Project Zeus: String Writing API

Overview

The eventual goal of Project Zeus is for a human participant to engage in writing word puzzles alongside the UR5 robot arm. The current stage of development provides an API for commanding the robot to write strings.

Set Up

Project Zeus is currently written using Python 3.7, making use of SintefManufacturing's URX library. The URX library provides an interface for connecting with the robot, pulling information, and sending commands.

Installation is simple. Locate the folder where your Python packages are stored, and make sure that both Python and the Python package manager PIP are installed and up to date. On the command line, run "pip install urx". Make sure that "import urx" is at the top of all code files, and Zeus is ready to use.

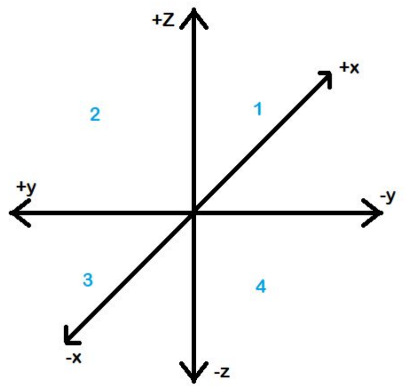
Development

The String Writing API enables users to write the 26 letters in the English alphabet using the UR Robot and a python environment. The exclamation point special character has been added to enhance the set of characters, and encourage the addition of special characters in the future.

To ensure efficiency in developing characters in the String Writing API, the following design standards must be present for all characters:

1) Each character must be enclosed in a 4 x 4 matrix
2) The initial position of the character is the most-bottom, and most-left point of the matrix
3) The final position of the character is the most-bottom, and most-right point of the matrix
4) A character must not draw any spaces beyond the 4 x 4 matrix, to enforce space independency
5) The programmer should assume that the robot arm starts away from the board

The 4 x 4 matrix is placed inside the y-plain and z-plain, within the first coordinate, as standardized by the following axis system.



Usage

A few common URX library calls include:

- urx.Robot
    - ex. rob = urx.Robot(“192.168.43.107”)
    - Connects to the robot at the specified IP address
- getl()
    - ex. pose = rob.getl()
    - Obtains information about the robot’s current position and angles
        - pose[0] is the x position
        - pose[1] is the y position
        - pose[2] is the z position
- movel(pose,acc,vel)
    - ex: rob.movel(pose, a, v)
    - Moves the robot from its current position to the position described by pose
    - The library makes the necessary angle and rotation calculations for you
    - **WARNING:** When a joint angle approaches 360°, movel() may cause the joint to rapidly flip around to 0°, potentially causing emergency stops due to the high velocity. Stand back from the robot while it is in motion, and be aware of exactly what the robot is doing when you call this function.

Main

The functionalities of the main function are as follows:

- Establishing the connection of the robot at the specified IP address (with help of urx.Robot(“192.168.43.107”) function).
- In try: The values of l(length/size),v(Velocity),a(acceleration) are assigned(all values are represented in meters).
- The main calls the draw_string(string, size, accel, velocity) function. This function takes in the string value to be printed along with the size, acceleration and the velocity for the robot to operate.