

GRAFFITI + Robots as Artists

Two robots explore their environment leaving colored marks or “tags” on the ground. When they leave marks in their environment, they announce to the world that they were there. A third robot navigates the area cleaning marks away, thus leaving evidence that it was there. This ongoing drawing and erasing exchange could continue indefinitely.

Overview

Background

Graffiti (strictly, as singular, "*graffito*," from the Italian – "graffiti" being the plural) are graphics applied without authorization to publicly viewable surfaces. They have been defined as being "a drawing or writing scratched on a wall or other surface."

There are two robots that explore their environment leaving colored marks, or graffiti, on the ground. These marks are a declaration of existence. These two robots also react to one another. They will not cross the other's drawn marks. This can lead to problems such as drawing themselves into a corner. A third robot travels through the environment erasing the marks left by the others. This also shows that the robot exists.

Project

The robots will work together to take input from their environment (human spectators) and manipulate their workspace. The drawing robots will also have a marker to leave a path. The erasing robot will take input from its environment as well, but it will erase lines instead of creating them. The drawing robots will work together to create paths. These paths are intended to be continuous without overlapping each other. The drawers will leave a path behind them, turning and accelerating depending on different environmental variables. When one of the drawers encounters a line it should reverse and turn in a different direction before continuing on. In the event that a drawer boxes itself in a line, it will become trapped and immobile. The robots will also have specific signals that will allow communications between each other. Utilizing Bluetooth technology to determine actual location, the eraser could come erase lines around a drawer to 'free' it from its boundaries.



Overview

Motivation

Robots could come together and express their reactions to their environment, simulating a 'creative' experiment. The result would be an unpredictable set of paths that visualize the history of exploration. Further more humans can act upon the environment by drawing or erasing, which engages viewers and gives them control. More often than not, robots are seen as unintelligent, uncreative automatons -- having a robot that reacts to humans while creating an artifact breaks down this stereotype. The drawing and erasing robots interact with humans and the environment in order to experience a reality where robots are more than workers behind the scene.

System Architecture

The system consists of humans and robots ("actors") and the arena ("stage"). The main actors are the robots--two drawers and one eraser. These robots work together to react to their environment as well as manipulate it.

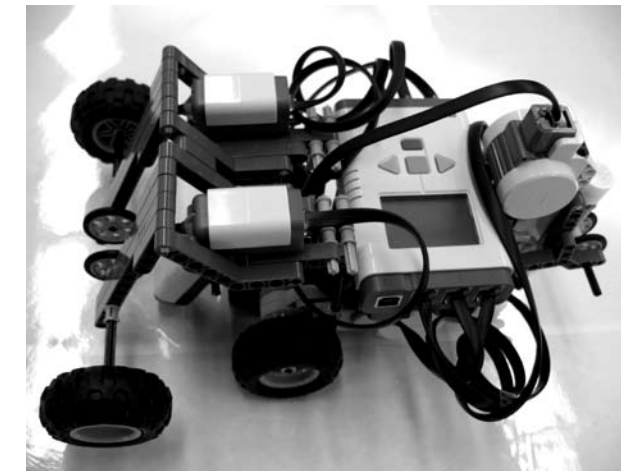
The closed system relies upon the robot's ability to discern colors:

- +white (empty area to be drawn on)
- +black (stage border)
- +red (border between the red sector and the white sector)
- +yellow (border between the white sector and the yellow sector)
- +blue (Malachi's pen)
- +green (Bob's pen)

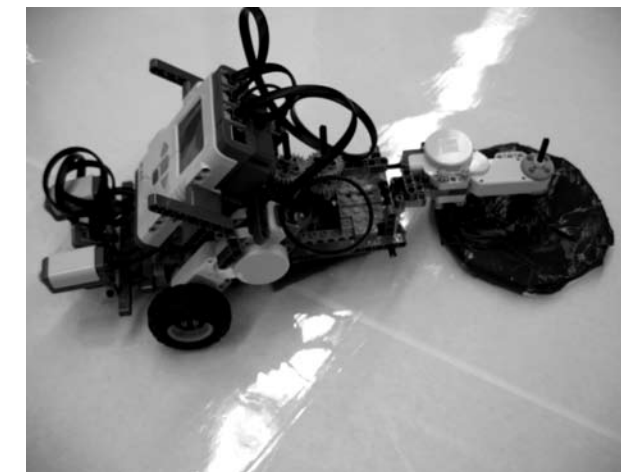
The stage is a large tabletop covered in white butcher paper. The paper is covered with clear vinyl that functions as an erasable writing surface. The stage is a large square made out of black duct tape. The square is subdivided into three sectors by red and yellow duct tape. The three actors are robots with programming to draw or erase. Observers to the demonstration are also free to pick up erasers or pens and become participants. The robot configurations and communication paths are further detailed in the following hardware sections.

* See following page for photo of arena ("stage")

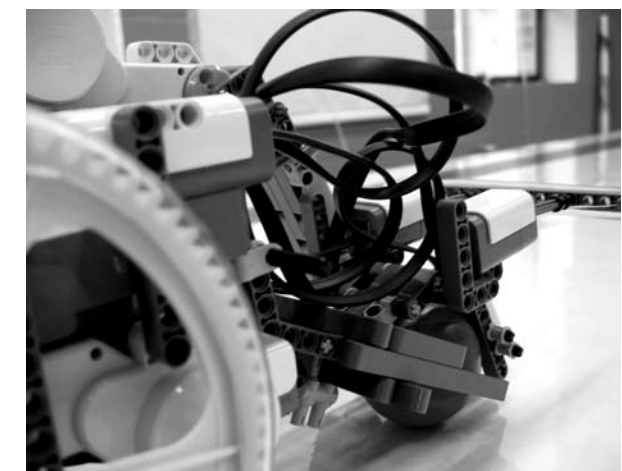
Introducing the Actors



Malachi
Drawing Robot (blue pen)

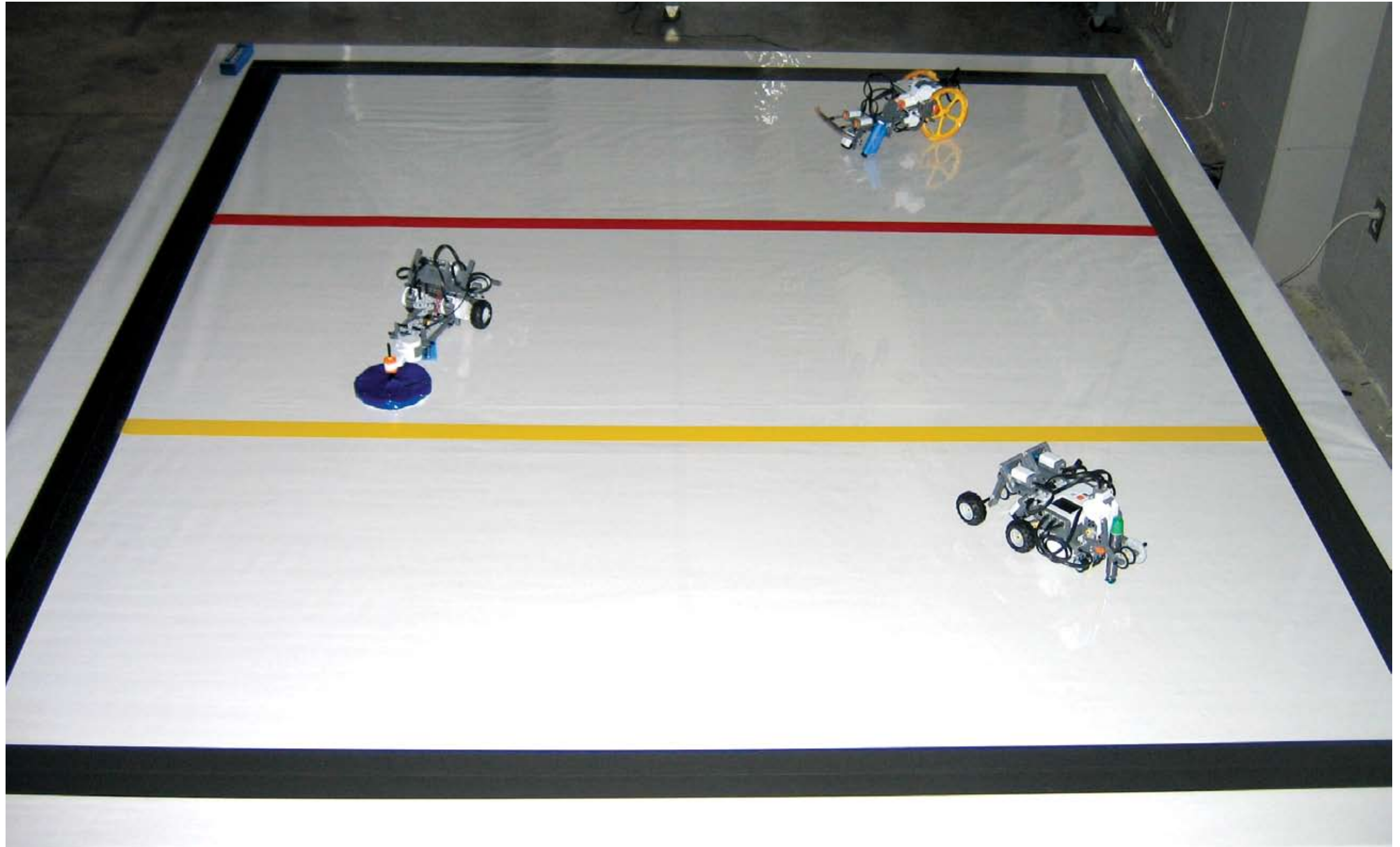


The Big Zamboni
Eraser Robot

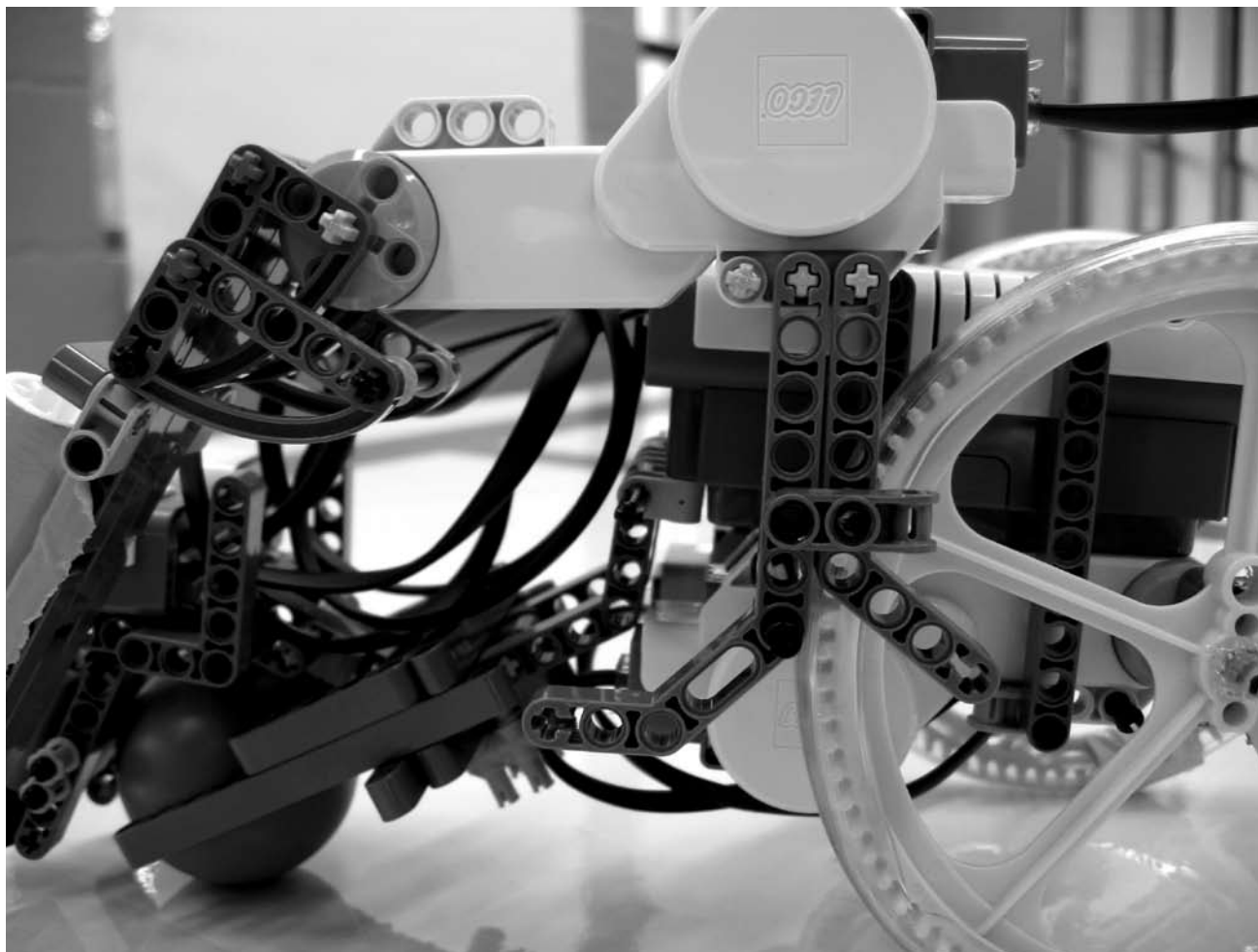


Tractor Bob
Drawing Robot (green pen)

Project:
System Architecture/Arena Layout



Project: Tractor Bob



Hardware

NXT Outputs

- A: Left Motor
- B: Drawing Hand (Motorized)
- C: Right Motor

NXT Inputs

- 1: Touch Sensor
- 2: Color Sensor
- 3: Touch Sensor
- 4: Empty

Color Sensor: Mounted on the front near surface level to detect lines drawn by the other robot, as well as borders and sectors. This input is sent to the NXT to make decisions on direction.

Touch Sensor: Mounted on the front as a bumper to detect collisions.

Drawing Hand: Mounted on the left side of the NXT brick. The hand controls whether or not the pen makes contact with the surface. Locomotion: The wheels work together with the ball to move the robot around the arena. The motors allow the robot to move forward, backward, left, and right. The ball acts as a freewheel to facilitate these motions while distributing the weight on the front of the robot.

Drawing: While the robot is moving forward, the robot presses its pen down on the surface, creating a line. The Drawing Hand comprises a marker, a housing for the marker, and a motor. Based on the software, the robot knows when to lift or raise its pen in order to show a history of where its been and its interaction with what it sees in the arena.

Detection: The robot uses its color sensor to see what is in front of it. Depending on the color that it sees, the robot carries out a set of instructions from the software.

Project: Tractor Bob

Software

The program for Tractor Bob determines his behavior as a drawer. The program initializes three motors and then runs four main processes. These processes never finish because they are set to loop infinitely. These threads are:

(1) Hit Detection: this thread is set to prevent the robot from running over an obstacle. Whenever the front touch sensors are pressed, the robot's movement is altered momentarily. The robot stops, backs up, turns left, and goes forward on its merry way.

(2) Sector Detection: this thread keeps track of where the robot has traveled in the arena. Whenever the color sensor crosses a red or a yellow border, it keeps track of the border it crosses in a number variable called "Sector". Then the next time it crosses one of those colors, it compares the new value against the old with a switch, and changes the variable according to the logic.

(3) Sector Display: this thread takes the information saved in the Sector Detection thread and displays it as output on the NXT Brick continuously.

(4) Border Line Detection: this thread gives the robot information on what to do when it sees the other robot's color or the borders of the arena. When the robot sees the color black, it knows that it's found the edge of the arena, stops, backs up, turns left and continues again. When it detects the color of the other robot, it stops before crossing, backs up, turns right, and continues again.

Project: Tractor Bob

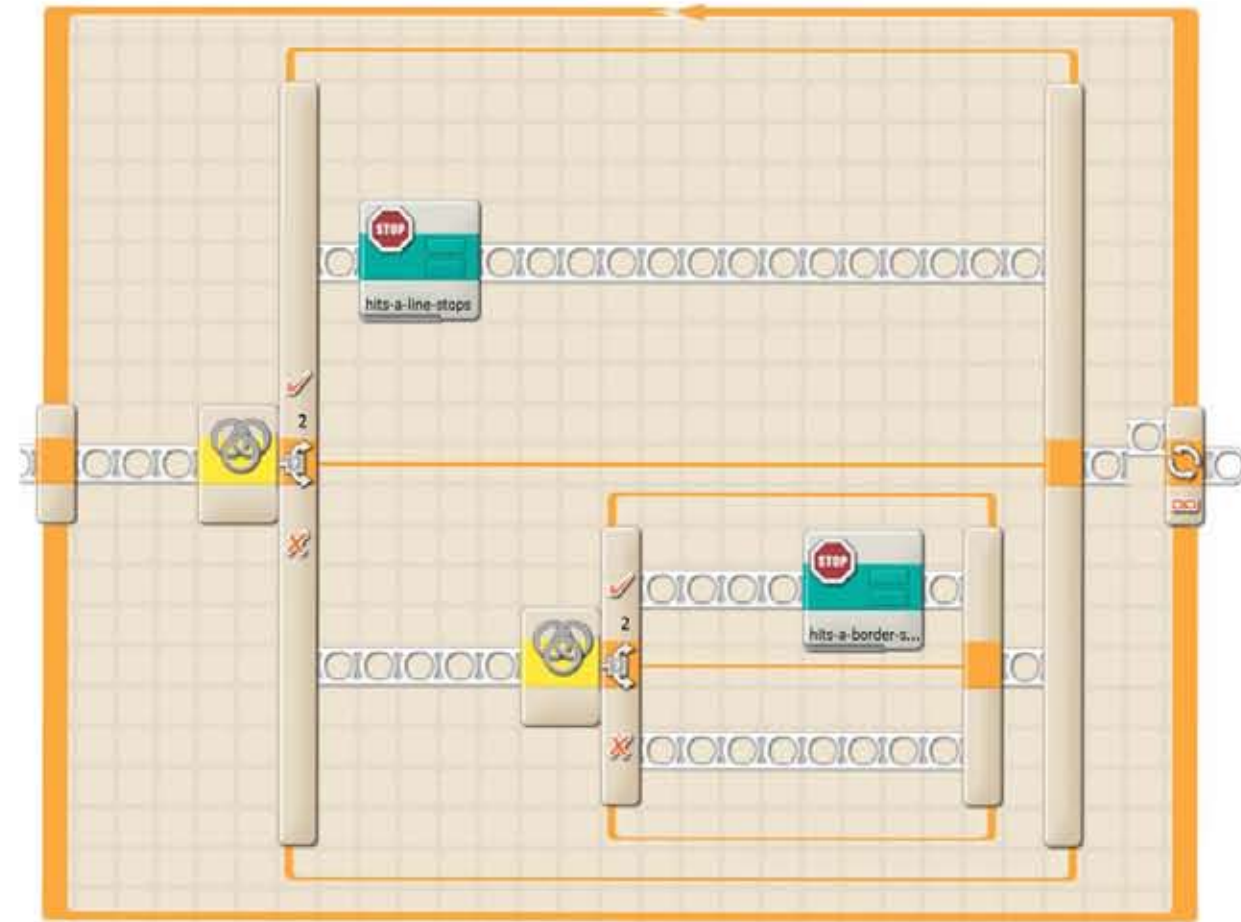
Initialization

The program begins by putting the wheels into motion, and putting the pen down to the surface.



Main Methods

Border Line Detection: this thread gives the robot information on what to do when it sees the other robot's color or the borders of the arena. When the robot sees the color black, it knows that it's found the edge of the arena, stops, backs up, turns left and continues again. When it detects the color of the other robot, it stops before crossing, backs up, turns right, and continues again.



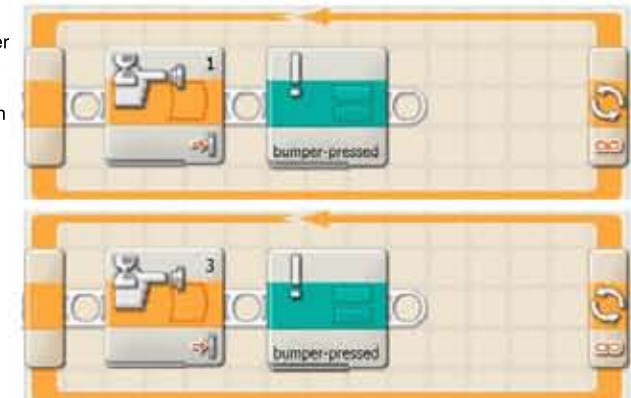
Sector Detection: this thread keeps track of where the robot has traveled in the arena. Whenever the color sensor crosses a red or a yellow border, it keeps track of the border it crosses in a number variable called "Sector". Then the next time it crosses one of those colors, it compares the new value



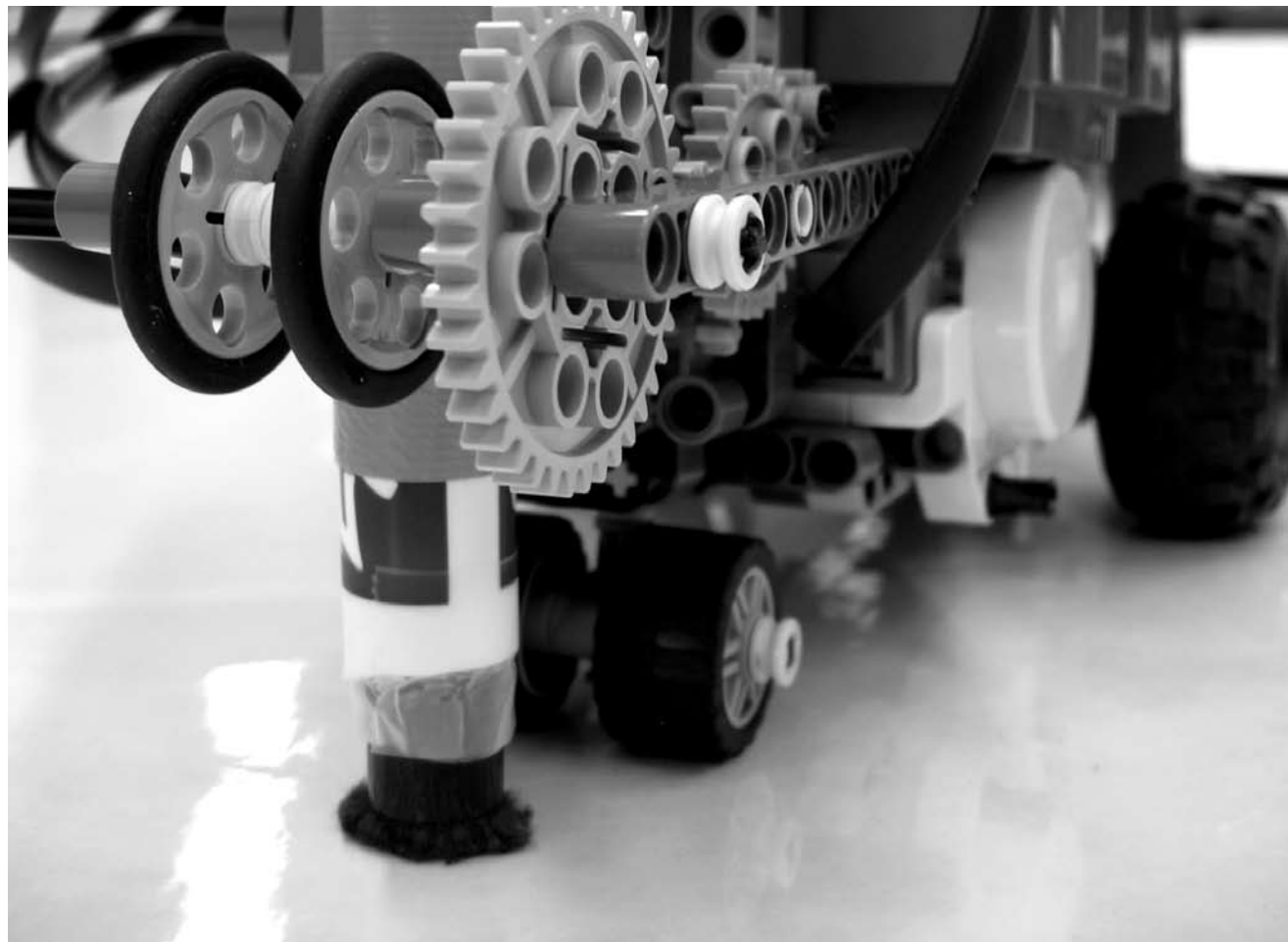
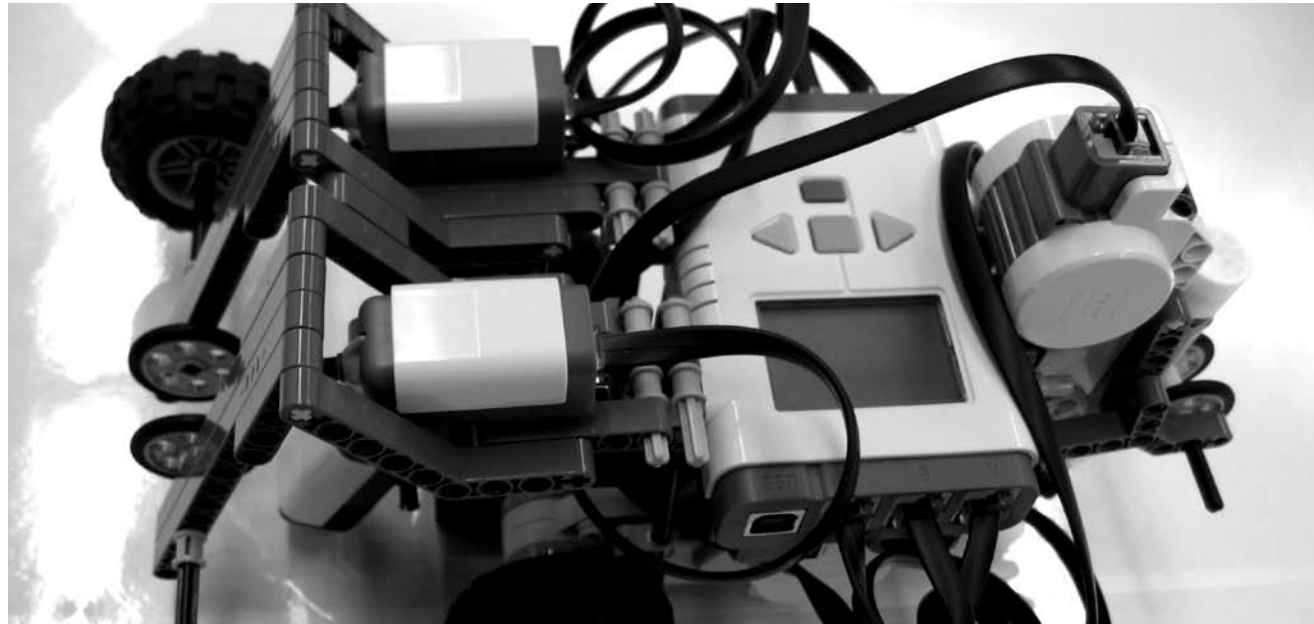
Sector Display: this thread takes the information saved in the Sector Detection thread and displays it as output on the NXT Brick continuously.



Hit Detection: this thread is set to prevent the robot from running over an obstacle. Whenever the front touch sensors are pressed, the robot's movement is altered momentarily. The robot stops, backs up, turns left, and goes forward on its merry way.



Project: Malachi



Hardware

NXT Outputs

- A: Left Motor
- B: Motor for marker
- C: Right Motor (and drive train for rotary cleaner)

NXT Inputs

- 1: Touch Sensor
- 2: Color Sensor
- 3: Touch Sensor
- 4: Empty

Locomotion: Malachi uses two independent motors for locomotion. The robot moves forward by applying both motors forward, backward by applying both motors backward, and turns by moving one forward while the other is stopped.

Detection: Malachi uses the color sensor to detect the external boundaries of the system as well as the arbitrary colors that signify the sector boundaries. Since Malachi doesn't want to make a mess of his art, he tries not to draw over his own or the other robot's lines. He does this by detecting blue or green lines. However, Malachi is a bit impatient, and gets frustrated when the other robot's lines get in his way. He jumps back and forth in an angry manner when he sees a blue line in his way. Malachi also uses its touch sensors to detect the physical presence of the other robots.

Drawing: Malachi draws in a green color, lifting his marker when he sees a blue or green line.

Project: Malachi

Software

The program for Malachi determines his behavior as an artist. It begins by running five simultaneous processes. Each of them will run infinitely, so the robot will have to be shut off manually. Each of the five processes are described below.

Eraser Caller: This program contains the logic that will tell The Big Zamboni that there is too much ink on the ground and that it should step up the cleaning. It sends a Bluetooth message to The Big Zamboni after it has seen 50 lines.

Display: This program reads the sector variable (determined in the Sector Analysis) and translates the number into a sector name. It then writes the sector name on the robot's display. For example, if the sector variable equals 2, then the robot's display will say "White Sector." Because this program is always running, when the sector variable changes the display will also change automatically.

The Line is Fine!: This program controls the standard motion of the robot. It begins by turning on the motors for locomotion (motors A and C). Then it waits until the color sensor detects black, blue, or green. If detected, it will chose a random direction to turn and continue moving.

Sector Detector: This program continually checks if the robot is seeing Red or Yellow. If it sees either of the colors, it waits until it doesn't see them anymore and then updates the sector variable to reflect the new sector.

Touch Detector: This program runs the left and right touch sensors, waiting until either is pressed. It will then say "Ow!", back up a short distance, and continue with the program.

Project: Malachi

Initialize

This step sets the initial sector position to the starting sector(white). It also sets the wheel in motion and puts the pen down.



Main Methods

This program runs the left and right touch sensors, waiting until either is pressed. It will then say "Owl", back up a short distance, and continue with the program.



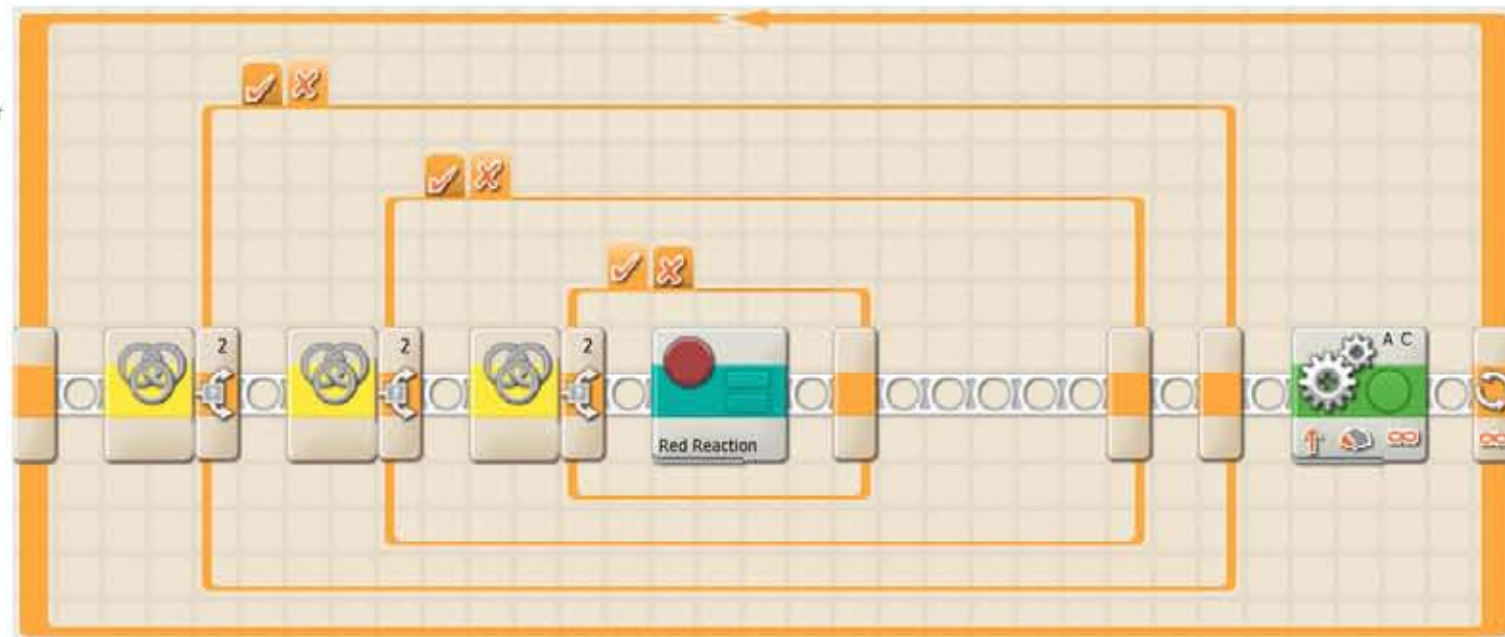
This program contains the logic that will tell The Big Zamboni that there is too much ink on the ground and that it should step up the cleaning. It sends a Bluetooth message to The Big Zamboni after it has seen 50 lines.



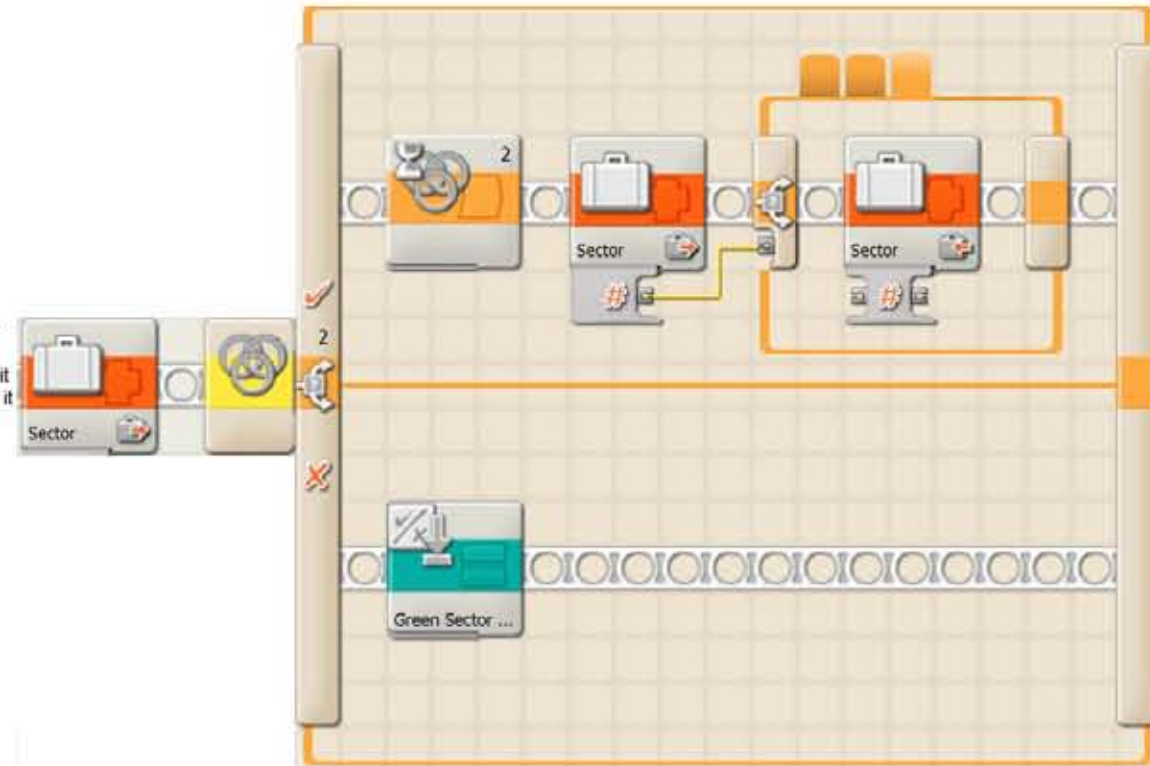
This program reads the sector variable (determined in the Sector Analysis) and translates the number into a sector name. It then writes the sector name on the robot's display. For example, if the sector variable equals 2, then the robot's display will say "White Sector." Because this program is always running, when the sector variable changes the display will also change automatically.



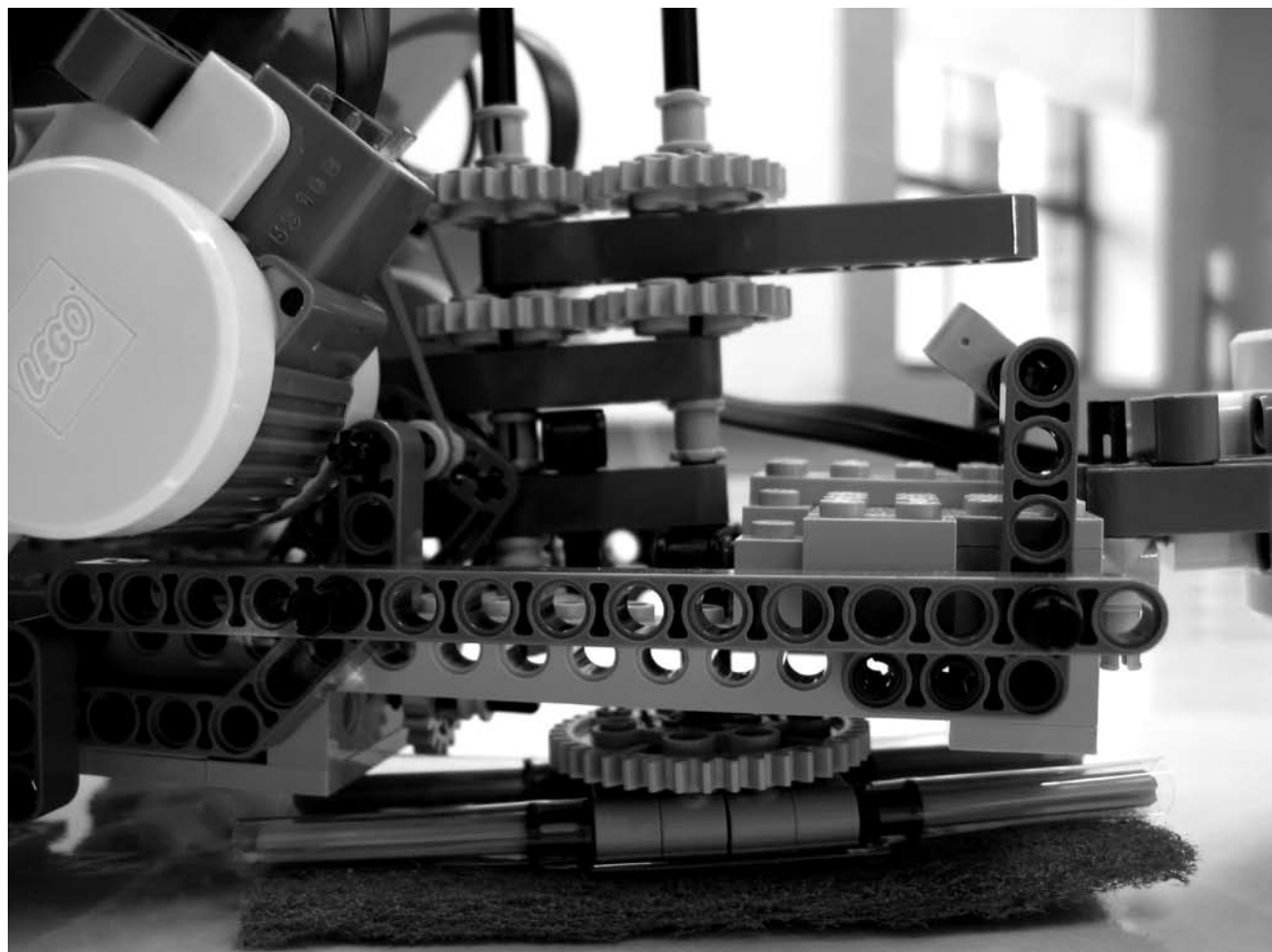
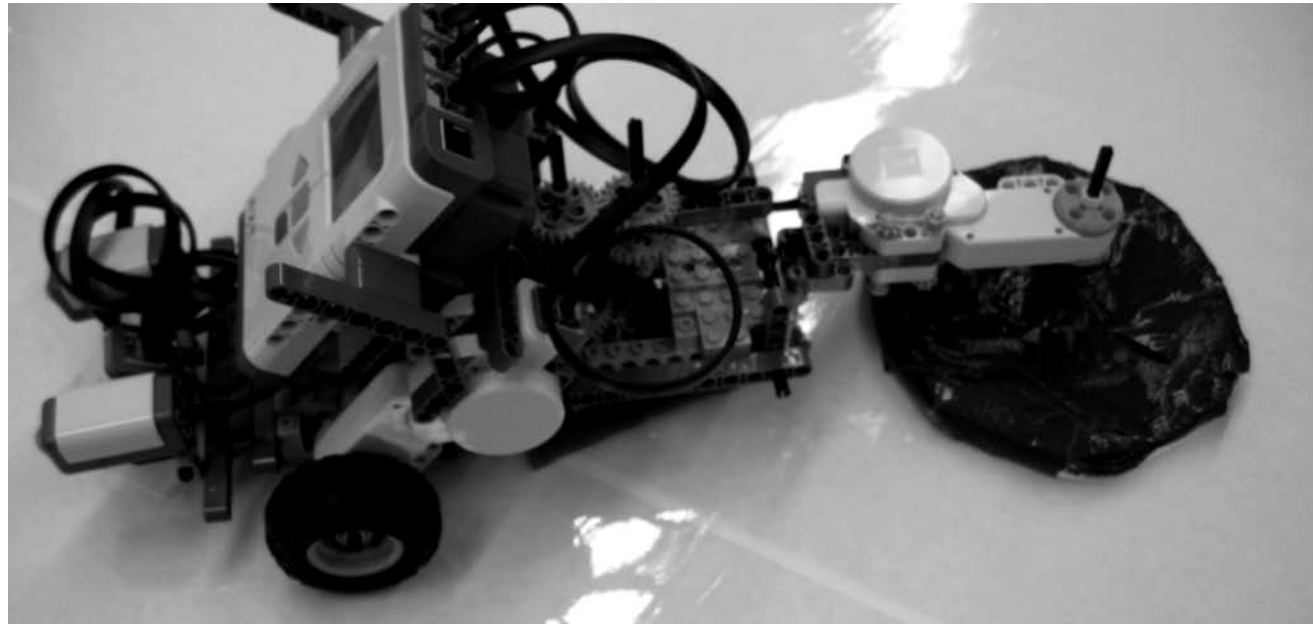
This program controls the standard motion of the robot. It begins by turning on the motors for locomotion (motors A and C). Then it waits until the color sensor detects black, blue, or green. If detected, it will chose a random direction to turn and continue moving.



This program continually checks if the robot is seeing Red or Yellow. If it sees either of the colors, it waits until it doesn't see them anymore and then updates the sector variable to reflect the new sector.



Project: The Big Zamboni



Hardware

NXT Outputs

- A: Left Motor
- B: Motor for Buffer
- C: Right Motor (and drive train for rotary cleaner)

NXT Inputs

- 1: Touch Sensor
- 2: Color Sensor
- 3: Empty
- 4: Touch Sensor

Locomotion: The Big Zamboni uses two independent motors for locomotion. The robot moves forward by applying both motors forward, backward by applying both motors backward, and turns by moving one forward while the other is stopped. The robot is further supported by the spinning cleaning apparatus and by the buffer which trails behind the body.

Detection: The Big Zamboni uses the color sensor to detect the external boundaries of the system as well as the arbitrary colors that signify the sector boundaries. So that it can erase freely, The Big Zamboni disregards the colors placed by the two drawing robots. The Big Zamboni also uses its touch sensors to detect the physical presence of the other robots.

Cleaning: The Big Zamboni has two mechanisms for cleaning: a wet rotary cleaner, and a dry rotary buffer.

The rotary cleaner is located underneath the NXT brick. It spins and moistens the surface to erase the lines drawn by other robots. The actual cleaner consists of an abrasive cleansing towel attached to an abrasive sponge via friction. A vinyl sleeve has been hot-glued to the abrasive sponge. This vinyl fits around four LEGO axles. The axle is moved by a system of gears attached to the right motor.

The buffer consists of a microfiber cloth attached to a large yellow LEGO wheel (the same wheel as Tractor Bob) by duct tape. The buffer has its own motor, and trails behind the rotary cleaner to dry the surface while spinning.

Project: The Big Zamboni

Software

The program for The Big Zamboni determines his behavior as an eraser. It begins by infinitely running two processes: Eraser Test II (which is explained in the subsequent paragraphs), and EmergencyCall (which is not used in the current system). Because the programs run indefinitely, the robot must be shut off manually.

Eraser Test II consists of four subroutines: Sector Analysis, Sector Display, Boundary, and TouchSensors. These four programs run at the same time over and over again.

Sector Analysis: This program uses the information from the color sensor to determine where The Big Zamboni is located in the system. The sectors are marked with red (sector 1) and yellow (sector 3) tape. The white sector (sector 2) lies between the red and yellow sectors. The program first determines the number of the sector where the robot is located, and then performs a different task based on the number it reads. For example, if the sector variable equals 2 (as in the accompanying schematic) then the current sector is the white sector. The robot will not change the sector variable until it passes either the red or yellow tape, at which point it will wait until it has finished passing the tape, and will then reset the sector variable to the new sector number.

Sector Display: This program reads the sector variable (determined in the Sector Analysis) and translates the number into a sector name. It then writes the sector name on the robot's display. For example, if the sector variable equals 2, then the robot's display will say "White Sector." Because this program is always running, when the sector variable changes the display will also change automatically.

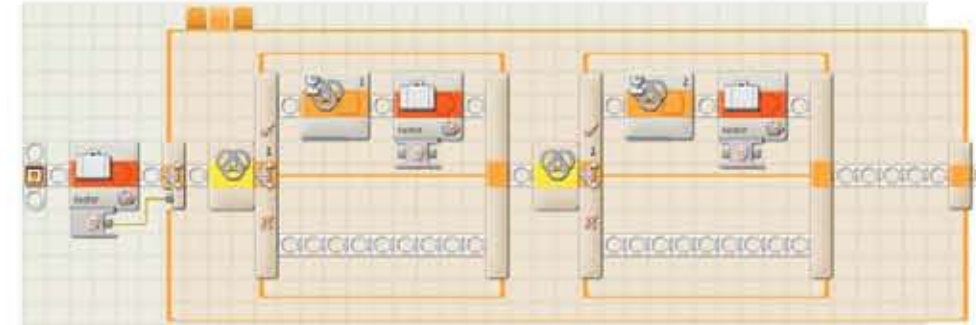
Boundary: This program controls the standard motion of the robot. It begins by turning on the motors for locomotion (motors A and C) and for the buffer (motor B). Then it waits until the color sensor detects black. When the color sensor detects black, the motors for movement (A and C) will move backward for one second, and then the robot will turn to the right. Then the program repeats.

TouchSensors: This program waits until one of the touch sensors is pressed, then backs up for one second and turns right (just like the behavior when it senses a boundary).

Project: The Big Zamboni

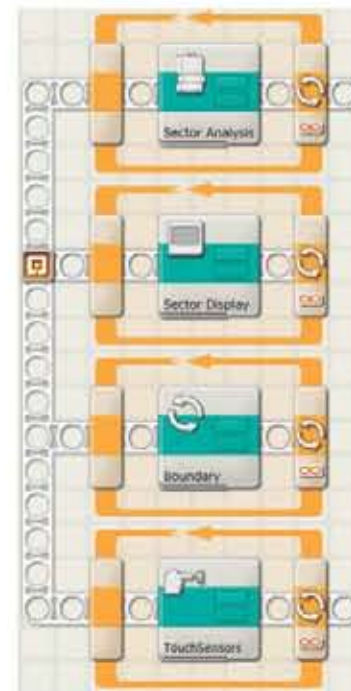
SECTOR ANALYSIS

Begins by analyzing the current sector. The color sensor looks for a specific color, waits until color changes back to white, then changes the sector variable to the new number. 1 = Red, 2 = White, 3 = Yellow.



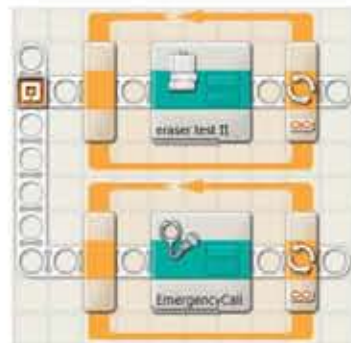
ERASER TEST II

Loops "Sector Analysis," "Sector Display," "Boundary," and "Touch Sensors" at the same time.



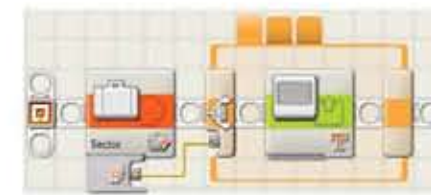
ERASER

Loops "Eraser Test II" and "Emergency Call" at the same time.



SECTOR DISPLAY

Reads the current sector number, and outputs the appropriate sector name to the display screen.



BOUNDARY

Moves all motors forward until it detects the black boundary. When the boundary is detected, the robot will back up for one second, and then turn around. When the robot has finished turning, the program repeats.



TOUCH1

Waits until the touch sensor on Port 1 is pressed, then backs up and turns around.



TOUCH4

Waits until the touch sensor on Port 4 is pressed, then backs up and turns around.



TOUCH SENSORS

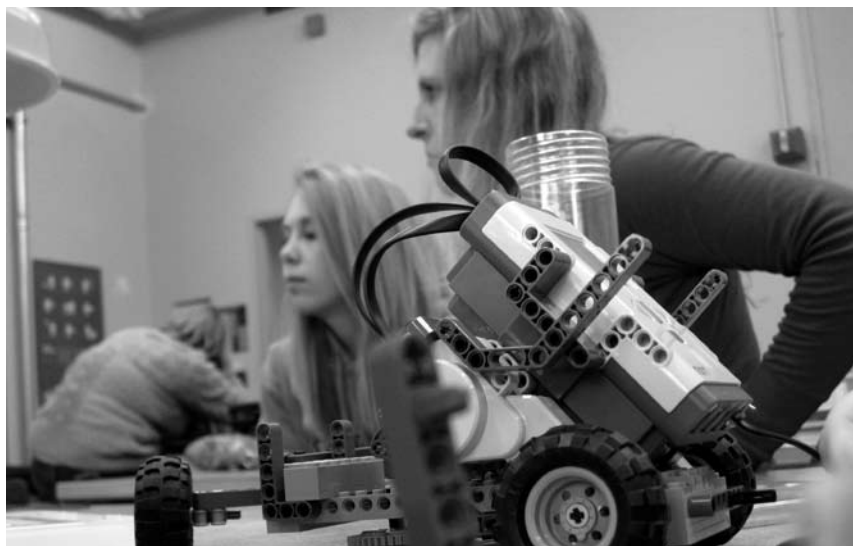
Runs "Touch1" and "Touch4" at the same time.



Results

Team

Kiira Hjert-Bernardi
Chris Govella
Kimberly Fulton
Tom Futrell
Amy leNoir
Christina Koehn
Chris Pritchard
Danielle Schechter



Process

The concept began with Tom's proposal to create robots that can draw.

During our brainstorming sessions, we came up with several different implementations, including a system of search and rescue robots, line-tracing and following robots, and line-removal robots.

After more practical analysis, the project evolved into a system where two robots function as artists. These artist robots would be aware of both the lines they draw and the lines drawn by the other robot, as well as their relative location within the system, and a boundary.

When we gave this idea more thought, we realized that a robot could easily become trapped within the bounds of its own lines or that of its counterpart, and that is how we developed the idea for the eraser robot. When a robot becomes trapped, it would "summon" the eraser robot via Bluetooth to travel to the appropriate sector, and free the robot by erasing lines. In the event that the eraser robot cannot free the other robot in a sufficient amount of time, the robot would exhibit additional behavior (such as frustration or anger) and would break through its bounds to continue.

While that was the ideal implementation of our system, we encountered several challenges when we received the color sensors that further narrowed the scope of our experiment:

(1) We were limited by the number of input ports on the robot. This forced us to eliminate our original goal of having the robots interact with humans through sound and light. Instead, interaction between the robots and humans is limited to manipulation of the drawing environment.

(2) We had trouble with consistency between the sensors and the environment, and had to change colors for boundary, sectors, and markers several times. As a result, we simply did not have enough time to program either the "stuck" algorithm or the communication to the eraser robot.

(3) Finally, we decided early on in the process to use Mindstorms as our programming language. We quickly realized the limitations of the software when it came to more complex programs, but did not have enough time to learn a new system. The more complex our programs got, the more problems we encountered with bugs, system crashes, and lack of memory. Ultimately, our programmers spent more time trying to work around the limitations of the software than actually writing the programs.

Because of these stresses, our current implementation is a more simple system, where two robots draw lines while one erases, with little direct communication between the robots. Instead, the robots interact with each other (and with humans) through their environment by drawing on and erasing the canvas. Throughout this process, each of the robots' "personalities" is revealed by its patterns of movement and manipulation of the drawing space.

Results

Summary

While the final result of our project is a complex system of interaction between three robots with different objectives, our project did not quite work as planned. Because we spent so much time troubleshooting hardware and software issues, we did not have as much time as we would have liked to test the programs and implement more complex behaviors.

Also, because we were limited to four input sensors per robot, we were unable to have the level of interaction with humans what we would have liked. With more time, we could split the input channels, and the robots could interact with the environment through sound and light.

We would have liked to spend more time with the Bluetooth communication. With better communication, we could have increased the amount of robots, and increased the complexity of the drawing patterns.

Evaluation

Our group worked together very well. We took a hands-on approach to the design process, as opposed to some other teams who appeared to be very conceptual. It was a good system for us because we encountered a lot of problems--things falling apart, working inconsistently, or not as we expected. By employing this method of trial-and-error we were able to reach our design goals and solve problems as a group.

One shortfall of this method however, is that because the physical design of the robots was constantly evolving, we were unable to begin programming early enough in the process to accomplish everything that we wanted to. Also, because there were so many people in our group, it was difficult to coordinate schedules and have everyone meet to work on the project together.

By working as a multi-disciplinary team, we were able to draw upon the expertise of each of our members to accomplish different goals. There was a lot of innovation in design from the design students, and the programmers were able to create complex code quickly. The different backgrounds of the team members also provided each of us with the opportunity to work with different personalities than we typically work with every day, on a project that is outside the norm. We were able to stretch our creativity, and create a better project overall.

If we were able to do this project all over again, we would have begun with a different programming language (such as C) that would enable us to program complex behaviors rapidly, and use less memory.

We might try to create a more structured timetable with deliverables at certain steps to make sure that we have adequate time for both design and programming.

We would also try to find a way to make the color sensors more reliable by experimenting with lighting conditions and different surfaces for the drawing area.

Despite these shortfalls, we are happy with the results of our project and the experience overall.