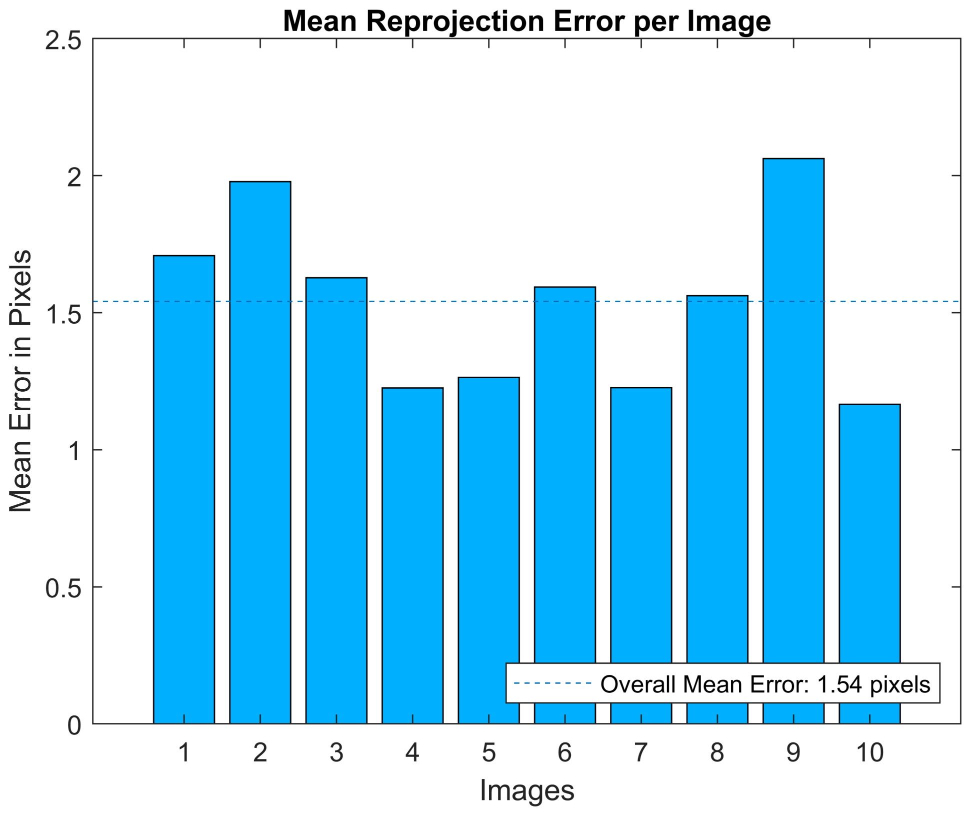
```
% Auto-generated by cameraCalibrator app on 07-Mar-2019
%-------------------------------------------------------


% Define images to process
imageFileNames = {'C:\Users\bengo\Downloads\Photos2\IMG_20190307_162109.jpg',...
    'C:\Users\bengo\Downloads\Photos2\IMG_20190307_162112.jpg',...
    'C:\Users\bengo\Downloads\Photos2\IMG_20190307_162121.jpg',...
    'C:\Users\bengo\Downloads\Photos2\IMG_20190307_162132.jpg',...
    'C:\Users\bengo\Downloads\Photos2\IMG_20190307_162150.jpg',...
    'C:\Users\bengo\Downloads\Photos2\IMG_20190307_162248.jpg',...
    'C:\Users\bengo\Downloads\Photos2\IMG_20190307_162303.jpg',...
    'C:\Users\bengo\Downloads\Photos2\IMG_20190307_163145.jpg',...
    'C:\Users\bengo\Downloads\Photos2\IMG_20190307_163152.jpg',...
    'C:\Users\bengo\Downloads\Photos2\IMG_20190307_163201.jpg',...
    };
% Detect checkerboards in images
[imagePoints, boardSize, imagesUsed] = detectCheckerboardPoints(imageFileNames);
```

```
Warning: The checkerboard must be asymmetric: one side should be even, and the other should be odd.
Otherwise, the orientation of the board may be detected incorrectly.
```

```
imageFileNames = imageFileNames(imagesUsed);

% Read the first image to obtain image size
originalImage = imread(imageFileNames{1});
[mrows, ncols, ~] = size(originalImage);

% Generate world coordinates of the corners of the squares
squareSize = 20;  % in units of 'millimeters'
worldPoints = generateCheckerboardPoints(boardSize, squareSize);

% Calibrate the camera
[cameraParams, imagesUsed, estimationErrors] = estimateCameraParameters(imagePoints, worldPoin
    'EstimateSkew', false, 'EstimateTangentialDistortion', false, ...
    'NumRadialDistortionCoefficients', 2, 'WorldUnits', 'millimeters', ...
    'InitialIntrinsicMatrix', [], 'InitialRadialDistortion', [], ...
    'ImageSize', [mrows, ncols]);

% View reprojection errors
h1=figure; showReprojectionErrors(cameraParams);
```

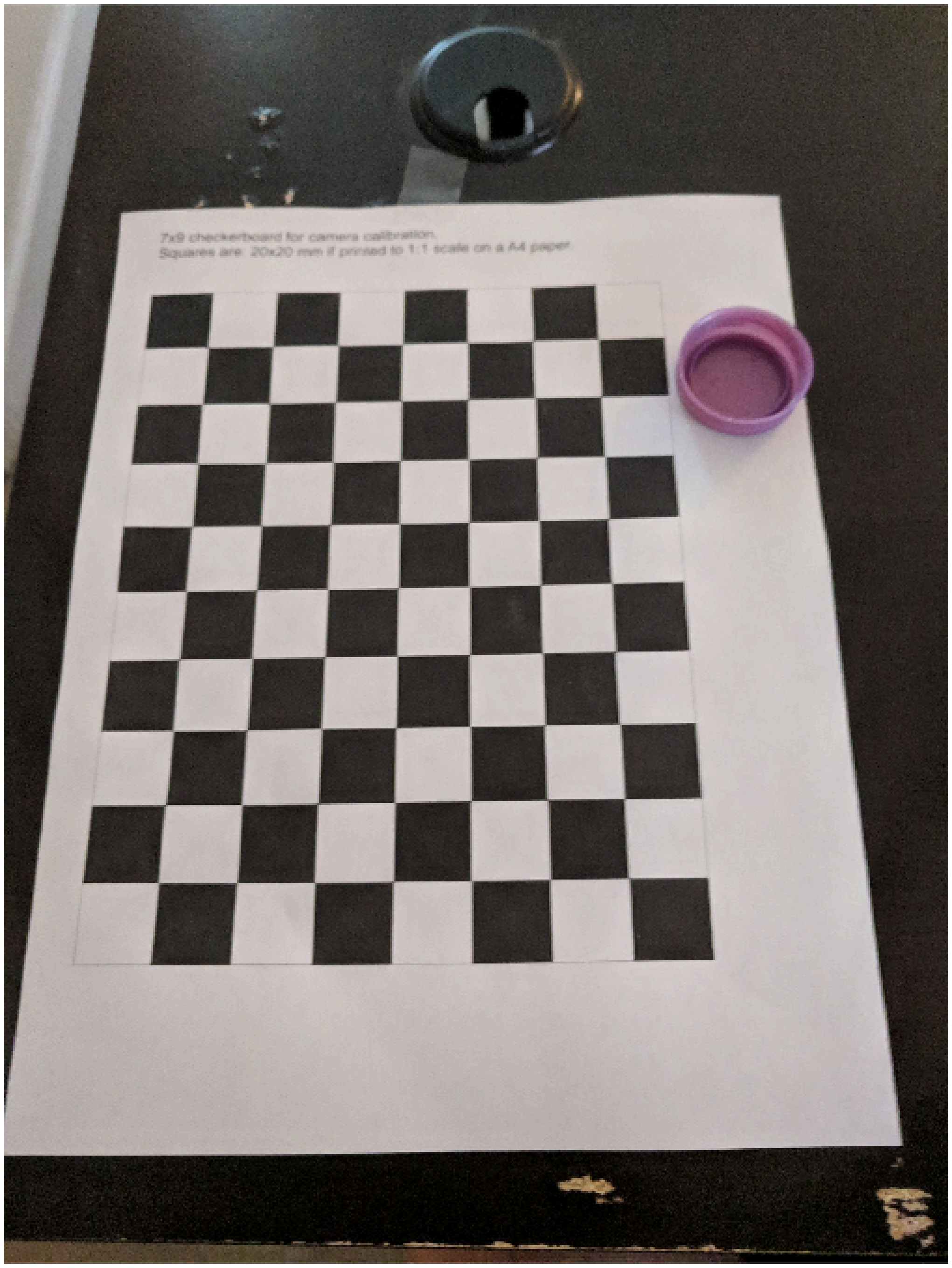


```
% Visualize pattern locations
h2=figure; showExtrinsics(cameraParams, 'CameraCentric');

% Display parameter estimation errors
displayErrors(estimationErrors, cameraParams);
```

```
   Standard Errors of Estimated Camera Parameters
   ----------------------------------------------

Intrinsics
----------
Focal length (pixels):   [ 3153.0762 +/- 13.7766    3154.4751 +/- 12.3332 ]
Principal point (pixels):[ 1548.2380 +/- 2.8118     2021.1713 +/- 10.7502 ]
Radial distortion:       [    0.1531 +/- 0.0085       -0.6476 +/- 0.0474  ]

Extrinsics
----------
Rotation vectors:
                         [   -0.2354 +/- 0.0017        0.2695 +/- 0.0017        1.5423 +/- 0.0004  ]
                         [   -0.3846 +/- 0.0017        0.9822 +/- 0.0033        2.1706 +/- 0.0013  ]
                         [   -0.6520 +/- 0.0029       -0.1041 +/- 0.0013        0.6639 +/- 0.0005  ]
                         [   -0.5338 +/- 0.0023        0.7998 +/- 0.0029        1.8635 +/- 0.0011  ]
                         [   -0.3287 +/- 0.0019        0.2364 +/- 0.0016        1.5478 +/- 0.0003  ]
                         [   -0.6874 +/- 0.0026       -0.3884 +/- 0.0017       -0.9063 +/- 0.0006  ]
                         [   -0.1386 +/- 0.0025        0.0266 +/- 0.0018       -0.0147 +/- 0.0003  ]
                         [   -0.5328 +/- 0.0023       -0.2770 +/- 0.0016       -0.9510 +/- 0.0005  ]
                         [   -0.3006 +/- 0.0022        0.5830 +/- 0.0021        0.8634 +/- 0.0011  ]
                         [   -0.3905 +/- 0.0024        0.6660 +/- 0.0024        1.3736 +/- 0.0010  ]

Translation vectors (millimeters):
```

```
[    34.9127 +/- 0.2636      -89.3882 +/- 0.9561      301.9741 +/- 1.3218  ]
[    72.7673 +/- 0.2934      -28.6637 +/- 1.1193      334.0843 +/- 1.4310  ]
[   -39.9551 +/- 0.3462      -85.6457 +/- 1.2902      395.1586 +/- 1.6629  ]
[   102.5519 +/- 0.4044      -12.0269 +/- 1.5500      457.8431 +/- 1.9342  ]
[    57.0909 +/- 0.2783      -88.7603 +/- 1.0197      321.3586 +/- 1.4011  ]
[   -99.5622 +/- 0.2861      -10.1706 +/- 1.0942      324.2631 +/- 1.4247  ]
[   -97.6645 +/- 0.2783      -19.9503 +/- 1.0854      321.0462 +/- 1.3975  ]
[   -95.1504 +/- 0.3038       -5.2549 +/- 1.1667      344.5307 +/- 1.5004  ]
[   -17.3876 +/- 0.4160     -131.3286 +/- 1.5326      471.5902 +/- 1.9295  ]
[    59.8192 +/- 0.4190      -73.7986 +/- 1.5660      474.2418 +/- 2.0213  ]
```

```
%Check image distortion
undistortedImage = undistortImage(originalImage, cameraParams);

magnification = 25;
imOrig = imread('C:\Users\bengo\Downloads\Photos2\IMG_20190307_162109.jpg');
figure; imshow(imOrig, 'InitialMagnification', magnification);
```


7x9 checkerboard for camera calibration.
Squares are: 20x20 mm if printed to 1:1 scale on a A4 paper

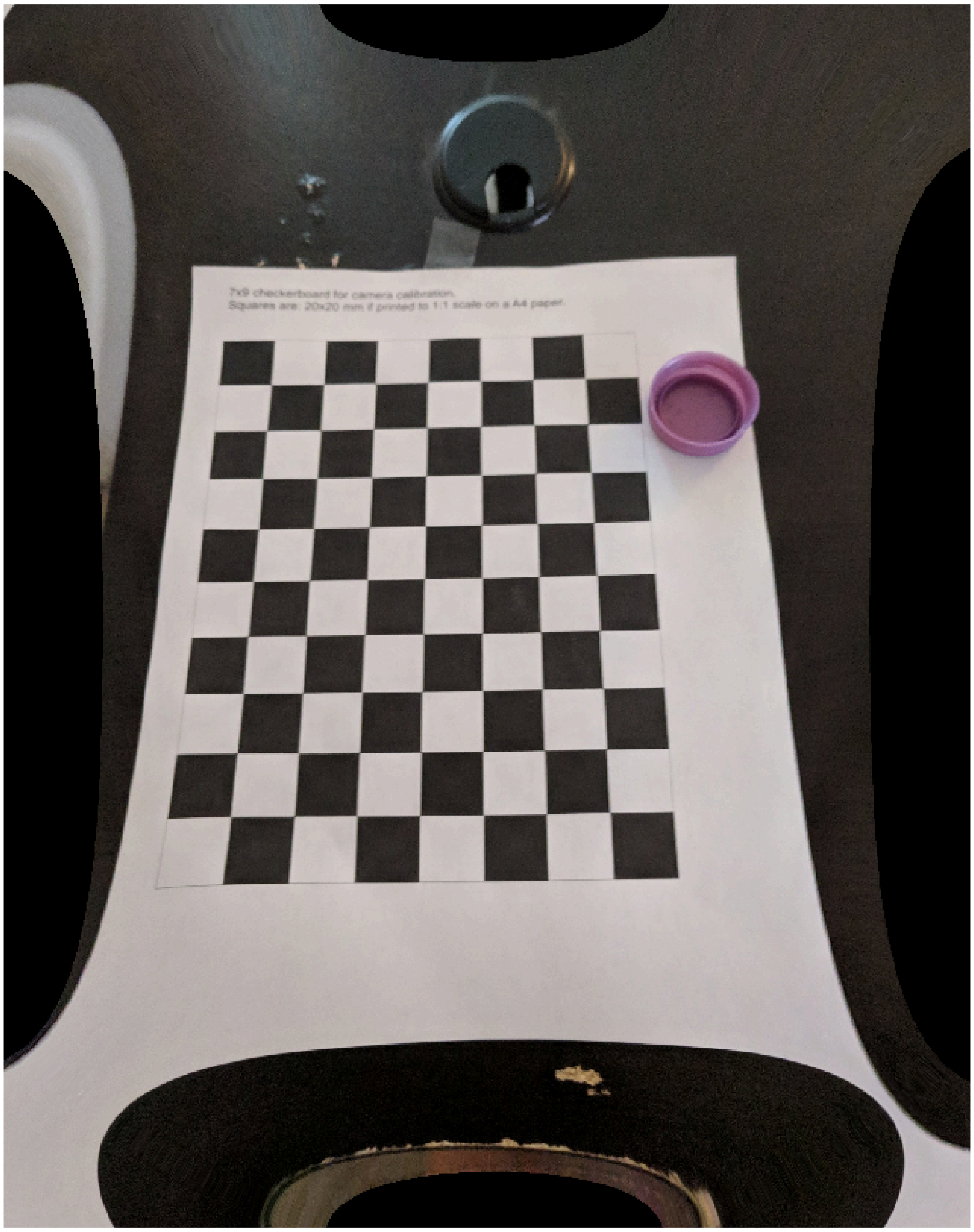
```
Warning: Image is too big to fit on screen; displaying at 17%
```

```
title('Input Image');

%Accounting for distortion/ mitigating
[im, newOrigin] = undistortImage(imOrig, cameraParams, 'OutputView', 'full');
figure; imshow(im, 'InitialMagnification', magnification);
```

```
Warning: Image is too big to fit on screen; displaying at 17%
```

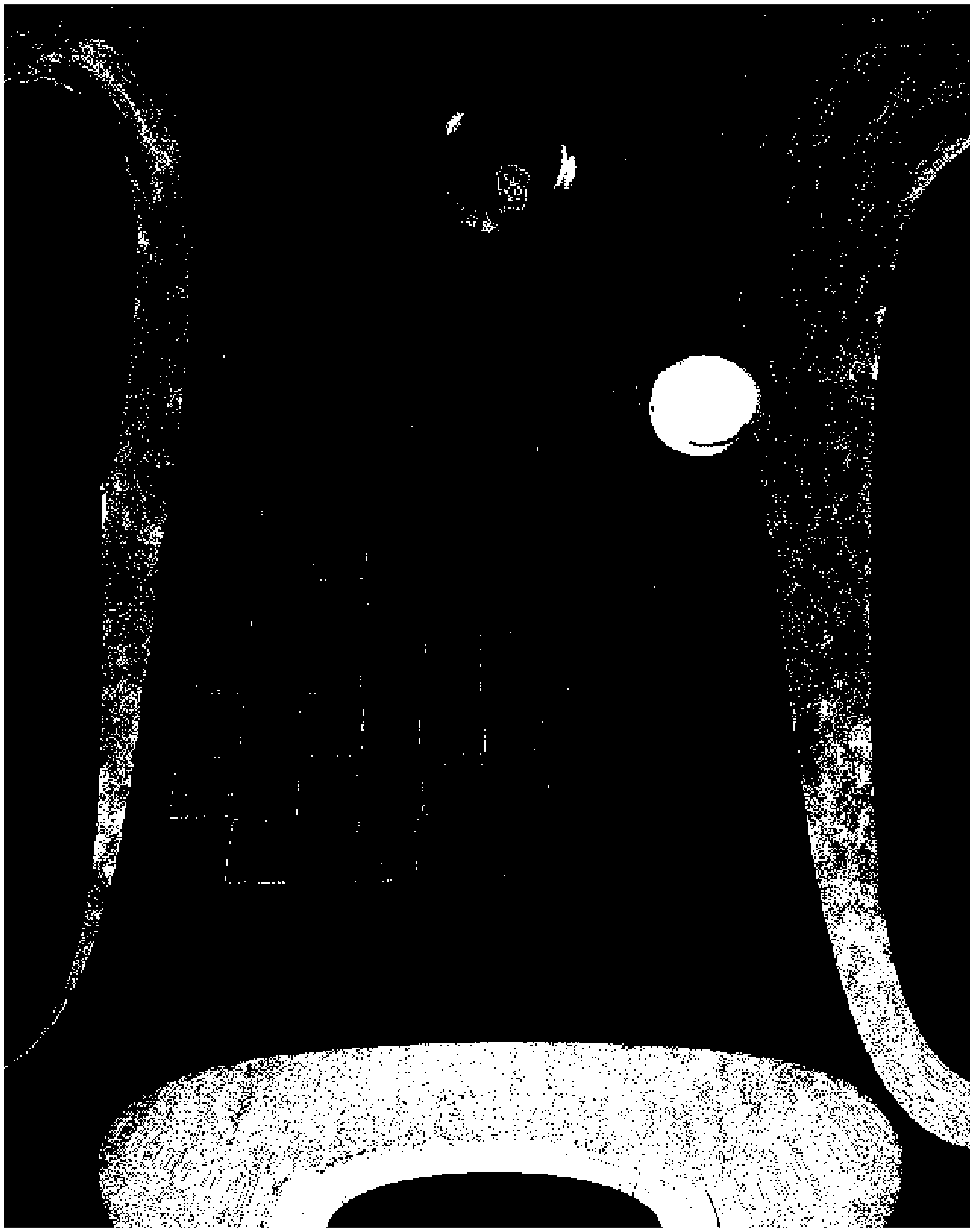
```matlab
title('Undistorted Image');
```

Undistorted Image


7x9 checkerboard for camera calibration.
Squares are: 20x20 mm if printed to 1:1 scale on a A4 paper

```matlab
%Convert image to grayscale to get a blob
imHSV = rgb2hsv(im);
```

```
saturation = imHSV(:, :, 2);
t = graythresh(saturation);
imCap = (saturation > t);

figure; imshow(imCap, 'InitialMagnification', magnification);
```



```
Warning: Image is too big to fit on screen; displaying at 17%
```

```
% Find blobs
blobAnalysis = vision.BlobAnalysis('AreaOutputPort', true,...
    'CentroidOutputPort', false,...
    'BoundingBoxOutputPort', true,...
    'MinimumBlobArea', 200, 'ExcludeBorderBlobs', true);
[areas, boxes] = step(blobAnalysis, imCap);

% Sort connected components in descending order by area
[~, idx] = sort(areas, 'Descend');

% Get largest blob
boxes = double(boxes(idx(1:1), :));

% Reduce the size of the image for display
scale = magnification / 100;
imDetectedCap = imresize(im, scale);
```