

Robotics: Follow a Line with One Sensor

Ages 10-14 · about 45 minutes, in one sitting

Your child runs a simulated robot that follows a black line using one light sensor, watches the sensor's number change between the line and the white mat, and finds the simple rule in the program that steers it. Then they predict what a gentler turn will do, test it, and test more values to find the one that works best.

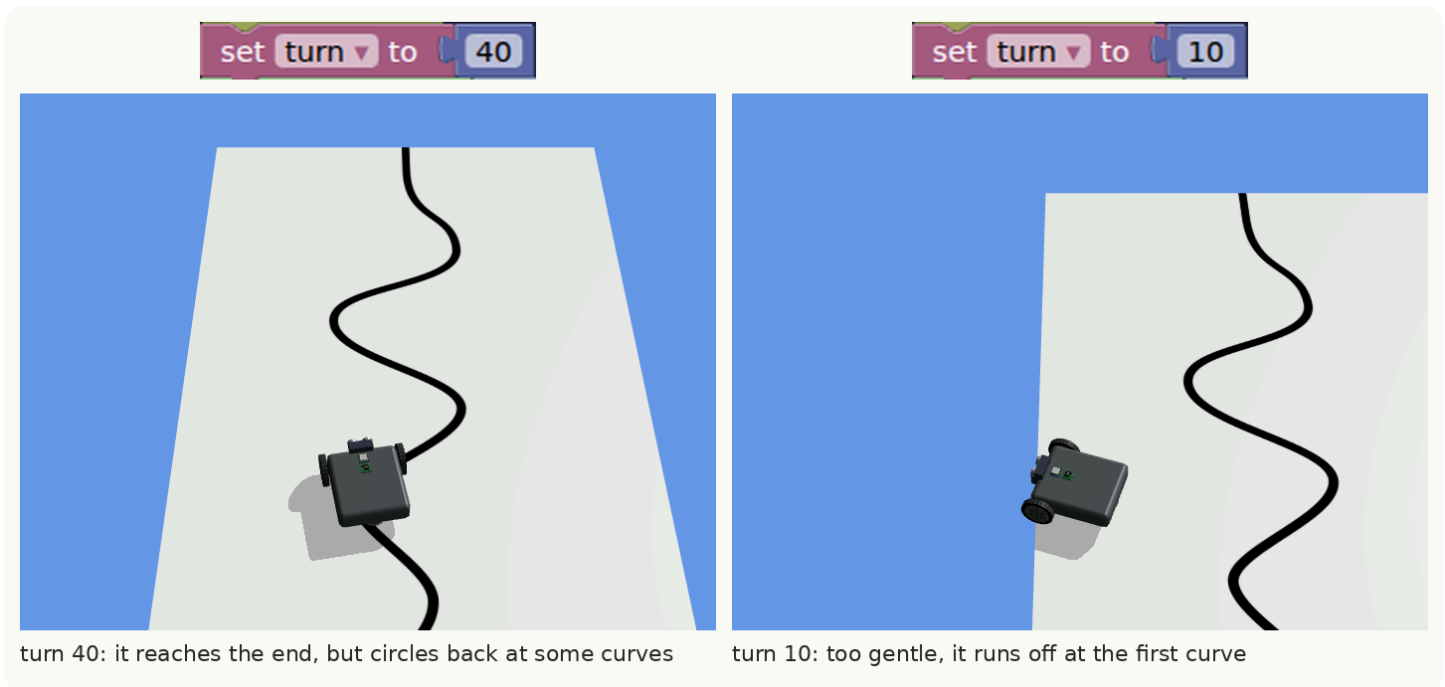
What you need

- A computer or Chromebook with a current browser. It draws in 3D, so an older computer runs it more slowly, and the robot may wobble more on it.
- The lesson link: mytekdev.com/try/robotics?lesson=follow-a-line-with-one-sensor&ref=fam-robotics-1. No account, no email and nothing to install.
- Optional: the activity sheet at the end of this plan, printed, and a pencil, for writing each prediction down before Run.

Before you start: try it once yourself

Run the robot and make the lesson's change yourself once before you start, about ten minutes.

1. **Open the lesson.** Go to mytekdev.com/try/robotics?lesson=follow-a-line-with-one-sensor&ref=fam-robotics-1. The app opens on the Simulator. Press Run, the play button at the top left. The robot follows the black line to the far end of the mat and stops.
2. **Watch the sensor.** Press reset, the circle arrow next to Run. Open the sensor readings with the sensors button at the top right (it looks like a page of lines, just left of the camera button), and press Run. Under `in1: Color` `Sensor`, `Intensity (%)` drops to the teens over the line and rises to about 100 over the mat.
3. **Read the program.** Open `Blocks`: set turn to 40, then repeat while gps y is less than 90, with an if-else inside it that compares color_sensor reflected light intensity to 58.
4. **Make it fail.** Change 40 to 10. Go back to `Simulator`, press reset, then Run. The robot runs off the mat at the first curve. Set it back to 40.
5. **Try one class value.** Try 80 the same way, so you have seen one of the values your child will test.



Turn 40 reaches the end, circling back to the line at some curves. Turn 10 is too gentle and runs off the mat at the first one.

Run and reset are only on the Simulator tab, so after every change in Blocks, you have to switch back. Run starts from wherever the robot is now, so a robot that was not reset starts at the far end or off the mat. Typing in the Python tab brings up a warning that Python changes cannot be converted back into blocks; after that, Run uses the Python and ignores changes to the blocks.

The ideas, in plain words

The robot has one light sensor under its front, pointing down. It shines on the mat and measures how much light comes back: black reflects little, so the reading is low (about 16), and white reflects a lot (about 100). The program picks a cut-off between them, 58. Below it counts as the line, and above it counts as the mat. That cut-off is called a threshold.

One sensor cannot tell which side of the line the robot is on, only whether it is over dark or light. So the program follows the edge of the line: over dark it steers one way, over light it steers the other way, and it checks again every few milliseconds. The robot never drives straight. It zigzags along the edge, correcting each time it crosses. The turn value is how hard each correction is. Too gentle, and the robot cannot keep up with a bend. Too hard, and the robot spends its time turning instead of driving: at 80 it takes about two minutes to reach the end, and at 100 the wheels turn opposite ways and it only spins in place.

How often the robot checks matters too. In this simulator, the robot reads its sensor once each time the screen is drawn, so a slower computer checks less often, and the robot drives further between corrections. On this line the difference is small: in our tests, each turn value in this plan gave the same result on fast and slow computers, and

only the times changed. Real robots have the same problem, which is why engineers test a program on more than one machine before they trust a number.

YOUR CHILD MIGHT ASK	A GOOD ANSWER
Why does it wiggle instead of driving straight?	It only knows dark or light. Each time it checks, it turns toward the other side, so it keeps crossing the edge of the line.
Why 58?	It sits between the line's reading, about 16, and the mat's, about 100. Readings in between happen when the sensor is half over the edge.
How does it know when to stop?	The loop repeats while the GPS y is less than 90. 90 is just short of the far end of the mat, so the loop ends there and the robot stops.
Why did my result not match the one in this plan?	First check that you both pressed reset before Run. Then check that neither of you typed in the Python tab, which makes Run ignore the blocks. In our tests, every value in the table gave the same result on fast and slow computers.

Do it together

1. Open mytekdev.com/try/robotics?lesson=follow-a-line-with-one-sensor&ref=fam-robotics-1 together. The app opens on the Simulator with a robot at the bottom of a white mat with a curvy black line. A short guide runs along the bottom; follow it together. It has your child press Run, open the sensor readings, and then look in Blocks. Then ask: the robot cannot see the line ahead of it, so how does it stay on the line?

You'll know it worked when: The robot follows the line to the far end of the mat and stops.

2. Go back to **Simulator**, press reset (the circle arrow next to Run), make sure the sensor readings are open (the sensors button at the top right, which looks like a page of lines), and press Run. Watch **Intensity (%)** under **in1: Color Sensor**.

You'll know it worked when: The number drops to the teens over the black line and rises to about 100 over the white mat.

3. Open **Blocks** and read the program together from the top: set turn to 40; repeat while gps y is less than 90, which is near the far end of the mat; if color_sensor reflected light intensity is less than 58, move steering with direction turn, else the same block with minus turn; then sleep for 10 milliseconds. The robot only ever knows dark or light, so it zigzags along the edge of the line.

You'll know it worked when: Your child can say what the robot does when the reading is below 58 and when it is above.

4. Predict, then test. Change set turn to 40 to 10. Before Run, your child writes or says what will happen. Then go to **Simulator**, press reset, then Run. Set it back to 40 afterward.

If you get stuck: Run and reset are only on the Simulator tab, so after every change in Blocks, switch back. Run starts from wherever the robot is now, so always press reset first.

You'll know it worked when: The robot runs off the mat at the first curve: its turns are too gentle to follow the bend.

5. Test more values: 20, 30, 50, 60, 70 and 80, one at a time. For each, predict first, then reset and Run, and let it go until the robot stops at the end or leaves the mat (up to about two minutes). Write down one of three results: reached the end, left the line and came back, or ran off the mat.

If you get stuck: Typing in the Python tab brings up a warning, and after that Run ignores the blocks. Keep changes in Blocks.

You'll know it worked when: A filled-in table, with a prediction written before every run.

6. Finish with one question: which turn value would you choose, and what is your evidence? Then: on a slower computer, the robot checks its sensor less often. How might that change your answer?

You'll know it worked when: Your child picks a value and points to the results that support it, and says that a robot that checks less often drives further between corrections.

Talk about it

Ask these while you work together. A child who has understood the lesson can answer them without help:

- The robot has one sensor. How does it know which way to turn?
- Why does it zigzag instead of driving straight along the line?
- Turn 10 missed the first curve. Why?
- Where does the number 58 come from?
- Two groups got different results for the same value. What would you check first?

The last one is the real test. A child who checks how a test was run (was the robot reset, were the blocks changed) before blaming the program is thinking like a robotics engineer.

For younger and older kids

Younger: For ages 8 and 9, do the turn 10 test side by side, test two more values together, and save the rest for another day.

Older: For teens, work in the Python tab, test the cut-off (try 90 and 30) as well as the turn value, and explain in a few sentences why a robot that checks less often needs gentler turns.

Keep going

- Change the cut-off from 58 to 90, predict what happens, and test it. Then try 30. Which readings count as the line now?
- Open the Worlds menu, choose Select World, pick Sharp Turns under Select Challenge, and press Ok. Before running, predict whether your chosen turn value will still reach the end, then test it.
- Open the Python tab and find the line that is the if block. Explain each part of it in your own words.

Bring it into other subjects

- Math: the line reads about 16 and the mat about 100. What number is exactly halfway? (58.) Why might a program use the halfway point as its cut-off? (A reading half over the edge lands near the middle.)
- Science: black things absorb more light than white things. Where else does a machine use reflected light? (Answers vary: barcode scanners, automatic doors, the sensor that turns on street lights at dusk.)
- Engineering: real line-following robots often use two sensors, one each side of the line. What could a robot learn from two sensors that it cannot learn from one? (Which side of the line it has drifted to.)
- Computer science: open the Python tab. The if block is one line that starts with if. Find it, and find the 58 in it.

For your records

Ready to copy into an hour log, a portfolio or a transcript.

LESSON	Robotics: Follow a Line with One Sensor
SUBJECTS	Computer science, Engineering
TIME	45 minutes
WHAT WE DID	Ran a simulated line-following robot, read its light sensor's values over the line and the mat, and explained the if-else rule inside a loop that steers it. Predicted and tested how changing the turn value changes the robot's path, recorded each result, and chose a value from the evidence.
SKILLS PRACTICED	sensors and thresholds; conditionals inside a loop; predicting before running; fair testing and recording results

Activity sheet (optional)

Name: _____ Date: _____



Open the app: scan the QR code, or go to mytekdev.com/try/robotics?lesson=follow-a-line-with-one-sensor

Essential question: How can a robot with one sensor stay on a line it cannot see ahead of?

1. **Run it.** Press Run, the play button. Watch the robot follow the line.
2. **Read the sensor.** Open the sensor readings. Intensity over the black line: _____ over the white mat: _____
3. **Reset first.** Before every Run, press reset, the circle arrow, so the robot starts over.
4. **Predict, then test.** For each turn value, write your prediction first. Then reset, Run, and write what happened: reached the end, left the line and came back, or ran off the mat.

TURN VALUE	I PREDICT	WHAT HAPPENED
10		
My first value: ____		
My second value: ____		

1. Which turn value would you choose? What results from your table support it?

2. Why does the robot zigzag instead of driving straight?

Quick check

Name: _____

1. The light sensor reads 20. Where is it most likely to be? (A) over the black line, (B) over the white mat, (C) you cannot tell.
 2. The program says: if intensity is less than 58, steer one way, else steer the other way. Why does the robot zigzag instead of driving straight? (A) the line is crooked, (B) it only knows dark or light, so it keeps turning back across the edge, (C) the motors are uneven.
 3. With turn set to 10, the robot ran off the mat at the first curve. What would you try next, and why?
-
-

Answers

Measured in the app: the color sensor's intensity reads about 16 over the black line and about 100 over the white mat. With turn 10 the robot runs off the mat at the first curve; with turn 40 it reaches the end of the line.

QUESTION	ANSWER	WHY
1	A	20 is close to the line's reading, about 16, and far below the cut-off of 58.
2	B	One sensor only tells dark from light. Each check turns the robot back across the edge, so it zigzags along it.
3	A bigger turn value, such as 40, because 10 turns too gently to follow the bend	A good answer names a larger value and a reason about how hard the robot turns.

Want the whole subject taught?

This is one lesson to do together. If you would rather have the whole subject taught, our Information Technology 1 and Information Technology 2 courses teach circuits and robotics as part of a full course, with an instructor, live and recorded classes, graded work and credit.

If your state or umbrella school asks, this lesson covers CSTA 2017 1B-AP-15, CSTA 2017 2-AP-17, NGSS 3-5-ETS1-3.

The Robotics simulator is built on GEARS (GearsBot) by Cort Wee, open source under the GNU General Public License 3.0. Its blocks are Blockly, its 3D view is Babylon.js and its Python runs in Skulpt.

Terms of use

This lesson was made by mytekDev. You may print and copy it for the children you teach at home, and change it for them. You may not sell it, share it with other families, or post it publicly.

The app this lesson uses is free and runs in the browser, with nothing to install and no account needed. It is at mytekdev.com.

© 2026 mytekDev. All rights reserved.