OzoBlockly Editor Challenge:

Intelligent Package Sorting – ORA Reads Shipping Labels

Grades 9-12

Teacher Guide

by Richard Born, Associate Professor Emeritus, Northern Illinois University

Introduction

This lesson challenge, "Intelligent Package Sorting," introduces students to the cutting-edge intersection of robotics and Artificial Intelligence through the lens of Vision-Language Models (VLMs). Students will simulate a modern automated warehouse scenario where an ORA robot acts as an intelligent sorter. The core objective is for students to understand how multimodal AI systems process visual data (shipping labels) and apply logical reasoning to make physical decisions. Students will observe how the VLM reads labels for critical keywords like "hazardous," "fragile," or specific destination cities. They must then apply a strict four-tier priority rulebook to program the robot's sorting actions. This hierarchy prioritizes safety (Hazardous/Flammable) over handling requirements (Fragile, Heavy/Oversize), with geographic sorting (East vs. West of the Mississippi) serving as the final default. By navigating these logic gates, students gain practical insight into how industrial AI combines computer vision, language processing, and decision-making algorithms to revolutionize logistics efficiency.

If you have not already done so, you should now read through the *Student Guide* for this challenge for details on what is expected from the students.

Example Solution to this Challenge

The solution to this challenge provides the opportunity for the students to develop an OzoBlockly Editor program that is modular, dividing the program logic into functions each having a specific purpose. Figure 1 shows the main program, which simply calls the major functions in the required order. As can be seen, it is assumed that twelve packages will be picked and placed at the appropriate factory locations.

Figure 1

Figure 2 shows the "*Initialization*" function. ORA's speed is set to 20% and is moved to the HOME location. The tool is set to the vacuum gripper and turned OFF.

Figure 2

Figure 3 shows the "*Use LLM with Camera to Image the Package*" function. The prompt was worded so that the string variable *LLMresponse* would ultimately contain the words East, West, Hazardous, Flammable, Fragile, Heavy, or Oversize when those words appear on the package. **Note that using the *Google (standard)* LLM seems to provide the most accurate determination of the regions east and west.**

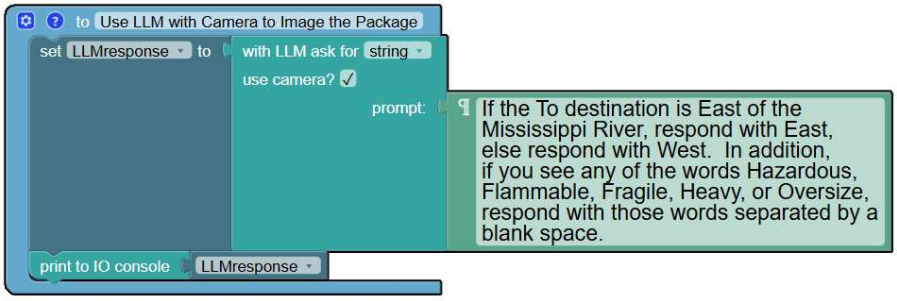

Figure 3

Figure 4 shows the "*Pick up the Package*" function. The first *move to coordinate* block moves ORA's arm to the location (225, 225), at a height of 30 mm. Then the arm is lowered to 2 mm so that it contacts the package. The vacuum gripper is turned ON, and the package is raised to 30 mm, making it ready to move to its final drop location.

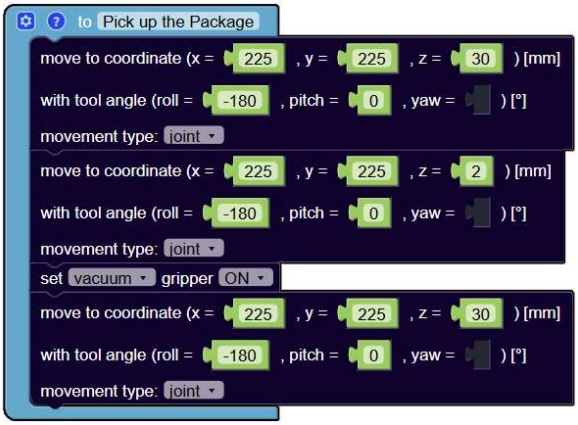

Figure 4

Figure 5 shows the "*Parse for Words on the Package*" function. The variables *Flammable, Fragile, Heavy, Oversize, East, and West* are all **logical** variables initialized as **false**. Then each of the following *if-do* blocks set the variable to true if the word is found in the variable *LLMresponse*. These *if-do* blocks make use of a text block for the first occurrence of each of these words. This text block returns *-1* if the word is *not* found. So, if the word *is* found, that is if the block returns a non-negative value, then the word is indeed in the variable LLMresponse.

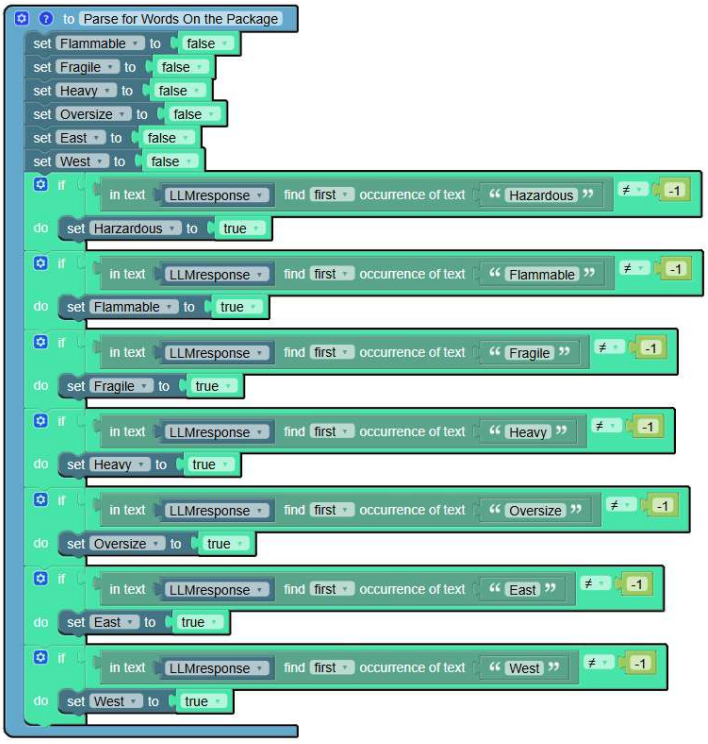

Figure 5

Figure 6 shows the "Move to Factory Rulebook Location" function. By using an if block with multiple else-ifs and properly ordering these else-ifs by priority, the required factory rulebook has been implemented.

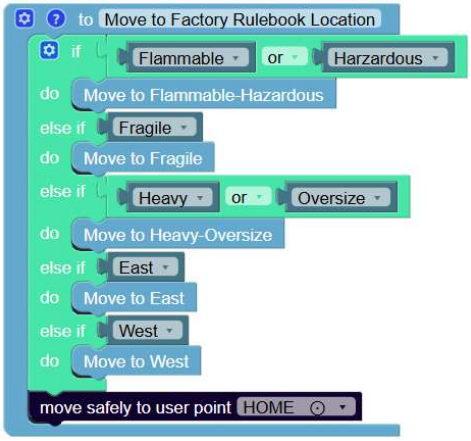

Figure 6

Note the use of **logical or** for Priority 1 Flammable/Hazardous, and priority 3 for Heavy/Oversize. **Logical or** is true if either one or both of its operands are true. Finally, appropriate *move* functions are called when needed. Figure 7 shows these move functions. Each of these five move functions sets the *xLoc* and *yLoc* location on the factory floor where the move is to occur. Then the "*Move to Place Location*" function is called. During the actual move process, the height is 30 mm. When the location has been reached, then the package is lowered to 5 mm to provide a precise drop when the vacuum is turned OFF.

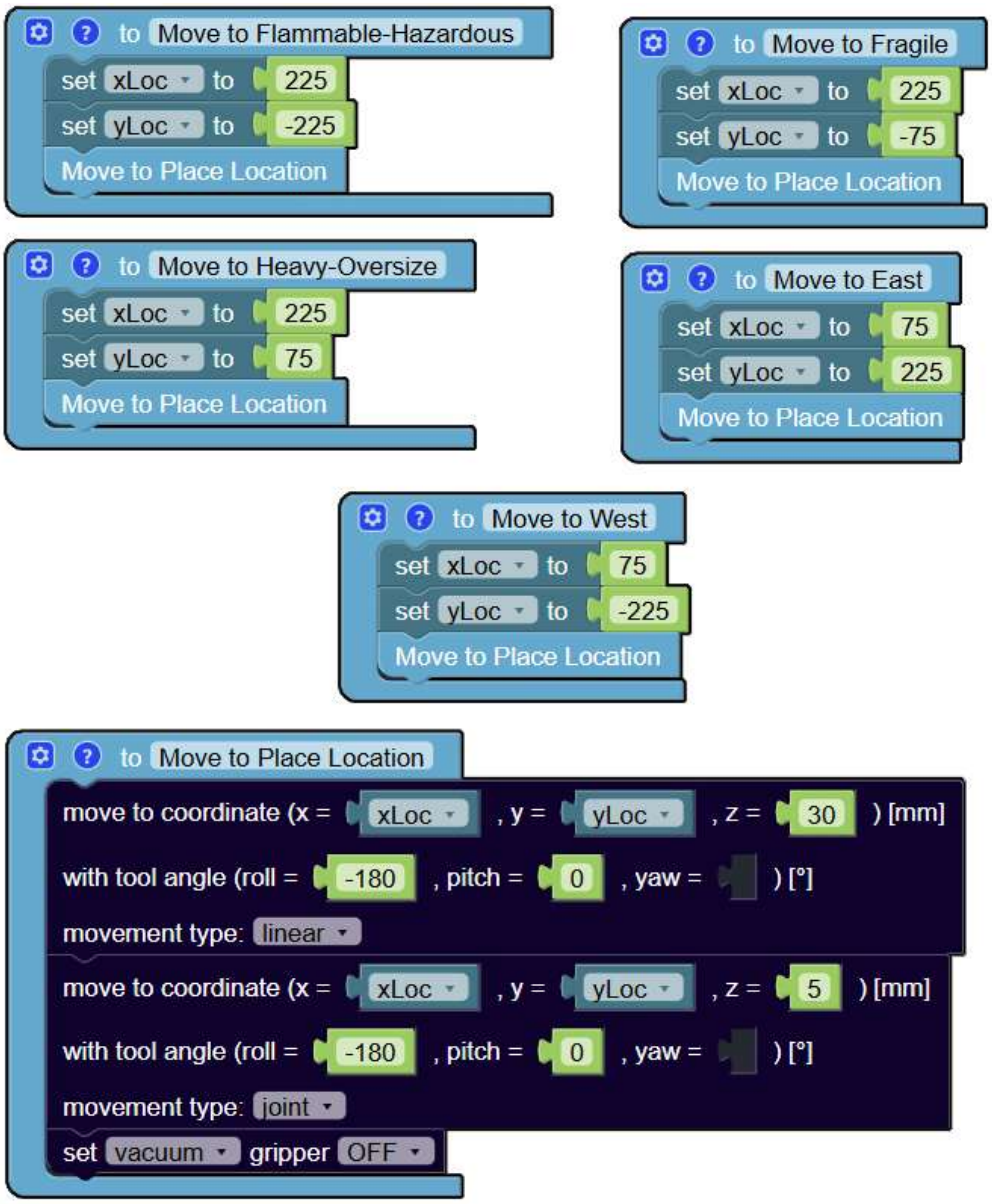

Figure 7

Share Code for the Solution - xaweqba

Link to 3D STL File - Package Holder

ISTE Standards (International Society for Technology and Education)

Empowered Learner 1.1.d: Students understand the fundamental concepts of technology operations, demonstrate the ability to choose, use and troubleshoot current technologies and are able to transfer their knowledge to explore emerging technologies.

Knowledge Constructor 1.3.d: Students build knowledge by actively exploring real-world issues and problems, developing ideas and theories and pursuing answers and solutions.

Computational Thinker 1.5.c: Students break problems into component parts, extract key information, and develop descriptive models to understand complex systems or facilitate problem-solving.

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.
3B-AP-14	Construct solutions to problems using student-created components, such as procedures, modules and/or objects.