

Ethan Furbert

5/03/2024

Ms. Diyva

Integrating Project

Lego Spike Demo Report



Table of Contents

Introduction	3
--------------------	---

Lego Spike Program	3
Goal / Expectations / Perimeters	3
The Build	4
Motors	4
Sensors	4
Code / Explanation of each Branch of the code	7
“Main Program”	8
“Following Line”	9
“Obstacle”	10
“Maze”	11
Tasks to Complete during the Demo = Solution / Code	12
Following the Line	13
Identifying / Maneuvering around the Obstacle	13
Maze	13
All the changes / implementations to build / code	14
Conclusion	14
Sources	15

Introduction

In this report, our assignment is on trying to build and code a robot to be able to successfully complete the designated course situated. While also being able to maneuver around obstacles without any major inconveniences. We've done the project in our respective groups or 3-4.

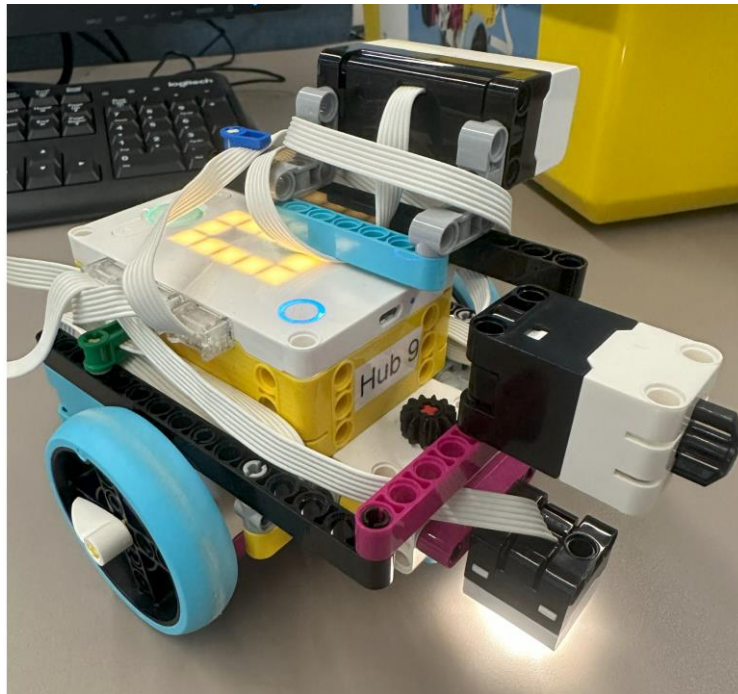
Lego Spike Program

The LEGO Spike program is an educational initiative designed to engage students in STEAM (Science, Technology, Engineering, Arts, and Mathematics) learning through hands-on building and coding experiences. It combines LEGO bricks with programmable electronic components, sensors, and motors, allowing users to create interactive projects and robots. With a focus on creativity, problem-solving, and collaboration, the Spike program encourages students to explore and understand complex concepts in a fun and intuitive way. It is commonly used in schools and educational settings to foster skills in technology and engineering while promoting teamwork and innovation.

Goal / Expectations / Perimeters

The goal for the groups was to build a robot that followed the past exercises in one full scope, such as the "Follow the Line Bot", "Bot that can Maneuver around Obstacles", and "Maze Solving Robot". Which specifically was to build a robot capable of exploring, following, maneuvering around and solving the maze autonomously using sensors and algorithms to navigate through dead ends and find the correct path to the exit. The Robot at the end of the demonstration should be able to successfully follow all perimeters.

The Build



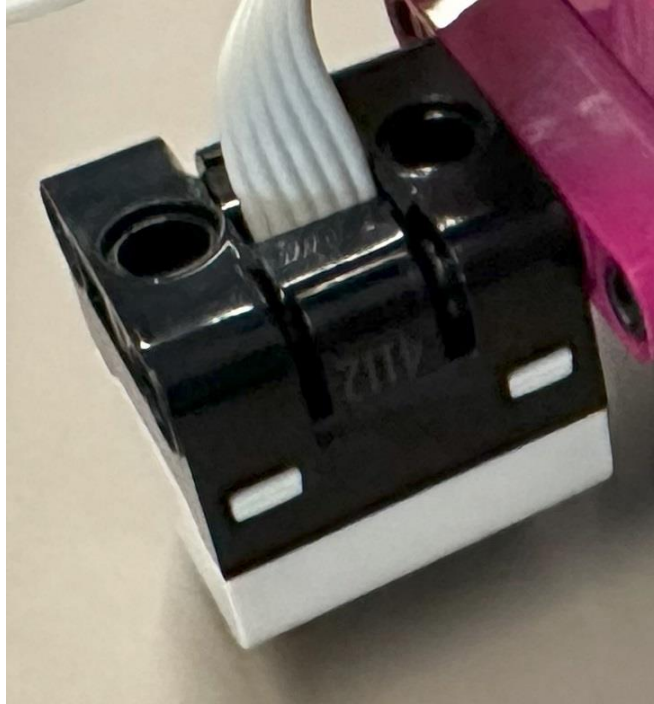
The building the group has come up with includes multiple sensors with the base of the structure coming from a past exercise called the “Driving Base”. The building is supposed to epitomize a vehicle or car in general. With additional Lego pieces to properly place certain sensors / motors in their optimal areas that can further our development in reaching our robot’s goal.

Motors

Motors were used in this building so the base or building can be able to operate as a vehicle / car and ultimately maneuver around the course.

Sensors

The sensors are specific pieces that help the static build move / operate accordingly depending on the extensions given to them.



The Color Sensor – A sensor that can recognize color

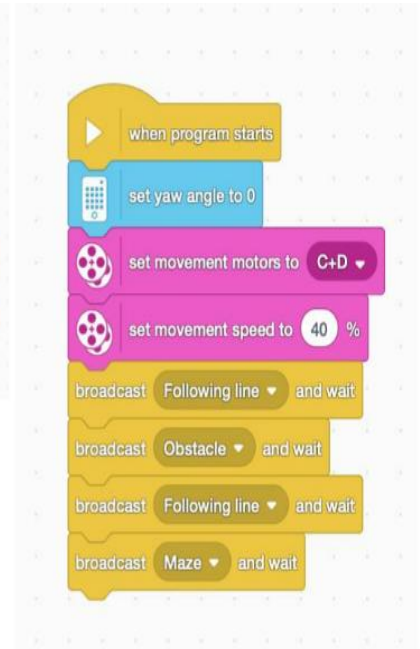
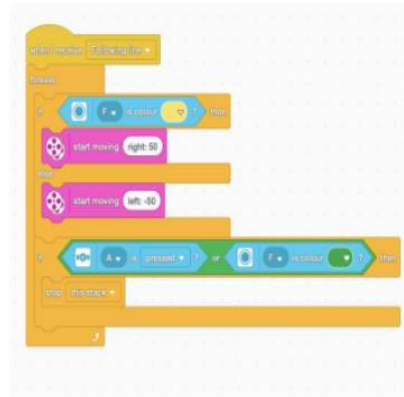
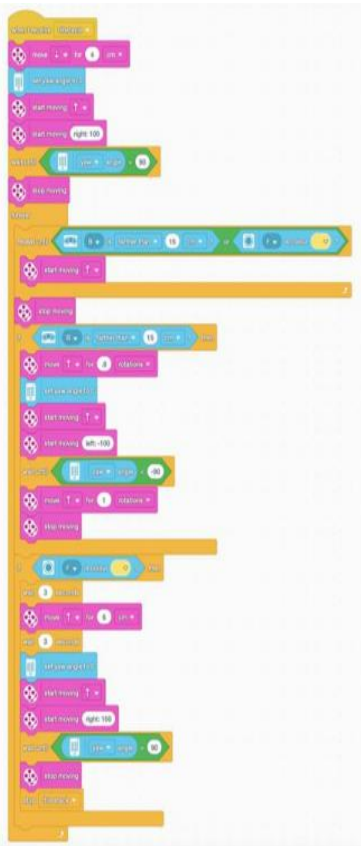


The Force Sensor – A sensor that sense force or pressure being applied

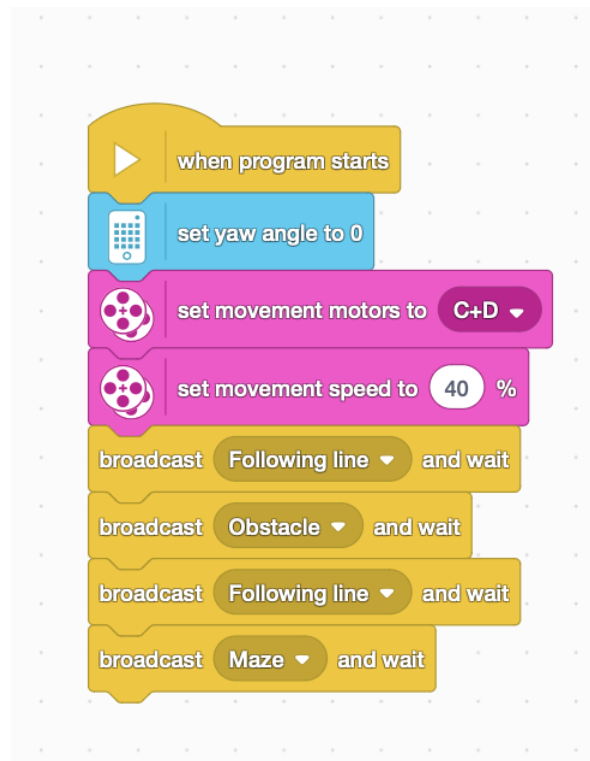


The Distance Sensor – A sensor that can gauge the distance nearby objects within its sight

Code / Explanation of each Branch of the code



“Main Program”



The program starts here

Set its origin (yaw angle)

Sets the motors its respective ports

Setting the movement speed (40%)

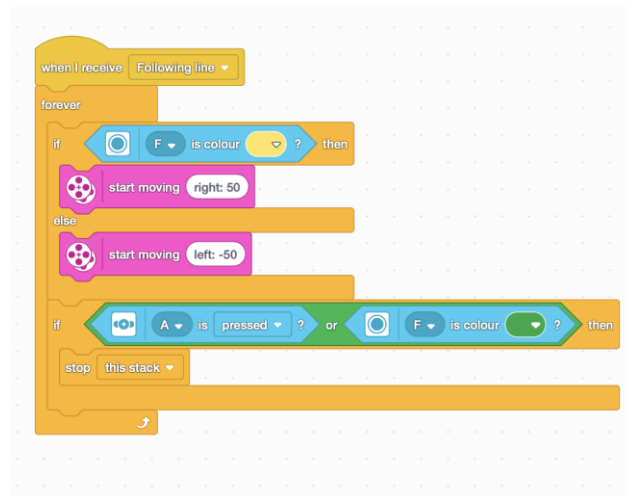
Jumps to “Following line” branch

Jumps to “Obstacle” branch

Jumps to “Following line” branch

Jumps to “Maze” branch

“Following Line”



Blocks start, coming from the “Main Program”

Forever – Its looping everything inside

If-statement = when the color sensor picks up said color (yellow)

If the sensor picks up the designated color, it commands motor to turn right

If the sensor doesn't pick up the designated color, it commands motor to turn left

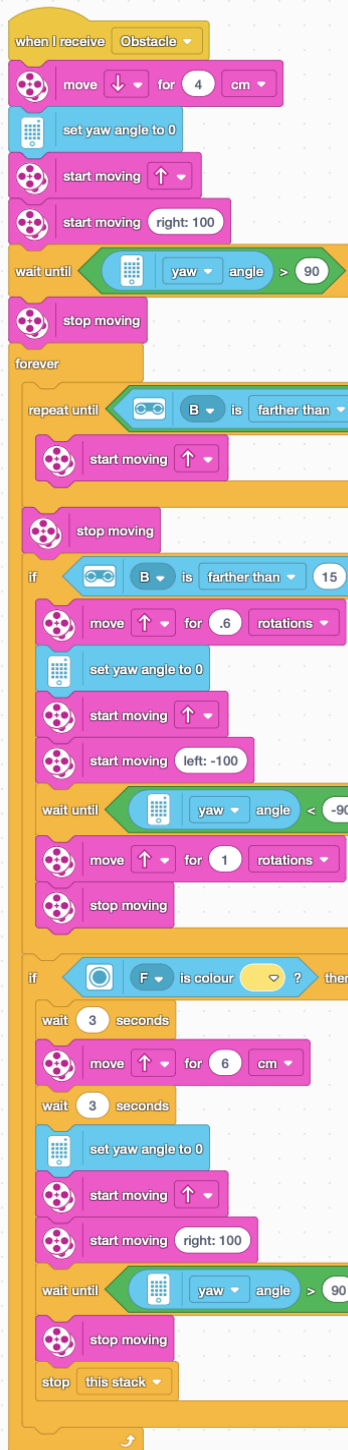
(This loop should warrant the robot to move like a snake)

If-statement 2 = when the force sensor is pressed, or the color sensor picks up said color
(green)

The current stack is stopped due to the conditions above being activated

(Loops, unless you the said condition were met to prevent such action)

“Obstacle”



Block starts, coming from the “Main Program”

Moves backwards (4 cm)

Sets origin

Moves forward

Turns right

(wait till condition) yaw angle less than 90

Stops moving

Loop

Repeat till distance sensor further than or color sensor finds color

Start moving

Stop moving

If-statement = distance sensor is further than

Motors move forward a bit

Yaw angle is resetted

Set to move forward

Turns left

(wait till condition) yaw angle less than 90

Moves forward

Stops moving

If-statement = color sensor picks up (yellow)

Wait for 3 secs

Moves forward

Wait for 3 secs

Yaw angle resetted

Moves forward

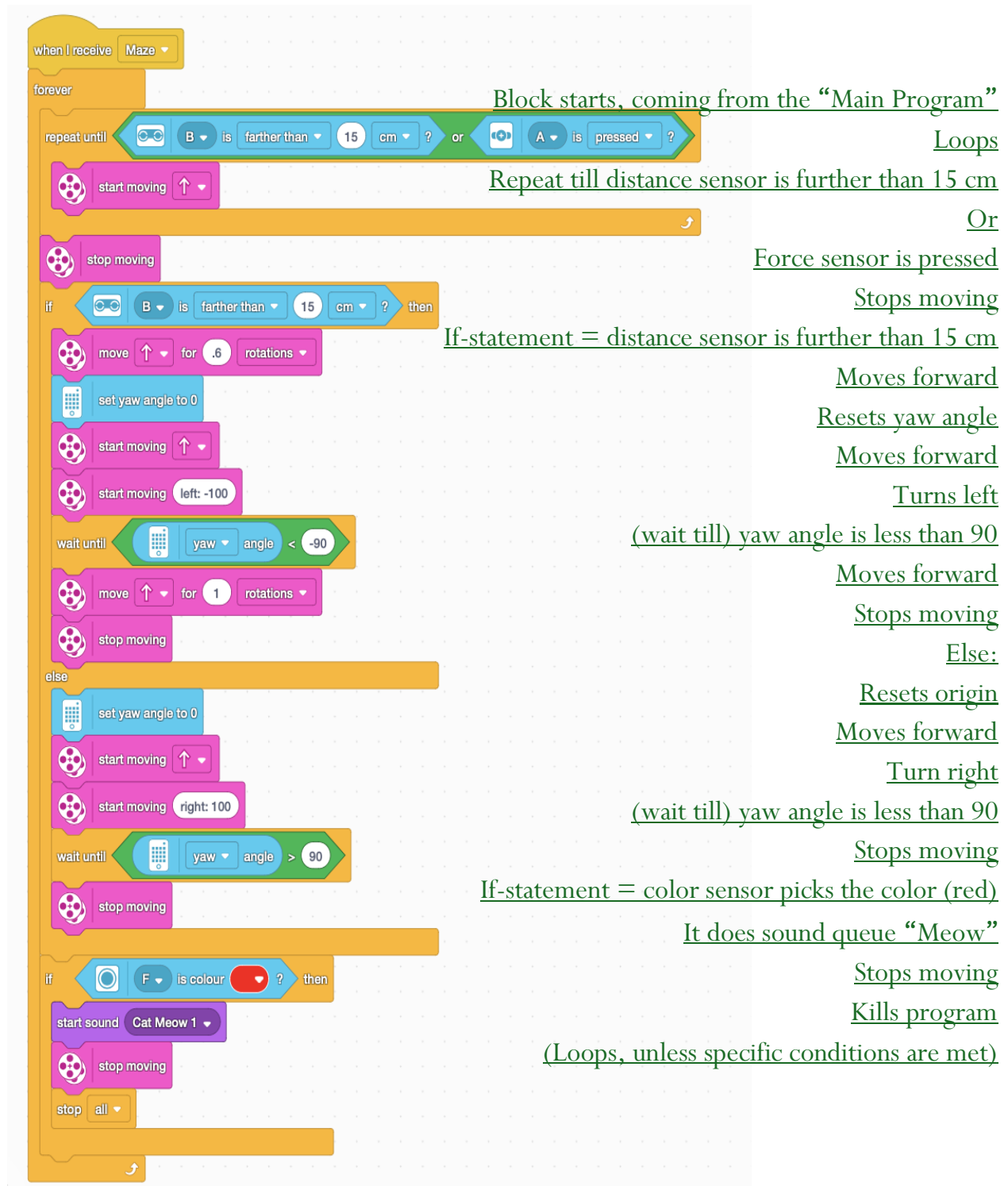
Turns right

Wait till yaw angle is less than 90

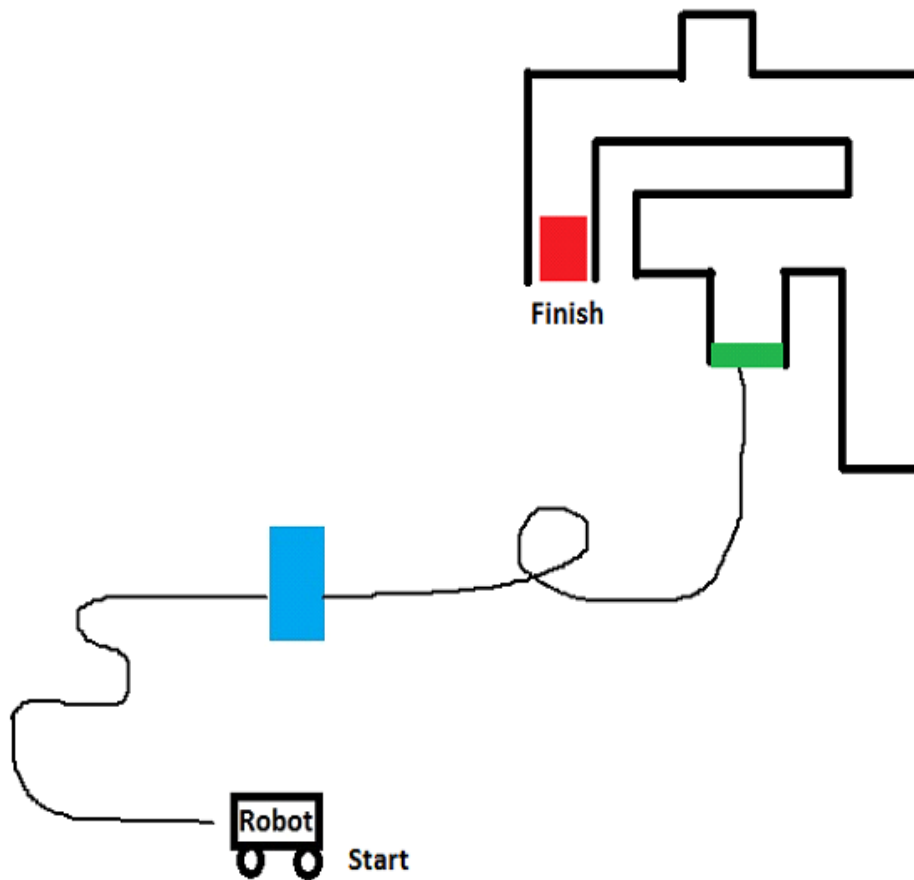
Stops moving

Stops branch

“Maze”



Tasks to Complete during the Demo = Solution / Code



Following the Line

At the beginning of the demonstration course, the robot is expected to follow the track/line to reach its next goal/ destination.

What our group decided to do to successfully complete this task was to have the robot use its color sensor to follow the track / line's specific color. By setting the color sensor to face the track / ground, it allowed the robot to efficiently pickup said color and help enable the robot to follow the line.

Identifying / Maneuvering around the Obstacle

While the robot is following the track / line, at some point in its journey there will be an obstacle for it to overcome.

We decided to sort of set a pre-determined code for the robot to make the necessary steps / movement distances to bypass whatever object was in front of it. Using the force sensor to initiate pre-determined code, and using the distance sensor which was placed on the side of the robot was used to scale / follow alongside the object to then later find its way back on the track / line to continue the "follow the line" branch. Main reason for this was that us, as the demonstrators, could decide/ choose the obstacle.

Maze

The final part of the robot demonstration, the maze is to force the robot into automatically trying to figure out a solution / find the exit.

The decision to make the robot autonomously move was standard procedure, it required all sensors. Firstly, the color sensor would be needed to initiate the "maze" branch due to at the start of the maze, there will be a green color platform to signify the entry way and later at the exit of end of such maze will be a red platform. Throughout the maze, the robot will utilize both the distance and force sensor to dictate which direction it should turn, this dependent on the distance of said wall to robot.

All the changes / implementations to build / code

Throughout this process, of learning the code style being word block and the use of each Lego piece there was going to be some struggles along the way.

The first bump was trying to figure out the necessary word blocks to complete some of the more rudimentary tasks, such as the identification of the color line or track to be specific. The realization of specific colors not accurately being picked up by the color sensor showed that outside conditions would be the hardest issues to identify. At that time, we drastically didn't really believe in our own code yet to believe in working appropriately in the first place.

After a while we've gotten a handle on how to code decently well, but the code seemed kind of untidy. So, the use of broadcasting / branches really helped us figure out which code was for.

The next stage was deciding on the going for an autonomous route or pre-determined code. It was quite easy to settle on a code that would automatically attempt to find the solution in said problem but at the same time, if we manage to have the specs of said object / obstacle it would also be straightforward to code the specifics to bypass the situation easily too. We decided it was best to go for the autonomous route due to the possibility of not knowing said demonstration quota till a later date, specifically for the maze. The obstacle has an essence of being able to solve the obstacle autonomously, but we needed it line up properly when evading such object to read the track again to fall back to the "follow the line" branch.

The last change was due to the size of the robot, since we were unsure of the overall size of the maze, we've decided to shorten the width of the bot to allocate more space for clearing turns while in an enclosed space.

Conclusion

In the end building and coding a Lego Spike robot involves combining creativity and technology to create interactive projects. It provides hands-on experience in robotics, engineering, and programming. The process involves assembling the robot using Lego Spike Prime components and then coding its behavior using the Spike app or other compatible

software. This experience helps us develop problem-solving skills, teamwork, and a basic understanding of coding principles, especially in group framework. Ultimately, it helps have a better understanding of how stressful and pointless a code can be if human error isn't accounted for.

Sources

“SPIKE Prime | Key Stage 2&3 STEAM Learning,” *LEGO® Education*.

<https://education.lego.com/en-gb/products/lego-education-spike-prime-set/45678/>

- Driving Base – for the Build
- Stopping at the line – for the color sensor / end of demonstration
- Follow the line – for the color sensor / 1st task
- Baseball runner – for the movement of the robot