Identifying Misplaced Shelf Items with an LLM Bot and Camera

Ozobot Editor Lesson—Student Guide

by Richard Born, Associate Professor Emeritus, Northern Illinois University

Introduction

An LLM equipped with a snapshot camera can significantly improve warehouse accuracy by analyzing real-time images of shelves. It detects discrepancies such as misplaced, missing, or incorrectly oriented items by comparing the captured visuals with the inventory database or predefined layout. When discrepancies are found, it can flag these issues and generate alerts or instructions for correction. This process streamlines inventory management, reduces human error, and enhances overall efficiency by ensuring items are correctly placed and easily retrievable, ultimately optimizing warehouse operations.

Workers at stores like Dollar Tree typically need to check shelves regularly, often daily, to ensure items are correctly located. This helps maintain organized displays, improves customer shopping experience, and prevents inventory issues. The exact frequency can vary depending on store policies, store size, and staffing, but daily shelf checks are common practice in retail to ensure proper merchandising and stock levels.

About this Lesson

In this lesson, you will deal with three shelves that might be found in a sporting goods store. The shelves contain soccer balls, footballs, and basketballs in that order from top to bottom, as shown in the Ozomap (labeled with X in the bottom left corner) of Figure 1. There are two similar Ozomaps labeled Y and Z that will be provided to you along with a larger copy of Ozomap X.

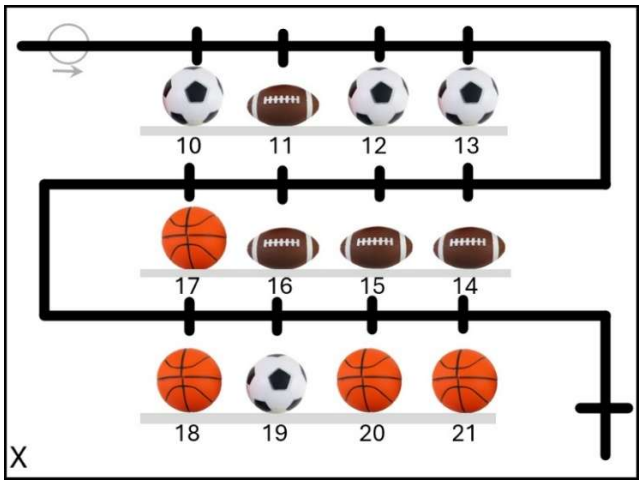


Figure 1

The shelves are shown as long horizontal gray lines, and product locations are indicated by the numbers 10 through 21 at the bottom of each shelf. You will note that each shelf has one item that appears to be misplaced. This will be the case in all three Ozomaps X, Y, and Z.

The goal of this lesson is to design an Ozobot Editor program that will accomplish the following two tasks:

1. Use an LLM Bot and connected camera, along with an appropriate prompt to the LLM, to determine the location of the misplaced item on each shelf.
2. Then place Ari or Evo at the start location in the upper-left corner of the Ozomap. Ozobot will then move from the start (10) to the end (21) locations, displaying a top green light at intersections if an item is not misplaced, and a top red light if an item is misplaced.

Example Video

Before we get started on these tasks, view the video provided by your teacher to see how Ozobot will traverse Ozomap X.

Accomplishing Task #1

Figure 1 shows the portion of the program that accomplishes the first task. As you can see, the LLM is asking for a string response, the variable *LLMresponse*, and using a camera. The first two paragraphs of the prompt describe major aspects of the image that are captured by the camera. You could probably get by without these, but it is always a good idea to set the stage by giving a *verbal* description of the image to help the *language* model understand the image. The **Google (mini)** LLM works well here and depletes LLM credits slowly.

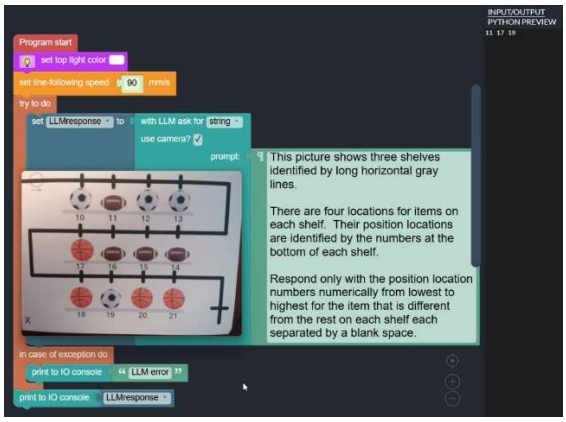


Figure 2

The final paragraph of the prompt is critical for two reasons. First, it specifies exactly what we want to know—the position location numbers for items that appear to be misplaced. Second, and most importantly, it tells the LLM how to format the response—numerically from lowest to highest, and with each number separated by a blank space. This is done so that the response will be in a format

that can be *easily dealt with* when programming the second task in which Ozobot traverses the Ozomap.

The first portion of the program ends by printing the LLM response to the IO console, as shown on the top-right side of Figure 2. The response for the image shown is the string ***11 17 19***, indicating the items at those three locations appear to be misplaced. We will want Ozobot to display a red top light at the intersection at those three locations.

One final note on Figure 1. The image of the Ozomap is a "Picture in picture". It is obtained by clicking on the *more options* vertical ellipsis at the far right of camera device window in the Ozobot Editor. The picture-in-picture view is quite helpful as it can be enlarged, dragged around in the Editor, and is a much clearer view of the image captured by the camera.

Accomplishing Task #2

The first thing that needs to be addressed in accomplishing the second task is to parse the LLM response string so that the three numbers can be extracted from the string. Let's look at the string ***11 17 19***, which we expect to get from Ozomap X, as shown in Figure 3.

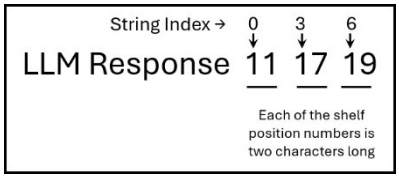
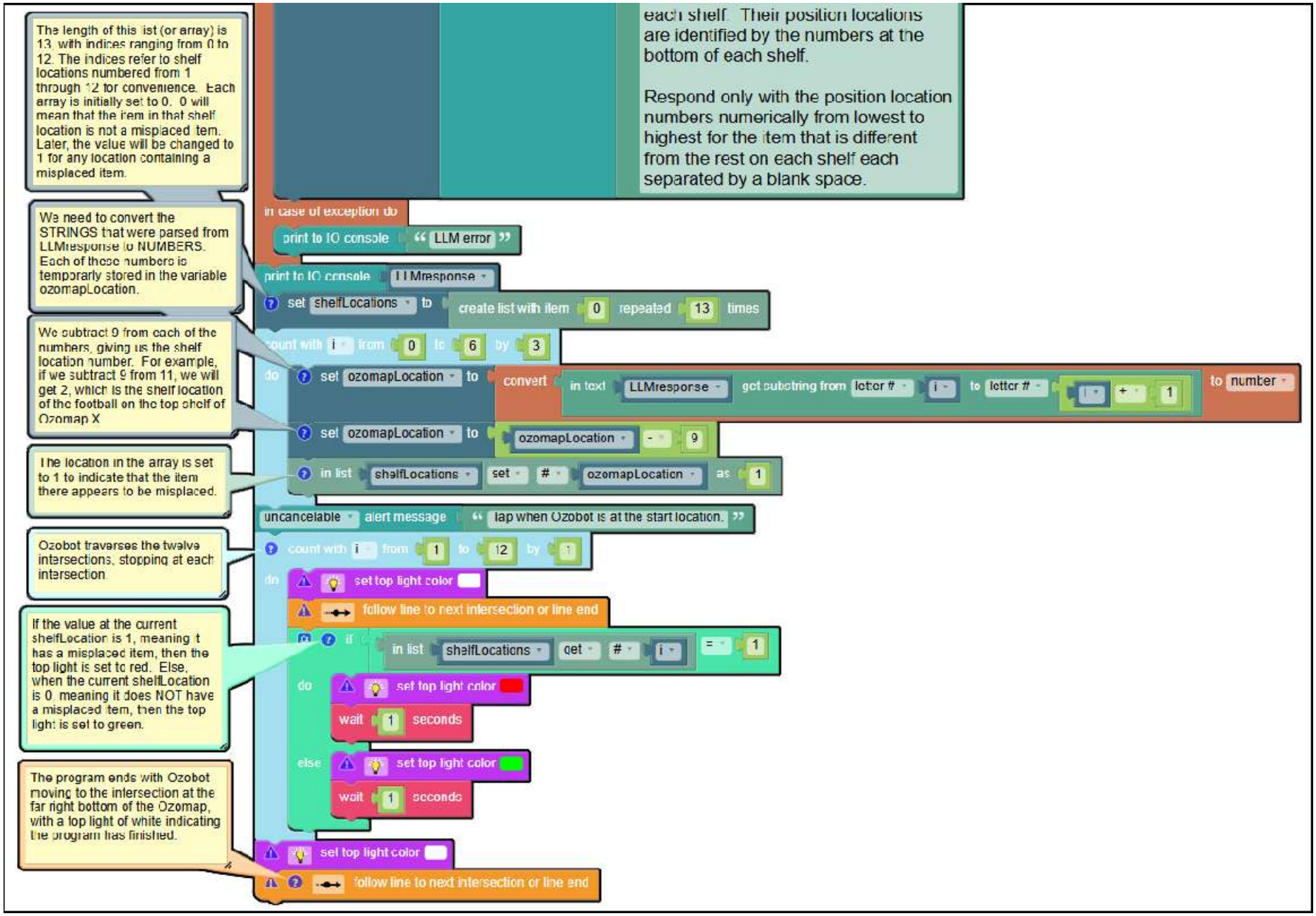


Figure 3

The indices of characters in a string start at 0. Therefore, the indices for the first character in each of the three position numbers are 0, 3, and 6. Each position number is two characters long. The perfect block to use for this situation is shown in Figure 4. The block would be in a loop in which the loop variable *i* varies from 0 to 6 by steps of 3. The first time through the loop the substring ***11*** would be parsed out, *i.e.,* characters with indices 0 and 1 from the LLMresponse. The second time through the loop the substring ***17*** would be parsed out, *i.e.,* characters with indices 3 and 4 from the LLMresponse. The third and final time through the loop the substring ***19*** would be parsed out, *i.e.,* characters with indices 6 and 7 from the LLMresponse.

Figure 4

With the above discussion on parsing complete, it is time to study the remainder of the program dealing with the second task. This is shown on the next page. Carefully read and study the comments to the left of the program. The comments detail what is happening in the blocks.


The length of this list (or array) is 13, with indices ranging from 0 to 12. The indices refer to shelf locations numbered from 1 through 12 for convenience. Each array is initially set to 0. 0 will mean that the item in that shelf location is not a misplaced item. Later, the value will be changed to 1 for any location containing a misplaced item.
We need to convert the STRINGS that were parsed from LLMresponse to NUMBERS. Each of these numbers is temporarily stored in the variable ozomapLocation.
We subtract 9 from each of the numbers, giving us the shelf location number. For example, if we subtract 9 from 11, we will get 2, which is the shelf location of the football on the top shelf of Ozomap X
The location in the array is set to 1 to indicate that the item there appears to be misplaced.
Ozobot traverses the twelve intersections, stopping at each intersection.
If the value at the current shelfLocation is 1, meaning it has a misplaced item, then the top light is set to red. Else, when the current shelfLocation is 0, meaning it does NOT have a misplaced item, then the top light is set to green.
The program ends with Ozobot moving to the intersection at the far right bottom of the Ozomap, with a top light of white indicating the program has finished.
each shelf. Their position locations are identified by the numbers at the bottom of each shelf.
Respond only with the position location numbers numerically from lowest to highest for the item that is different from the rest on each shelf each separated by a blank space.
in case of exception do
print to IO console "LLM error"
print to IO console LLMresponse
set shelfLocations to create list with item 0 repeated 13 times
count with i from 0 to 6 by 3
do
set ozomapLocation to convert in text LLMresponse get substring from letter # i to letter # i + 1 to number
set ozomapLocation to ozomapLocation - 9
in list shelfLocations set # ozomapLocation as 1
uncancelable alert message "lap when Ozobot is at the start location."
count with i from 1 to 12 by 1
do
set top light color
follow line to next intersection or line end
if in list shelfLocations get # i = 1
do
set top light color
wait 1 seconds
else
set top light color
wait 1 seconds
set top light color
follow line to next intersection or line end

Working with the Program

Your teacher will provide you with three different Ozomaps, labeled X, Y, and Z in the lower-left corner. X and Y have shelves with sporting goods, while Z has shelves with Ozobot products. Your program should work with all three of the Ozomaps, with no changes to the prompt. Recall, as mentioned earlier, it is best to work with the **Google (mini)** LLM.