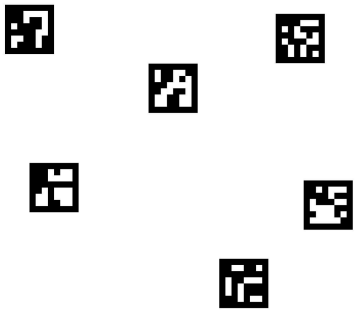
CS 5330 Project 4
Yunyu Guo
March 9 2025

This project is to calibrate a camera and then use the calibration to generate virtual objects in a scene. The program can detect a target and then place a virtual object in the scene relative to the target that moves and orients itself correctly given motion of the target.

1. **Detect and Extract Target Corners**

   Indicate in your report which type of target you are using. Describe any limitations or challenges the system has in locating the target.

   I use the ArUco Markers image from:
   https://docs.opencv.org/4.x/d5/dae/tutorial_aruco_detection.html



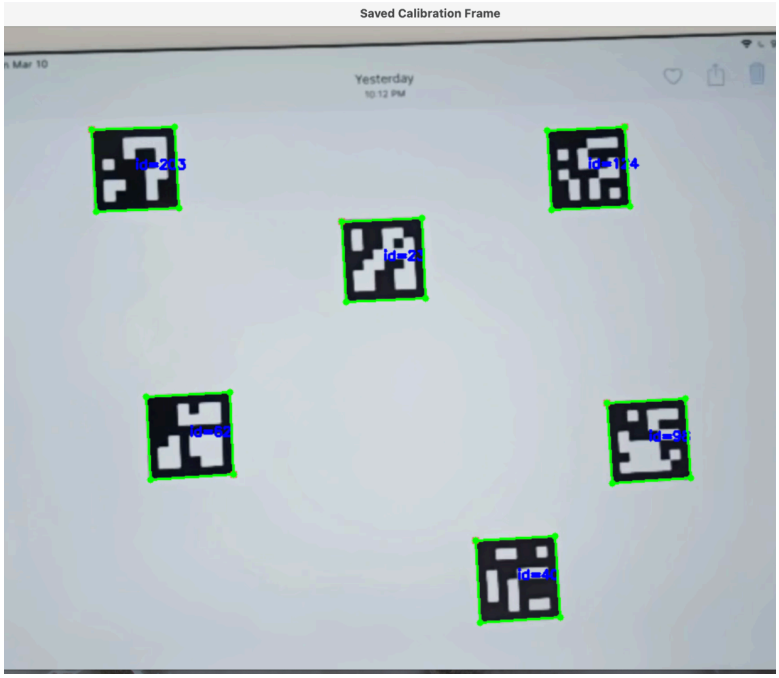
```
Number of markers detected: 6
First marker's first corner at: (40, 64)
```

It is able to identify the rotated markers. The small red square indicates the marker's top left corner. The challenges of locating the targets: the lighting conditions can affect detection; also motion blur can make detection difficult.

2. Select Calibration Images

Include a calibration image with chessboard/marker features highlighted in your project report.

```
mac@Mac build % ./calibration
2025-03-10 22:44:14.783 calibration[39378:21709969] +[IMKClient subclass]: chose IMKClient_Modern
2025-03-10 22:44:14.783 calibration[39378:21709969] +[IMKInputSession subclass]: chose IMKInputSess
ion_Modern
Saved calibration frame 1 with 24 corners
Collected 1 calibration frames
```



3. **Calibrate the Camera**

   Include your calibration matrix in your report, along with the corresponding re-projection error.

   3.1 uses standard cv::calibrateCamera in camera_caliration.cpp

```
// Perform calibration
std::vector<cv::Mat> rvecs, tvecs;
last_error = cv::calibrateCamera(
    point_list,
    corner_list,
    image_size,
    camera_matrix,
    distortion_coeffs,
    rvecs,
    tvecs,
    flags
);
```


```
Initial camera matrix:
[1, 0, 320;
 0, 1, 240;
 0, 0, 1]
Initial distortion coeffs:
[0;
 0;
 0;
 0;
 0]

Final camera matrix:
[1313.919463151876, 0, 432.3459586721955;
 0, 1313.919463151876, 140.2019995251815;
 0, 0, 1]
Final distortion coeffs:
[1.080819269118794;
 -6.238913559836588;
 0.032716049776395 29;
 -0.097696684592629 85;
 11.38716964279258]
Reprojection error: 0.159169 pixels
Calibration saved to camera_calibration.yml
Final calibration saved with error: 0.159169 pixels
```

```
Frame 5 saved.
Per-frame reprojection errors:
Frame 1: 1.06607 pixels
Frame 2: 0.199455 pixels
Frame 3: 8.682 pixels
Frame 4: 4.78391 pixels
Frame 5: 1.86705 pixels
```

Camera matrix: fx = fy = 1313.91: focal length in pixels same in x and y
Principal point: cx = 432.34, cy = 140.20, which are not very close to the initial estimates (324, 240) of the center of the image.
Distortion Coefficients [k1, k2, p1, p2, k3]:
k1 = 1.08
k2 = -6.24
p1 = 0.033
p2 = -0.098
k3 = 11.39
k1, k2, k3: Radial distortion coefficients
p1, p2: Tangential distortion coefficients
k2 and k3 are quite hight, indicating significant lens distortion.
Reprojection error: 0.159169 pixels, indicating high accuracy in the calibration.

However, when try to calibrate using 6 markers together, the reprojection error was more than 100 pixels.

3.2 use GridBoard in camera.cpp

```
// Calibrate camera using ArUco markers
double repError = cv::aruco::calibrateCameraAruco(
    allCornersConcatenated,
    allIdsConcatenated,
    markerCounterPerFrame,
    gridboard,
    image_size,
    camera_matrix,
    dist_coeffs,
    rvecs,
    tvecs,
    cv::CALIB_FIX_ASPECT_RATIO
);
```

```
=== Frame 7 captured ===

=== Camera Parameters before calibration ===
Camera Matrix:
[474.036543, 0.000000, 954.206731]
[0.000000, 460.037006, 507.320522]
[0.000000, 0.000000, 1.000000]

Distortion Coefficients:
[0.552695, -1.851799, 0.000000, 0.000000, 3.030175]
Calibration complete with error: 0.352179

=== Camera Parameters after calibration ===
Camera Matrix:
[474.036543, 0.000000, 954.206731]
[0.000000, 460.037006, 507.320522]
[0.000000, 0.000000, 1.000000]

Distortion Coefficients:
[0.552695, -1.851799, 0.000000, 0.000000, 3.030175]
Calibration updated with 7 frames
Current reprojection error: 0.352179 pixels

Final calibration results:
Total frames used: 7
Final reprojection error: 0.352179 pixels
```

flags: 8 (CALIB_ZERO_TANGENT_DIST flag was used)
Camera Matrix:
[fx 0 cx]
[0 fy cy]
[0 0 1]

fx = 474.04 (focal length in x direction)
fy = 460.04 (focal length in y direction), focal lengths in x and y are different, suggesting non-square pixels.
cx = 954.21 (principal point x-coordinate)
cy = 507.32 (principal point y-coordinate)

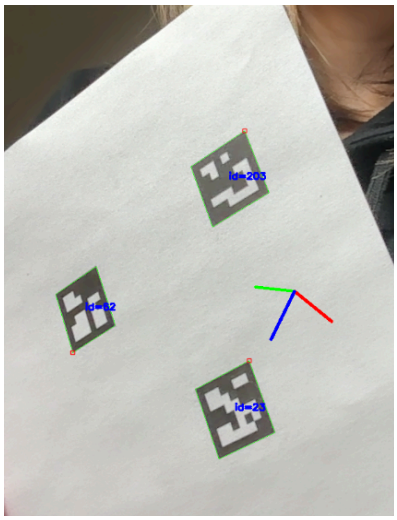
Distortion Coefficients (k1, k2, p1, p2, k3):
k1 = 0.5527 (1st radial distortion coefficient)
k2 = -1.8518 (2nd radial distortion coefficient)
p1 = 0.0000 (1st tangential distortion coefficient)
p2 = 0.0000 (2nd tangential distortion coefficient)
k3 = 3.0302 (3rd radial distortion coefficient)

Camera has significant radial distortion (k1, k2, k3 ≠ 0)
Tangential distortion is zero (p1 = p2 = 0) as specified in calibration flags

avg_reprojection_error: 0.3522 pixels

4. **Calculate Current Position of the Camera**

   Include in your report a description of how these values change as you move the camera side to side. Do they make sense?

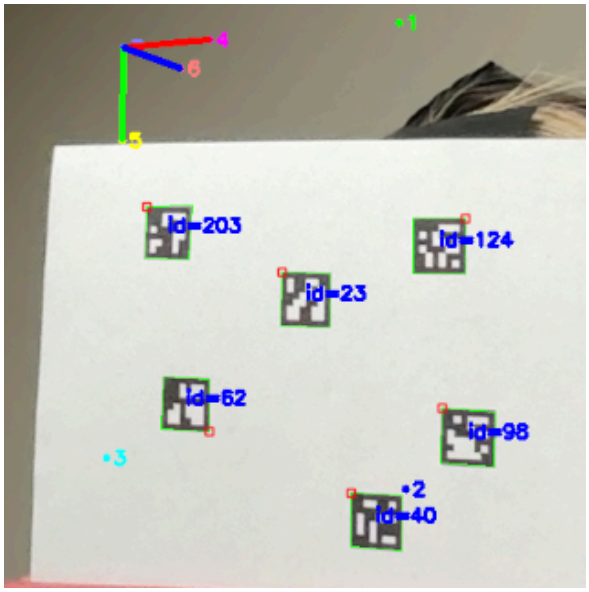



```
Camera Position (meters):
X: -0.413
Y: 0.136
Z: 0.455

Camera Rotation (degrees):
Roll : -39.4
Pitch: 9.1
Yaw  : -76.4

Camera Position (meters):
X: -0.068
Y: 0.107
Z: -0.040

Camera Rotation (degrees):
Roll : 111.7
Pitch: -21.1
Yaw  : -99.1

Camera Position (meters):
X: -0.075
Y: 0.108
Z: -0.024

Camera Rotation (degrees):
Roll : 113.5
Pitch: -21.7
Yaw  : -100.7
```

   The values make sense because:
   X position changes most significantly during side-to-side movement
   Yaw angle changes correspond to camera rotation

5. **Project Outside Corners or 3D Axes**

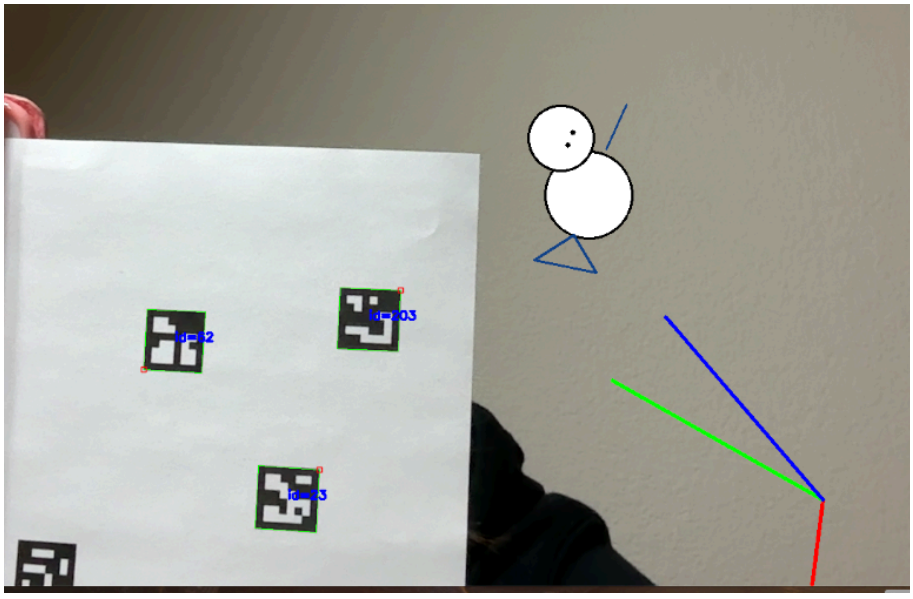
   Do the reprojected points/axes show up in the right places? Include at least one image from this step in your report.



6. **Create a Virtual Object**

   Describe your virtual object and take some screen shots and/or videos of your system in action for your project report.

   Snowman with 2 eyes, body filled with white colour, one hand straight and one hand on waist. 🙂



7. **Detect Robust Features**

   In your report, explain how you might be able to use those feature points as the basis for putting augmented reality into the image.

Harris Corner Detection Parameters:

1. minResponse (Threshold, default=200)

```
double minResponse = 200;  // Higher value = fewer corners
```

Higher values:

More selective corner detection

Only strongest corners detected

Better noise rejection

Lower values:

More corners detected

May include weaker corners

More susceptible to noise

2. k (Harris parameter, default=0.04)

```
double k = 0.04;  // Harris corner sensitivity
```

Higher values:

More permissive corner detection

May detect edges as corners

Lower values:

Stricter corner detection

Better edge rejection

3. blockSize and apertureSize

```
int blockSize = 2;      // Neighborhood size
int apertureSize = 3;   // Sobel operator size
```

Larger values:

Better for larger features

More robust to noise

But less precise locations

Smaller values:

Better for fine details

More precise locations

More sensitive to noise

SURF Feature Parameters:

1. minHessian (Threshold, default=800)

```
int minHessian = 800;  // Higher value = fewer features
```

Higher values:

More distinctive features

Fewer but more reliable points

Better for strong textures

Lower values:

More features detected

Includes weaker features

Better for subtle textures

2. SURF Settings

surf_detector->setExtended(true); // More distinctive features

surf_detector->setUpright(false); // Rotation invariant

Extended:

true = 128-dimensional descriptors

false = 64-dimensional descriptors

Upright:

true = faster but no rotation invariance

false = slower but handles rotation

Reference:

https://docs.opencv.org/4.x/d5/dae/tutorial_aruco_detection.html

https://docs.opencv.org/4.x/d2/d1a/classcv_1_1aruco_1_1ArucoDetector.html#a0c1d14251bf1cbb06277f49cfe1c9b61

https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html#ga3207604e4b1a1758aa66acb6ed5aa65d

https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html#ga3207604e4b1a1758aa66acb6ed5aa65d

https://github.com/Firifire/OpenCvSharp-UWP/blob/master/OpenCV/opencv-hololens-contrib/modules/aruco/samples/calibrate_camera.cpp