MRT Assignment 2

Harshit Somani

October 21, 2024

1 Introduction

Learnt to implement interfaces, clients, servers and ArUco marker detection using OpenCV.

The game plan was to implement the task for images and then extend the same logic to videos frame-by-frame.

2 Working with Images

2.1 Making the Interface

This includes two main things, the `msg` and `srv` files.

- The `ProcessImage.srv` file is

  ```
  sensor_msgs/Image img
  string path
  bool imgorvid #0 for img, 1 for vid
  uint16 frame
  ---
  ImageBounds imgbounds
  ```

  The request from the client to the server includes the Image data `img`, the path of the file being read `path`, whether the request is for a image or video `imgorvid` and the frame number of the video `frame`.

  The response from the server is a custom ROS2 message `ImageBounds` which we will define in the `msg` directory.

- The `ImageBounds.msg` message includes two arrays, `uint64[] ids` which stores the IDs of the ArUco markers detected in the image, and `PointArray[]` which is an array of the custom `PointArray` messages.

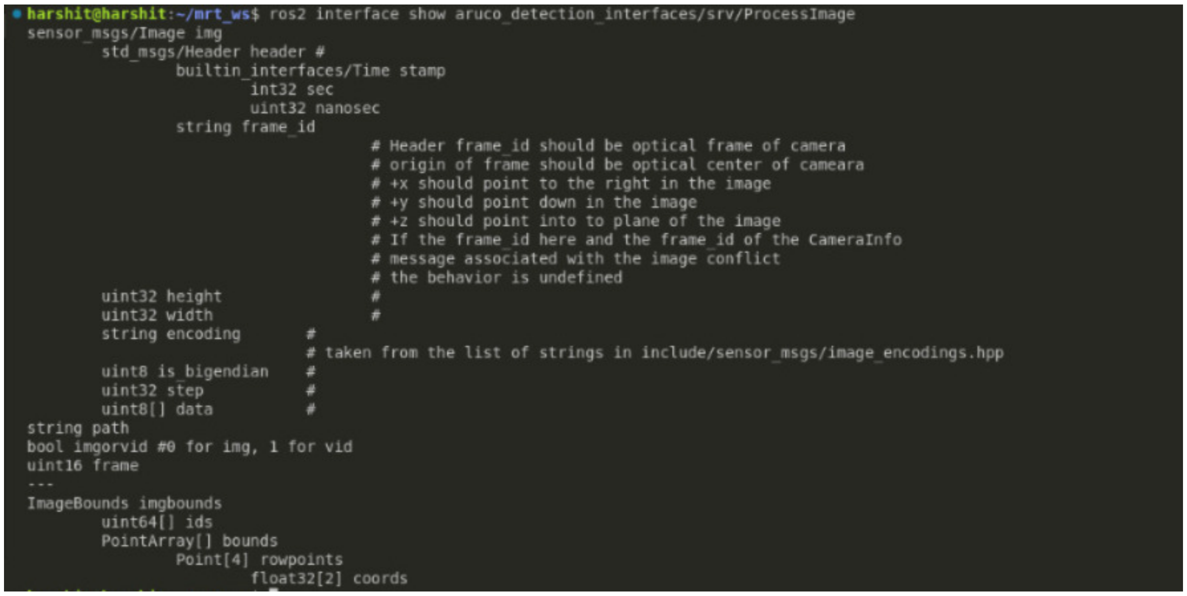
  ```
  uint64[] ids
  PointArray[] bounds
  ```

- The `PointArray.msg` message is a 4-element array of the custom `Point` messages. This defines the bounding co-ordinates for a specific ArUco marker.

  ```
  Point[4] rowpoints
  ```

- The `Point.msg` message is a 2-element array of `float32` values which define the x, y - co-ordinates of a point in the image.

  ```
  float32[2] coords
  ```



```
harshit@harshit:~/mrt_ws$ ros2 interface show aruco_detection_interfaces/srv/ProcessImage
sensor_msgs/Image img
        std_msgs/Header header #
                builtin_interfaces/Time stamp
                        int32 sec
                        uint32 nanosec
                string frame_id
                                     # Header frame_id should be optical frame of camera
                                     # origin of frame should be optical center of cameara
                                     # +x should point to the right in the image
                                     # +y should point down in the image
                                     # +z should point into to plane of the image
                                     # If the frame_id here and the frame_id of the CameraInfo
                                     # message associated with the image conflict
                                     # the behavior is undefined
        uint32 height                #
        uint32 width                 #
        string encoding       #
                              # taken from the list of strings in include/sensor_msgs/image_encodings.hpp
        uint8 is_bigendian    #
        uint32 step           #
        uint8[] data          #
string path
bool imgorvid #0 for img, 1 for vid
uint16 frame
---
ImageBounds imgbounds
        uint64[] ids
        PointArray[] bounds
                Point[4] rowpoints
                        float32[2] coords
```

Figure 1: Summary of the interface

2.2 Making the Server

The main format of making the server (`service.py`) is the same as the one in this tutorial.

First, we convert the incoming image message (`request.img`) to a OpenCV image for processing.

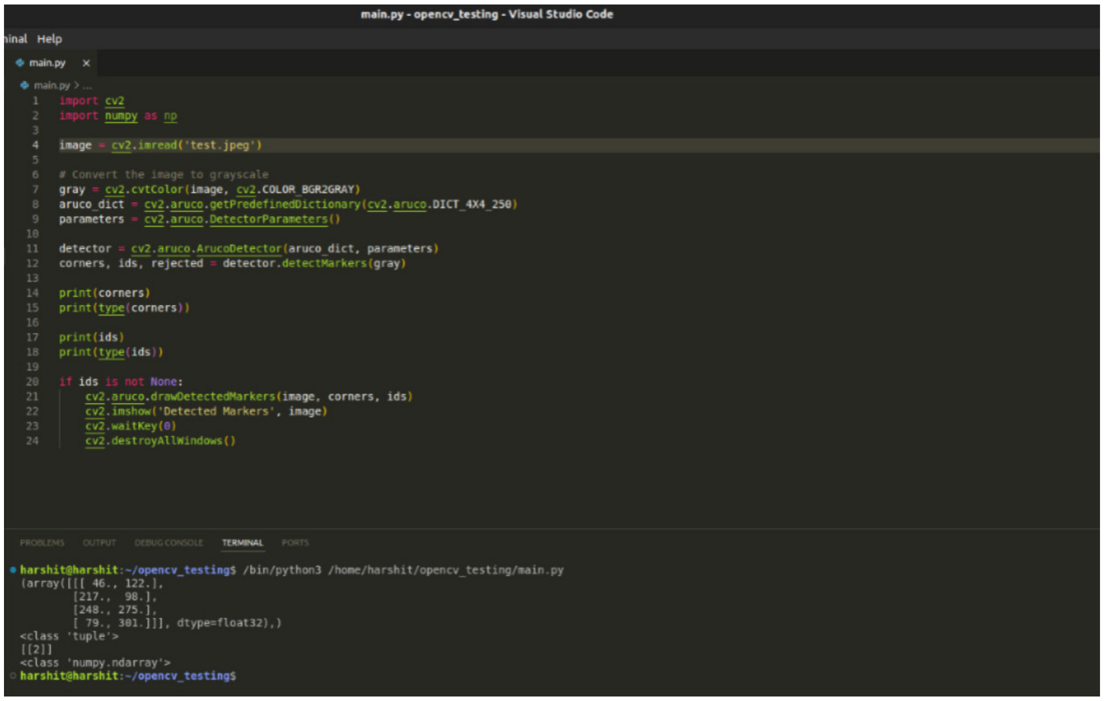
```
cv_image = bridge.imgmsg_to_cv2(request.img, desired_encoding='passthrough')
```

The line

```
corners, ids, rejected = detector.detectMarkers(cv_image)
```

defines the variables `corners` and `ids` as the arrays which store the bounding boxes and the IDs of the ArUco markers detected.

The structure of `corners` and `ids` can be checked using a OpenCV testing code.



```python
import cv2
import numpy as np

image = cv2.imread('test.jpeg')

# Convert the image to grayscale
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
aruco_dict = cv2.aruco.getPredefinedDictionary(cv2.aruco.DICT_4X4_250)
parameters = cv2.aruco.DetectorParameters()

detector = cv2.aruco.ArucoDetector(aruco_dict, parameters)
corners, ids, rejected = detector.detectMarkers(gray)

print(corners)
print(type(corners))

print(ids)
print(type(ids))

if ids is not None:
    cv2.aruco.drawDetectedMarkers(image, corners, ids)
    cv2.imshow('Detected Markers', image)
    cv2.waitKey(0)
    cv2.destroyAllWindows()
```

```
harshit@harshit:~/opencv_testing$ /bin/python3 /home/harshit/opencv_testing/main.py
(array([[[ 46., 122.],
        [217.,  98.],
        [248., 275.],
        [ 79., 301.]]], dtype=float32),)
<class 'tuple'>
[[2]]
<class 'numpy.ndarray'>
harshit@harshit:~/opencv_testing$
```

Figure 2: The data types and structure of the arrays `corners` and `ids`

The `corners` array is a tuple of 3D `numpy` arrays and `ids` is a 2D `numpy` array of the IDs.

The following code sets the `response.imgbounds.bounds` in the desired format:

```
for i in range(len(corners)):
    box = PointArray()
    for j in range(4):
        p = Point()
        p.coords = corners[i][0][j]
        box.rowpoints[j] = p
    response.imgbounds.bounds.append(box)
```

This obviously requires including the `PointArray` and `Point` classes.

```
from aruco_detection_interfaces.msg import Point
from aruco_detection_interfaces.msg import PointArray
```

The following code sets the `response.imgbounds.ids` in the desired format:

```
if ids is None:
    pass
else:
    response.imgbounds.ids = ids.ravel().tolist()
```

Checking if `ids` is crucial because if there are no ArUco images in the given image, the OpenCV detector returns an object of type `None` instead of `numpy.ndarray` for `ids`. (Note that this is not required for `corners` because in that case, it returns an empty tuple)

The following code tells the server to log the incoming response.

```
output_string = 'Incoming Request: '
if(request.imgorvid):
    output_string += "Video Path: " + request.path + " Frame Number: " + str(request.frame)
else:
    output_string += "Image Path: " + request.path

self.get_logger().info(output_string)
```

For example:

```
[INFO] [1729534848.565129227] [service]: Incoming Request: Image Path: /home/harshit/mrt_ws/
content/test.jpg
```

or

```
[INFO] [1729534903.776707848] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/
content/Aruco_1.MOV Frame Number: 60
```

2.3 Making the Client

The main format of making the client (`client_img.py`) is the same as the one in this tutorial

The following code requests the user for the image path as the input and converts the image to a ROS2 image message and sets the parameters to their required values.

```
def send_request(self):
    img_path = str(sys.argv[1])
    gray = cv2.cvtColor(cv2.imread(img_path), cv2.COLOR_BGR2GRAY)
    image_message = bridge.cv2_to_imgmsg(gray, encoding="passthrough")
    self.req.img = image_message
    self.req.path = img_path
    self.req.imgorvid = False
    self.future = self.cli.call_async(self.req)
    rclpy.spin_until_future_complete(self, self.future)
    return self.future.result()
```

We also convert the image to `GRAY` encoding to make the detection of ArUco markers easier.

The following code just tells the client to log the request and the received response from the server.

```
numAruco = len(response.imgbounds.ids)
output_string = "Number of ArUco markers detected: " + str(numAruco) + '\n'
for i in range(numAruco):
    output_string += "Marker " + str(i+1) + ": ID = " + str(response.imgbounds.ids[i])
                        + " Bounding Borders = "
    for j in range(4):
        output_string += '(' + ', '.join(map(str, response.imgbounds.bounds[i].rowpoints[j].coords))
                            + ') '
    output_string += '\n'

minimal_client.get_logger().info(output_string)
```

For example:

```
[INFO] [1729534848.577088680] [client_img_async]: Number of ArUco markers detected: 1
Marker 1: ID = 17 Bounding Borders = (370.0, 67.0) (392.0, 71.0) (389.0, 89.0) (368.0, 85.0)
```

Or

```
[INFO] [1729534903.777717312] [client_vid_async]: Frame 60
Number of ArUco markers detected: 4
Marker 1: ID = 1 Bounding Borders = (310.0, 456.0) (431.0, 470.0) (415.0, 584.0) (295.0, 572.0)
Marker 2: ID = 3 Bounding Borders = (1386.0, 491.0) (1504.0, 490.0) (1508.0, 611.0) (1392.0, 606.0)
Marker 3: ID = 0 Bounding Borders = (655.0, 475.0) (771.0, 468.0) (776.0, 581.0) (660.0, 590.0)
Marker 4: ID = 2 Bounding Borders = (957.0, 486.0) (1067.0, 509.0) (1045.0, 621.0) (935.0, 599.0)
```

3 Working with Videos

See client_vid.py

The main idea was to open a video using OpenCV in Python and feed the frames of the video into the same server one by one (in the same way we processed images).

This required some changes to `client_img.py`.

The following code sends the request to the server for the given frame and sets the parameters to their required values.

```
def send_request(self, image_message, path, current_frame):
    self.req.img = image_message
    self.req.path = path
    self.req.frame = current_frame
    self.req.imgorvid = True
    self.future = self.cli.call_async(self.req)
    rclpy.spin_until_future_complete(self, self.future)
    return self.future.result()
```

The `image_message, path, current_frame` variables change for each frame, and hence are set as parameters to the `send_request` function.

In the `main` function, we request the path to the video from the user and start looping through the video frame-by-frame.

```
vid_path = str(sys.argv[1])
cap = cv2.VideoCapture(vid_path)
if cap.isOpened():
    current_frame = 0
    while True:
        ret, frame = cap.read()
        if ret:
            if current_frame % 15 == 0:
                gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
                image_message = bridge.cv2_to_imgmsg(gray, encoding='passthrough')

                response = minimal_client.send_request(image_message, vid_path, current_frame)

                while rclpy.ok():
                    rclpy.spin_once(minimal_client)
                    if minimal_client.future.done():
                        try:
                            response = minimal_client.future.result()
                        except Exception as e:
                            minimal_client.get_logger().info('Service call failed %r' % (e,))
                        else:
                            numAruco = len(response.imgbounds.ids)
                            output_string = loggingData(numAruco, current_frame, response)
                            minimal_client.get_logger().info(output_string)
                        break
        else:
            break
        current_frame += 1
    cap.release()
cv2.destroyAllWindows()
```

Note that the request is sent to the server only when `current_frame % 15 == 0` (every 15 frames). This is to prevent logging spam and overloading the server.

The function `loggingData` returns the `output_string` which is logged.

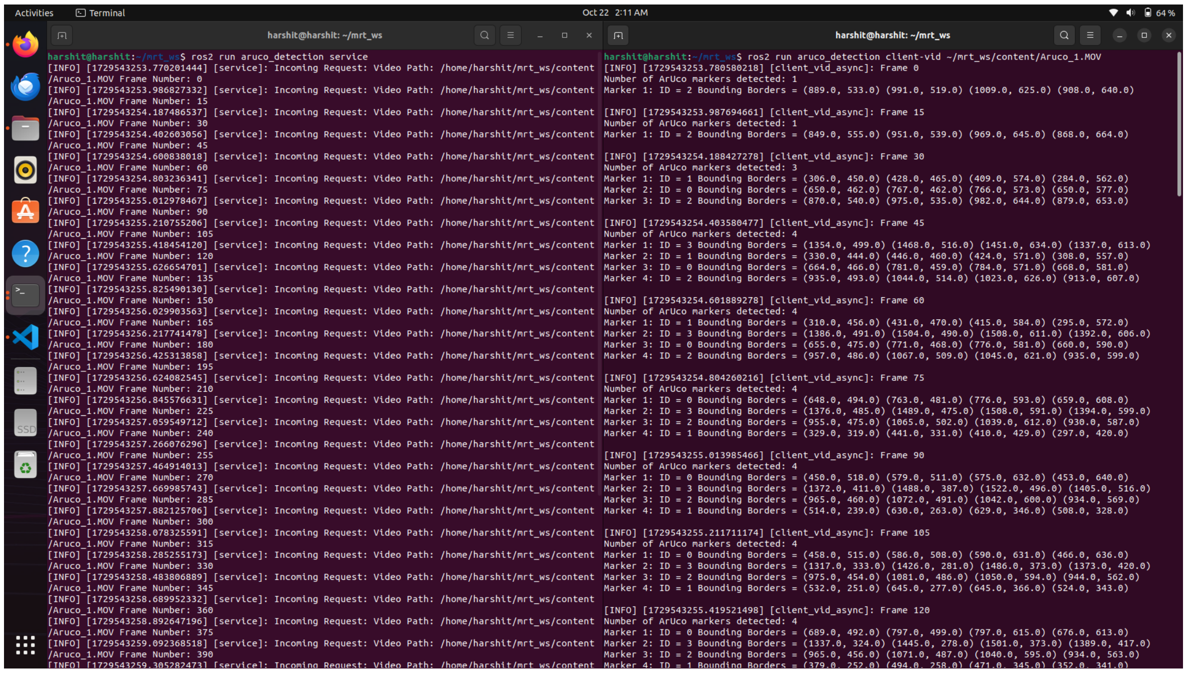
```
def loggingData(numAruco, current_frame, response):
    output_string = "Frame " + str(current_frame) + "\nNumber of ArUco markers detected: "
                    + str(numAruco) + '\n'
    for i in range(numAruco):
        output_string += "Marker " + str(i+1) + ": ID = " + str(response.imgbounds.ids[i])
                            + " Bounding Borders = "
        for j in range(4):
            output_string += '(' + ', '.join(map(str,
                            response.imgbounds.bounds[i].rowpoints[j].coords)) + ') '
        output_string += '\n'
    return output_string
```

A sample log of `client_vid.py` is

```
...
[INFO] [1729541625.405310906] [client_vid_async]: Frame 15
Number of ArUco markers detected: 1
Marker 1: ID = 2 Bounding Borders = (849.0, 555.0) (951.0, 539.0) (969.0, 645.0) (868.0, 664.0)

[INFO] [1729541625.607731882] [client_vid_async]: Frame 30
Number of ArUco markers detected: 3
Marker 1: ID = 1 Bounding Borders = (306.0, 450.0) (428.0, 465.0) (409.0, 574.0) (284.0, 562.0)
Marker 2: ID = 0 Bounding Borders = (650.0, 462.0) (767.0, 462.0) (766.0, 573.0) (650.0, 577.0)
Marker 3: ID = 2 Bounding Borders = (870.0, 540.0) (975.0, 535.0) (982.0, 644.0) (879.0, 653.0)

[INFO] [1729541625.815185483] [client_vid_async]: Frame 45
Number of ArUco markers detected: 4
Marker 1: ID = 3 Bounding Borders = (1354.0, 499.0) (1468.0, 516.0) (1451.0, 634.0) (1337.0, 613.0)
Marker 2: ID = 1 Bounding Borders = (330.0, 444.0) (446.0, 460.0) (424.0, 571.0) (308.0, 557.0)
Marker 3: ID = 0 Bounding Borders = (664.0, 466.0) (781.0, 459.0) (784.0, 571.0) (668.0, 581.0)
Marker 4: ID = 2 Bounding Borders = (935.0, 493.0) (1044.0, 514.0) (1023.0, 626.0) (913.0, 607.0)

[INFO] [1729541626.024209102] [client_vid_async]: Frame 60
Number of ArUco markers detected: 4
Marker 1: ID = 1 Bounding Borders = (310.0, 456.0) (431.0, 470.0) (415.0, 584.0) (295.0, 572.0)
Marker 2: ID = 3 Bounding Borders = (1386.0, 491.0) (1504.0, 490.0) (1508.0, 611.0) (1392.0, 606.0)
Marker 3: ID = 0 Bounding Borders = (655.0, 475.0) (771.0, 468.0) (776.0, 581.0) (660.0, 590.0)
Marker 4: ID = 2 Bounding Borders = (957.0, 486.0) (1067.0, 509.0) (1045.0, 621.0) (935.0, 599.0)
...
```

Both `client_img.py` and `client_vid.py` are added as nodes.

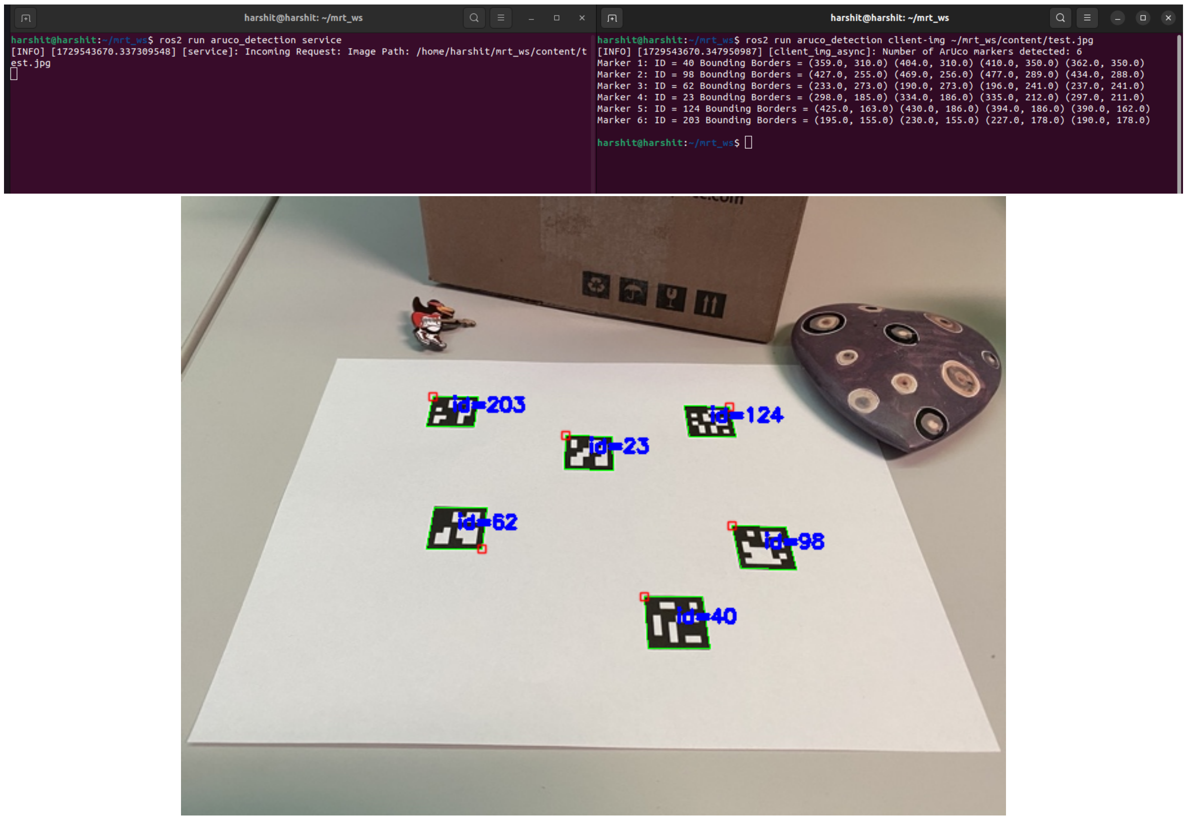
```
'console_scripts': [
    'service = aruco_detection.service:main',
    'client-vid = aruco_detection.client_vid:main',
    'client-img = aruco_detection.client_img:main',
],
```



```
harshit@harshit:~/mrt_ws$ ros2 run aruco_detection service
[INFO] [1729543253.770201444] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 0
[INFO] [1729543253.986827332] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 15
[INFO] [1729543254.187486537] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 30
[INFO] [1729543254.402603056] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 45
[INFO] [1729543254.600838018] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 60
[INFO] [1729543254.803236341] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 75
[INFO] [1729543255.012978467] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 90
[INFO] [1729543255.210755206] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 105
[INFO] [1729543255.418454120] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 120
[INFO] [1729543255.626654701] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 135
[INFO] [1729543255.825490130] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 150
[INFO] [1729543256.029903563] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 165
[INFO] [1729543256.217741478] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 180
[INFO] [1729543256.425313858] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 195
[INFO] [1729543256.624082545] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 210
[INFO] [1729543256.845576631] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 225
[INFO] [1729543257.059549712] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 240
[INFO] [1729543257.266076296] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 255
[INFO] [1729543257.464914013] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 270
[INFO] [1729543257.669985743] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 285
[INFO] [1729543257.882125706] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 300
[INFO] [1729543258.078325591] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 315
[INFO] [1729543258.285255173] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 330
[INFO] [1729543258.483806889] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 345
[INFO] [1729543258.689952332] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 360
[INFO] [1729543258.892647196] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 375
[INFO] [1729543259.092368518] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
/Aruco_1.MOV Frame Number: 390
[INFO] [1729543259.305282473] [service]: Incoming Request: Video Path: /home/harshit/mrt_ws/content
```

```
harshit@harshit:~/mrt_ws$ ros2 run aruco_detection client-vid ~/mrt_ws/content/Aruco_1.MOV
[INFO] [1729543253.780580218] [client_vid_async]: Frame 0
Number of ArUco markers detected: 1
Marker 1: ID = 2 Bounding Borders = (889.0, 533.0) (991.0, 519.0) (1009.0, 625.0) (908.0, 640.0)

[INFO] [1729543253.987694661] [client_vid_async]: Frame 15
Number of ArUco markers detected: 1
Marker 1: ID = 2 Bounding Borders = (849.0, 555.0) (951.0, 539.0) (969.0, 645.0) (868.0, 664.0)

[INFO] [1729543254.188427278] [client_vid_async]: Frame 30
Number of ArUco markers detected: 3
Marker 1: ID = 1 Bounding Borders = (306.0, 450.0) (428.0, 465.0) (409.0, 574.0) (284.0, 562.0)
Marker 2: ID = 0 Bounding Borders = (650.0, 462.0) (767.0, 462.0) (766.0, 573.0) (650.0, 577.0)
Marker 3: ID = 2 Bounding Borders = (870.0, 540.0) (975.0, 535.0) (982.0, 644.0) (879.0, 653.0)

[INFO] [1729543254.403580477] [client_vid_async]: Frame 45
Number of ArUco markers detected: 4
Marker 1: ID = 3 Bounding Borders = (1354.0, 499.0) (1468.0, 516.0) (1451.0, 634.0) (1337.0, 613.0)
Marker 2: ID = 1 Bounding Borders = (330.0, 444.0) (446.0, 460.0) (424.0, 571.0) (308.0, 557.0)
Marker 3: ID = 0 Bounding Borders = (664.0, 466.0) (781.0, 459.0) (784.0, 571.0) (668.0, 581.0)
Marker 4: ID = 2 Bounding Borders = (935.0, 493.0) (1044.0, 514.0) (1023.0, 626.0) (913.0, 607.0)

[INFO] [1729543254.601889278] [client_vid_async]: Frame 60
Number of ArUco markers detected: 4
Marker 1: ID = 1 Bounding Borders = (310.0, 456.0) (431.0, 470.0) (415.0, 584.0) (295.0, 572.0)
Marker 2: ID = 3 Bounding Borders = (1386.0, 491.0) (1504.0, 490.0) (1508.0, 611.0) (1392.0, 606.0)
Marker 3: ID = 0 Bounding Borders = (655.0, 475.0) (771.0, 468.0) (776.0, 581.0) (660.0, 590.0)
Marker 4: ID = 2 Bounding Borders = (957.0, 486.0) (1067.0, 509.0) (1045.0, 621.0) (935.0, 599.0)

[INFO] [1729543254.804260216] [client_vid_async]: Frame 75
Number of ArUco markers detected: 4
Marker 1: ID = 0 Bounding Borders = (648.0, 494.0) (763.0, 481.0) (776.0, 593.0) (659.0, 608.0)
Marker 2: ID = 3 Bounding Borders = (1376.0, 485.0) (1489.0, 475.0) (1508.0, 591.0) (1394.0, 599.0)
Marker 3: ID = 2 Bounding Borders = (955.0, 475.0) (1065.0, 502.0) (1039.0, 612.0) (930.0, 587.0)
Marker 4: ID = 1 Bounding Borders = (329.0, 319.0) (441.0, 331.0) (410.0, 429.0) (297.0, 420.0)

[INFO] [1729543255.013985466] [client_vid_async]: Frame 90
Number of ArUco markers detected: 4
Marker 1: ID = 0 Bounding Borders = (450.0, 518.0) (579.0, 511.0) (575.0, 632.0) (453.0, 640.0)
Marker 2: ID = 3 Bounding Borders = (1372.0, 411.0) (1488.0, 387.0) (1522.0, 496.0) (1405.0, 516.0)
Marker 3: ID = 2 Bounding Borders = (965.0, 460.0) (1072.0, 491.0) (1042.0, 600.0) (934.0, 569.0)
Marker 4: ID = 1 Bounding Borders = (514.0, 239.0) (630.0, 263.0) (629.0, 346.0) (508.0, 328.0)

[INFO] [1729543255.211711174] [client_vid_async]: Frame 105
Number of ArUco markers detected: 4
Marker 1: ID = 0 Bounding Borders = (458.0, 515.0) (586.0, 508.0) (590.0, 631.0) (466.0, 636.0)
Marker 2: ID = 3 Bounding Borders = (1317.0, 333.0) (1426.0, 281.0) (1486.0, 373.0) (1373.0, 420.0)
Marker 3: ID = 2 Bounding Borders = (975.0, 454.0) (1081.0, 486.0) (1050.0, 594.0) (944.0, 562.0)
Marker 4: ID = 1 Bounding Borders = (532.0, 251.0) (645.0, 277.0) (645.0, 366.0) (524.0, 343.0)

[INFO] [1729543255.419521498] [client_vid_async]: Frame 120
Number of ArUco markers detected: 4
Marker 1: ID = 0 Bounding Borders = (689.0, 492.0) (797.0, 499.0) (797.0, 615.0) (676.0, 613.0)
Marker 2: ID = 3 Bounding Borders = (1337.0, 324.0) (1445.0, 278.0) (1501.0, 373.0) (1389.0, 417.0)
Marker 3: ID = 2 Bounding Borders = (965.0, 456.0) (1071.0, 487.0) (1040.0, 595.0) (934.0, 563.0)
Marker 4: ID = 1 Bounding Borders = (379.0, 252.0) (494.0, 258.0) (471.0, 345.0) (352.0, 341.0)
```

Figure 3: Testing out Aruco_1.MOV using `client-vid`



```
harshit@harshit:~/mrt_ws$ ros2 run aruco_detection service
[INFO] [1729543670.337309548] [service]: Incoming Request: Image Path: /home/harshit/mrt_ws/content/t
est.jpg
```

```
harshit@harshit:~/mrt_ws$ ros2 run aruco_detection client-img ~/mrt_ws/content/test.jpg
[INFO] [1729543670.347950987] [client_img_async]: Number of ArUco markers detected: 6
Marker 1: ID = 40 Bounding Borders = (359.0, 310.0) (404.0, 310.0) (410.0, 350.0) (362.0, 350.0)
Marker 2: ID = 98 Bounding Borders = (427.0, 255.0) (469.0, 256.0) (477.0, 289.0) (434.0, 288.0)
Marker 3: ID = 62 Bounding Borders = (233.0, 273.0) (190.0, 273.0) (196.0, 241.0) (237.0, 241.0)
Marker 4: ID = 23 Bounding Borders = (298.0, 185.0) (334.0, 186.0) (335.0, 212.0) (297.0, 211.0)
Marker 5: ID = 124 Bounding Borders = (425.0, 163.0) (430.0, 186.0) (394.0, 186.0) (390.0, 162.0)
Marker 6: ID = 203 Bounding Borders = (195.0, 155.0) (230.0, 155.0) (227.0, 178.0) (190.0, 178.0)

harshit@harshit:~/mrt_ws$
```

Figure 4: Testing out an image using `client-img`