Object Detection and Localization

Computer Vision Blockweek Project

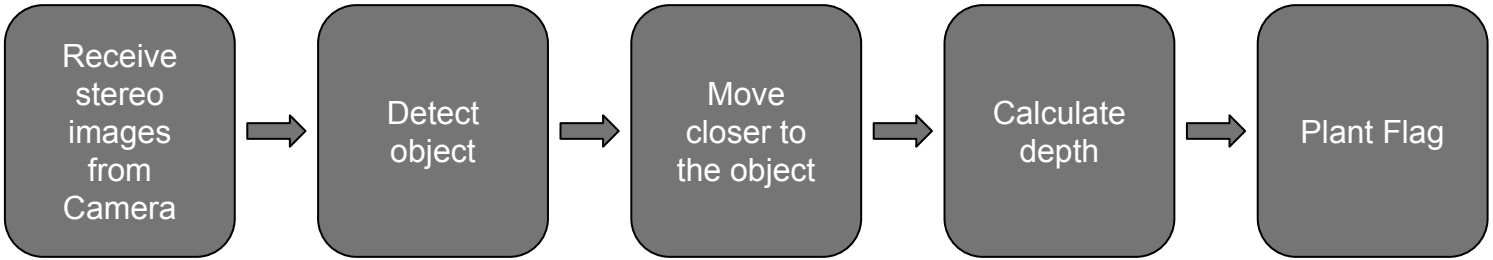
Project Overview

2 cans, red and green are placed in the arena. 2 Robots - search and pickup are present in the arena. The search robot maps the arena and takes pictures of the scene. The problem statement requires us to

1. Identify the cans from the image
2. Locate the identified cans in the arena *(x,y,z)*

Once the cans are located, the search robot transmits the coordinates to the ROS network and the pickup robot drives to the location and collects the can.

Approach


Receive stereo images from Camera
Detect object
Move closer to the object
Calculate depth
Plant Flag

These links will take you to your respective content

Aaron

Rushab

Saul

Shreya

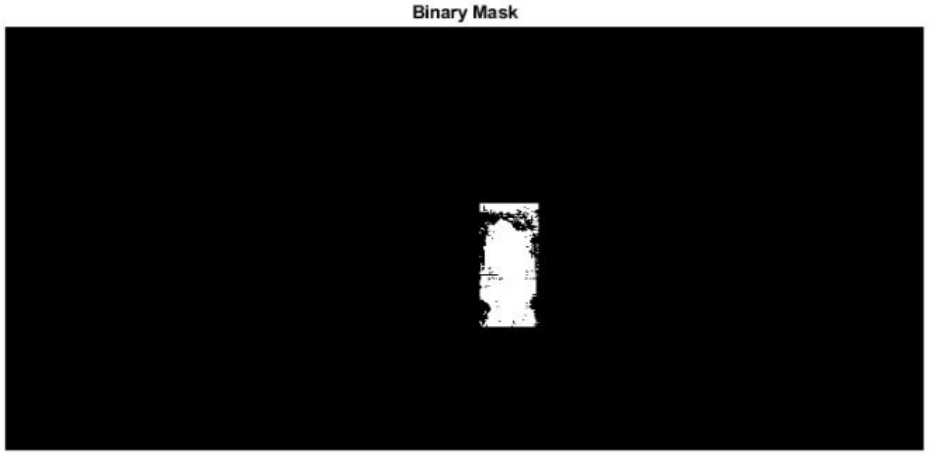
Vishnu

Saul

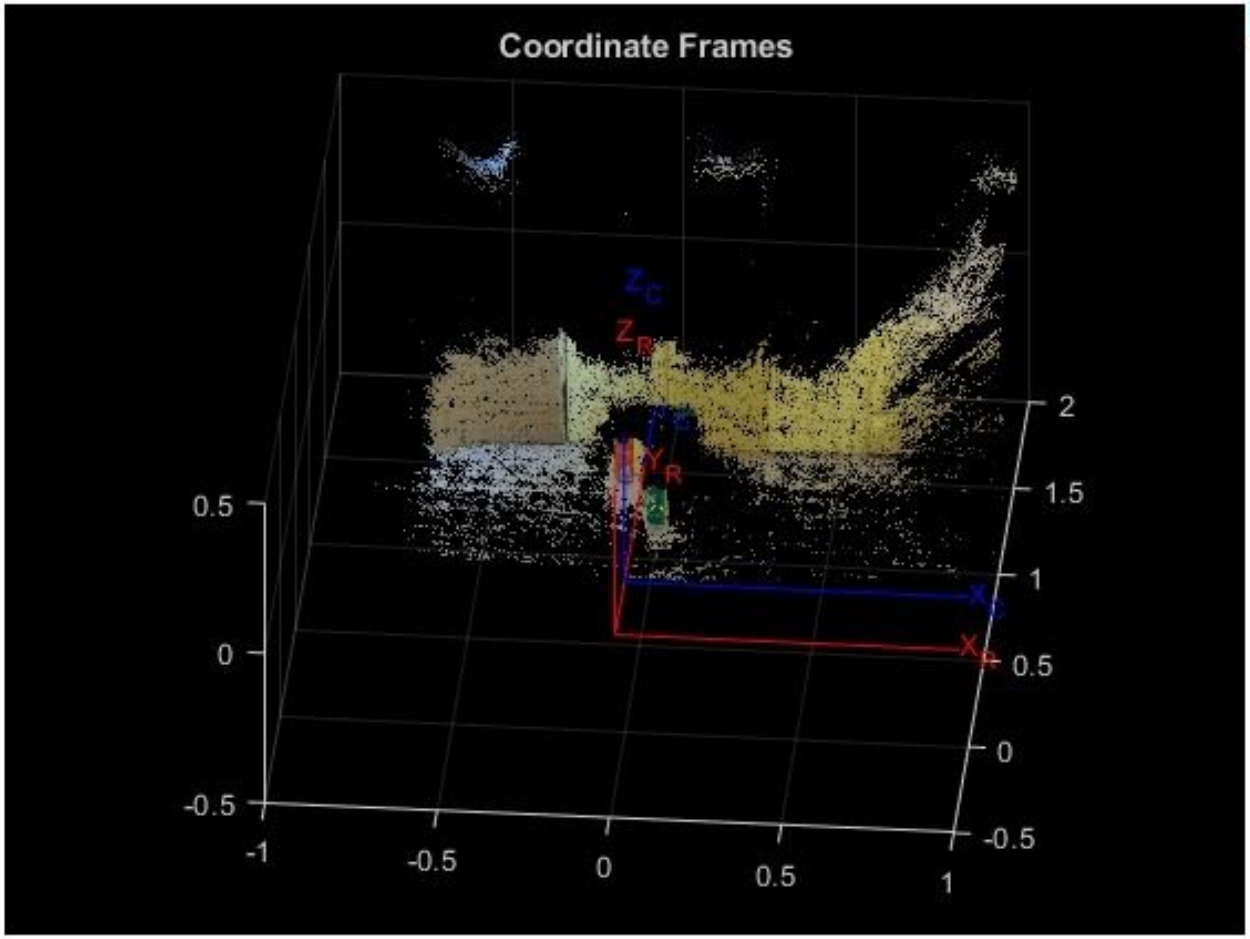
Responsibilities

- Find 3-D world coordinates of centroids.
- Heterogeneous Transformation Matrix for Camera and Robot
- Calculation of distance from camera to find closest object
- Creation and implementation of masks
- Visualization of images
- Image extraction from Rosbag files

Masks Implementation



Extraction based on depth


Coordinate Frames
Z_C
Z_R
Y_R
X_R
0.5
0
-0.5
-1
-0.5
0
0.5
1
2
1.5
1
0.5
0
-0.5

```
msg =
  ROS Point message with properties:

    MessageType: 'geometry_msgs/Point'
              X: 0.0458
              Y: 0
              Z: 0

  Use showdetails to show the contents of the message
```

```
msg.Y = yr
```

```
msg =
  ROS Point message with properties:

    MessageType: 'geometry_msgs/Point'
              X: 0.0458
              Y: 0.5261
              Z: 0

  Use showdetails to show the contents of the message
```

```
msg.Z = zr
```

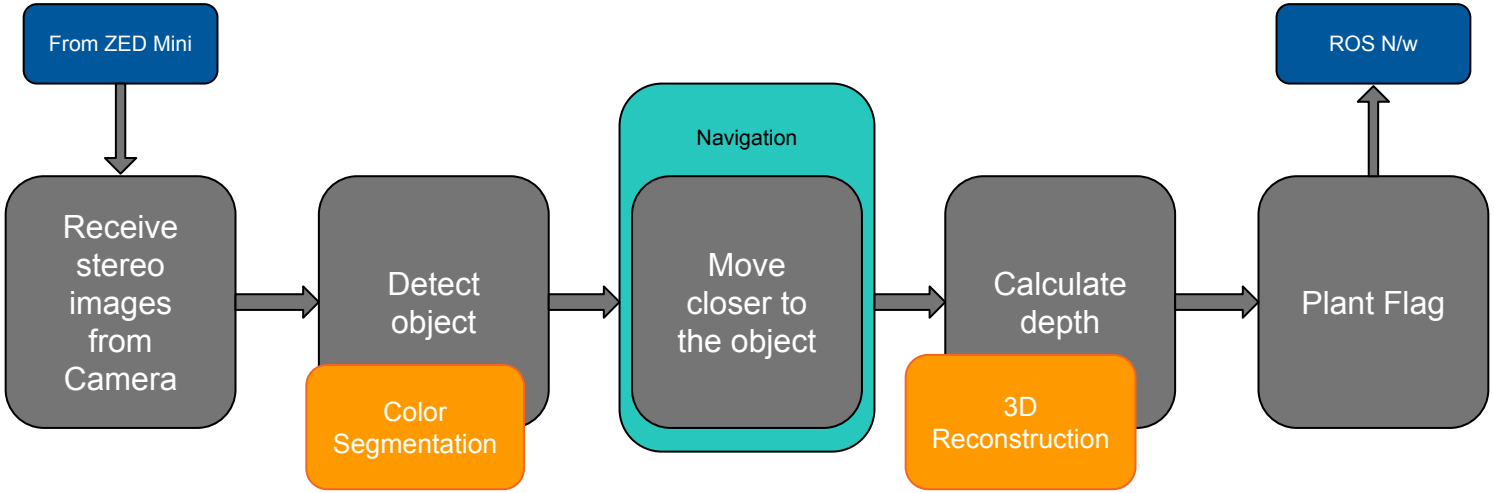
```
msg =
  ROS Point message with properties:

    MessageType: 'geometry_msgs/Point'
              X: 0.0458
              Y: 0.5261
              Z: -0.0400

  Use showdetails to show the contents of the message
```

Aaron

Responsibilities

- Pipeline design for object detection
- Image acquisition from the ROS Network
- Sending the coordinates to the ROS Network
- Recording ROS Bags for storage and postprocessing

Vision System Pipeline



Connecting to ROS Network

- The Jetson Nano running the ZED Wrapper acts as the ROS Master and the topics are published.
- The MATLAB instance running the pipeline subscribes to these topics and the images are acquired live .

```
rosinit('10.0.101.120')
```

```
Initializing global node /matlab_global_node_82173 with NodeURI http://Aaron-XPS:52058/


imS_sub =
  Subscriber with properties:

       TopicName: '/zedm/zed_node/stereo_raw/image_raw_color'
   LatestMessage: [0×1 Image]
     MessageType: 'sensor_msgs/Image'
      BufferSize: 1
   NewMessageFcn: []
      DataFormat: 'object'
```

Image Acquisition

- A ZED Mini stereo camera is used as the image and depth sensor.
- We acquire only the left and right images using the camera and the pipeline performs depth calculation and the 3D reconstructions using MATLAB functions

```
imL_sub=rossubscriber('/zedm/zed_node/left_raw/image_raw_color');
imR_sub=rossubscriber('/zedm/zed_node/right_raw/image_raw_color');
if(useSystemStereo)
    imS_sub=rossubscriber('/zedm/zed_node/stereo_raw/image_raw_color')
```

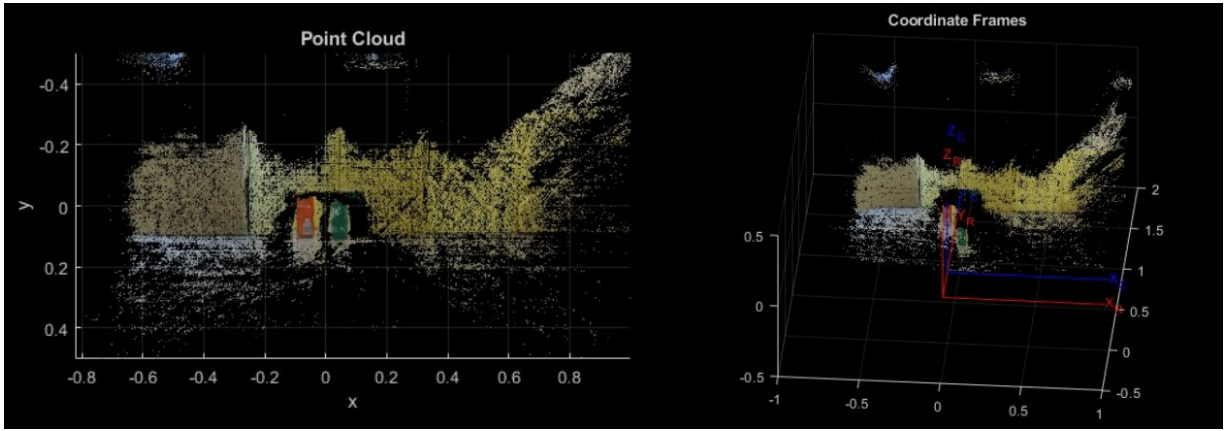
Acquired Stereo Images from the Arena

Object Detection

- Object Detection is done using color segmentation as the objects of interest have a prominent color of red and green on them.

Calculate Depth

- Stereo vision is used to extract the depth from the scene and calculate the distance to the object.



Calculating the 3D Coordinates

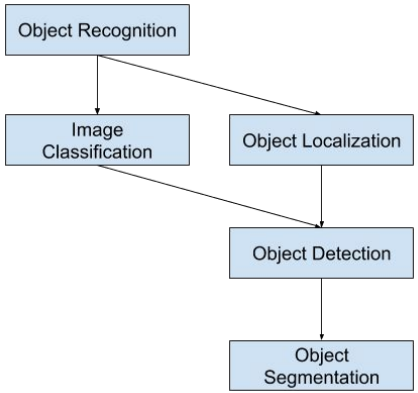
- The 3D coordinates in the world frame are back calculated with the given transforms from the world frame to robot frame and the robot frame to the camera frame.

Send Location to ROS Network

- The final location in terms of x,y,z coordinates are published onto the ROS network via the topic /vision_team/target_pos

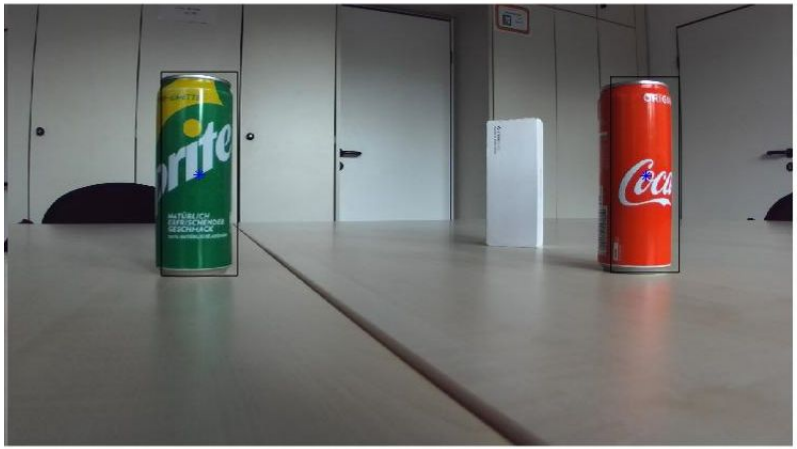
Object Localization

- Identifying the location of one or more objects in an image and drawing a bounding box around their extent.
- Object detection combines these two tasks and localizes and classifies one or more objects in an image

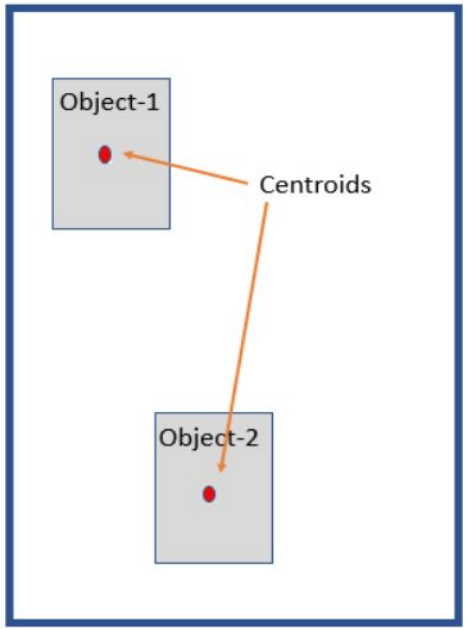


Bounding box

- to bind or identify a target and serve as a reference point for object detection and create a collision box for that object.



Centroids



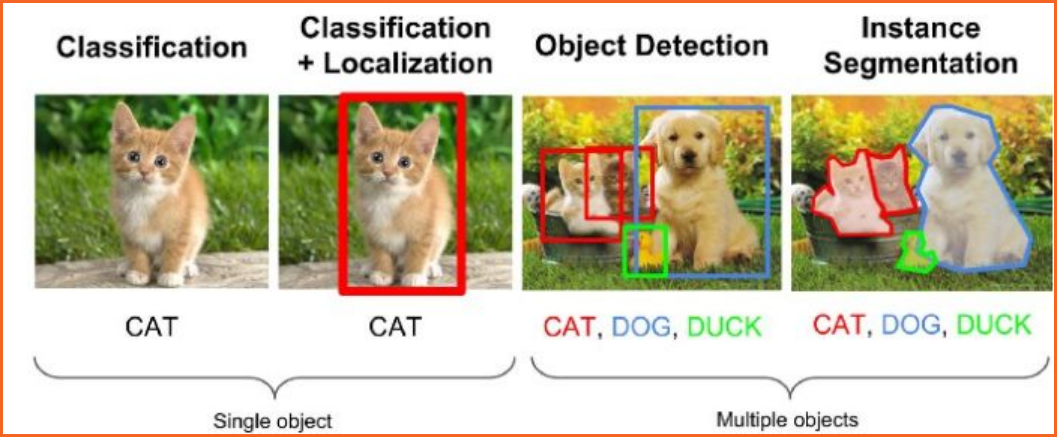
- **Object Detection and Recognition:** A technique to identify and localize the objects of interest by drawing bounding boxes around them in each frame.
- Once object detection and recognition is done tracking could be done using Centroid-based Object Tracking
- This centroid acts as a x and y coordinates of detected object.

Histogram Equalization

Shreya

Object Detection

Is a computer vision technique for locating instances of objects in images.
Used in various applications such as ADAS, video surveillance.

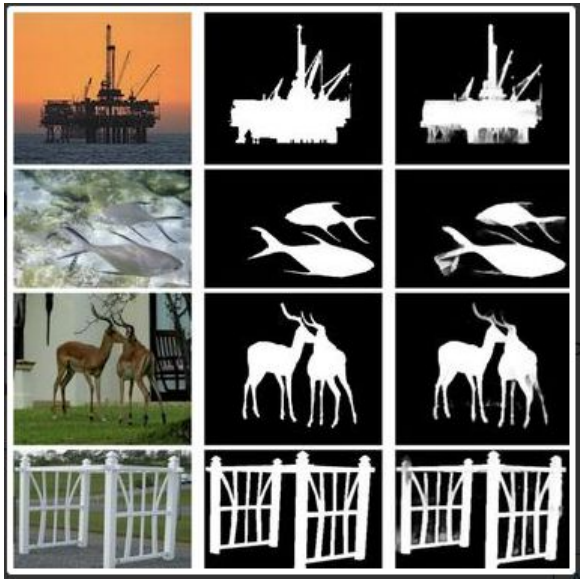


Object Detection using Color segmentation

- Segmentation is based on color feature of image pixels.
- Each pixel in an image is extracted with given concepts.
- Extraction of pixels that share similar color properties or illumination/luminosity.
- Segmentation can be done using various color spaces.

Advantages of using Color segmentation

- Color images can provide more information than gray level images.
- Color helps to recognize objects that are presented without surface details (texture or shadow).
- Useful when the background of an image needs to be ignored.

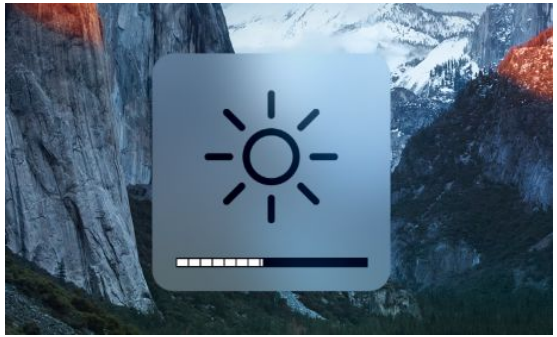


HSV Color Space

- Color detection algorithm searches for an image for pixels that have a specific color.

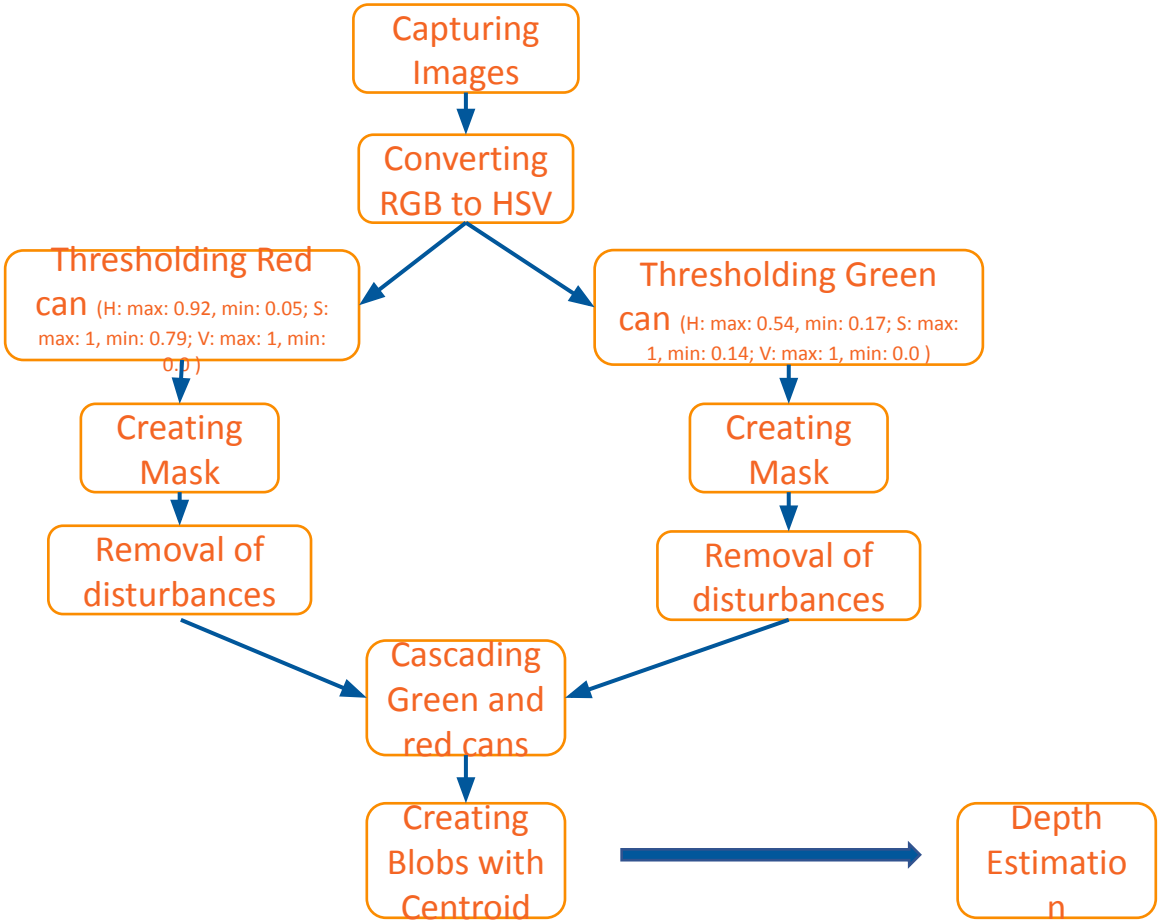
H(ue) represents color

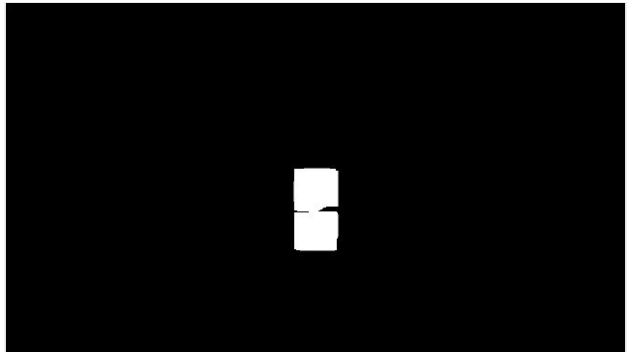
S(aturation) represents dominance of that color

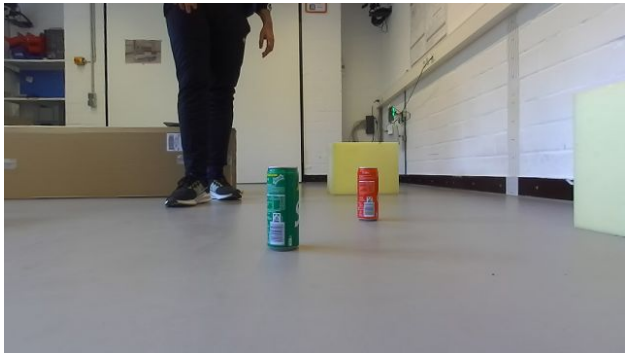


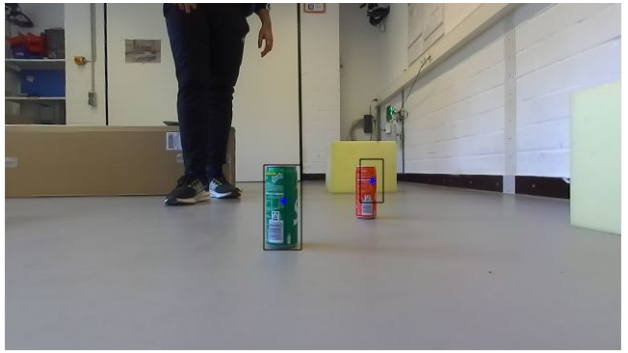
V(alue) represents brightness

Algorithm


Capturing Images
Converting RGB to HSV
Thresholding Red can (H: max: 0.92, min: 0.05; S: max: 1, min: 0.79; V: max: 1, min: 0.0)
Thresholding Green can (H: max: 0.54, min: 0.17; S: max: 1, min: 0.14; V: max: 1, min: 0.0)
Creating Mask
Creating Mask
Removal of disturbances
Removal of disturbances
Cascading Green and red cans
Creating Blobs with Centroid
Depth Estimation





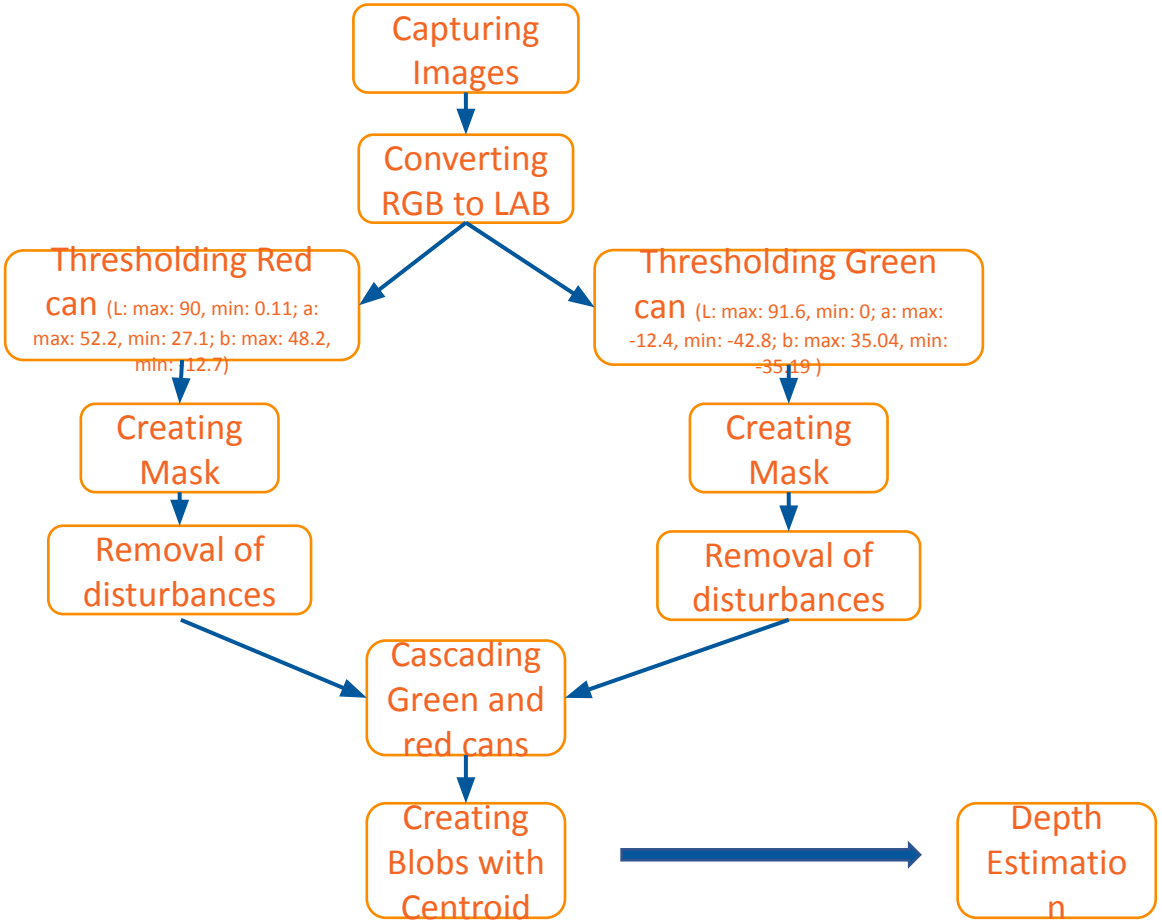
Lab Color Space

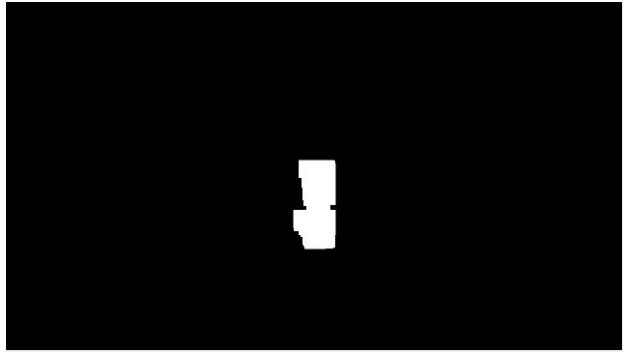
- In general, color detection algorithm searches for an image for pixels that have a specific color.
- Lab includes all colors of the spectrum as well as colors that are beyond human perception.

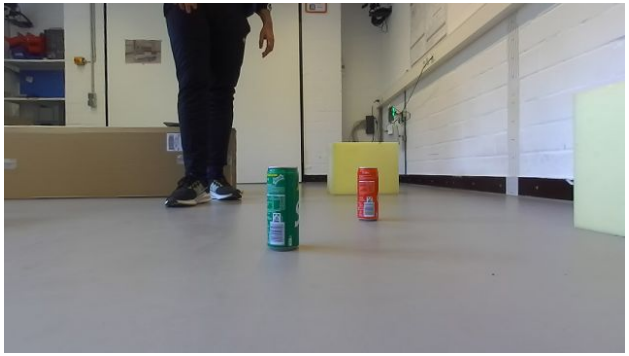
L(ightnesss) represents color

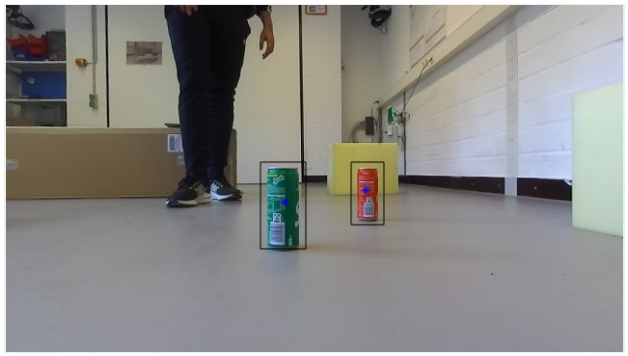
a and **b** represents color dimension

Algorithm


Capturing Images
Converting RGB to LAB
Thresholding Red can (L: max: 90, min: 0.11; a: max: 52.2, min: 27.1; b: max: 48.2, min: 12.7)
Thresholding Green can (L: max: 91.6, min: 0; a: max: -12.4, min: -42.8; b: max: 35.04, min: -35.19)
Creating Mask
Creating Mask
Removal of disturbances
Removal of disturbances
Cascading Green and red cans
Creating Blobs with Centroid
Depth Estimation





LAB

HSV

It includes all colors of the specturum as well as colors that are beyond human perception.

Most accurate and device independent.

Better control over sharpening the image.

Considers luminosity factor of the image.

Attributes corresponds directly to the basic color concepts. Hence conceptually simple.

Cannot use real brightness of the color.

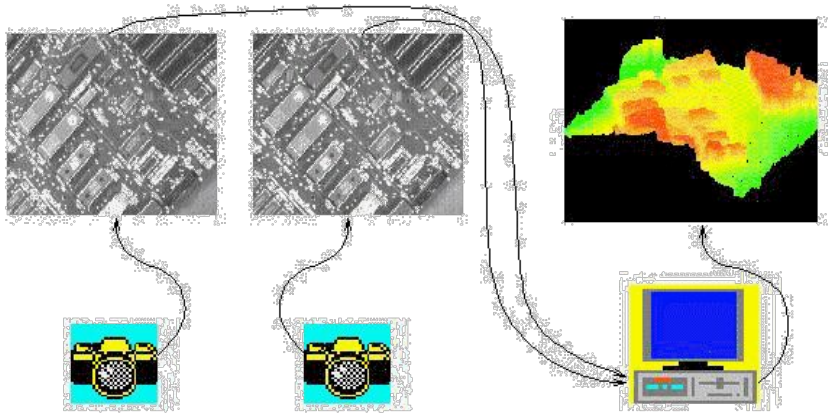
Doesn‘t consider luminosity factor of the image.

Stereo vision

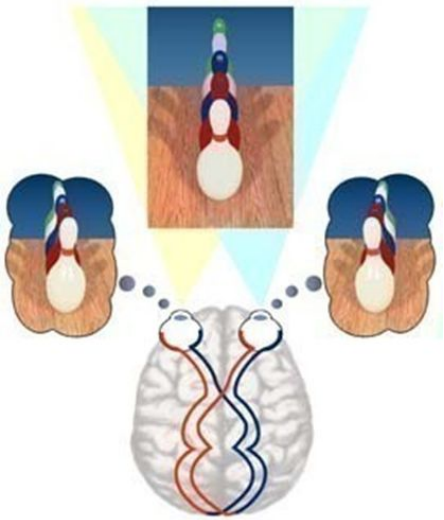
- Stereo vision is comparing 2 or more images of same scene and recovering depth information from the camera images by estimating relative depth of points in the scene.
- The output of the computation is 3D point cloud where each point cloud corresponds to pixel in one of the images.

Application of stereo image

- ADAS
- Robot navigation

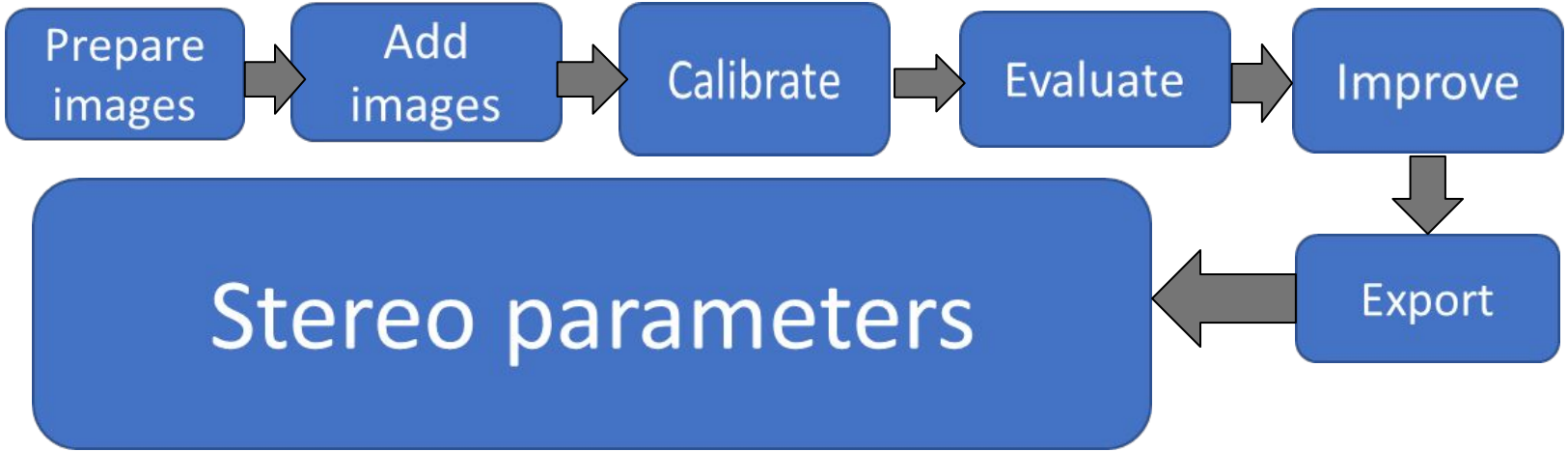


Stereo vision



Stereo vision is the computation of depth based on the binocular disparity between the images of an object in left and right eyes.

Stereo camera calibration

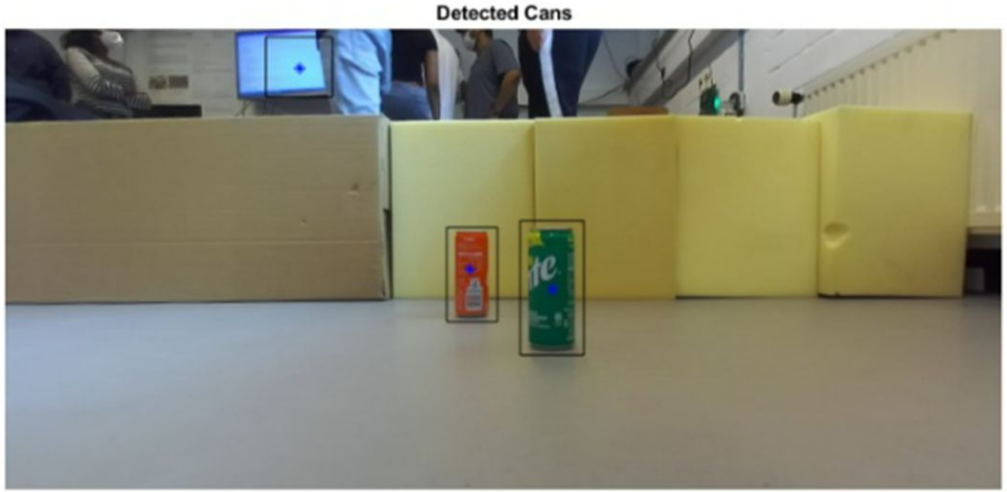


Depth estimation

- From the given left and right image, rectifying is done to remove distortion and horizontally align the image pairs.
- Creating disparity map by matching every pixel in the left hand image to the corresponding pixel in right hand image.
- With depth estimation 3D reconstruction can be done
- Applying mask

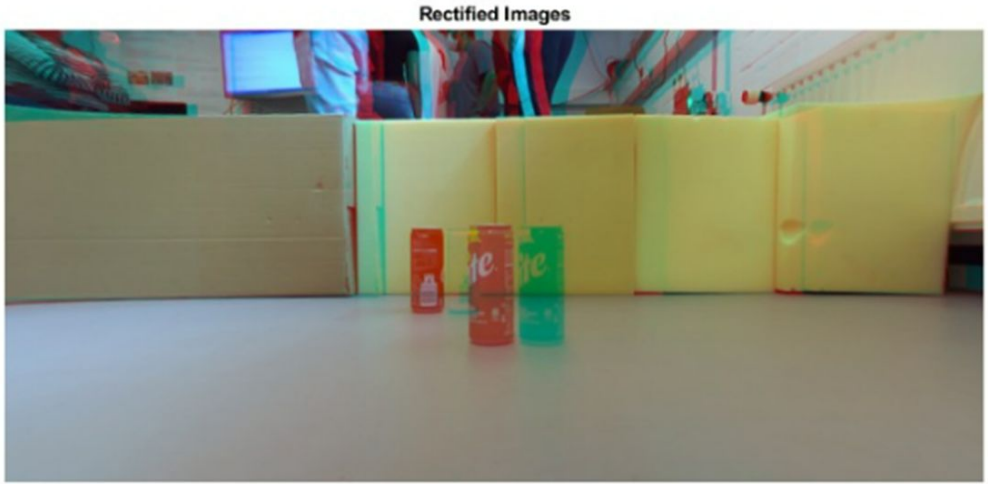
Depth extraction

Detected image



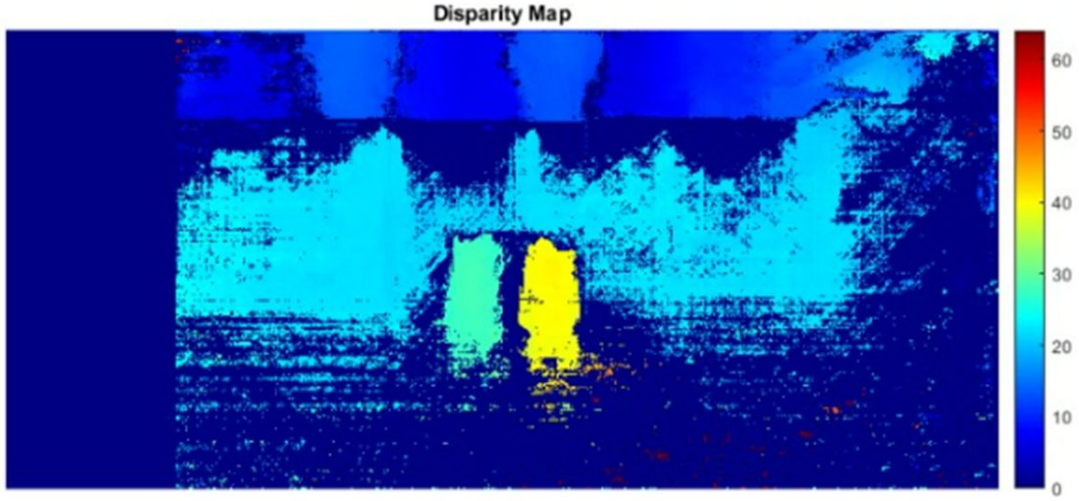
Depth extraction

Rectified image



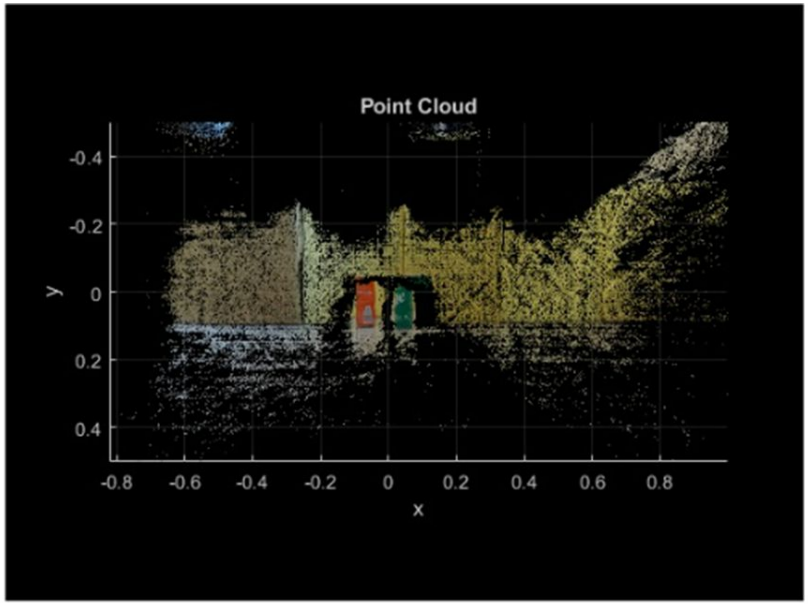
Depth extraction

Disparity map



Depth extraction

Point cloud



Depth extraction

- Thresholding point cloud to remove noise
- Converting to meters and creating point cloud object
- Finding the centroids of detected people
- Finding the 3D coordinates of the centroid

Appendix

Github Repository link

Bag files link