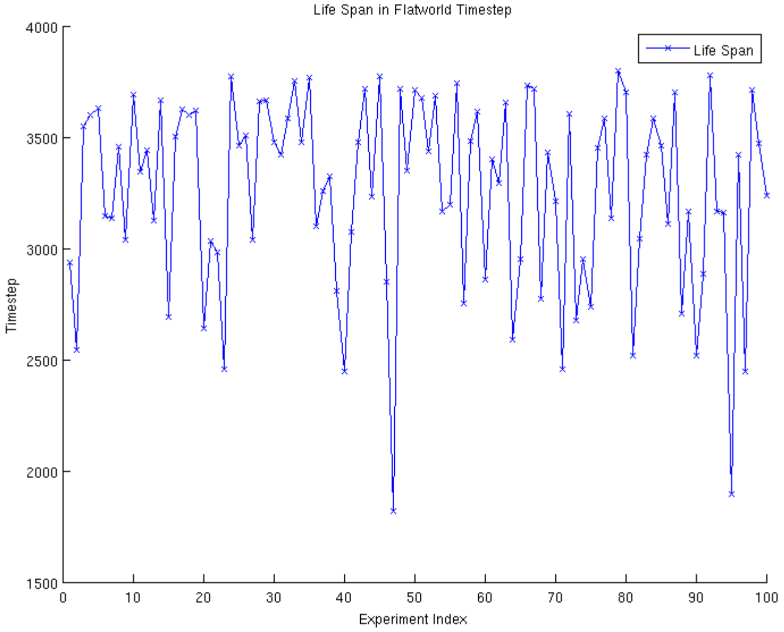
A.III.7b Model 3.2 Mileage



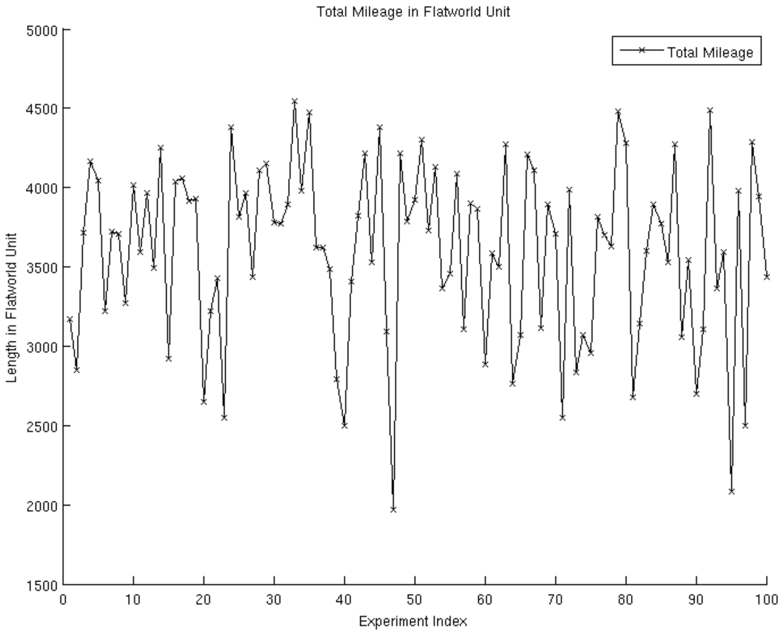
A.III.7c Model 3.2 Obtained Objects



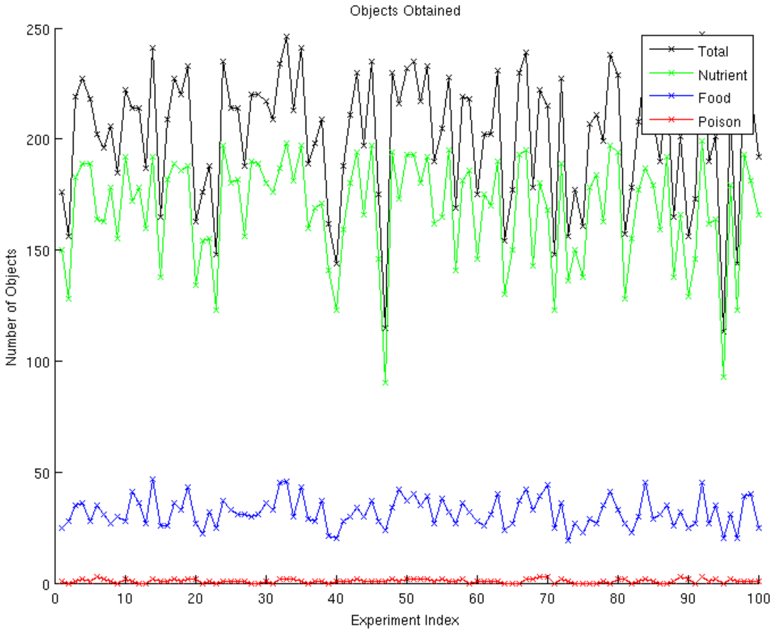
A.III.7d Model 3.2 Average Distance Before Eating Objects



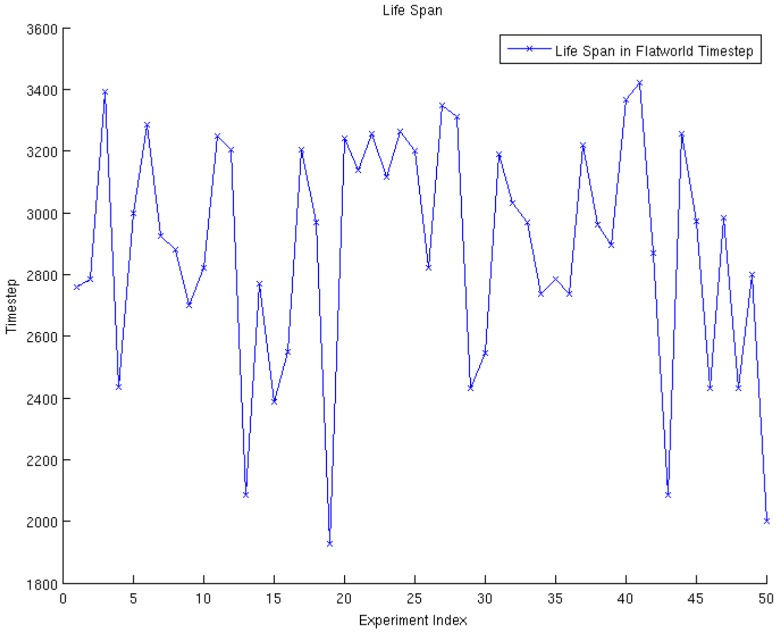
A.III.8a Model 4.0 Life Span



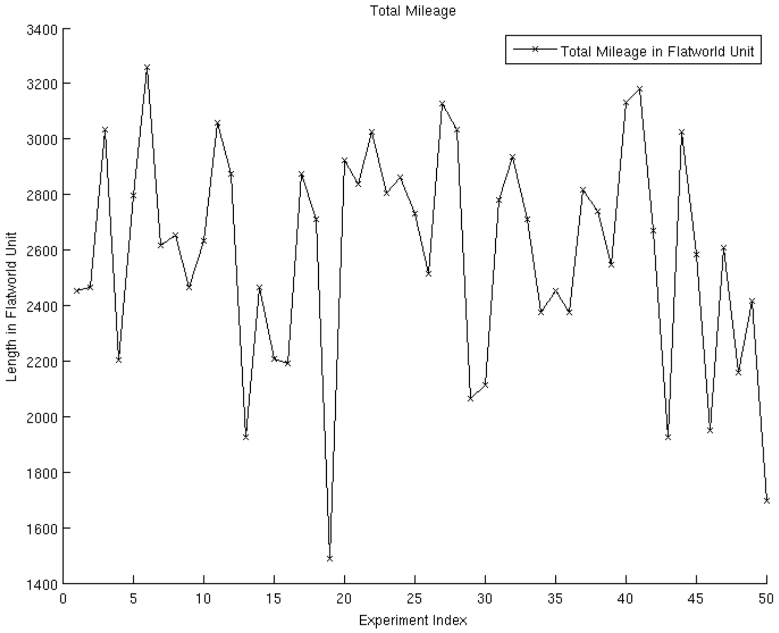
A.III.8b Model 4.0 Mileage



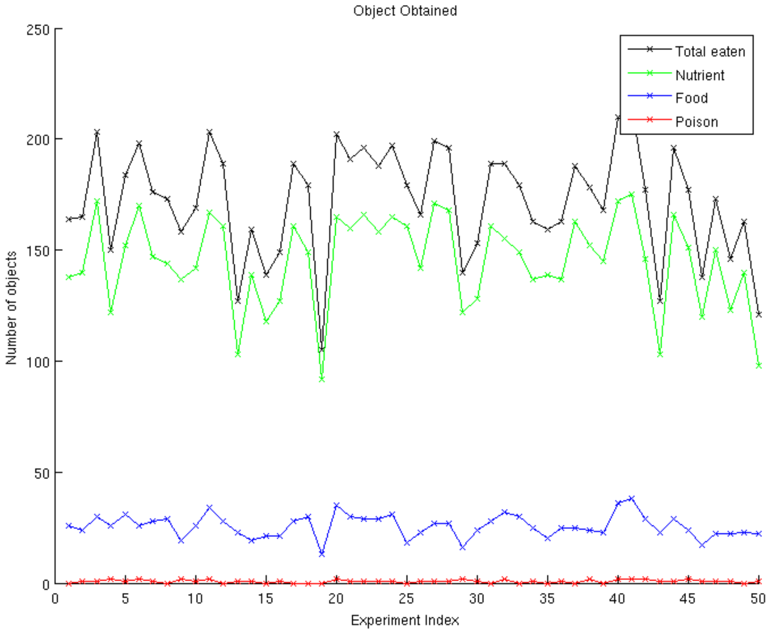
A.III.8c Model 4.0 Obtained Object



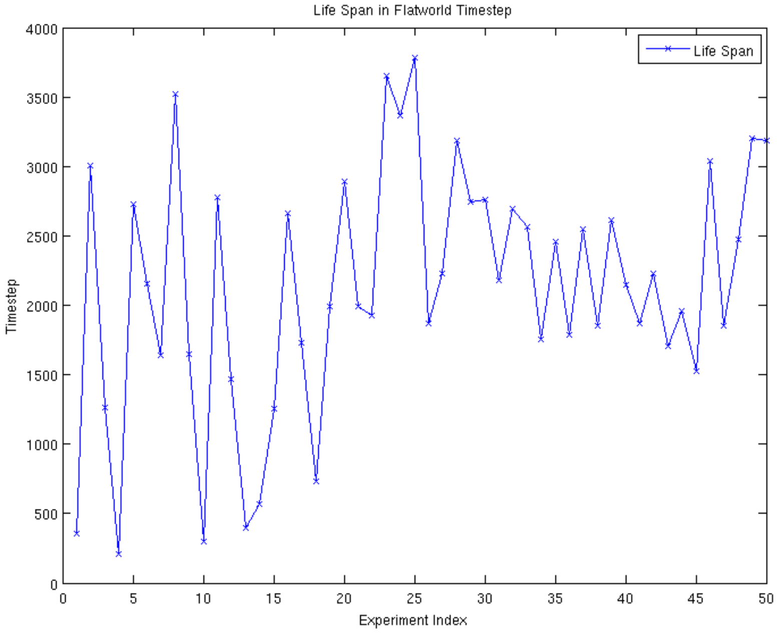
A.III.9a Model 4.1 Life Span



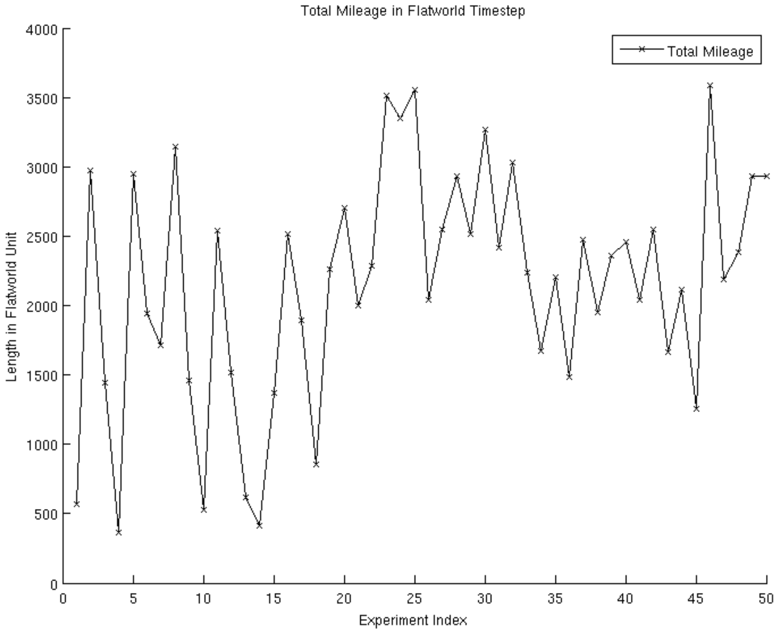
A.III.9b Model 4.1 Mileage



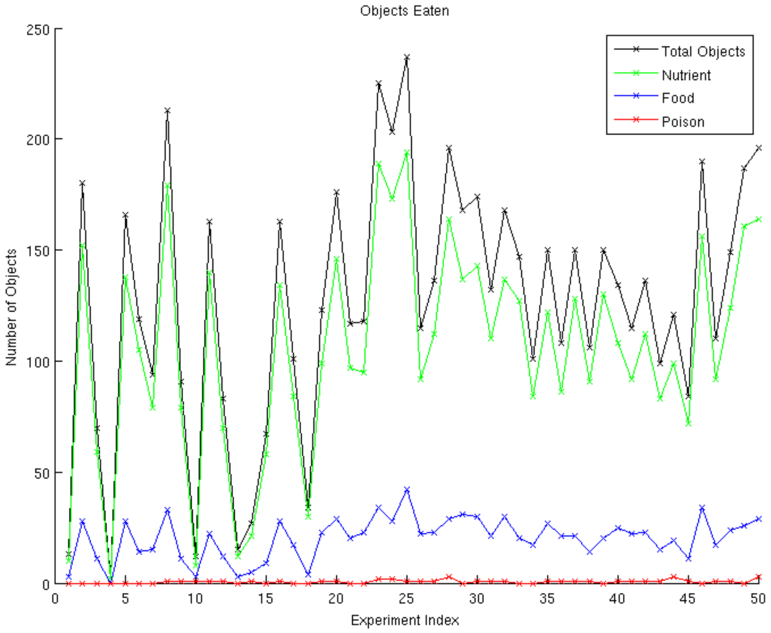
A.III.9c Model 4.1 Obtained Object



A.III.10a Model 4.2 Life Span



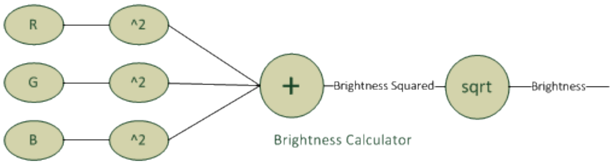
A.III.10b Model 4.2 Mileage



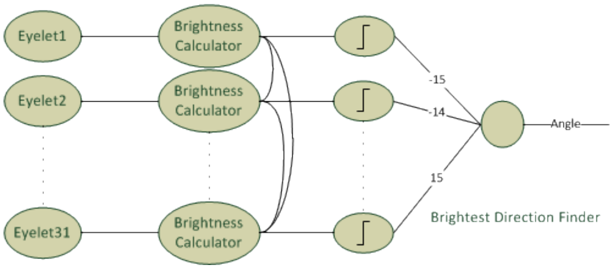
A.III.10c Model 4.2 Obtained Object

IV. Module Diagrams

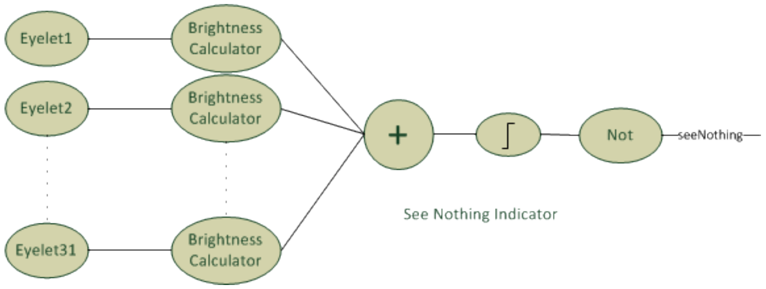
In milestone1:



A.IV.1.1 Brightness Calculator

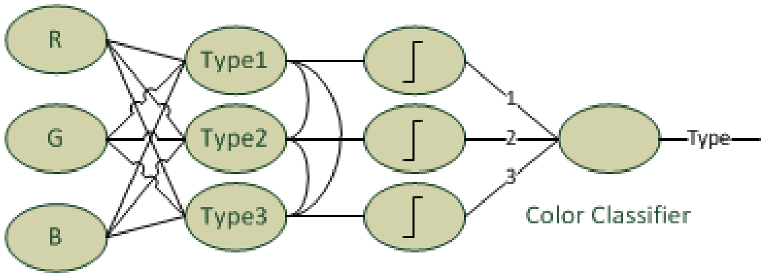


A.IV.1.2 Brightest Direction Finder

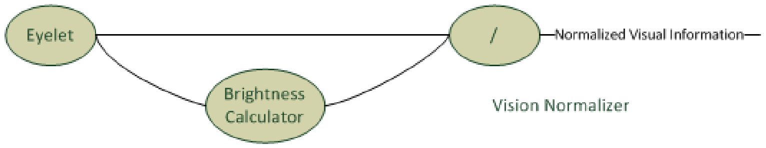


A.IV.1.3 Brightest Direction Finder

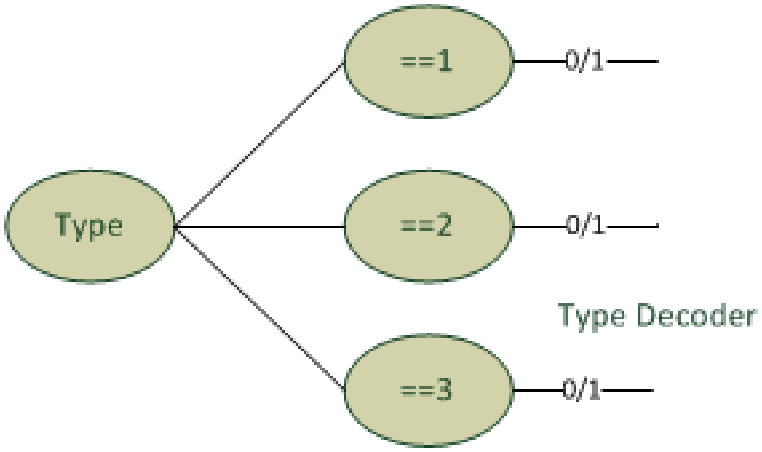
In milestone2:

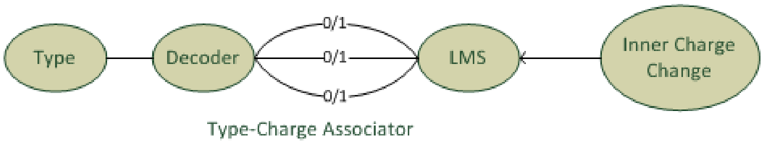


A.IV.2.1 Color Classifier

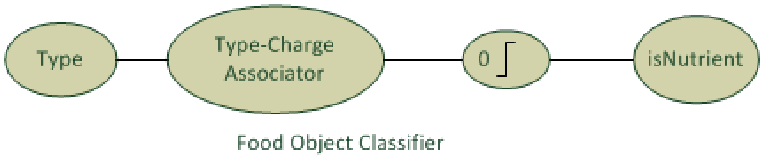


A.IV.2.2 Vision Normalizer

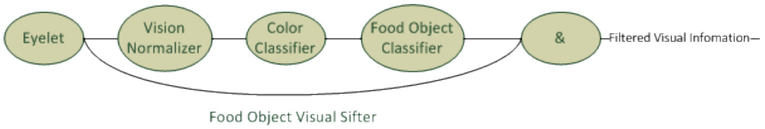




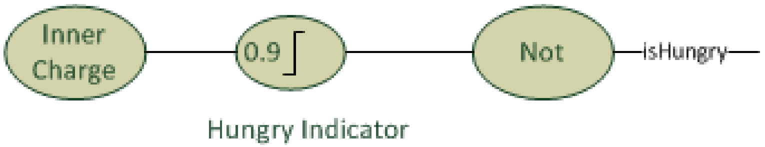
A.IV.2.3 Type Decoder



A.IV.2.4 Type Associator



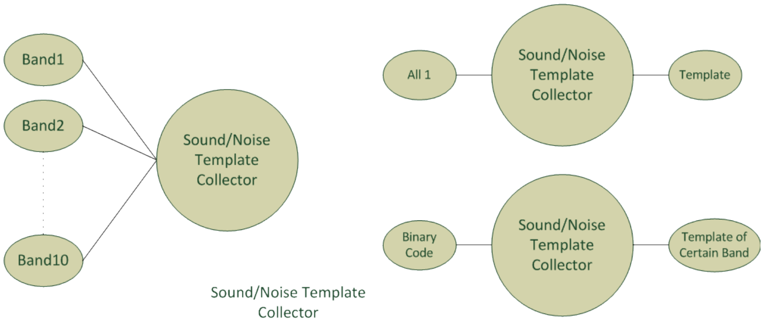
A.IV.2.5 Food Object Classifier



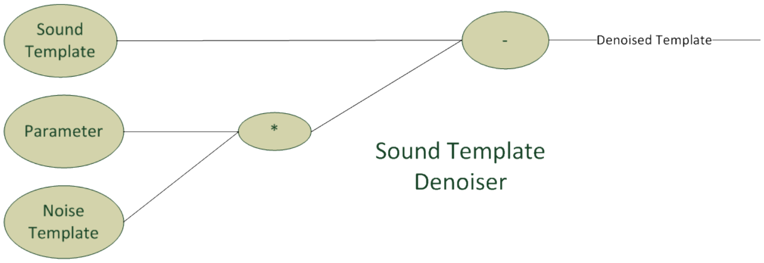
A.IV.2.6 Food Object Visual Sifter

A.IV.2.7 HungryIndicator

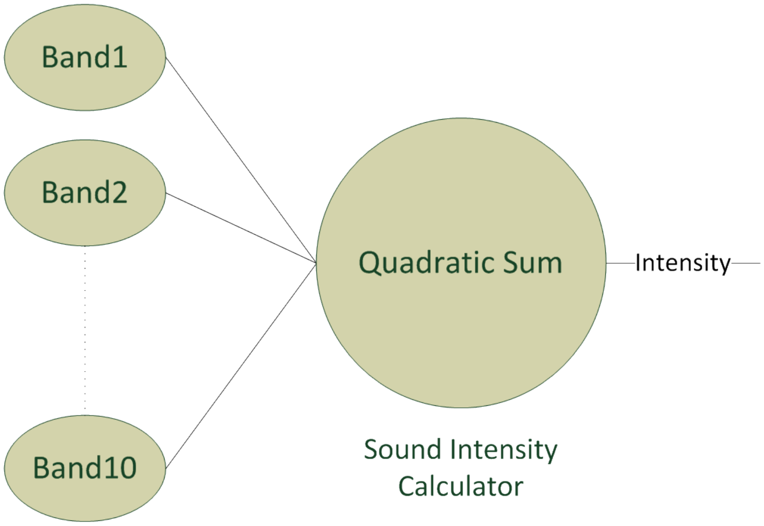
In milestone3:


Band1
Band2
Band10
Sound/Noise Template Collector
Sound/Noise Template Collector
All 1
Sound/Noise Template Collector
Template
Binary Code
Sound/Noise Template Collector
Template of Certain Band

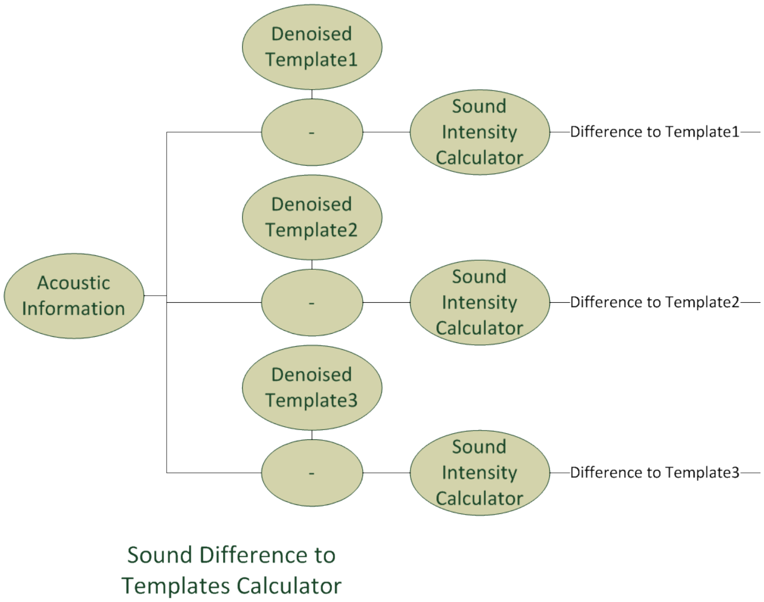
A.IV.3.1 Sound/Noise Template Collector



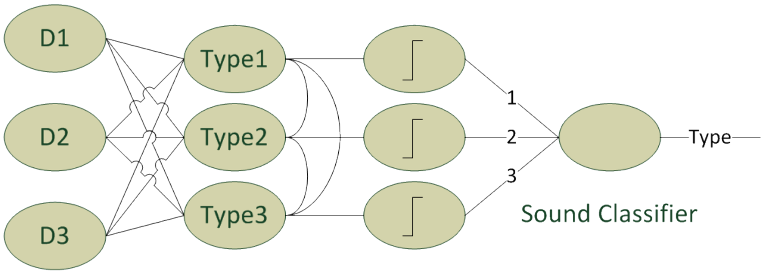
A.IV.3.2 Sound Template Denoiser



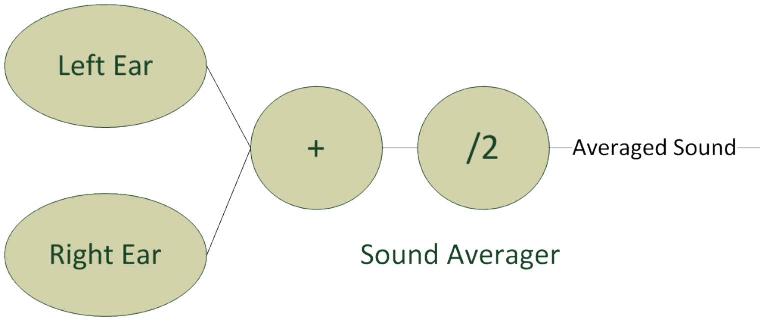
A.IV.3.3 Sound Intensity Calculator



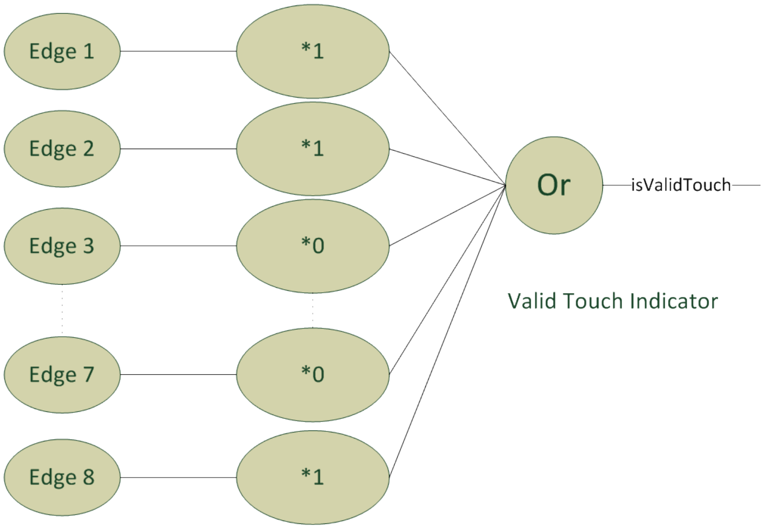
A.IV.3.4 Sound Difference To Templates Calculator



A.IV.3.5 Sound Classifier

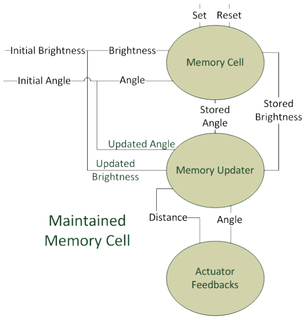


A.IV.3.6 Sound Averager

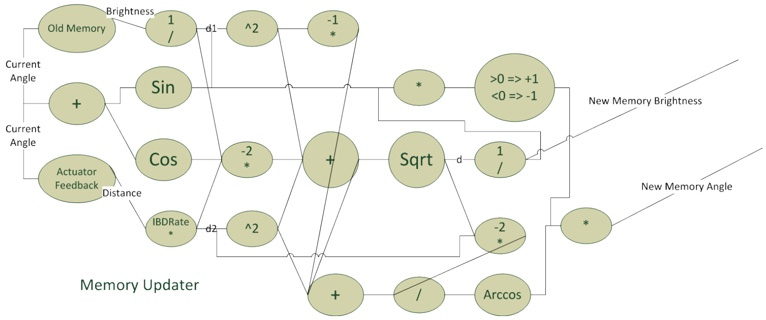


A.IV.3.7 Valid Touch Indicator

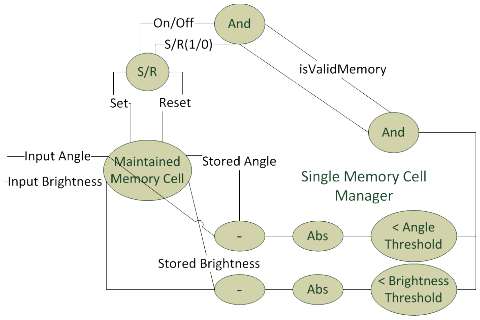
In milestone4:



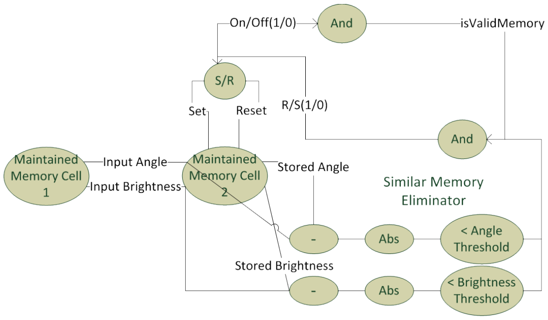
A.IV.4.1 Maintained Memory Cell



A.IV.4.2 Memory Updater



A.IV.4.3 Single Memory Cell Manager



A.IV.4.4 Similar Memory Eliminator