Teaching the Mechanisms of Evolution Through the Use of Genetic Algorithms in LEGO© Mindstorms™ Robots

Linda Smith
Teresa Pegors
Indiana University

Abstract

It is often difficult for students to grasp how nature is able to evolve creatures with complex survival strategies without a pre-planned design. This curriculum presents a one-class-period workshop on using genetic algorithms to demonstrate the basic mechanisms of evolution. The workshop begins with a lecture on the role of genes in evolution across generations and how genetic algorithms exploit these same principles in goal-seeking strategies. Then, by one or more LEGO© Mindstorms™ robots, students participate in a simulated evolutionary process of evolving a strategy for a pre-specified goal. The students experience first-hand how selection, reproduction, and development can take place in nature. Students also learn how genetic algorithms are being used in many fields an as alternative method of problem solving, and they gain experience working with the popular LEGO© Mindstorms™.

Teaching the Mechanisms of Evolution Through the Use of Genetic Algorithms in LEGO© Mindstorms™ Robots

The world contains an incredible variety of living organisms, from massive, slow-moving blue whales in the Pacific, to small spiders that scuttle away at the slightest disturbance, or eagles, which soar in spectacular majesty above the earth. Most students have been taught a bit about the theory of evolution, which proposes that the variety in species of animals comes about through a process of selection, reproduction, and mutation. Because this process takes place over such a great amount of time, students have a difficult time conceptualizing the ways in which evolution can find simple solutions to issues of survival.

In 1975, John Holland first devised genetic algorithms (GA's) as a way in which the biological processes of evolution could be modeled computationally. In the last quarter century, the use of genetic algorithms has become a popular way of using many of the simple mechanisms of evolution to practically evolve solutions to problems.

This workshop allows the students to participate in the use of a genetic algorithm strategy to evolve the structure of LEGO robots for achieving a specific goal. The students are first introduced to the basics of how genes are involved in biological evolution and how genetic algorithms use these processes computationally. Groups are then formed in which members are responsible for the tasks associated with evolving their group's robot. The robots' performance in the environment is measured according to a pre-specified goal, and selection and reproduction take place manually between the groups.

Not only do students become more aware of the capabilities of evolution, but they get a different perspective on how problems can be solved, and gain experience working with LEGO© Mindstorms™.

Methods

Resources

The optimal class size for this workshop is 20-25 students. This allows for small group sizes, depending on the number of robots being used. It is expected that only one LEGO© Mindstorms™ kit will be available in the classroom. (See note at end of Methods section if more kits are available). These kits are normally priced at around $199 (www.legomindstorms.com), and each includes an onboard computer, light and touch sensors, motors, and a large variety of building pieces. Programming and building instructions are included as well. While only one robot at a time will be in physical form, there should be 5 or 6 groups of students, each responsible for keeping their own robot's traits on paper.

A flat area, such as a table top, should be allocated for the robot to move about in. Cardboard or some sort of boundary or barrier walls should be set up around the perimeter of the space. A stopwatch, a tape measure, and dice are also needed.

Setup

The robot will be given two programs on its brick which can be switched as gene choice dictates throughout the demonstration. The LEGO© Mindstorms™ kit comes with the necessary software and hardware for creating and transferring programs. These programs are similar and can be created with the included software (See Appendix C). Because the basic structure of each robot will be the same, making it easy to switch each robot to its physical form. The variability between robots will be in the wheel type, sensors used, and the order of connection wires. (See Appendix A for more detail on robot structure, building instructions, and trait differences.) Use dice to determine the beginning traits of each robot, and record these on the table found in Appendix B.

Procedure

Ask students at the beginning of the lecture how they might design a robot which is able to continually move in its environment without becoming stuck and needing outside help. Have students

discuss various strategies and write them down on the board for future reference. Characteristics of these robots should include some type of sensors, a program which interprets these signals, and motor devices.

Students, depending on their background knowledge, should then be introduced to or given a review of such concepts (in a very simplified form) as chromosomes, genes, traits, genotypes, phenotypes, recombination, mutation, and fitness. Compare the differences between the ways an engineer might design a solution and how nature might find a solution.

Explain the idea of a fitness function and state space and how an optimization strategy attempts to climb the largest “hill” within this space. Discuss how random mutation allows strategies to not become stuck on sub-optimal solutions.

Give a basic idea of what will be done in the workshop and that the genetic algorithm will be using the fitness function of the total amount of time the robot is active before it is incapacitated.

Form groups and assign each group a number or name for their robot and give them one of the tables on which a pre-determined set of traits has been recorded. Each group will be responsible for measuring their robot’s fitness, keeping a chart of its genotype, and helping in the selection and reproduction process.

Even though in theory, all robots in a generation should be run at once, each robot will have to be run separately. Begin a run by placing the robot in the environment and letting it run for one minute. Record should be kept of the amount of time each robot spends in movement. When all robots have been run, rank the robots from longest to shortest times (best to worst.)

“Mating” among individuals should take place in a process more specifically described in Appendix B. After a new generation is formed, these robots will be measured on their performance also. Continue cycling through the process of selection and reproduction six or seven times.

If time allows, other fitness functions could be proposed and tested, such as total distance moved from starting place or total amount of time touching another robot (this can only be done with multiple robots).

Note: If more than one robot is available, they can be run at the same time. Simply place them at intervals in a larger environment and have each group keep track of the time for their own robot. The optimal demonstration would have 5 or 6 kits available so that an entire generation could run at once. In this situation, other fitness functions could be devised which rely on interactions between robots.

Discussion

In the beginning trials, most robots will probably have very low fitnesses. Over generations, students should be able to observe that the average length of time the robots move increases. Ask the students which, if any, traits were removed from the gene pool and why this might be. Compare the final strategies that the robots exhibited to those strategies which the students proposed at the beginning of the workshop to accomplish the same goal. Mention that the program in each robot was the exact same, but how the robot's structure had a significant effect on how the program played out in the robot's behavior.

Also point out that while the robots in this demonstration moved towards one solution, the vast, almost limitless variability in nature allows for millions of strategies to arise, hence the thousands of species found around the world. Subtle environmental differences may cause great changes in the types organism needed to survive in those circumstances.

It should be emphasized that the fitness function in nature is the ability to pass on genes, or reproduce, and that it is implicitly enforced simply by the fact that the genes of an organism which cannot reproduce do not get carried on. In genetic algorithms, the fitness function is usually explicitly stated.

By interacting and being a part of a process using evolutionary techniques, students will develop a greater understanding of the mechanisms in natural search strategies. They are also given an introduction to genetic algorithms and the uses of LEGO© Mindstorms™. Students will discover

that having a better understanding of evolutionary techniques allows them to come up with more creative solutions to all sorts of problems they come across in the future.

References

Holland, John. (1975). Adaptation in Natural and Artificial Systems. Ann Arbor, MI: University of Michigan Press.

Further Readings/Resources

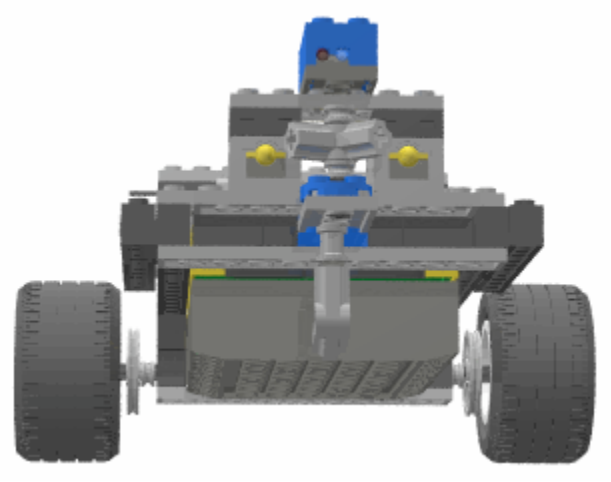
http://geneticalgorithms.ai-depot.com/ - a good collection of books, tutorials, papers, etc. on genetic algorithms.

http://www.mae.cornell.edu/bongard/MorphEngine/ - a very neat program which allows the user to build a structure, define a fitness function, and watch it evolve the controllers necessary.

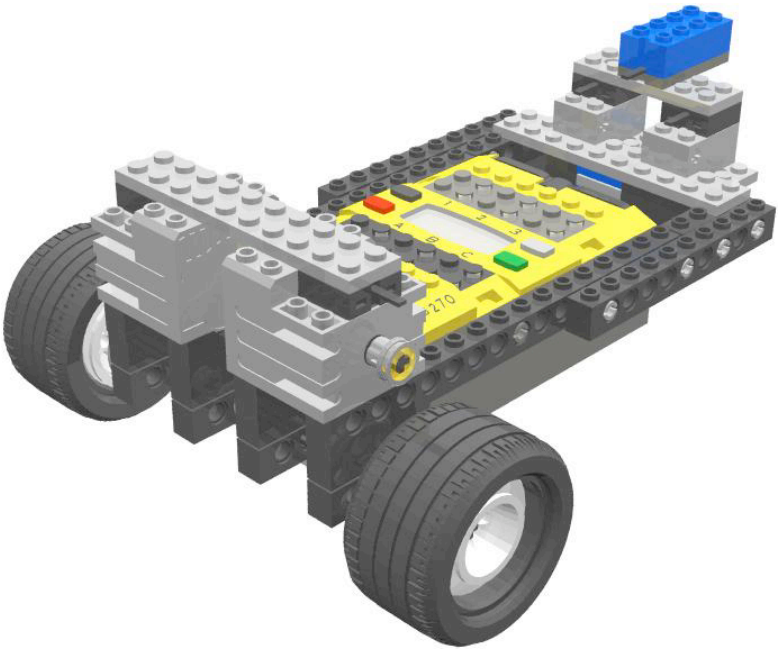
http://www.biota.org/ksims/ - contains some great demonstrations of Karl Sims' work with evolving simulated creatures through genetic algorithms.

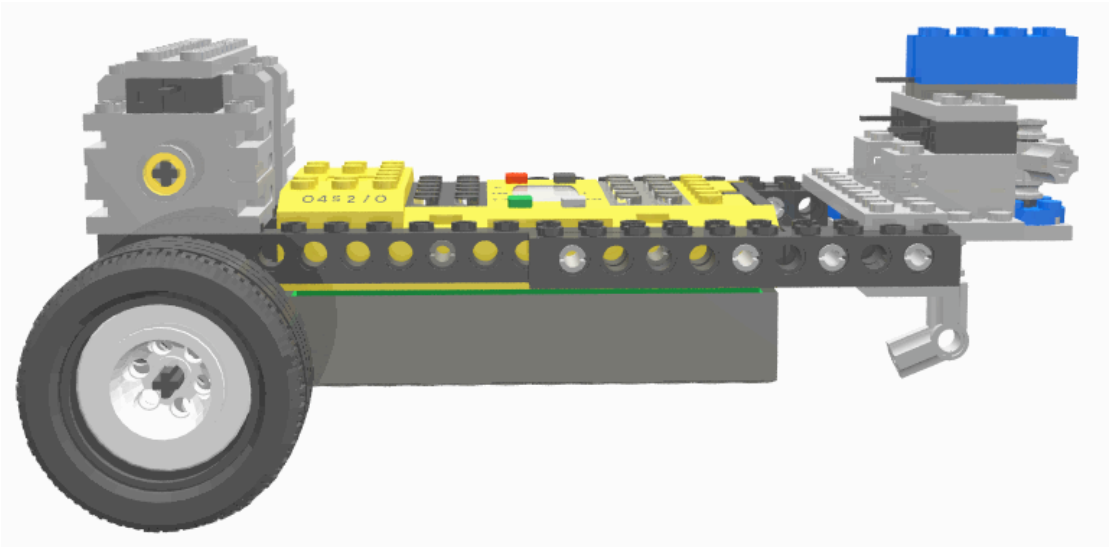
http://www.spiderland.org/breve/ - A simulated environment built from the ODE physics simulator, which has genetic algorithm capabilities built into the code.

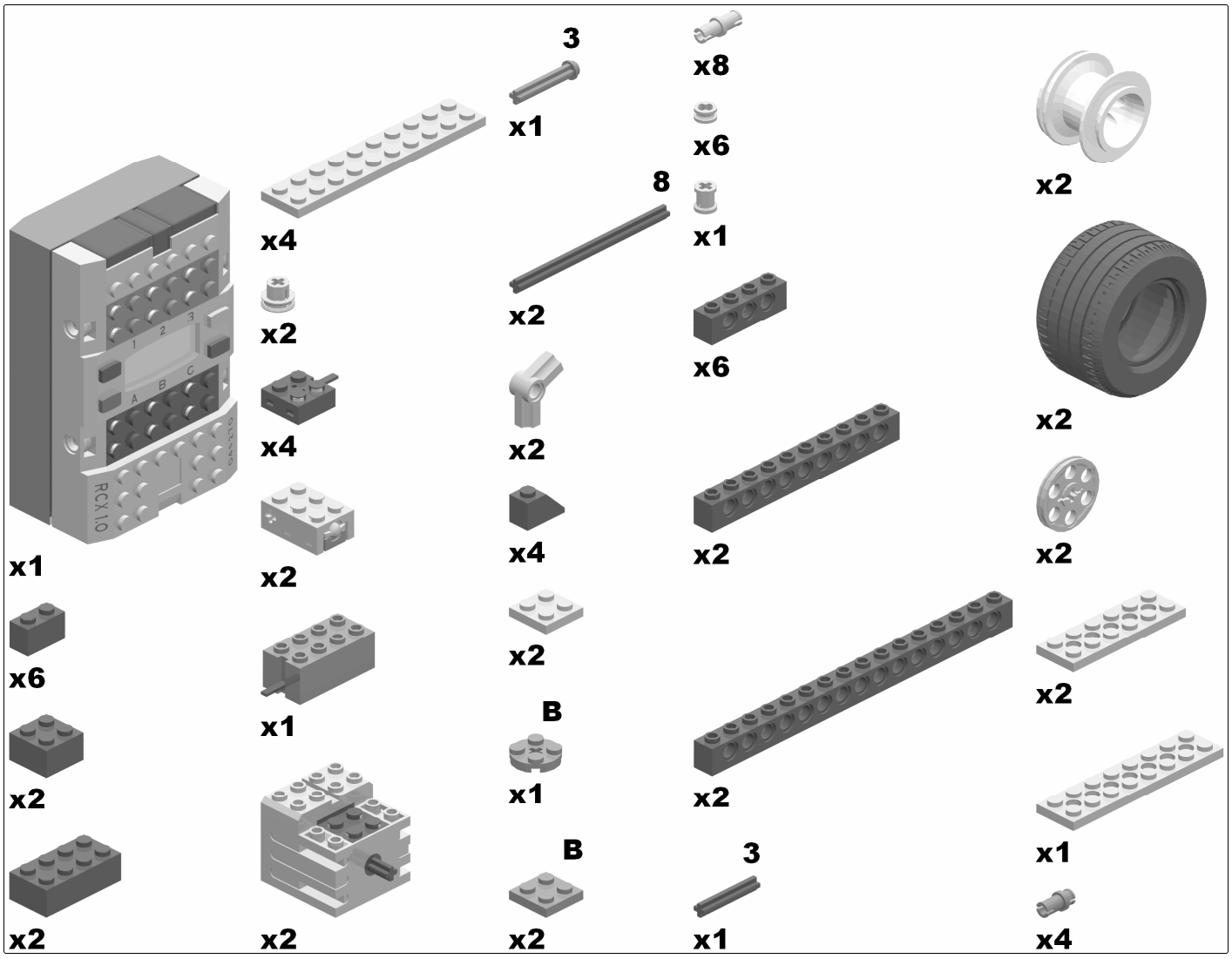
Appendix A - Robot Design/Building Instructions

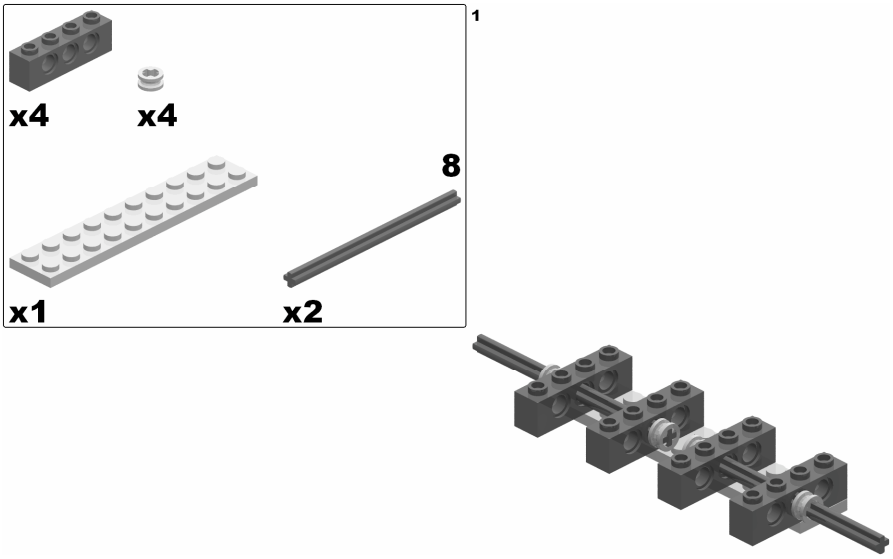


These pictures show the basic design of a robot which can be used for the workshop. All traits mentioned in Appendix C can be applied to this model. (Instructions for building the back of the robot can be found in the building instructions book which comes with a basic kit). The most significant change takes place when switching wheels to a "**ski**." A central support in the front, such as a wheel connected to the joint shown, should be changed to two supports, one on each side. An example would be two wheels similar to the back wheels, or yellow "skis" instead of wheels. The "**tentacles**" trait refers to having either long, yellow feelers a pair of black axles in the front. The "**touch sensor**" trait refers to having one sensor moved slightly more forward than the other as opposed to being symetric. The "**light sensor**" trait places the sensor in either the front or back of the robot.

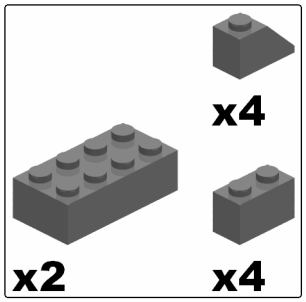


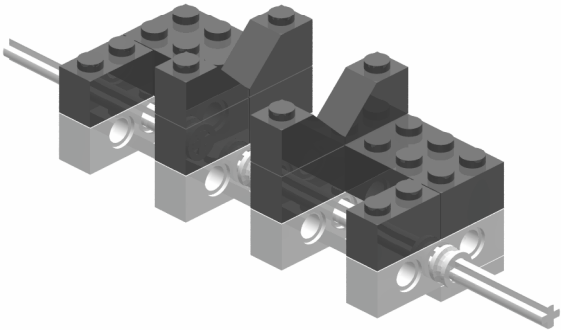



x1
x6
x2
x2
x4
x2
x4
x2
x1
x2
3
x1
8
x2
x2
x4
x2
B
x1
B
x2
x8
x6
x1
x6
x2
3
x1
x2
x2
x2
x2
x2
x2
x1
x4

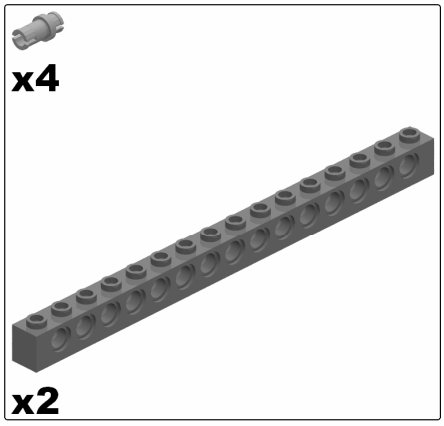

1
x4
x4
x1
8
x2

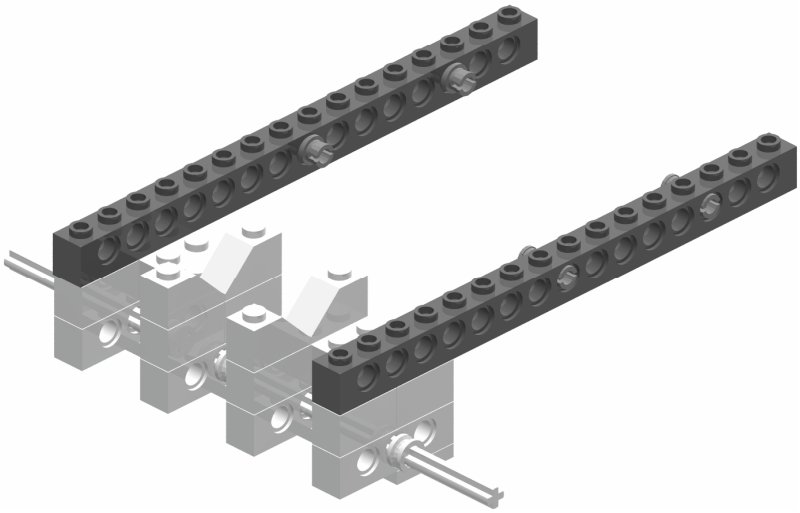
2



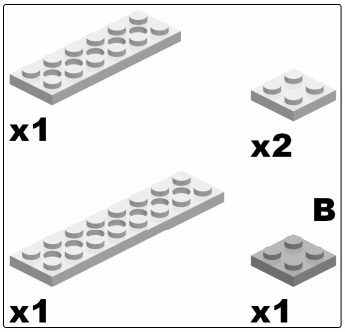


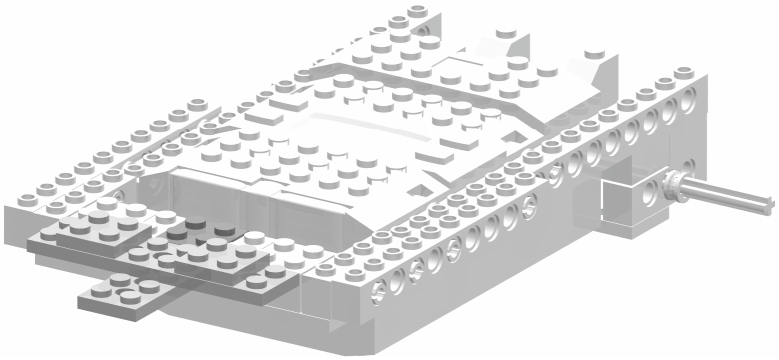
3



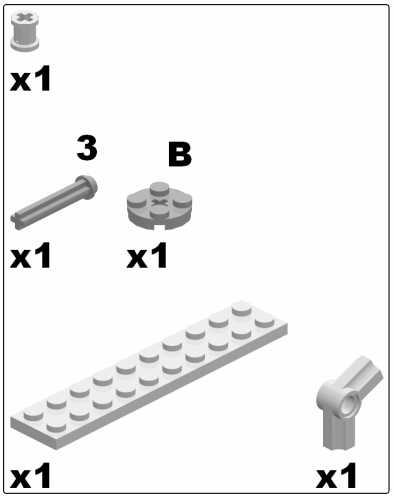


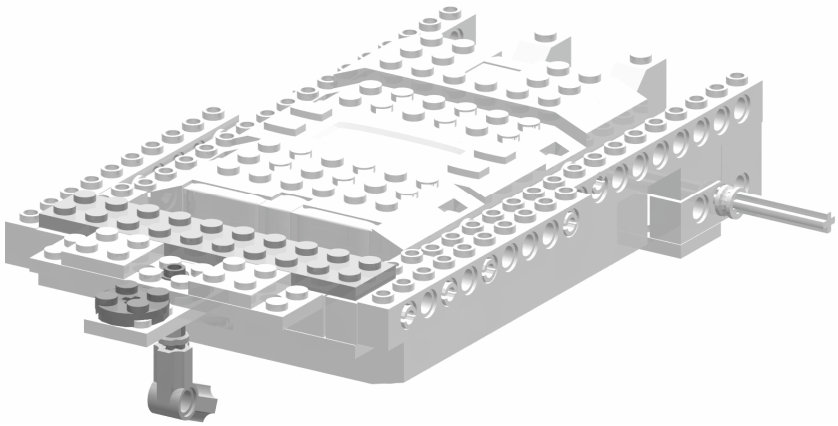
8



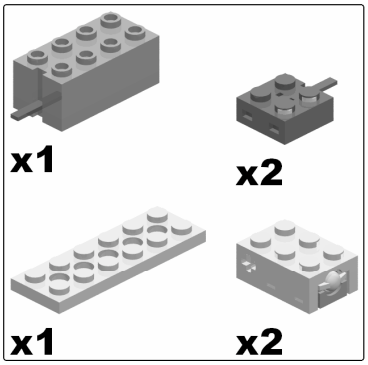


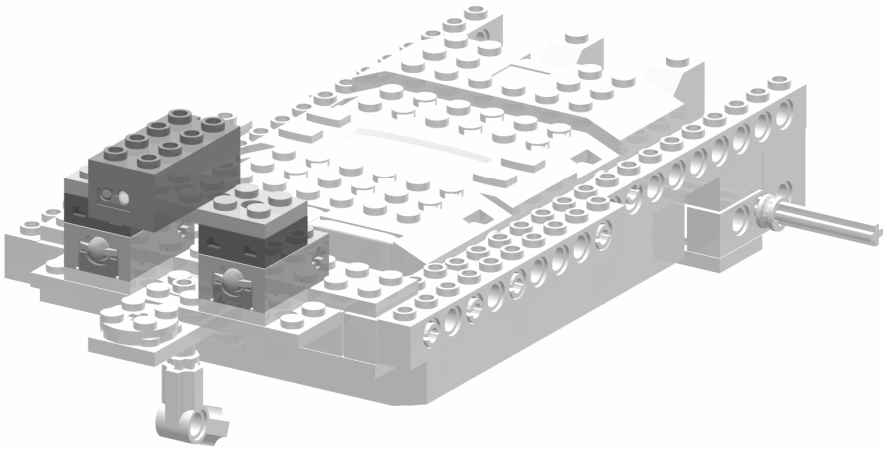
9



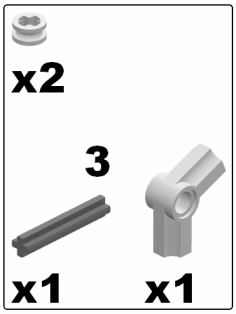


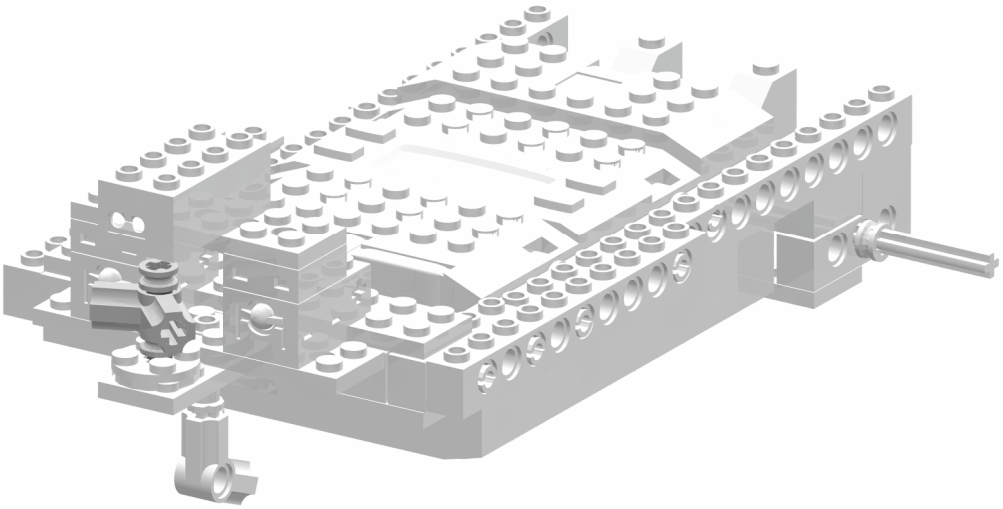
10

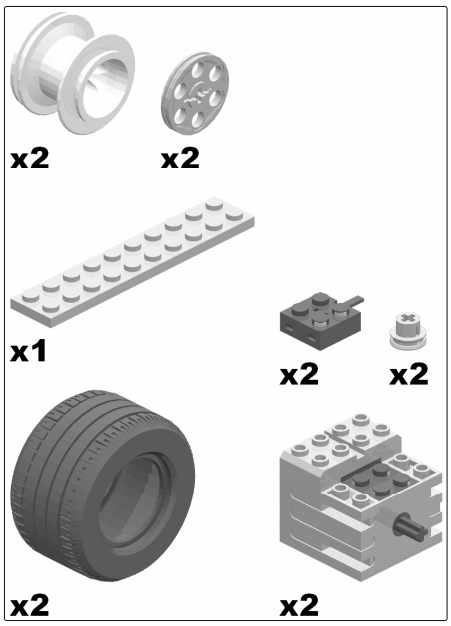


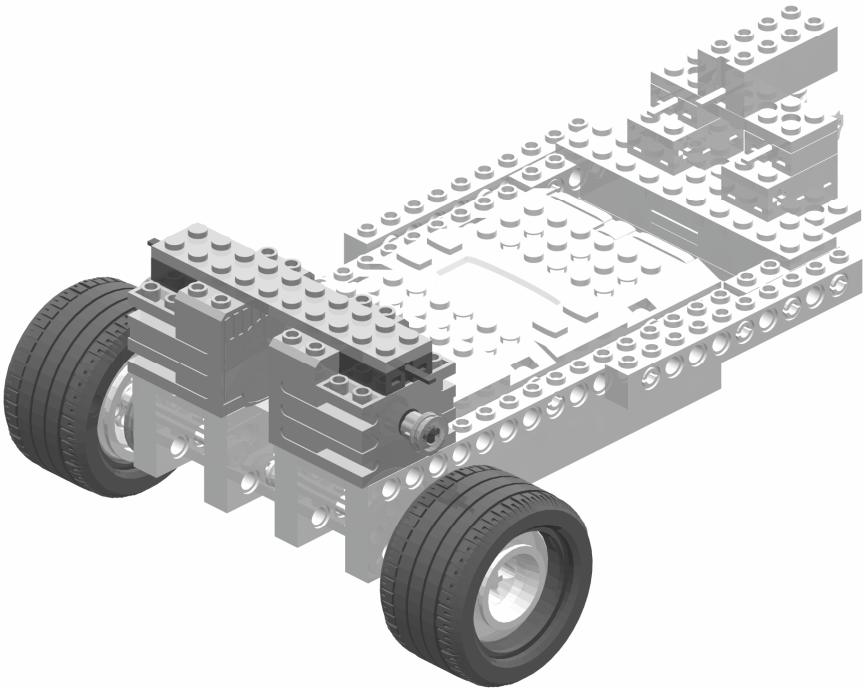


11









From this step, add or remove traits to complete each separate robot.

Appendix B – Mating Instructions

Give each team a chart where they can keep track of their robot's traits for each successive generation. When students are first given their robot, have them mark down their robot's original traits in the "Generation 1" column. After a generation is run, students should record the performance of the robot in distance/time/etc., and based on those values then record its rank compared with the other robots.

Robot _____							
	Generation 1	Generation 2	Generation 3	Generation 4	Generation 5	Generation 6	Generation 7
Program							
Tentacles							
Front wheel/ski							
Touch Sensor							
A wire							
C wire							
Light Sensor							
Performance							
Rank							

Once a generation of robots has been ranked according to the specific fitness function, use a pre-specified set of mating rules to determine which robots reproduce with whom. In most cases, each couple will have one "kid," and the "parent" which takes on the new properties is determined by rolling a die. Below are some examples of mating rules. (Numbers correspond with rank).

Mating Rules - Option 1	Mating Rules - Option 2
1 with 2	1 with 2 (two kids)
1 with 3	3 with 4
1 with 5	2 with 5
2 with 6	1 with 6
3 with 4	roll die for last couple
roll die for last couple	

After couples are decided, give each pair a chart with a set of randomly chosen traits to take from either parent. There should be a large number of these combinations available. Once a team gets a paper with a chart such as the following on it, they should role to decide for kid 1 or kid 2. In very few combinations, a mutation will appear, which is a trait different that those normally available.

Kid 1		Kid 2
1	Program	1
1	Tentacle	2
2	Front	2
1	Touch Sensor	1
2	A wire	1
1	C wire	1
1	Light Sensor	2

*mutate C wire to side

Once all reproductions have taken place, the next generation is

Appendix C – RCX Program

Commands

Default
If no sensors are activated, the robot moves forward and turns at random intervals. The left wheel turns off at random intervals between 1 and 4 seconds and continues to remain off for a random interval of between 1 and 6 seconds. (The only difference in the second program is switching the interval lengths between these two variables)

Sensors
- When a touch sensor is activated, the opposite wheel turns off.

When the light sensor reaches a certain threshold, the robot will move straight forward.

Sounds
- When the left wheel turns off, both in the random case and in the case of the right touch sensor being activated, the robot emits a pair of low tones.

When the light sensor reaches a certain threshold, the robot emits a warbling sound.

Produced Observable Behavior

Avoid Obstacles
- The touch sensors make robot turn away from obstacles.

Exception: If something is straight ahead, neither “whisker” will activate a touch sensor.

Explore
The robot turns at random intervals when nothing else is happening.

Attraction to Light
When the light is strong, the robot moves straight.