Q1.

A black square with a white dot

AI-generated content may be incorrect.

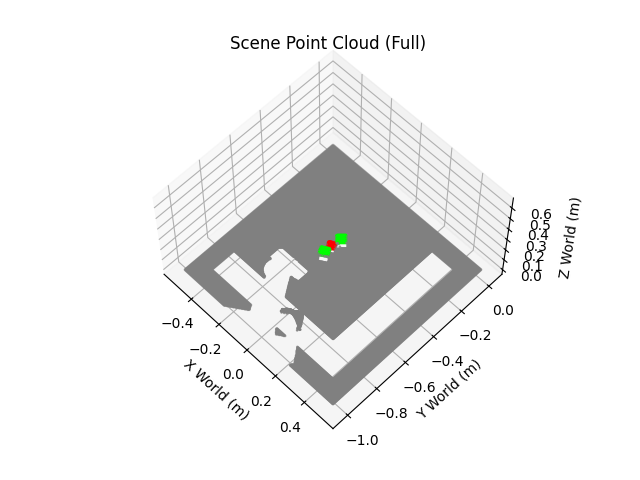
Q2.

1. Equations for calculating 3D point cloud (camera frame) from a depth image

Where and are the focal lengths, and are the principal point.

1. Equations for transforming the obtained point cloud to the world coordinate frame

Where is the inverse of the extrinsic matrix.



Q3.

**Grasping Strategy**: It begins by processing the image to resist noise through color segmentation, morphological filtering, and selecting the largest component, then precisely calculates the red cube's 3D centroid and top surface height using the median and percentile of its point cloud coordinates. Subsequently, the robot plans three critical waypoints which are a safe hover position directly above the cube, a grasp position for acquisition, and a lift position for retreat. The entire motion sequence strictly follows a vertical descend, grasp, and vertical lift pattern to ensure no side collisions occur while entering or exiting the confined space surrounded by the green cubes.

The grasping strategy is quite effective and robust within a controlled environment. Rather than just performing simple color detection, it reconstructs a 3D point cloud and uses robust statistical methods like medians and percentiles to precisely calculate each object's 3D center, top surface height, and size. This approach is highly resistant to sensor noise and outliers, and its intelligent sorting logic—which accurately distinguishes between multiple similar objects—provides the reliable data necessary for a robot to execute precise grasping motions.

**Failure modes:** First, it is highly dependent on hardcoded color ranges, meaning significant changes in ambient lighting or the introduction of similarly colored objects would cause detection to fail. Second, the code rigidly assumes there is only one red and exactly two green blocks; any deviation in this count will cause a program error. Furthermore, if objects occlude one another, the calculated center and size will be inaccurate, and the strategy does not compute object orientation, which is a critical failure for a gripper that needs to align with the object's faces.

Q4.

**Green cube poses:** First, an overhead camera image is processed to create a binary mask isolating green pixels. The *scipy.ndimage.label* function then identifies each separate green cube as a distinct blob. For each blob, the corresponding 3D points are extracted from the scene's point cloud. Finally, the cube's position is calculated by finding the median of its points' X and Y coordinates and its top surface height by using the 95th percentile of the Z coordinates.

**Selection of base green cube:** The primary criterion is to choose the cube closest to the center of the workspace (table), as this ensures the final stack is in a stable and easily reachable location for the robot. As a tie-breaker, if two cubes are equidistant from the center, the one with the larger pixel area in the 2D camera view is selected.

Q5.

**Improvement**: First, Its perception can be made more robust by replacing the fragile hardcoded color thresholds with adaptive techniques or a deep learning object detection model, which is resilient to lighting changes. The rigid, linear script can also be upgraded to a flexible state machine that includes error handling, such as confirming successful grasps and retrying if they fail. Finally, the robot's physical interactions can be enhanced by using force feedback for precise placements instead of relying on estimated heights, and by implementing smoother trajectory planning for more fluid and reliable movements.