Ozobot Follows a Path Based on an LLM Bot and Camera Image

Ozobot Editor Lesson—Teacher's Guide

by Richard Born, Associate Professor Emeritus, Northern Illinois University

Introduction

If you haven't already done so, you should read the student guide that accompanies this lesson. It provides all of the information needed by the student to thoroughly understand how to get Ozobot to follow a path based on an LLM Bot and camera image of the path. Students also learn how to parse a string using appropriate blocks from the "Text" category of blocks. The entire Ozobot Editor program, with detailed comments, is presented to the student in the student guide.

The **share code** for this program is **n2r95aw**. You may elect to share this code with your students, particularly if classroom time is an issue. Alternatively, the students can create the entire program by entering blocks since an image of the program is included in the student guide. A ten-question multiple choice quiz accompanies this lesson. You may want to consider providing students with the share code if they can ace the quiz. *[This quiz was created by supplying a Google Gemini LLM model with the entire student guide, minus the program blocks.]*

Computer Science Programming Concepts in This Lesson

- The use of variables to hold the values of program constants
- Subroutines to break up code into logical segments
- The use of loops to handle tasks that must be repeated several times
- The use of conditionals, as needed, to control the flow of logic
- Subroutine and variable names that support self-documentation
- Extracting information from a string by parsing
- The use of LLM (Large Language Models)

Applicable CSTA (Computer Science Teachers Association) Standards

2-CS-02	Design projects that combine hardware and software components to collect and exchange data.
2-AP-11	Create clearly named variables that represent different data types and perform operations on their values.
2-AP-12	Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.
2-AP-13	Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.

2-AP-19	Document programs in order to make them easier to follow, test, and debug.
3A-DA-11	Create interactive data visualizations using software tools to help others better understand real-world phenomena.
3A-AP-17	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
3B-CS-02	Illustrate ways computing systems implement logic, input, and output through hardware components.
3B-AP-08	Describe how artificial intelligence drives many software and physical systems