LESSON 41

Line Following Car

60 minutes | Designing and Building for the Future

WHAT THIS LESSON DOES

In this step-by-step lesson, you'll learn to program a Move Motor Car to follow a line track using a Microbit. You'll create a new project, add an extension, create variables, and set up LEDs. You'll then program the car to turn right, left, and move forward. Finally, you'll test your car on a track and tweak the code for optimal performance.

WHAT STUDENTS SHOULD BE ABLE TO DO

- Understand and apply the concept of programming a Microbit to control a Move Motor Car.
- Create and manipulate variables to store sensor values and control the car's movements.
- Implement conditional logic to guide the car's movements based on sensor readings.
- Use LEDs for visual feedback and enhance the functionality of the car.
- Experiment with code modifications to optimise the car's performance on different tracks.

BEFORE THE LESSON

- Confirm classroom access to https://makecode.microbit.org and that Microbits can be flashed over USB.
- Per pair or group: a Microbit, a Move Motor car with 4 AA batteries, and a USB cable.
- Print tracks from mbmmlfc tiles.pdf, or provide paper and thick black markers so lines are clear and bold.
- Test one car on a sample track so batteries, sensors and motors work before pupils begin.

Line Following Car

HOW THE LESSON RUNS

Introduction
Create a new project
Add the extension
Create the variables
Setup the LEDs
Turn right
Turn left
Go forwards
Test it on a track
Tweak the code

TEACHING MOVES

- 1. Introduction. Show the Microbit, Move Motor car, USB cable and track. State the goal: sensors read the line and the code steers the car along it.
- 2. Create a new project. Open makecode.microbit.org together and have every pair create a new project with a clear name before adding any blocks.
- 3. Add the extension. Demonstrate adding the kitronik-move-motor extension. Check that MOVE Motor blocks appear in the toolbox before moving on.
- 4. Create the variables. Create left, right and difference with the class. Stress left and right hold sensor brightness and difference is left minus right.
- 5. Setup the LEDs. Add the LED setup so colour can show whether the car is going straight, turning left or turning right.
- 6. Turn right. When difference is greater than 10, the left sensor is darker: stop the right motor, spin the left, set LEDs yellow.
- 7. Turn left. In the else if, when difference is less than -10, stop the left motor, spin the right, set LEDs red.
- 8. Go forwards. In the else branch, similar sensor readings mean drive both motors forwards so the car stays on the line.
- 9. Test it on a track. Download to the Microbit, seat it in the car, centre the line between the two sensors, then switch on and watch.
- 10. Tweak the code. Once it follows reliably, raise speed, time laps and try fiddlier tracks. Aim for faster runs without leaving the line.

WHAT TO WATCH FOR

- Pupils reverse the turn logic, turning left when the left sensor is darker and right when the right sensor is darker. Retell it: the darker sensor is over the line, so steer toward the brighter side. Walk through one greater-than-10 and one less-than--10 case on the board.
- They place the car with both sensors on the same side of the line, so readings never balance and it spins or leaves the track. Before powering on, check the line sits in the middle under the car with one sensor either side.
- They treat left, right and difference as set once, not updated every loop, so they expect the car to decide only at the start. Point at the forever block and say the three values refresh continuously as the car moves.

DIFFERENTIATION

- Emerging: Pair them with a partner who drives MakeCode while they handle the car, track and sensor placement. Stay with them for extension add and the three variable names before any motor blocks.
- Developing: Ask them to explain each branch (turn right, turn left, forwards) aloud before downloading. Have them test on a simple straight or gentle curve before a full loop.
- Proficient: Challenge them to raise motor speed and time laps without leaving the line. Invite more complex tracks and small code changes to keep the car stable on tighter bends.