Intel Unnati industrial training program 2024-2025

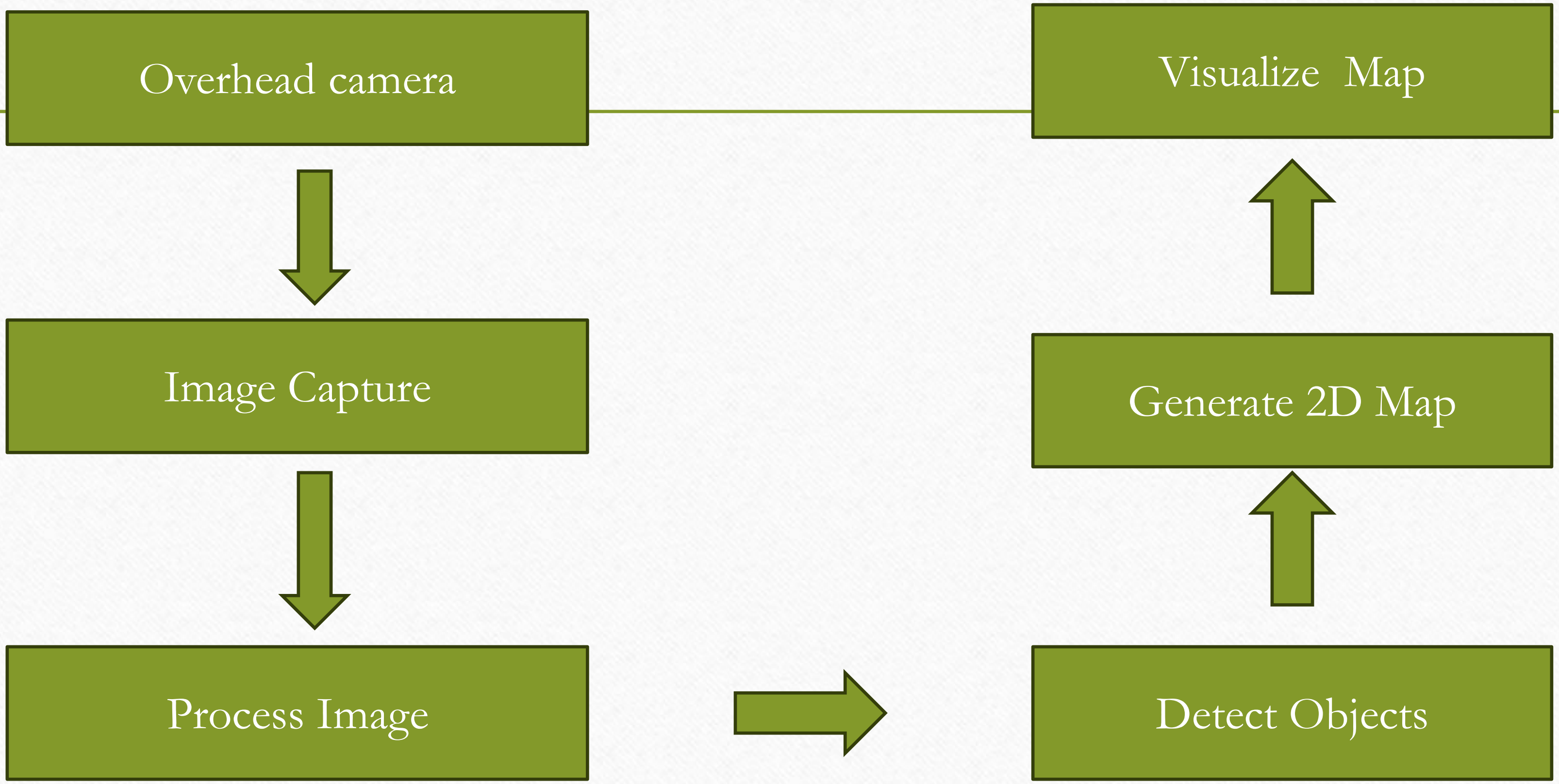
Problem statement

A 2D occupancy grid map of a room using overhead cameras refers to the creation of a two-dimensional representation of a room's floor plan, where each grid cell indicates whether it is occupied or free. This map is generated using data captured from cameras placed above the room. The occupancy status of each grid cell is determined by processing the images or video feeds from the cameras, allowing for the detection and mapping of objects or obstacles within the room. This technique is often used in robotics and automation for navigation and environment understanding.

Solution approach

- The problem we are trying to solve is the accurate and efficient mapping of indoor environments for the purpose of navigation and environment understanding, particularly in the context of robotics and automation. In many indoor settings, such as homes, offices, and warehouses, it is crucial for autonomous systems to have a clear understanding of the layout and the location of obstacles to navigate safely and effectively. Traditional methods of mapping, such as manual surveying or using ground-level sensors, can be time-consuming, labor-intensive, and sometimes impractical in cluttered or dynamic environments.

Flow Chart


Overhead camera
Image Capture
Process Image
Detect Objects
Generate 2D Map
Visualize Map

1. **Overhead Cameras**
    - **Function:** Capture images or video of the gazebo from an overhead perspective.
    - **Details:** Cameras are mounted above the gazebo to provide a clear, unobstructed view.
2. **Image Capture**
    - **Function:** Acquire images or video frames from the overhead cameras.
    - **Details:** The cameras continuously capture images or video frames at regular intervals.
3. **Preprocess Images**
    - **Function:** Prepare the captured images for object detection.
    - **Details:** This step includes noise reduction and rectification to ensure the images are clear and accurately represent the gazebo.
4. **Detect Objects**
    - **Function:** Identify and locate objects in the preprocessed images.

5. **Details:** Techniques such as background subtraction and edge detection are used to identify objects within the gazebo. **Map Coordinates**
    - **Function:** Convert the positions of detected objects from image coordinates to real-world coordinates.
    - **Details:** Using camera calibration data, object positions in the images are translated to real-world coordinates within the gazebo.

6. **Update Grid**
   - **Function:** Update the occupancy grid based on the positions of detected objects.
   - **Details:** The gazebo is divided into a grid of cells, each marked as occupied or unoccupied based on object detection.
7. **Generate 2D Map**
   - **Function:** Create a 2D occupancy grid map using the updated grid data.
   - **Details:** The occupancy grid is compiled into a map that shows the layout of the gazebo and the locations of objects.
8. **Visualize Map**
   - **Function:** Display the 2D occupancy grid map for users to interpret.
   - **Details:** The generated map is visualized, typically on a computer screen, showing the gazebo's layout and the positions of objects.

Methodology

- **Step 1: Software installation**

Install ubuntu and gazebo in our computers

- **Step 2: Camera Setup**

Select Cameras: Use high-resolution cameras with a wide field of view

Install Cameras: Position cameras overhead to cover the entire room.

- **Step 3: stitching**

Intrinsic Calibration: Use a calibration target (like a checkerboard) to determine each camera's internal parameters. Tools like OpenCV can help with this.

Extrinsic Calibration: Calculate each camera's position and orientation relative to a global coordinate system using known reference points

- **Step 4: Image Acquisition**

Capture Images/Video: Use a software like OpenCV to acquire synchronized images or video streams from the cameras

- **Step 5: Image Processing**

Background Subtraction: Remove the static background using methods like Gaussian Mixture-based Background/Foreground Segmentation (MOG2) in OpenCV.

Segmentation: Use image processing techniques to isolate the occupied regions. This can be done using methods like thresholding, contour detection, or deep learning-based segmentation.

- **Step 6: Homographs Transformation**

Transform Perspective: Apply holography using OpenCV to convert the camera's perspective to a top-down view of the room.

- **Step 7: Occupancy Grid Creation**

Define Grid: Set the size and resolution of the occupancy grid.

Map Occupancy: Mark grid cells as occupied based on the transformed segmented regions.

- **Step 8: Visualization**

Display Grid: Use a visualization tool to display the occupancy grid, representing the room's layout and occupied areas.

Pros And Cons

- **Pros**

Comprehensive Spatial Coverage:

Top-Down Perspective: Eliminates blind spots, providing a complete view of the room.

Accurate Mapping:

Holography Transformation: Maintains spatial integrity by accurately converting perspectives.

- Cons

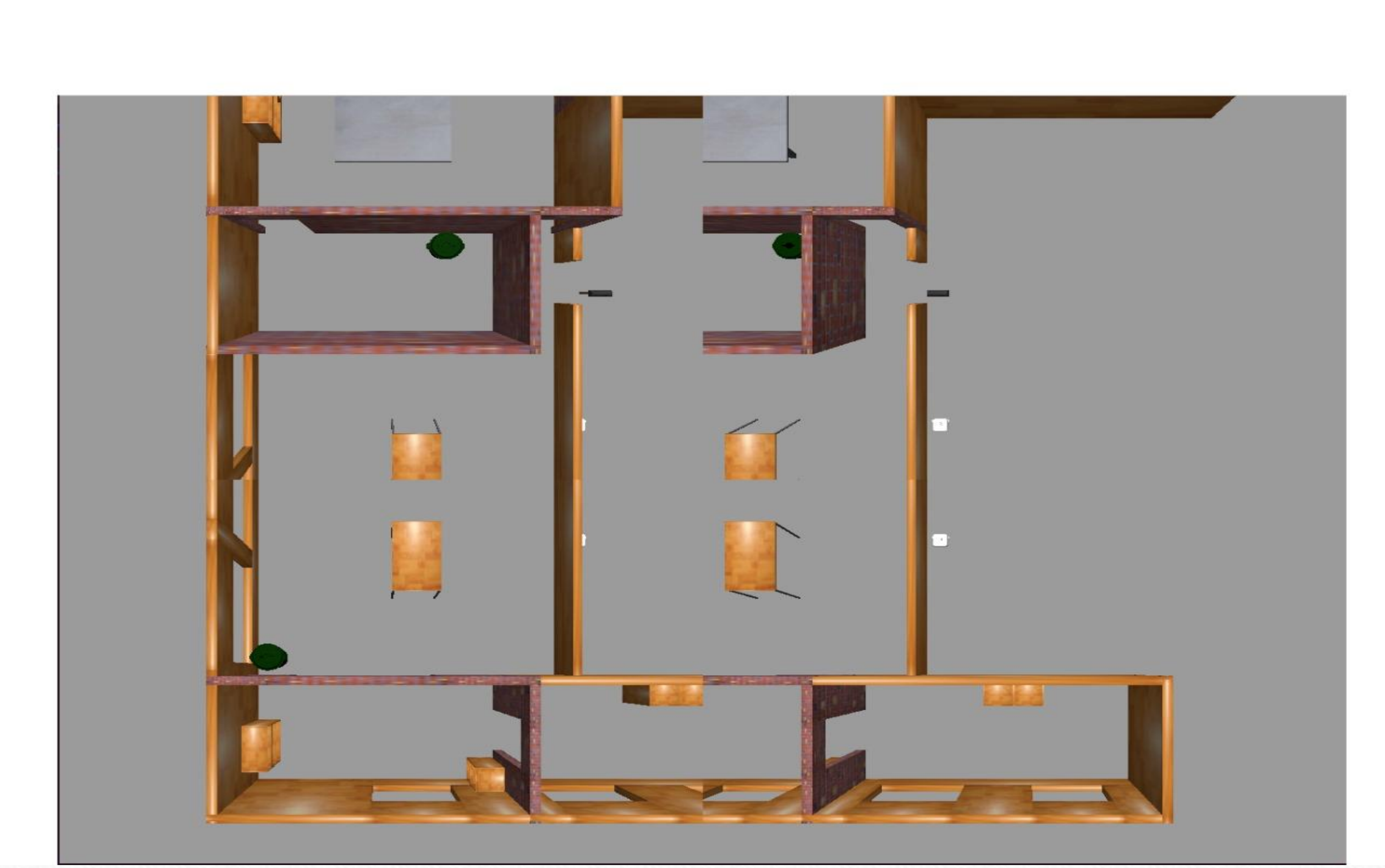
Processing Overhead:

Latency: Ensuring low latency in real-time applications can be challenging.

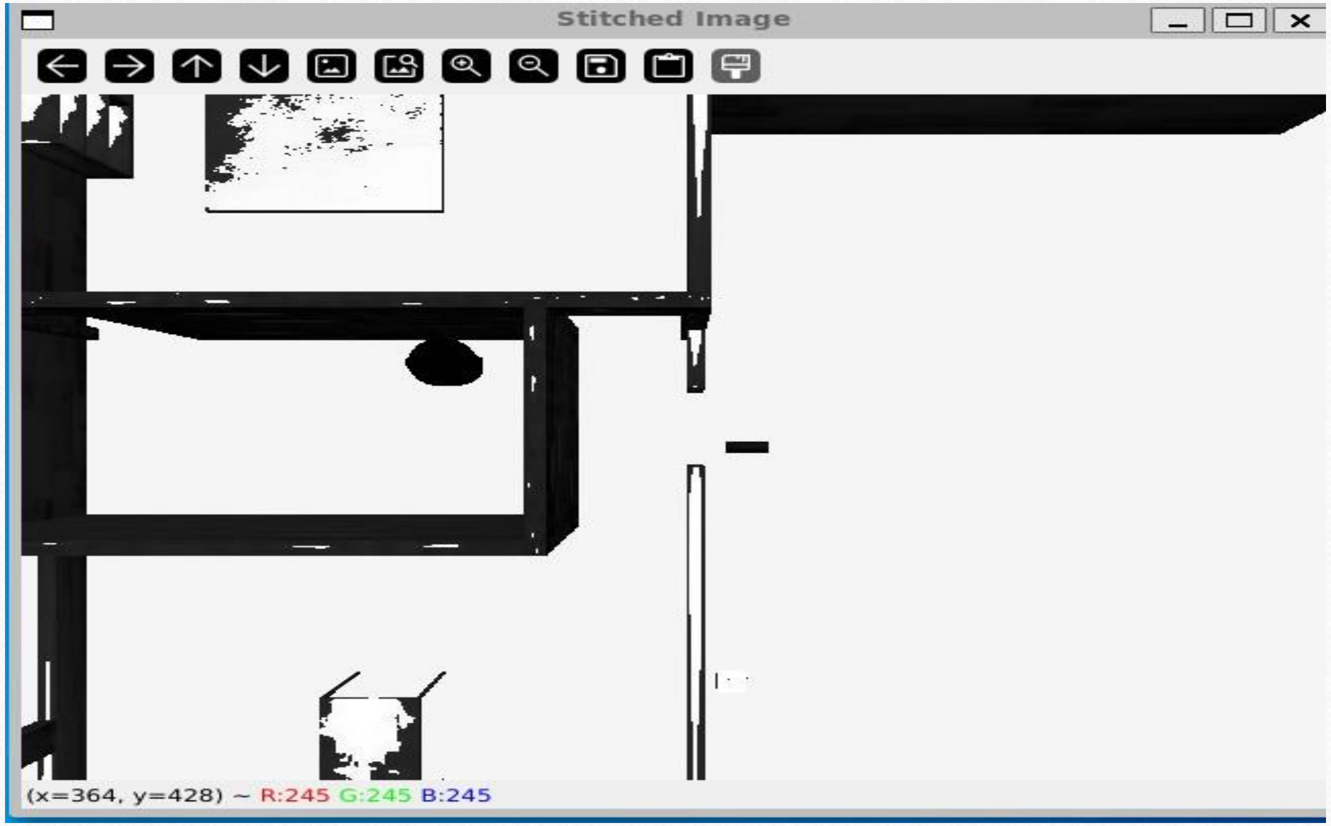
Technical Challenges:

Objects or people overlapping can complicate segmentation and occupancy detection.

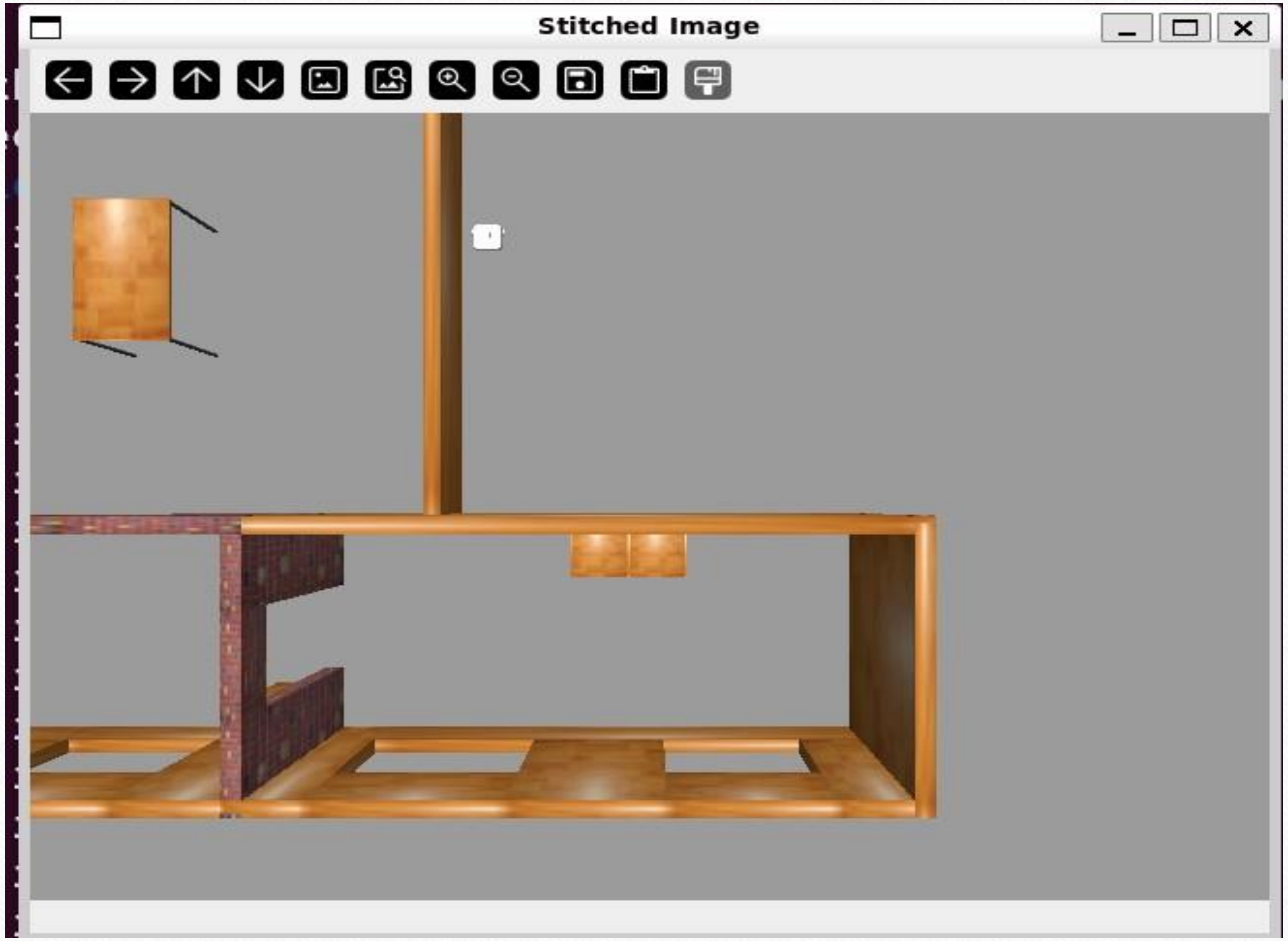
Result



https://github.com/Spriya-12/PS_10/blob/main/code/stitch_1.py



https://github.com/Spriya-12/PS_10/blob/main/Stitch_imga1.py
https://github.com/Spriya-12/PS_10/blob/main/image_viewer.py



https://github.com/Spriya-12/PS_10/blob/main/code/stitch_2.py

- Team members

- Santhi Priya
- N.Keerthi
- Reddy Soma Sai Sathya Snigdha