CAMERA CALIBRATION

Charmaine S. Tolledo

CONTENTS

Objectives

Methods

Results

Analysis

Reflection

OBJECTIVES

01

Calibration

To calibrate the camera accurately using the Camera Calibration Toolbox by Jean Bouguet in Matlab.

02

Intrinsic Parameters

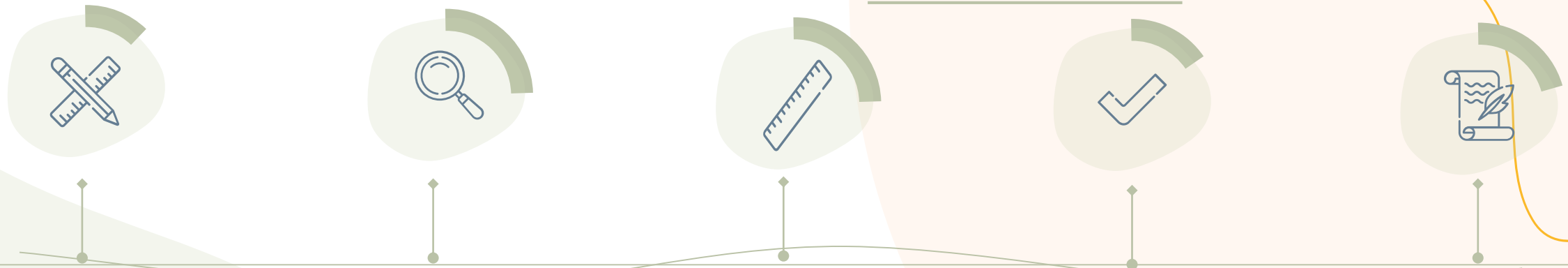
To determine the intrinsic camera parameters through the calibration process.

03

Distortion Correction

To rectify any distortion present in the images taken by the camera, if required.

METHODOLOGY



TOOLBOX INSTALLATION

- Download and install the Camera Calibration Toolbox by Jean Bouguet.
- Ensure proper configuration in the Matlab environment.

DATA COLLECTION AND DOCUMENTATION

- Capture multiple images of the calibration checkerboard with the target camera.
- Vary the checkerboard's positions and orientations within the camera's field of view.
- Save these images in a single folder.
- Use filenames that begin with a common base name followed by sequential numbers and the file extension.

TOOLBOX USAGE & CALIBRATION PROCESS

- Launch the Camera Calibration Toolbox in Matlab.
- Within the toolbox, load the collected images.
- The toolbox will automatically detect and extract checkerboard corners in the images.
- Proceed with the camera calibration process, which includes estimating intrinsic parameters and evaluating calibration quality.

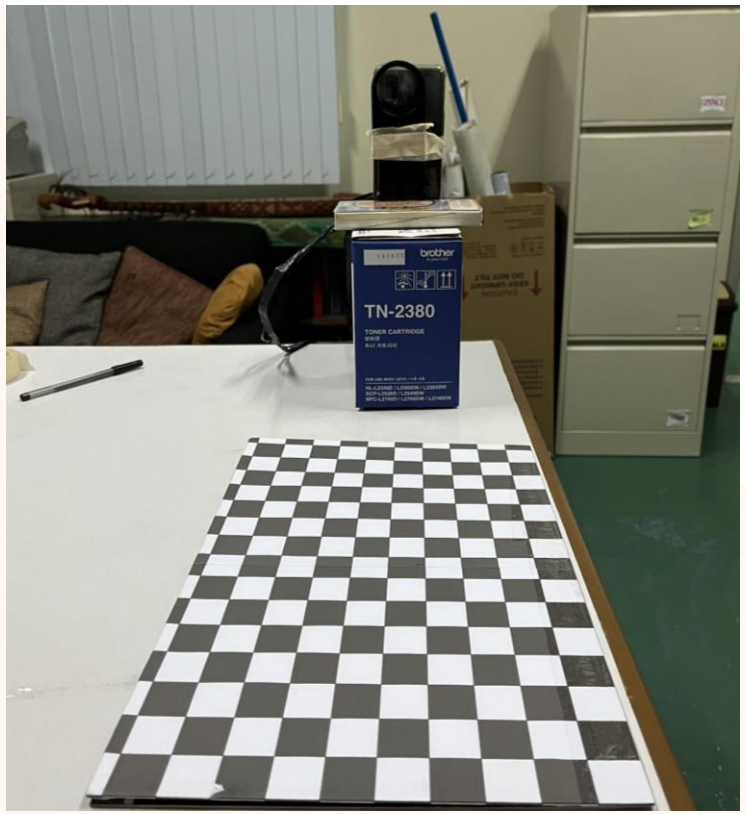
SAVE PARAMETERS

- After successful calibration, save the intrinsic camera parameters for analysis.

DISTORTION CORRECTION

- If the images have pincushion or barrel distortion, undistort the image using the toolbox.

SETUP



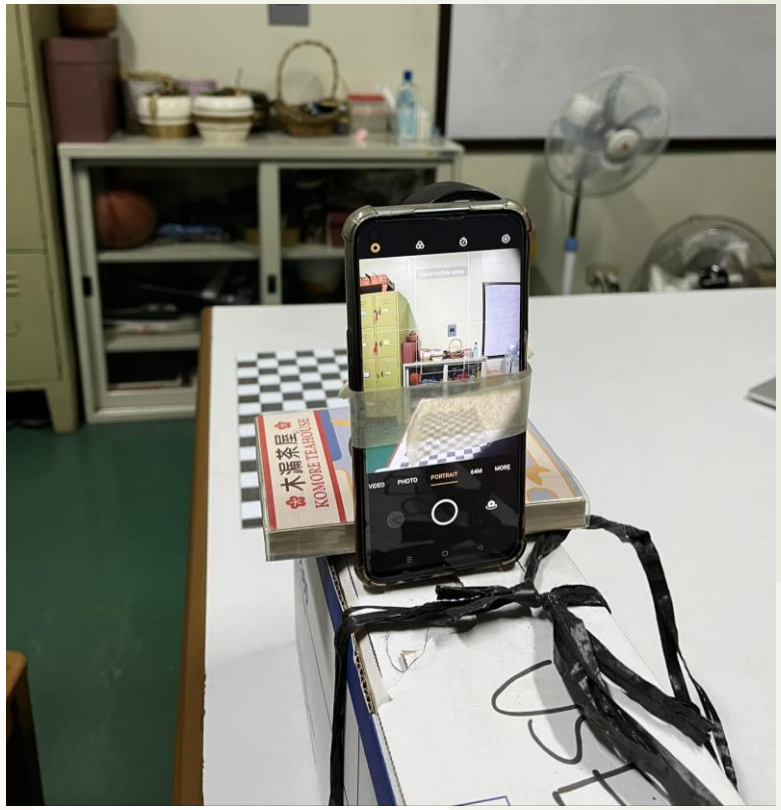


Figure 1
Camera Calibration Setup

In my setup, I used my smartphone's camera for calibration. To maintain a consistent perspective, I elevated the phone on a sturdy box and placed a book underneath for added stability. I intentionally introduced distortion by taping the convex lens at the front of my camera lens. On the same table as the elevated phone, I positioned a Tsai grid/checkerboard, which served as my calibration target with known geometric features.

SAMPLE IMAGES COLLECTED

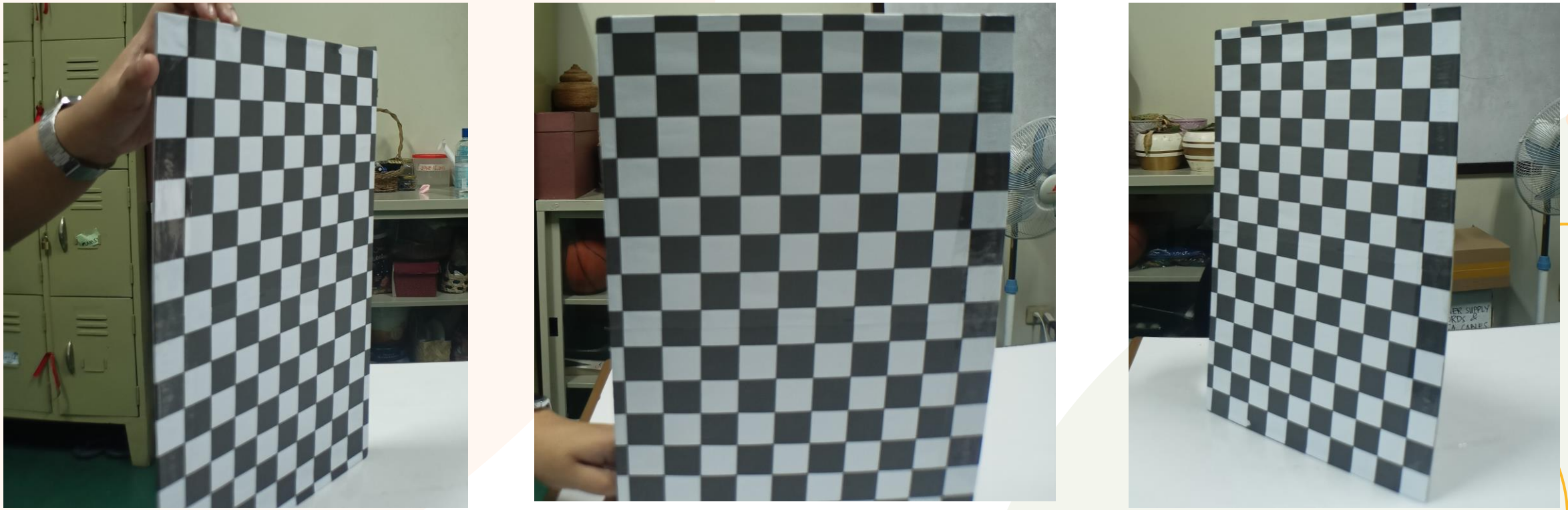


Figure 2
(a) Image 10, (b) Image 12, and (c) Image 15 from the dataset

These three images—numbers 10, 12, and 15—demonstrate visible distortions intentionally introduced through a convex lens. They are part of the 20 images collected, and these three specifically showcase the pronounced distortions. In the end, I'll also show you the undistorted versions of these images.

MAIN CALIBRATION PROCESS USING MATLAB CAMERA CALIBRATION TOOLBOX

READING IMAGES

```
>> calib_gui

.                 camera_calib11.jpg  camera_calib15.jpg  camera_calib19.jpg  camera_calib4.jpg   camera_calib8.jpg
..                camera_calib12.jpg  camera_calib16.jpg  camera_calib2.jpg   camera_calib5.jpg   camera_calib9.jpg
camera_calib1.jpg   camera_calib13.jpg  camera_calib17.jpg  camera_calib20.jpg  camera_calib6.jpg
camera_calib10.jpg  camera_calib14.jpg  camera_calib18.jpg  camera_calib3.jpg   camera_calib7.jpg


Basename camera calibration images (without number nor suffix): camera_calib
Image format: ([]='r'='ras', 'b'='bmp', 't'='tif', 'p'='pgm', 'j'='jpg', 'm'='ppm') j
Loading image 1...2...3...4...5...6...7...8...9...10...11...12...13...14...15...16...17...18...19...20...
done
```

Figure 3
Screenshot of the calib_gui interface used

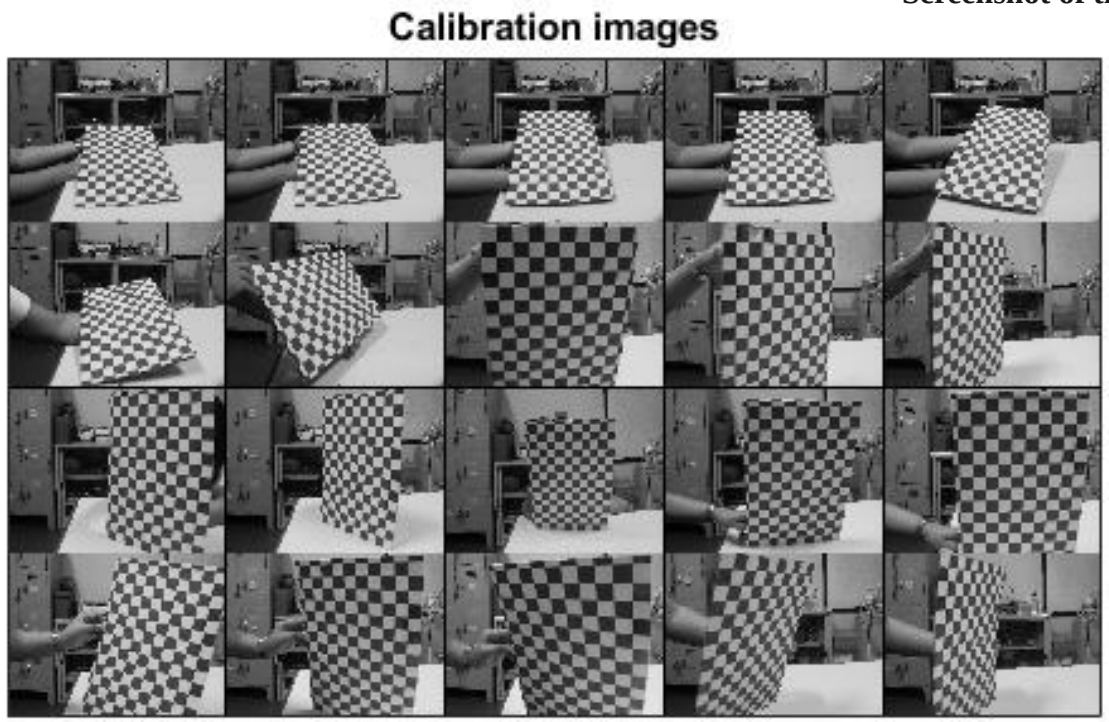


Figure 4
Thumbnail view of the 20 images used for camera calibration

To start, I clicked on "Image Names" in the Camera Calibration Tool. A window popped up where I named my calibration images as “camera_calib" and specified their format as “jpg." When I hit "OK," the tool loaded all 20 images into memory and labeled them as I_1, I_2, and so on, which made them easy to access. (shown in figure 3a) Plus, MATLAB displayed a handy thumbnail preview of my images for a quick overview (shown in figure 3b).

EXTRACTING GRID CORNERS

```
Extraction of the grid corners on the images
Number(s) of image(s) to process ([] = all images) =
Window size for corner finder (wintx and winty):
wintx ([] = 24) =
winty ([] = 24) =
Window size = 49x49
Do you want to use the automatic square counting mechanism (0=[]=default)
  or do you always want to enter the number of squares manually (1,other)?

Processing image 1...
Using (wintx,winty)=(24,24) - Window size = 49x49      (Note: To reset the window size, run script clearwin)
Click on the four extreme corners of the rectangular complete pattern (the first clicked corner is the origin)...
```

Figure 5
screenshot of the corner extraction process

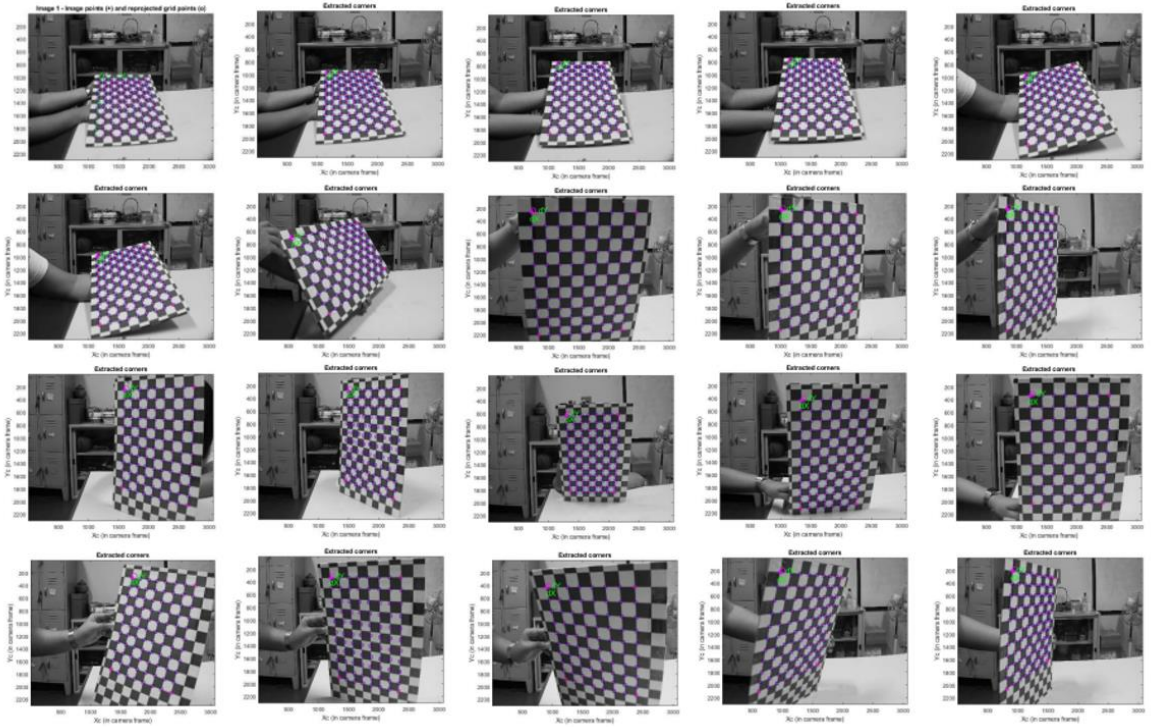


Figure 6
Corners successfully extracted from all 20 calibration images

After loading the images, the next step was to extract the grid corners. I initiated this process by clicking on the "Extract Grid Corners" function within the Camera Calibration Tool. Personally, I found this to be a meticulous process. It required me to manually identify and mark the four corners of the calibration grid within each image. Achieving precision was crucial, and I remember clearly that I spent time on this step. In some instances, I had to repeat the process up to 10 times to be satisfied. This manual process can indeed be quite challenging and time-consuming, and I believe that this is one of the primary sources of potential errors in the calibration procedure.

FIRST CALIBRATION

```
Aspect ratio optimized (est_aspect_ratio = 1) -> both components of fc are estimated (DEFAULT).
Principal point optimized (center_optim=1) - (DEFAULT). To reject principal point, set center_optim=0
Skew not optimized (est_alpha=0) - (DEFAULT)
Distortion not fully estimated (defined by the variable est_dist):
    Sixth order distortion not estimated (est_dist(5)=0) - (DEFAULT) .
Initialization of the principal point at the center of the image.
Initialization of the intrinsic parameters using the vanishing points of planar patterns.

Initialization of the intrinsic parameters - Number of images: 20


Calibration parameters after initialization:

Focal Length:          fc = [ 3020.04880   3020.04880 ]
Principal point:       cc = [ 1535.50000   1151.50000 ]
Skew:             alpha_c = [ 0.00000 ]   => angle of pixel = 90.00000 degrees
Distortion:            kc = [ 0.00000   0.00000   0.00000   0.00000   0.00000 ]

Main calibration optimization procedure - Number of images: 20
Gradient descent iterations: 1...2...3...4...5...6...7...8...9...10...11...12...13...done
Estimation of uncertainties...done
```

```
Calibration results after optimization (with uncertainties):

Focal Length:          fc = [ 3070.83620   3130.58245 ] ± [ 18.52862   21.73556 ]
Principal point:       cc = [ 1475.31397   498.86527 ] ± [ 27.72938   38.28153 ]
Skew:             alpha_c = [ 0.00000 ] ± [ 0.00000 ]   => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:            kc = [ -0.13222   -0.02046   -0.01005   -0.01660  0.00000 ] ± [ 0.02359   0.03876   0.00337   0.00177  0.00000 ]
Pixel error:          err = [ 3.03518   4.92637 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).
```

Figure 7
Calibration parameters result in (a) initialization step and (b) non-linear optimization step

After extracting the grid corners, I clicked on the "Calibration" button in the toolbox to initiate calibration process. This calibration has two steps: an initialization step and a non-linear optimization step. During initialization, calibration parameters are determined without considering lens distortion (as shown above), while the subsequent optimization phase accounts for distortion effects to minimize reprojection errors. At this stage, we can observe that the obtained error is relatively high which indicates the need for further refinement to achieve optimal calibration accuracy (we'll visualize this error on the next slide)

VISUALIZING CALIBRATION RESULTS AND EXTRINSIC PARAMETERS

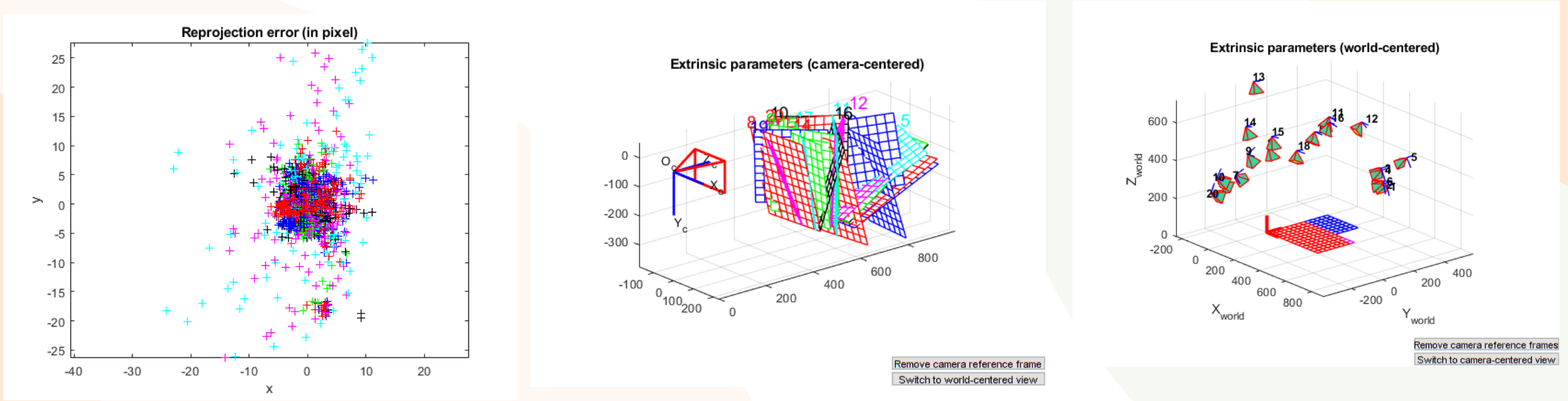


Figure 8
(a) Reprojection Errors of the calibration parameters; the extrinsic parameters for (b) camera-centered view and (c) world-centered view

Here, I used the 'Reproject on Images' option to visualize the calibration accuracy through the reprojection error. The plot (as shown above) displayed a certain degree of scatter, indicating that there were discrepancies between the predicted and observed image points. I also clicked the ‘Show extrinsic’ in which gave me these two figures above — camera-centered and world-centered views of extrinsic parameters. This is precisely what I pointed out earlier – the manual extraction of corners can introduce errors, particularly on highly distorted images. Look at the next slides to see how we address this.

REFINING CALIBRATION

```
Re-extraction of the grid corners on the images (after first calibration)
Window size for corner finder (wintx and winty):
wintx ([] = 5) =
winty ([] = 5) =
Window size = 11x11
Number(s) of image(s) to process ([] = all images) =
Use the projection of 3D grid or manual click ([]=auto, other=manual):
Processing image 1...2...3...4...5...6...7...8...9...10...11...12...13...14...15...16...17...18...19...20...
done

Aspect ratio optimized (est_aspect_ratio = 1) -> both components of fc are estimated (DEFAULT).
Principal point optimized (center_optim=1) - (DEFAULT). To reject principal point, set center_optim=0
Skew not optimized (est_alpha=0) - (DEFAULT)
Distortion not fully estimated (defined by the variable est_dist):
    Sixth order distortion not estimated (est_dist(5)=0) - (DEFAULT) .

Main calibration optimization procedure - Number of images: 20
Gradient descent iterations: 1...2...3...4...5...6...7...8...9...done
Estimation of uncertainties...done


Calibration results after optimization (with uncertainties):

Focal Length:          fc = [ 3059.21356   3121.61349 ] ± [ 7.87406   9.40182 ]
Principal point:       cc = [ 1483.90464   470.04699 ] ± [ 11.55036   16.23853 ]
Skew:             alpha_c = [ 0.00000 ] ± [ 0.00000  ]   => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:            kc = [ -0.12383   -0.02489   -0.01187   -0.01577  0.00000 ] ± [ 0.01012   0.01574   0.00151   0.00073  0.00000 ]
Pixel error:          err = [ 1.66677   1.81821 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).
```

Figure 9
Screenshot of the second calibration process and the calibration parameters result

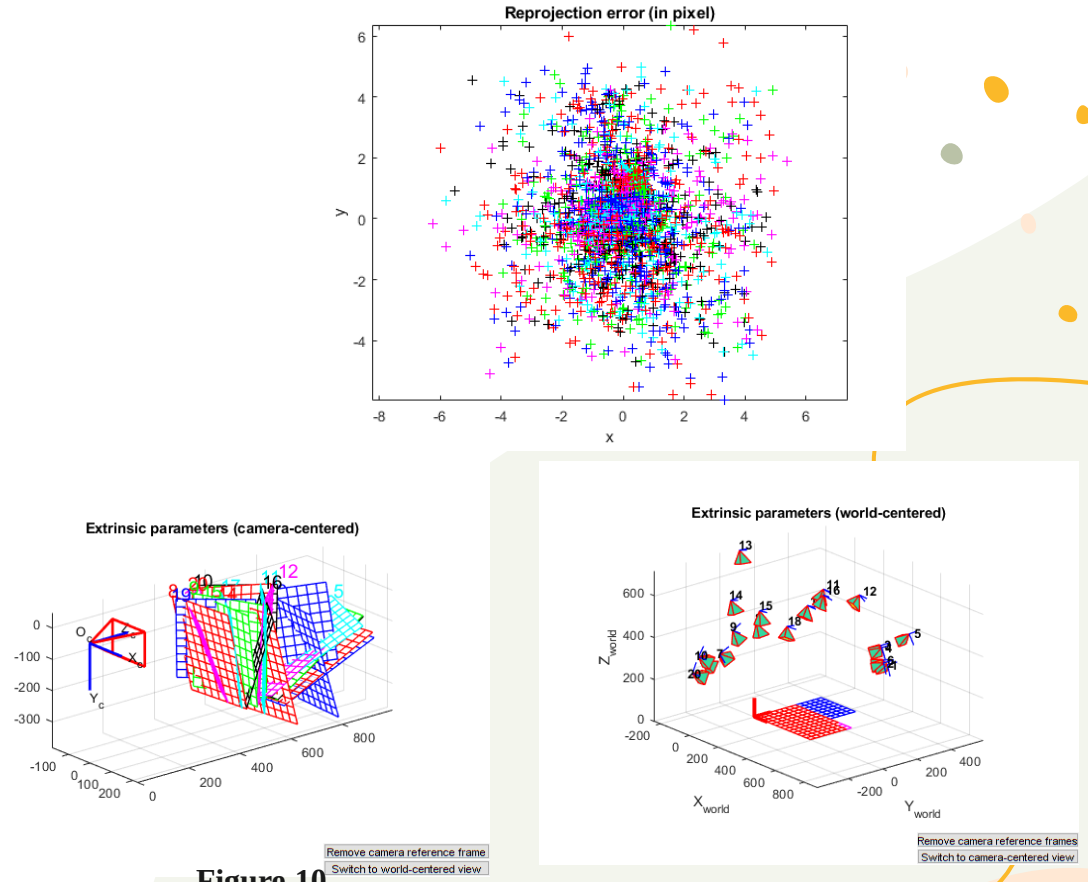


Figure 10
(a) Reprojection Errors of the calibration parameters; the extrinsic parameters for (b) camera-centered view and (c) world-centered view

To refine calibration, we clicked on the 'Recomp. Corners' button in the Camera Calibration tool for us to recompute the image corners on all images automatically. Afterward, I repeated the calibration process. As seen above, our calibration refinement has resulted in a reduced pixel error compared to the initial calibration round. We can also see the intrinsic and extrinsic parameters above.

ERROR ANALYSIS AND REFINEMENT

```
Selected image: 9
Selected point index: 86
Pattern coordinates (in units of (dX,dY)): (X,Y)=(8,1)
Image coordinates (in pixel): (1168.32,1775.64)
Pixel error = (4.89128,4.23248)
Window size: (wintx,winty) = (5,5)
```

Figure 11
Screenshot of the result of analyse error process

```
Re-extraction of the grid corners on the images (after first calibration)
Window size for corner finder (wintx and winty):
wintx ([] = 5) = 9
winty ([] = 5) = 9
Window size = 19x19
Number(s) of image(s) to process ([] = all images) = [1:4 6 9:17]
Use the projection of 3D grid or manual click ([]=auto, other=manual):
Processing image 1...2...3...4...6...9...10...11...12...13...14...15...16...17...
done
```

Figure 12a
First call of Corner Re-computation

```
Re-extraction of the grid corners on the images (after first calibration)
Window size for corner finder (wintx and winty):
wintx ([] = 5) = 8
winty ([] = 5) = 8
Window size = 17x17
Number(s) of image(s) to process ([] = all images) = 18
Use the projection of 3D grid or manual click ([]=auto, other=manual):
Processing image 18...
done
```

Figure 12b
Second call of Corner Re-computation

```
Re-extraction of the grid corners on the images (after first calibration)
Window size for corner finder (wintx and winty):
wintx ([] = 5) = 7
winty ([] = 5) = 7
Window size = 15x15
Number(s) of image(s) to process ([] = all images) = [5 7 8 19]
Use the projection of 3D grid or manual click ([]=auto, other=manual):
Processing image 5...7...8...19...
done
```

Figure 12c
Third call of Corner Re-computation

```
Aspect ratio optimized (est_aspect_ratio = 1) -> both components of fc are estimated (DEFAULT).
Principal point optimized (center_optim=1) - (DEFAULT). To reject principal point, set center_optim=0
Skew not optimized (est_alpha=0) - (DEFAULT)
Distortion not fully estimated (defined by the variable est_dist):
    Sixth order distortion not estimated (est_dist(5)=0) - (DEFAULT) .

Main calibration optimization procedure - Number of images: 20
Gradient descent iterations: 1...2...3...4...5...6...7...8...9...10...done
Estimation of uncertainties...done


Calibration results after optimization (with uncertainties):

Focal Length:          fc = [ 3066.63841   3115.94163 ] ± [ 10.69053   12.74303 ]
Principal point:       cc = [ 1472.14389   461.85299 ] ± [ 15.67905   22.04797 ]
Skew:             alpha_c = [ 0.00000 ] ± [ 0.00000  ]   => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:            kc = [ -0.12603   -0.02255   -0.01242   -0.01427  0.00000 ] ± [ 0.01351   0.02054   0.00206   0.00098  0.00000 ]
Pixel error:          err = [ 2.18663   2.52824 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).
```

Figure 13
Intrinsic parameter result after re-calibration

For error analysis and refinement, I used 'Analyse Error' to inspect points with large errors, and it revealed 'Image 9' as having significant errors. To address this, I performed 'Corner Re-computation' three times for all images using various window sizes, as shown above. After multiple iterations, I re-calibrated the camera, reducing errors, particularly in 'Image 9.' When I was satisfied with the improved accuracy, I saved the calibration results.

…Now, I’ll just add here the next steps for which we run another sets of calibration with additional parameters (these are optional and depends on your calibration goals and requirements)

```
Loading calibration results from Calib_Results.mat
done
>> est_dist = [1;1;1;1;1];
>> est_alpha = 1;
```

Figure 14
Loading the last calibration results

```
Aspect ratio optimized (est_aspect_ratio = 1) -> both components of fc are estimated (DEFAULT).
Principal point optimized (center_optim=1) - (DEFAULT). To reject principal point, set center_optim=0
Skew optimized (est_alpha=1). To disable skew estimation, set est_alpha=0.

Main calibration optimization procedure - Number of images: 20
Gradient descent iterations: 1...2...3...4...5...6...7...8...9...10...done
Estimation of uncertainties...done


Calibration results after optimization (with uncertainties):

Focal Length:          fc = [ 3059.41920   3107.10978 ] ± [ 10.96221   13.07468 ]
Principal point:       cc = [ 1486.53511   478.53737 ] ± [ 17.96605   22.73212 ]
Skew:             alpha_c = [ -0.00315 ] ± [ 0.00210  ]   => angle of pixel axes = 90.18058 ± 0.12006 degrees
Distortion:            kc = [ -0.09647   -0.19402   -0.01103   -0.01349  0.25295 ] ± [ 0.02302   0.09056   0.00236   0.00121  0.12837 ]
Pixel error:          err = [ 2.17752   2.50560 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).
```

Figure 15
Re-calibrating

```
>> est_dist(5) = 0;
```

Figure 16
Disabling estimation

```
>> est_dist(5) = 0;

Aspect ratio optimized (est_aspect_ratio = 1) -> both components of fc are estimated (DEFAULT).
Principal point optimized (center_optim=1) - (DEFAULT). To reject principal point, set center_optim=0
Skew optimized (est_alpha=1). To disable skew estimation, set est_alpha=0.
Distortion not fully estimated (defined by the variable est_dist):
     Sixth order distortion not estimated (est_dist(5)=0) - (DEFAULT) .

Main calibration optimization procedure - Number of images: 20
Gradient descent iterations: 1...2...3...4...5...6...7...8...9...10...11...12...done
Estimation of uncertainties...done


Calibration results after optimization (with uncertainties):

Focal Length:          fc = [ 3062.99912   3110.01252 ] ± [ 10.89033   13.13378 ]
Principal point:       cc = [ 1486.07447   472.49088 ] ± [ 18.16356   23.02263 ]
Skew:             alpha_c = [ -0.00329 ] ± [ 0.00210  ]   => angle of pixel axes = 90.18864 ± 0.12019 degrees
Distortion:            kc = [ -0.13278   -0.01989   -0.01054   -0.01318  0.00000 ] ± [ 0.01436   0.02130   0.00235   0.00119  0.00000 ]
Pixel error:          err = [ 2.18930   2.51493 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).
```

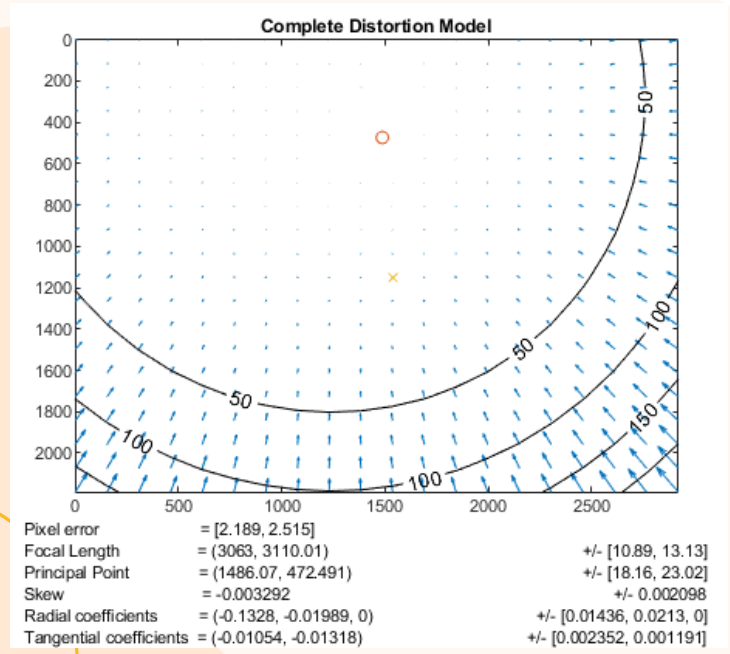
Figure 17
Re-calibrating

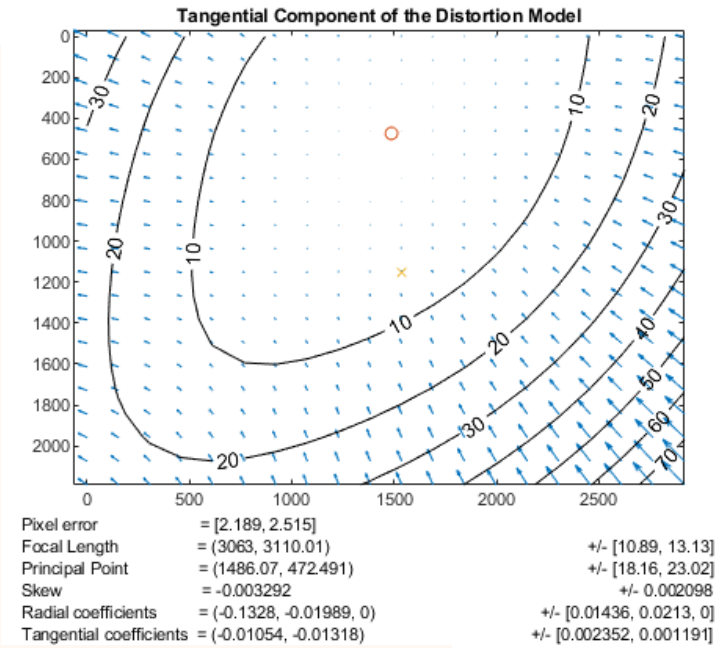
Set the variable est_alpha to one; set the last entry of the vector est_dist to one:

Run the calibration; after optimization, the skew coefficient is very close to zero. This justifies the previous assumption of rectangular pixels (alpha_c = 0)

Since the uncertainty on the 6th order radial distortion coefficient is very large, the estimation was disabled by set the last entry of est_dist to zero

Another round of calibration; We then saved this resulting calibration parameters.





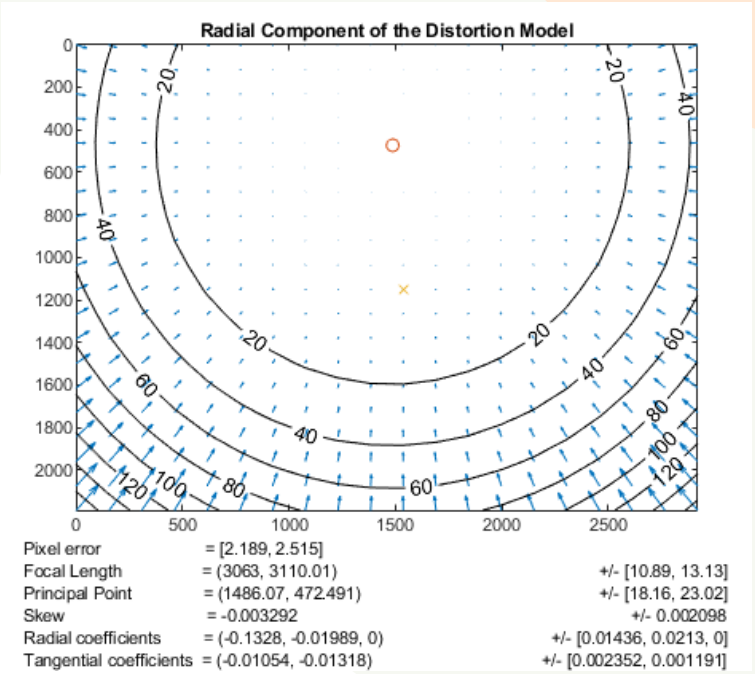


Figure 18
(a) Complete Distortion Model; (b) Tangential Component of the Distortion Model; and (c) Radial Component of the Distortion Model

I also tried running the script visualize_distortions (this is not required; I just wanted to do it and above are the three distortion models produced.

UNDISTORTED IMAGES

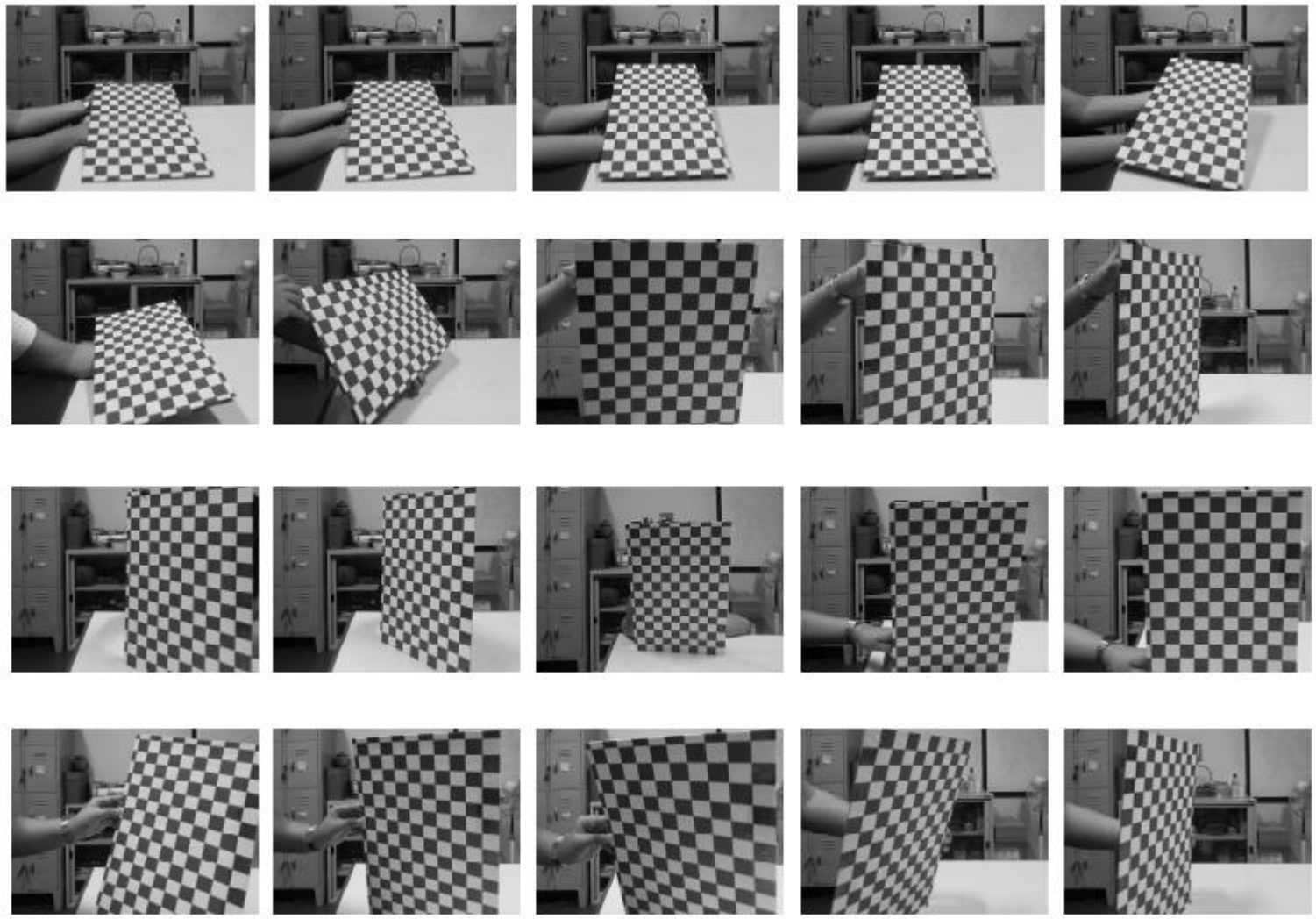
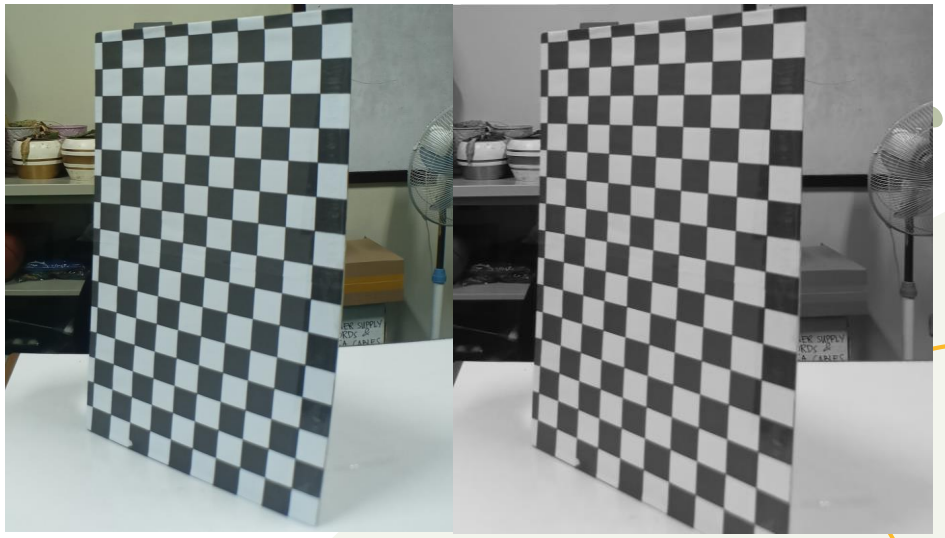


Figure 19
Undistorted versions of the 20 images

By meticulously fine-tuning the calibration parameters, including intrinsic and extrinsic factors, we were able to substantially reduce lens distortion effects and finally able to undistort our 20 images. And as we can see, the undistorted images now better represent the true geometry of the object we captured.

...GOING BACK TO THE FIRST THREE



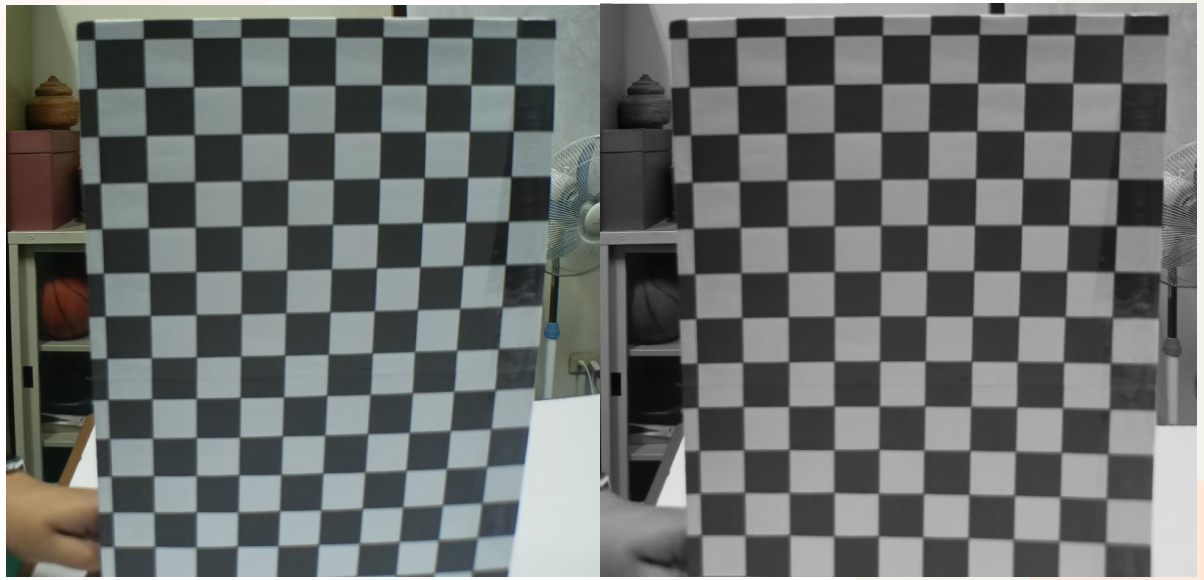


Figure 20

Comparison of the distorted images (left) and the undistorted images (right) of images 10, 12, and 15

REFLECTION

This whole process felt like a bit of an epic journey for me. I tackled it offline and I diligently followed Jean-Yves Bouguet's detailed instructions on camera calibration using the toolbox. There were a few hiccups along the way because my Matlab version wasn't perfectly with the toolbox, so I had to do some code tweaking to make things work. But thank goodness, it eventually clicked into place. I had to go through the calibration process twice because I got a bit lost the first time around. And then, the kicker: I discovered that the camera calibrator online is way quicker and more user-friendly than what I used, which left me feeling a bit frustrated. Still, I can't deny that I enjoyed the journey, especially the part where I was able to undistort the distorted images. The report, too, was a challenged to me as I don’t know how to organize all the things I obtained, but I managed to do it.

All in all, considering the effort I poured into this, I’m giving myself a **100/100.**

THANK YOU!