

SE 3XA3: User Guide

GrateBox

Team 8, Grate
Kelvin Lin (linkk4)
Eric Chaput (chaputem)
Jin Liu (liu456)

1 How to access GrateBox

GrateBox is a web application created to educate and entertain users by the use of genetic algorithms. There are two ways to access and use the GrateBox application. The first method is to navigate to the url to access the online version of GrateBox. To do so, open your internet browser and enter the following url <http://ugweb.cas.mcmaster.ca/linkk4/>. GrateBox can also be run locally on your machine. To do this first navigate to GrateBox's GitHub repository found at the following url <https://gitlab.cas.mcmaster.ca/linkk4/GrateBox/tree/master>. Then install the zip file by clicking the install button near the top right of the screen to download the zip file containing GrateBox. Extract the zip file to a location of your choosing and navigate to the src folder and double select the GrateBoxBootstrap.html file to open it.

2 How to use Gratebox

Once you access GrateBox, you will see a number of things. All of GrateBox's functionality is contained on one screen. Under the simulation title, you will see a simulation of a randomly created generation of cars. By default, this simulation will generate an initial population of three cars for the first generation, with the best two cars of each generation selected for breeding the next generation, and a mutation rate of two percent. These values can be seen under the Parameters title, and can be edited by the user. To edit a parameter, simply enter a new value under the corresponding parameter title and click enter. The simulation will then be run from the start with the new values. Also under the Parameters title can be seen a pause button. To pause the simulation at any time click the pause button. To unpaue the simulation, simply select the pause button again. Further down the page you will see text giving you information about the simulation and about GrateBox and Genetic Algorithms in general.

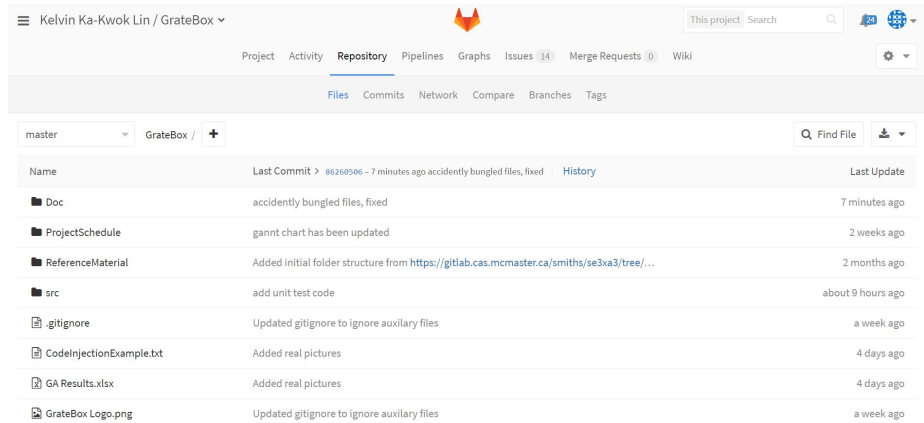


Figure 1: The view of the GrateBox repository on GitHub. You can see the download button in the top right next to the Find File button.

<input checked="" type="checkbox"/>	Doc	2016-12-02 1:13 PM	File folder	
<input type="checkbox"/>	ProjectSchedule	2016-12-02 1:13 PM	File folder	
<input type="checkbox"/>	ReferenceMaterial	2016-12-02 1:13 PM	File folder	
<input type="checkbox"/>	src	2016-12-07 9:19 AM	File folder	
<input type="checkbox"/>	.gitignore	2016-12-02 1:13 PM	Text Document	1 KB
<input type="checkbox"/>	CodeInjectionExample.txt	2016-12-05 6:48 PM	Text Document	1 KB
<input type="checkbox"/>	commit.sh	2016-12-02 1:13 PM	Shell Script	1 KB
<input type="checkbox"/>	eric.jpg	2016-12-05 6:48 PM	JPG File	13 KB
<input type="checkbox"/>	GA Results.xlsx	2016-12-05 6:48 PM	Microsoft Excel Work...	13 KB
<input type="checkbox"/>	GrateBox Logo.png	2016-12-02 1:13 PM	PNG File	28 KB
<input type="checkbox"/>	jln.jpg	2016-12-05 6:48 PM	JPG File	17 KB
<input type="checkbox"/>	kelvin.jpg	2016-12-05 6:48 PM	JPG File	7 KB
<input type="checkbox"/>	LICENSE.txt	2016-12-02 1:13 PM	Text Document	1 KB
<input type="checkbox"/>	Presentation.ppsx	2016-12-02 1:13 PM	Microsoft PowerPoint...	817 KB
<input type="checkbox"/>	Presentation.pptx	2016-12-05 6:48 PM	Microsoft PowerPoint...	1,452 KB
<input type="checkbox"/>	README.md	2016-12-02 1:13 PM	MD File	1 KB

Figure 2: The contents of the GrateBox repository. The src folder can be seen clearly as the forth from the top.

test	2016-12-07 9:19 AM	File folder	
Box2D.js	2016-12-02 1:13 PM	JS File	393 KB
CarObject.js	2016-12-07 9:19 AM	JS File	7 KB
GrateBox.js	2016-12-07 9:19 AM	JS File	17 KB
GrateBoxBootStrap.html	2016-12-02 1:13 PM	Chrome HTML Docu...	5 KB
heap.js	2016-12-02 1:13 PM	JS File	9 KB
MakeCar.js	2016-12-02 1:13 PM	JS File	13 KB
Path.js	2016-12-02 1:13 PM	JS File	6 KB
style.css	2016-12-02 1:13 PM	Cascading Style Shee...	1 KB
test.js	2016-12-02 1:13 PM	JS File	4 KB
title.png	2016-12-02 1:13 PM	PNG File	40 KB

Figure 3: The contents of the src folder. Open the GrateBoxBootStrap.html file to access GrateBox.



Figure 4: The initial view of GrateBox.

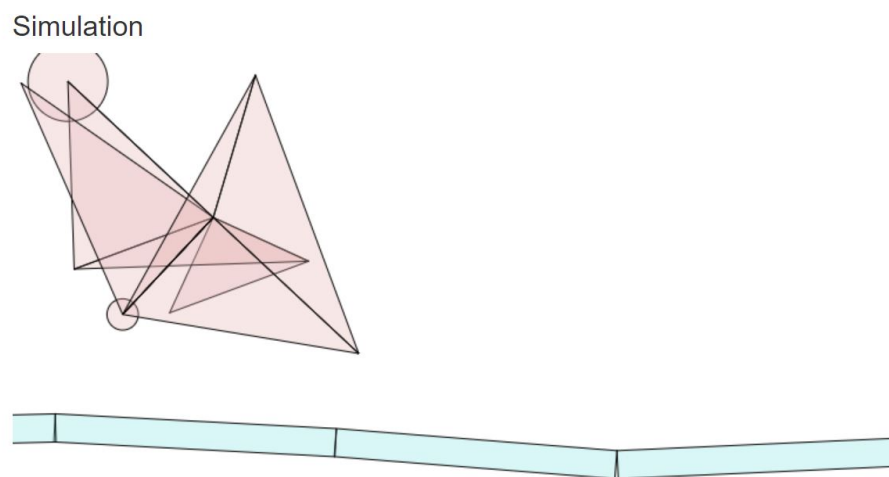


Figure 5: The simulation display.

Parameters

Pause

Population Size:

3

Enter the desired number of cars per generation here.

Number of Parents:

2

Enter the number of cars chosen from each generation to create the next here.

Mutation Rate:

0.02

Enter the degree of mutation as a percent here (i.e. the initial mutation rate of 2% is entered as 0.02).

Figure 6: The parameters display.

Now Running Generation 10, Car Number 2

Current Generation:

Population Size: 3

Parent Pool: 2

Mutation Rate: 0.02

Results

Top car of generation 1 ran for 280 cycles

Top car of generation 2 ran for 297 cycles

Top car of generation 3 ran for 296 cycles

Top car of generation 4 ran for 381 cycles

Top car of generation 5 ran for 423 cycles

Top car of generation 6 ran for 422 cycles

Top car of generation 7 ran for 422 cycles

Top car of generation 8 ran for 422 cycles

Top car of generation 9 ran for 428 cycles

Figure 7: The info at the bottom of the screen showing simulation results.

What is a Genetic Algorithm?

Genetic algorithms (GAs) search for near-optimal solutions to a wide variety of problems with incomplete or imperfect information by emulating the process of natural selection. As the algorithm is used, the solution will be more and more optimized. They are well suited to solving open ended problems with a variety of possible solutions. Applications of genetic algorithms include automated design, bioinformatics, economics, game theory, and training neural networks.

What is this program's purpose?

Presented here is an application of genetic algorithms. This program will randomly generate an initial car with a polygon body and two wheels randomly placed. Observe how the car evolves towards a more successful iteration to move farther and farther along the road.

How performance is calculated

Under the **Results** heading, you will see that the best car of each generation is displayed with a corresponding value giving its "cycles". Each car possesses a health value to determine if it is still moving. As the car moves, a thread is constantly running to check the car's speed. If the car is not making forwards advancement the health of the car decreases. When the car's health reaches zero it is determined to be dead, and the number of times the thread cycled is recorded as the number of cycles. The greater the amount of cycles, the better performing the car is.

GitHub

The full code and documentation for the Genetic Cars project can be found on a [Github repository](#).

Figure 8: The info at the bottom of the screen describing GrateBox and Genetic Algorithms.