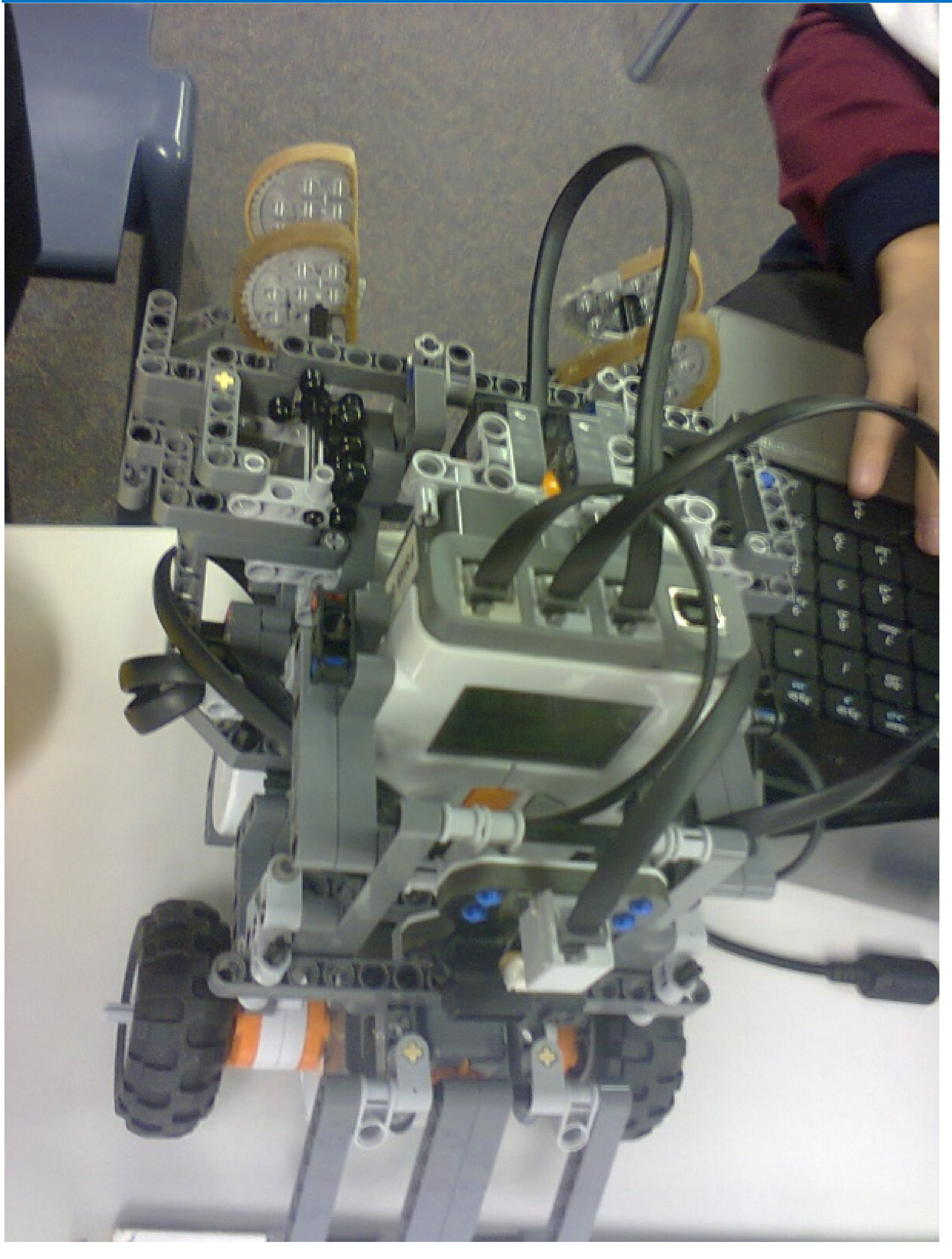
Open Rescue Logbook 2014



Team Captain: Andrew Wu

Jason Wooi

Jackson Zheng

Problem Definition

3/4/2014 T1W10

Broadly, our goal is to construct and code a robot that can navigate the course, pick up the can, place it onto the upraised block and return to the Spill Access Point.

After examining the Rescue Rules and due to the team members' previous experiences with Rescue, the team was able to designate the following requirements;

- For the robot:
  - The robot need to be balanced, especially for the sloped tiles
  - External interference of light should be minimised to ensure correct colour feedback from tiles
  - The touch sensor can be easily activated
  - The distance sensor is positioned without obstructions
  - The mechanism by which the robot picks up the can has sufficient friction to pick up the can reliably
  - The size of the robot is small enough to be able to move through doorways and tunnels
  - The robot does not easily fall apart
- For the code:
  - The green, white, black and silver colour values are correctly defined
  - The robot can manoeuvre along black lines
  - The robot moves toward the direction of the detected green square
  - The robot navigates around the bottle once the touch sensor is activated in a circular motion
  - The robot can reliably detect and move the can to the upraised block

Jason and Andrew produced most of the criteria since Jackson was the least experienced member.

Planning

10/4/2014 T1W11

In previous Rescue events, major problems the team members encountered were-

- Insufficient traction on sloped tiles
- Insufficient friction to lift the can
- The robot detecting erroneous reflective material when searching for the victim
- Knocking over the water tower
- Failure to navigate through the gridlock

Therefore, the team aims to remedy these errors in their new design.

The robot will be coded in RobotC and constructed with Lego and Lego Mindstorms NXT components.

1/5/2014 T2W1

The goals outlined during defining stage were expanded upon;

- The robot needs at least four wheels to ensure balance (particularly for the see-saw and sloped tiles)
  - Weight should be evenly distributed
- The wheels that are attached to the motors are located close to the light sensors to synchronise movement
- The wheels need sufficient traction (in particular for navigating on sloped surfaces)
  - Masses can be added on top of the wheels if they skid
- Large "shields" can be attached to the sensor to increase chance of contact between the bottle or can
  - However, the shields cannot be too heavy, otherwise the sensor may spontaneously activate on a sloped surface
- Due to the different colour values in different lighting environments, the robot will need to be calibrated when the environment is changed
- Two light sensors are positioned as close to the ground as possible to inhibit external light interference
  - Light shields can also be used, composed of paper taped around the sides of the sensors
- The robot can continue following the line after the black line after it navigates around the water tower
- In the previous RoboCup, the team members had used cogs with rubber rims added to lift the can, however, they were not reliable
  - Andrew decided to implement a claw mechanism which uses one motor to simultaneously lift the and grasp the can

Due to resource constraints, the team is restricted to three motors, one distance sensor, two light sensors, one touch sensor and one NXT Intelligent Brick.

Once the robot detects silver, the code initiates a new module in which the robot searches for the can with the distance sensor;

- The robot "remembers" its movements when this module is activated
- Once the can is located, the robot moves toward it
- The can should activate the touch sensor
- When the can is picked up, the robot reverses its "remembered" movements to return to the starting position at the silver rectangle
- The robot then moves forward until the upraised block activates the touch sensor
- The robot places the can onto the block
- The robot exits via the Spill Access Point and terminates

8/5/2014 T2W2

The mentor, Mr. Grant was contacted to understand time constraints. Basic modules could be coded initially, although specifics can only be coded once the robot was constructed as the code could only be tested afterward. Below is the Gantt chart we developed for our project.

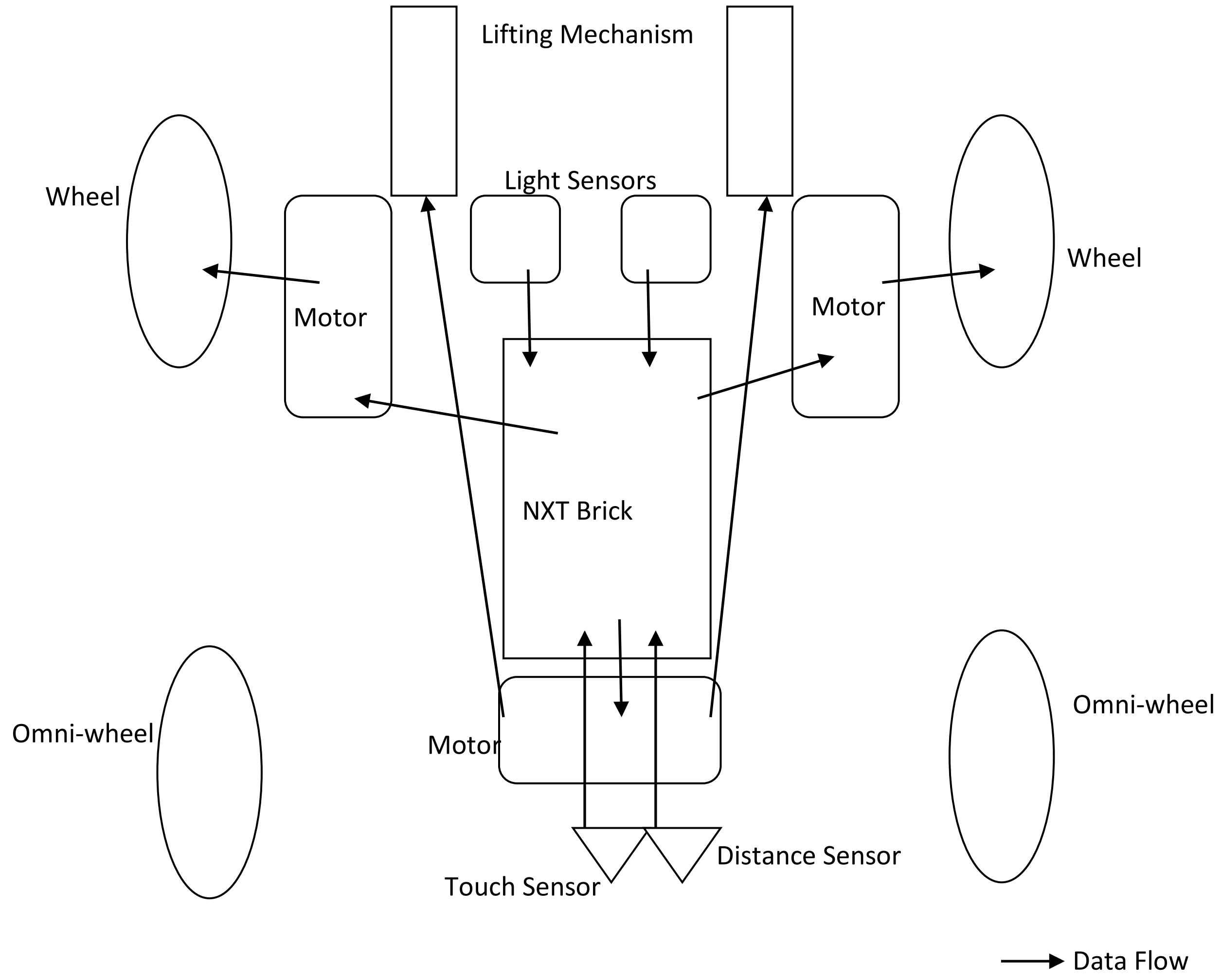
Since Andrew was the most logic-oriented member and had the most experience building in the past, he tasked himself with constructing the robot. Jason will be giving Andrew specific instructions to as he was designated to be the main programmer; Jason should have the optimal robot to work with when he programs. Jackson's most significant task will be writing the documentation. Since this requires significant communication with Jason and Andrew, he will learn their coding and constructing thought processes and hence improve his skills as he is the least experienced member. Otherwise, Jackson acts as a handyman and an occasional conduit between Jason and Andrew.

Development Arc	Task	Time Period																	
		T1W10	T1W11	T2W1	T2W2	T2W3	T2W4	T2W5	T2W6	T2W7	T2W8	T2W9	T3W1	T3W2	T3W3	T3W4	T3W5	T3W6	T3W7
Defining	Problem Outlining	■																	
Planning	Goal Specifications		■	■															
	Time Schedule				■														
	Role Allocation				■														
Design	Robot Plan					■													
	Coding Plan						■	■											
Implementation	Basic Frame							■	■										
	Wheel Motors								■	■									
	Touch and Distance Sensor									■	■								
	Lifting Mechanism										■	■	■						
	Line Following Code												■	■					
	Green Turning Code													■					
	Water Tower Code													■	■				
	Can Lifting Code														■	■	■		
Evaluation	Confirmation of Compliance to Rules																	■	
	Explanation of Deviation from Design																		■

Solution Design

15/5/2014 T2W3

Andrew planned to construct a robot with the following design.



The light sensors would relay their detected colour values to the brick, which the brick interprets and sends appropriate commands to the motors controlling the frontal wheels. The touch sensor sends a value to the brick if it is pushed, which the brick initially interprets as the bottle being contacted, thus it begins to manoeuvre around it using the motors. However, after the bottle has been encountered, activating the touch sensor would instead cause the brick to interpret the value as contacting the can, causing it to rotate and pick up the can using the third motor. The distance sensor is used for searching for the can once the Spill Access Point has been detected.

22/5/2014 T2W4

The pseudocode below is heavily simplified compared to the version of the code Jason described; different movements referred to different modules. The values of a particular colour range from halfway to the next darkest from halfway the next lightest (e.g. the colour range for green is defined as the average of black and green to the average of white and green). Furthermore, several commands to the wheel motors will be given time values (not indicated in the pseudocode) such that they are active for that amount of time. If the pseudocode is unsuccessful, Jackson and Jason have agreed to hard-code the robot.

Jason decided to take a module-based approach to code the robot. Jackson simplified his description of his predicted code as thus;

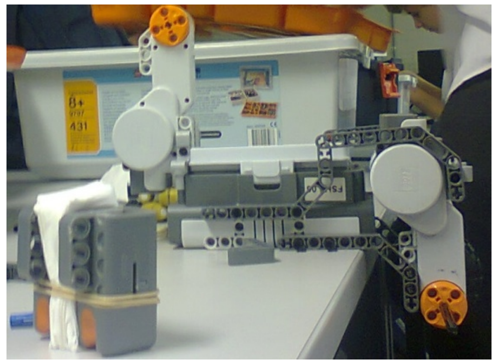
```
BEGIN Main
	WHILE(1)
		BEGIN Values
		IF CanOperation=0 THEN
			BEGIN L	ineCode
		END IF
		IF CanOperation=7 THEN
			BREAK
		END IF
	END WHILE
END Main

BEGIN Values
	GET RightLightSensorValue
	GET LeftLightSensorValue
	GET DistanceSensorValue
	GET TouchSensorValue
END VALUES

BEGIN LineCode
	IF RightLightSensorValue=Black AND LeftLightSensorValue=White THEN
		RightMotor=Backward
		LeftMotor=Forward
	END IF
	IF RightLightSensorValue=White AND LeftLightSensorValue=Black THEN
		RightMotor=Forward
		LeftMotor=Backward
	END IF
	IF RightLightSensorValue=White AND LeftLightSensorValue=White THEN
		RightMotor=Forward
		LeftMotor=Forward
	END IF
	IF RightLightSensorValue=Black AND LeftLightSensorValue=Black THEN
		RightMotor=Forward
		LeftMotor=Forward
	END IF
	IF RightLightSensorValue=Green THEN
		LeftMotor=Forward
	END IF
	IF LeftLightSensorValue=Green THEN
		RightMotor=Forward
	END IF
	IF TouchSensorValue=True AND BottleEncountered=False THEN
		FOR t=0 TO 25
			LeftMotor=Backward
			RightMotor=Backward
		NEXT t
		FOR t=0 TO 50
			LeftMotor=Backward
			RightMotor=Forward
		NEXT t
		WHILE(1)
			IF LeftLightSensorValue=Black OR RightLightSensorValue=Black THEN
				FOR t=0 TO 25
					RightMotor=Forward
				NEXT t
				BottleEncountered=True
				BREAK
			ELSE
				RightMotor= Slow Forward
				LeftMotor=Forward
			END IF
		ENDWHILE
	END IF
	IF RightLightSensorValue=Silver THEN
		CanOperation=1
		WHILE(LeftLightSensorValue<>Silver)
			LeftMotor=Forward
		END WHILE
		BEGIN CanRescue
```

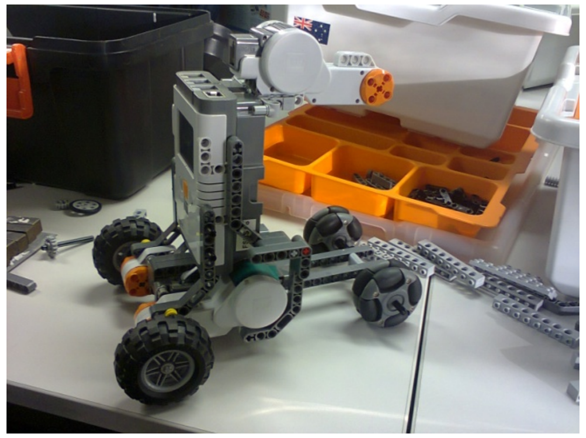
```
	END IF
	IF LeftLightSensorValue=Silver THEN
		CanOperation=1
		WHILE(RightLightSensorValue<>Silver)
			RightMotor=Forward
		END WHILE
		BEGIN CanRescue
	END IF
END LineCode

BEGIN CanRescue
	WHILE(1)
		IF CanOperation=1 THEN
			FOR t=0 TO 50
				RightMotor=Forward
				LeftMotor=Backward
			NEXT t
			CanOperation=2
		ELSE IF CanOperation=2 THEN
			Rotation=100
			BEGIN Values
			FOR t=0 TO Rotation
				IF DistanceSensorValue<=500 THEN
					t=Rotation
					CanOperation=3
				ELSE
					RightMotor=Backward
					LeftMotor=Forward
				END IF
			NEXT t
		ELSE IF CanOperation=3 THEN
			IF TouchSensorValue=True THEN
				CanOperation=4
			ELSE
				RightMotor=Forward
				LeftMotor=Forward
				TimeTaken=TimeTaken+1
			END IF
		ELSE IF CanOperation=4 THEN
			FOR t=0 TO 150
				CanMotor=Forward
			NEXT t
			FOR t=0 TO TimeTaken
				RightMotor=Backward
				LeftMotor=Backward
			NEXT t
			CanOperation=5
		ELSE IF CanOperation=6 THEN
			IF Rotation>=50 THEN
				FOR t=0 TO Rotation-50
					RightMotor=Forward
					LeftMotor=Backward
				NEXT t
			ELSE
				FOR t=0 TO 50-Rotation
					RightMotor=Backward
					LeftMotor=Forward
				NEXT t
			END IF
			FOR t=0 TO 300
				RightMotor=Forward
				LeftMotor=Forward
			NEXT t
			FOR t=0 TO 150
				CanMotor=Backward
			NEXT t
			CanOperation=7
			BREAK
		END IF
END CanRescue
```

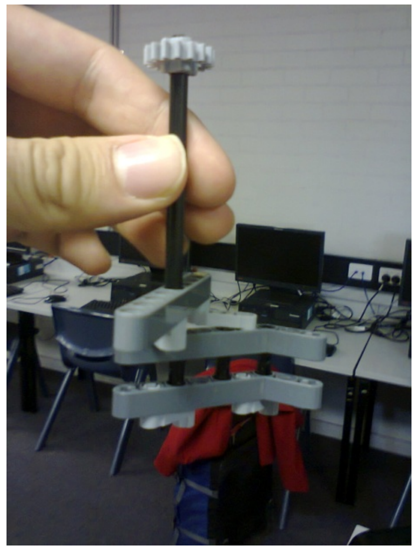
Implementation

29/5/2014 T2W5

Andrew assembled a tank-resembling structure onto the NXT brick, building according to advice from Jason and the guidelines in the solution design. Andrew decided that the light sensors could not be affixed to the robot at his desired length by only using Lego pieces. Thus, tissue paper was added around the light sensors with a rubber band to increase their distance from each other. Jason said that large distances would decrease movement synchronisation while the gap would need to be larger than the width of the line, at 15mm. Andrew finally attached the two light sensors onto the anterior of the robot, near the anterior wheels. Before this session had ended, Andrew had begun constructing the claw mechanism. We are pleased with our progress as we are ahead of our time schedule. A member from another team, Kris, had suggested that we should place the light sensor beside the motors.

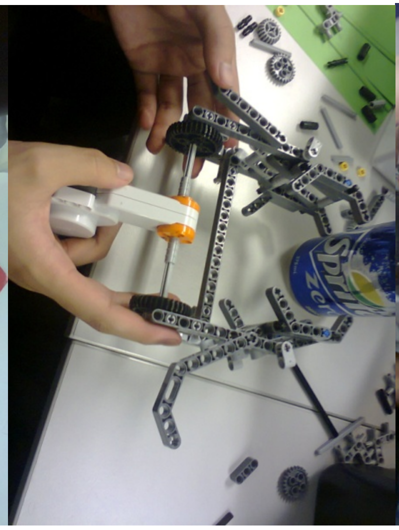


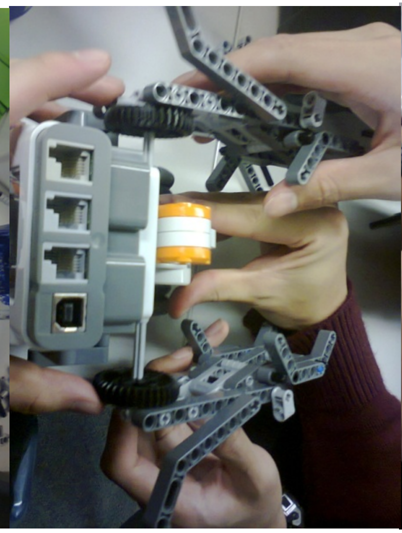


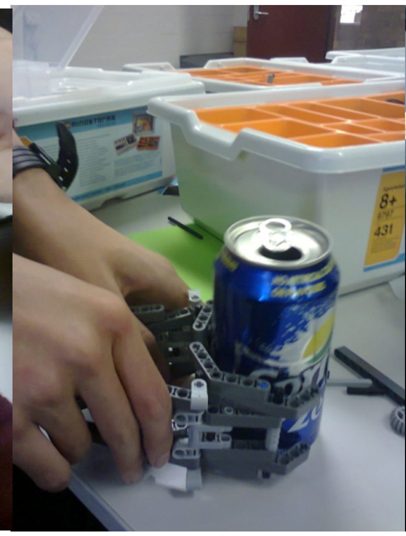


5/6/2014 T2W6

Today the team is constructing the claw-like lifting mechanism. Andrew plans to link both the clawing and lifting mechanism to a single motor with cogs. We used significant amounts of time attempting to find appropriate cogs. We found an insufficient amount of cogs, so we had to resort to using half-cogs, coated in glue, which had been part of the lifting mechanism in another robot. Hence, we used even more time attempting to remove the glue. By the end of the session, we deemed the cogs to be useable. Kris also helped to add aesthetics to the logbook.

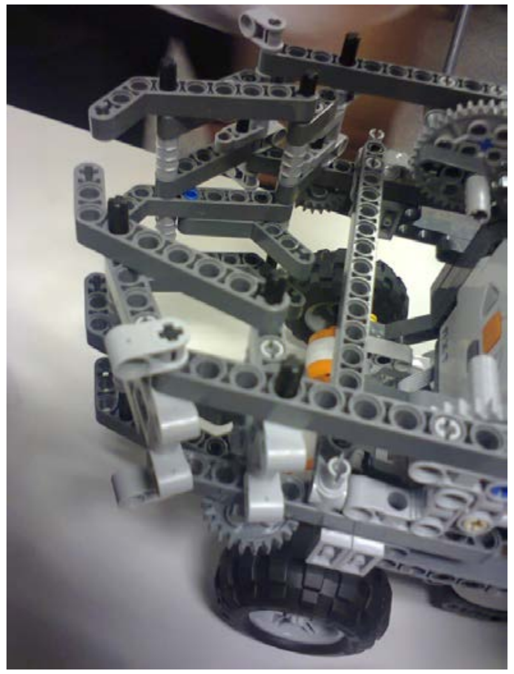


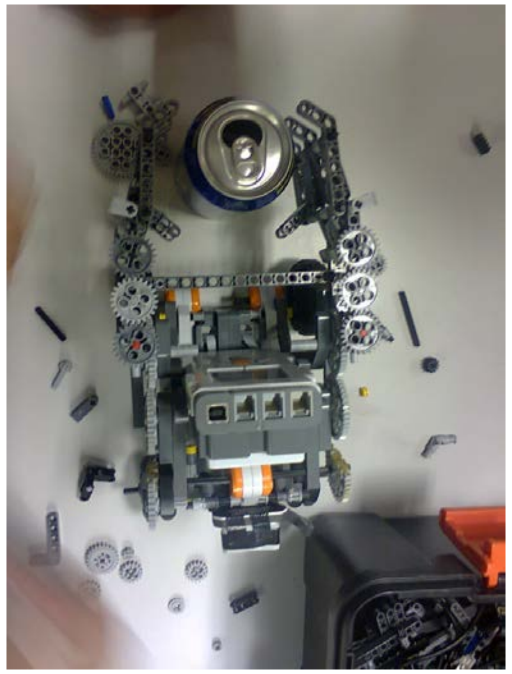




12/6/2014 T2W7

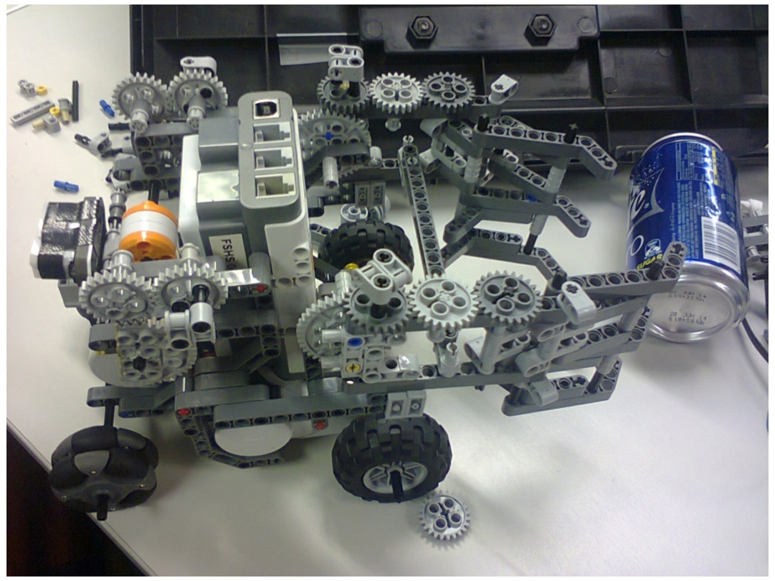
The team finished constructing the lifting claws, so Andrew started constructing an overhead cog system which could simultaneously lift and grip the can when the motor is activated. We experimented with various cog combinations, however, as of yet, none were successful. Darren, another member from a different team, suggested that we should place the distance sensor on the anterior of the robot, so we did. Despite these setbacks, we were still ahead of schedule.

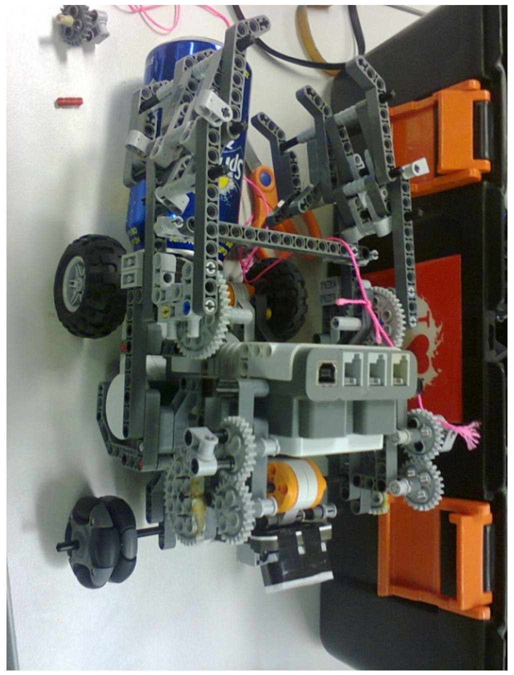


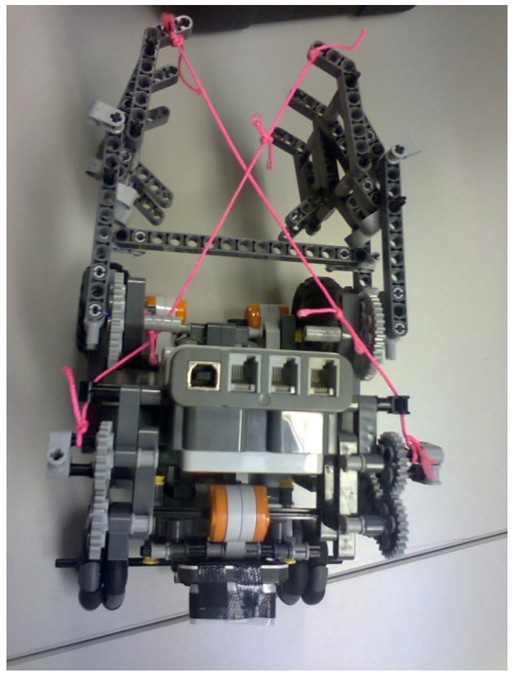


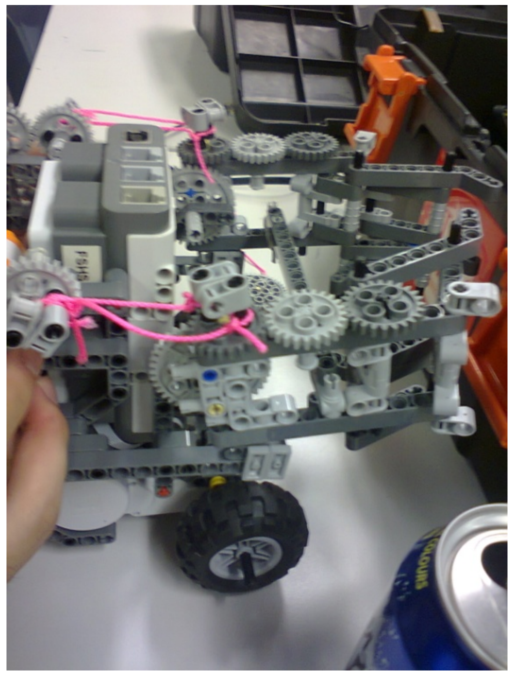
19/6/2014 T2W8

Andrew continued to develop the cog system. Due no suitable pieces, Andrew could not directly connect the two halves of the cog system, picture below, so he tasked Jason and Jackson to find a means of turning the cogs at a distance. Jason and Jackson tested indirectly turning the cogs with rubber bands; however, they lacked sufficient tension. Hence, Andrew resorted to using string to turn the cogs. The string was adjusted such that the claws could simultaneously grip both sides of the can. We failed to achieve a simultaneous claw gripping, thus Andrew rearranged the cogs so that they would lift the can with one turn of the cogs. Andrew tried different cog combinations, all to no success.



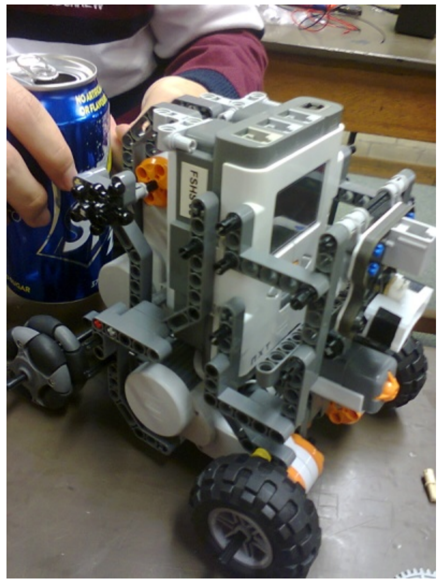


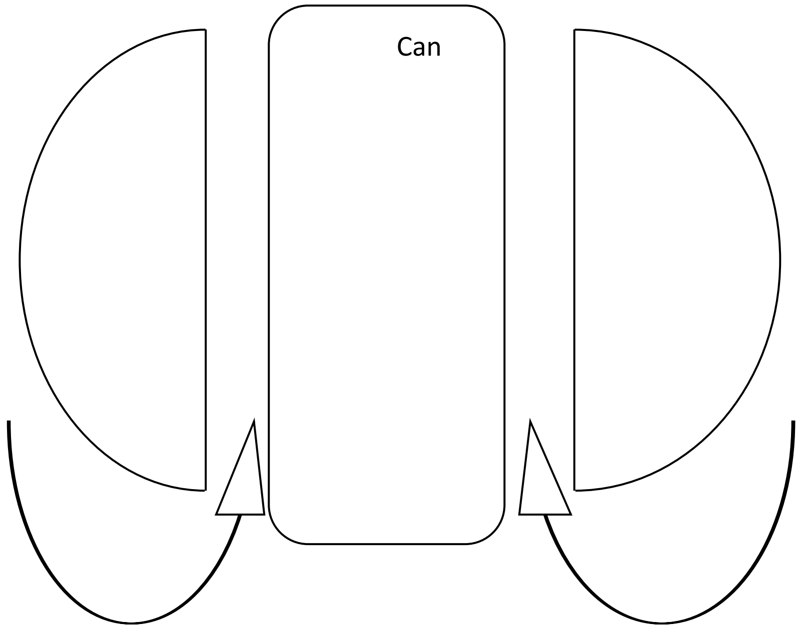




26/6/2014 T2W9

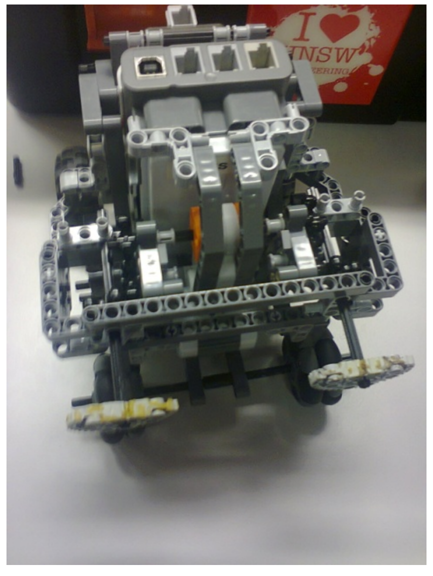
Today the team deemed the gripping mechanism to be too inconsistent and unreliable so we abandoned it. However, in doing so, we had to disassemble the majority of our robot, putting us behind schedule. The team decided to use last year's lifting mechanism, which consisted of two pairs of semicircle cogs rotating to lift the can.

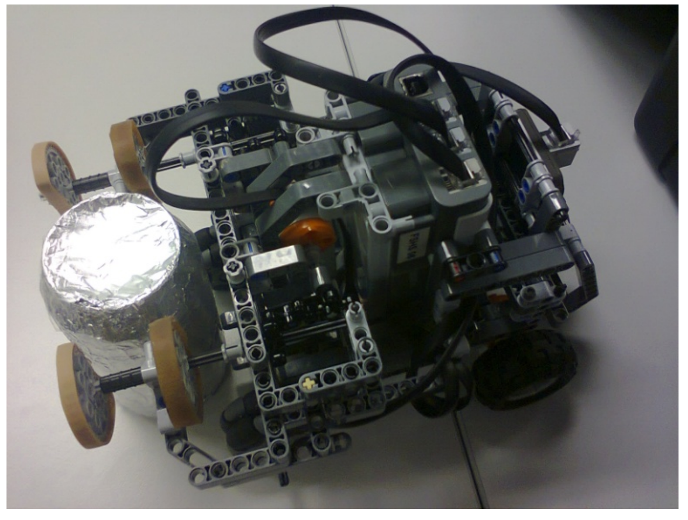




17/7/2014 T3W1

Since we should have started the line following code, Andrew and Jason worked simultaneously on the robot; Andrew constructed the lifting mechanism on the posterior, while Jason attached he distance and touch sensors on the anterior. Jason attached an array of L-shaped Lego pieces onto the touch sensor to increase the potential surface area of contact, however, the team thereafter decided that the array should be made smaller, since it had too much inertia. Jason deemed the robot to be (somewhat) functional and has decided to start programming in the next session.





31/7/2014 T3W3

Two weeks were missed due to HSC Trial examinations. Since the hardware was deemed satisfactory, Jason started optimising the software. Jason encountered various light calibration problems. Even after re-calibrating the light values, the robot ran off-course. Thus, Jason asked Andrew to install light shields/reflectors around the sensors in the form of a tape-paper hybrid. Nevertheless, the robot still failed to follow the black line. Jason then identified that the sensors were located in close proximity together and hypothesised that the feedback from the black line to the sensors was overlapping.

7/8/2014 T3W4

After the robot did not follow the black line correctly, Darren identified that the problem was likely to be the sensors being situated behind the wheels, such that the robot reacted more slowly than others. Thus, Andrew had to reposition the sensors in front of the wheels; however, the sensors were very close to each other. The wheels also had to be replaced, since they were either too large (jamming with the can and other parts of the robot) or frictionless. To increase traction, Andrew hot-glued rubber bands onto the wheels. When this failed, Jason attached weights above the wheels. When this failed, Jason slightly pushed the wheels out, which somewhat alleviated the problem. After these fixes, the robot could follow the line, albeit not smoothly.

8/8/2014 T3W4

Our mentor instigated a robotics incursion in which we had 7 hours to work on our robot at school.

The code for the green turning points was completed without significant problems. The code has the robot turn for a period of time then move forward for a period of time.

The first major problem encountered when coding for the bottle is the robot's posterior colliding with the bottle as it readjusts itself onto the line, so Jason increased the radius of the turn. Furthermore, after touching bottle, the robot can stop executing the bottle code as it detects the original black line before it departs.

After the bottle code was deemed satisfactory, Jason noticed that the colour values for different tiles were different, and thus we had to calibrate the robot for individual tiles. Later, Jason replaced the less/more or equal signs to just less/more signs, to stop contradictions when the light sensors detect a boundary value. Andrew also improved the light shields around the sensors.

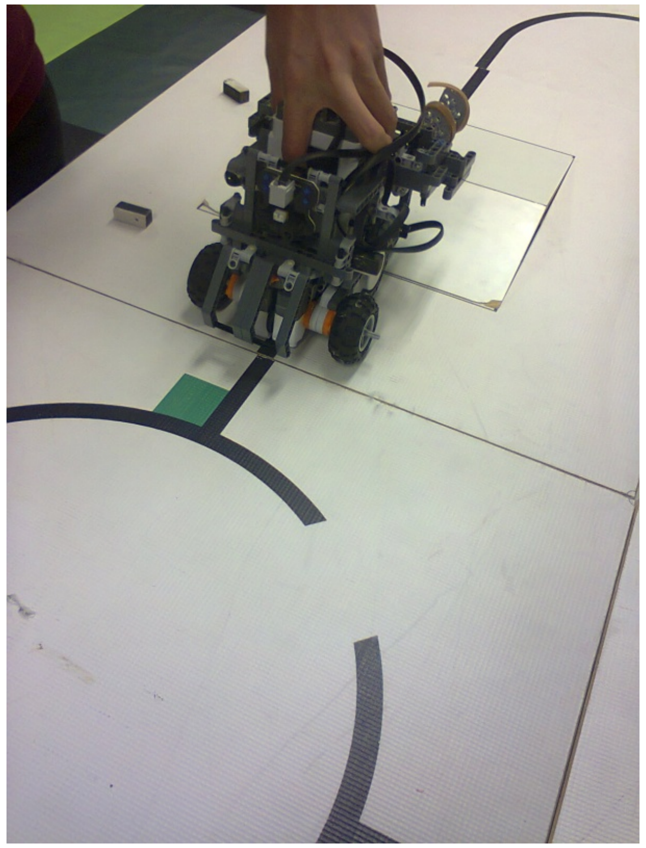
Jackson helped optimise the code by contracting or removing unnecessary parts of the code. For instance the previous design was changed-

OLD CODE

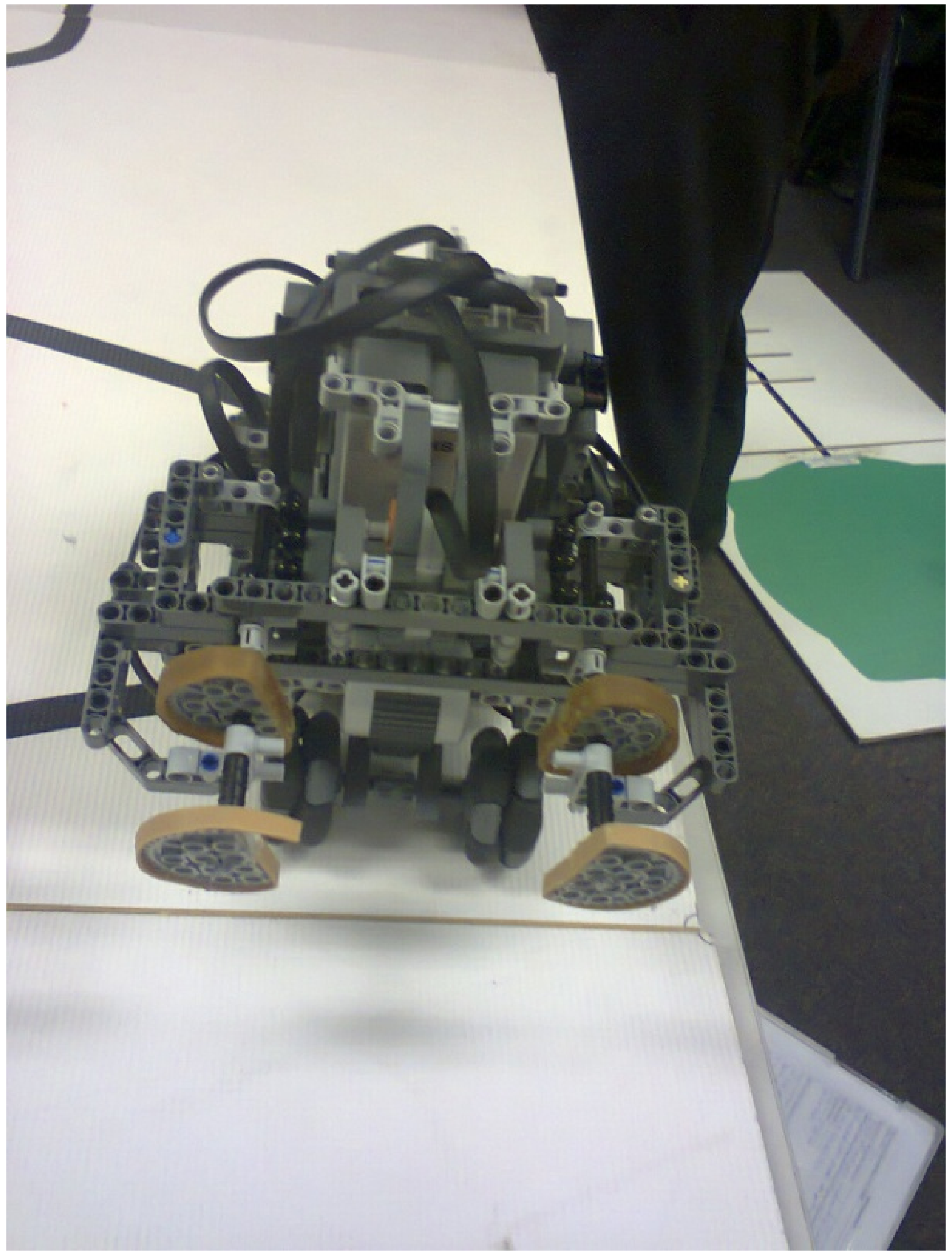
```
BEGIN LineCode
	IF RightLightSensorValue=Black AND LeftLightSensorValue=White THEN
		RightMotor=Backward
		LeftMotor=Forward
	END IF
	IF RightLightSensorValue=White AND LeftLightSensorValue=Black THEN
		RightMotor=Forward
		LeftMotor=Backward
	END IF
	IF RightLightSensorValue=White AND LeftLightSensorValue=White THEN
		RightMotor=Forward
		LeftMotor=Forward
	END IF
	IF RightLightSensorValue=Black AND LeftLightSensorValue=Black THEN
		RightMotor=Forward
		LeftMotor=Forward
	END IF
	IF RightLightSensorValue=Green THEN
		LeftMotor=Forward
	END IF
	IF LeftLightSensorValue=Green THEN
		RightMotor=Forward
	END IF
```

```
BEGIN LineCode
	IF RightLightSensorValue=Black AND LeftLightSensorValue=White THEN
		RightMotor=Backward
		LeftMotor=Forward
	ELSE IF RightLightSensorValue=White AND LeftLightSensorValue=Black THEN
		RightMotor=Forward
		LeftMotor=Backward
	ELSE IF RightLightSensorValue=White AND LeftLightSensorValue=White THEN
		RightMotor=Forward
		LeftMotor=Forward
	ELSE IF RightLightSensorValue=Black AND LeftLightSensorValue=Black THEN
		RightMotor=Forward
		LeftMotor=Forward
	ELSE IF RightLightSensorValue=Green THEN
		LeftMotor=Forward
	ELSE IF LeftLightSensorValue=Green THEN
		RightMotor=Forward
	END IF
```

This new code forgoes reading subsequent lines of code if any previous IF statement was true.



Evaluation



Since the logbook is to be submitted before our robot is to be finished, this evaluation only applies to robot and code's state as of Term 3 Week 5. The logbook will continue to be updated after this submission.

Statement of Compliance with Competition Rules

Jason and Andrew have competed in the RoboCup for at least two years and hence, this team will be entering in the Open Rescue division. We are prepared to deal with slight imperfections in the tiles and unideal lighting conditions.

The robot is to be started by humans and will act autonomously thereafter. Our robot has a demonstrable mechanism by which to rescue the victim. Our robot is constructed from mostly the efforts of the team and is not a replica of any previous competing robot.

The robot has no means to damage or interfere with the performance of other robots. Our mentor has played no part in the programming of our robot.

Evaluation

- For the robot:
    - The robot's mass is not evenly distributed; the lifting mechanism weighs a significant portion of the robot
        - Hence, a wheel occasionally skids when ascending slopes
    - The light sensors' close proximity to the ground and light shields help inhibit external light interference
    - The touch sensor can be activated without difficulty
        - The water tower is only moved slightly
    - The distance sensor is positioned without obstructions
    - The cog wheels can (unreliably) pick up the can
        - The team reverted to using cog wheels from the claw mechanism as the latter was found to be too complex to be implemented
    - The size of the robot is small enough to be able to move through doorways and tunnels
    - The robot is robust
- For the code:
    - The green, white, black and silver colour values will need to be calibrated for changing environments
    - The robot can manoeuvre along black lines
    - The robot occasionally can manoeuvre through a green turning point
        - It has not succeeded manoeuvring past the gridlock
    - The robot navigates around the bottle once the touch sensor is activated in a circular motion
    - The code for rescuing the victim after the Spill Access Point is yet to be implemented