Table of Contents


```
clc;
clear;
close all;
```

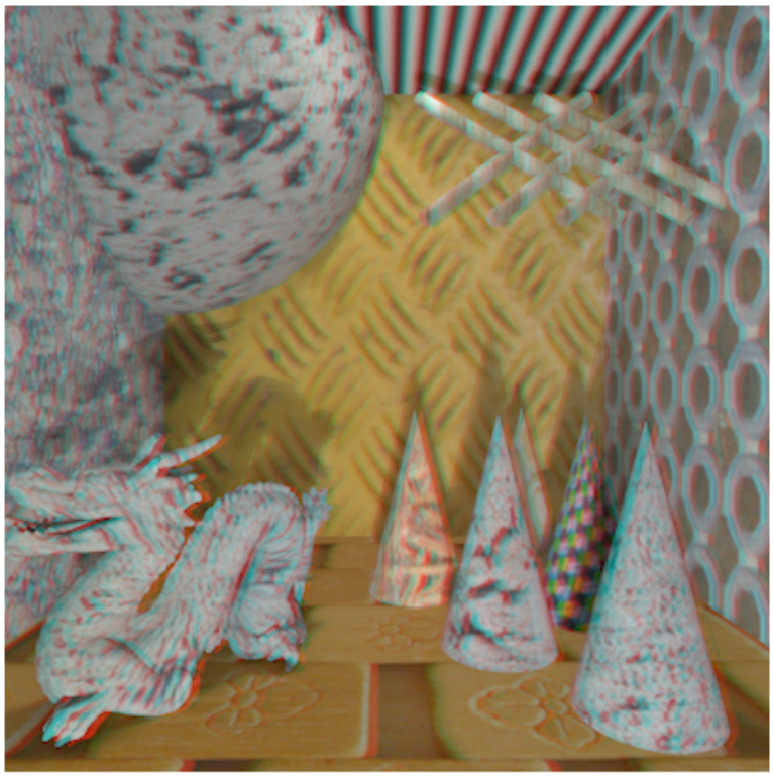
Initial pre-processing

Read the object image

```
LeftImage = imread('Left.png');
LeftImageGray = rgb2gray(LeftImage);
```

```
% Read the scene image
RightImage = imread('Right.png');
RightImageGray = rgb2gray(RightImage);
```

```
figure;
imshowpair(LeftImage, RightImage,'montage');
title('I1 (left); I2 (right)');
figure;
imshow(stereoAnaglyph(LeftImage,RightImage));
title('Composite Image (Red - Left Image, Cyan - Right Image)');
```



Composite Image (Red - Left Image, Cyan - Right Image)

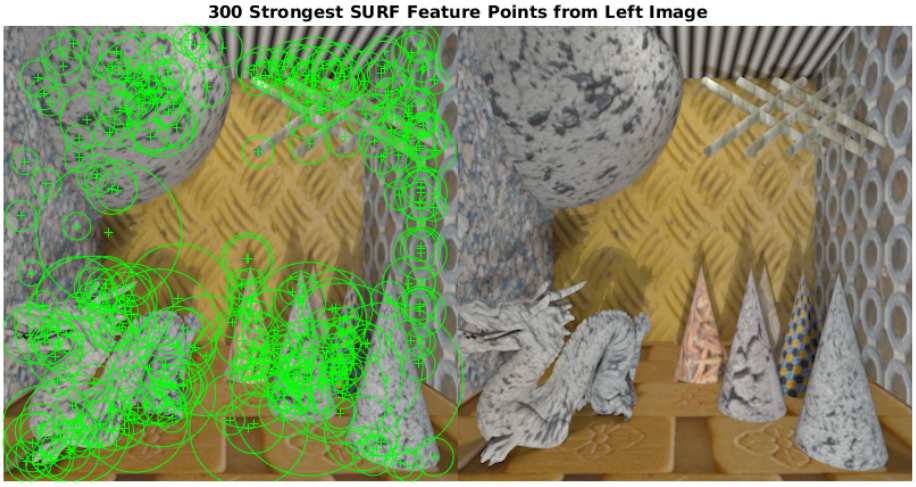
Implementation of SURF based feature detection

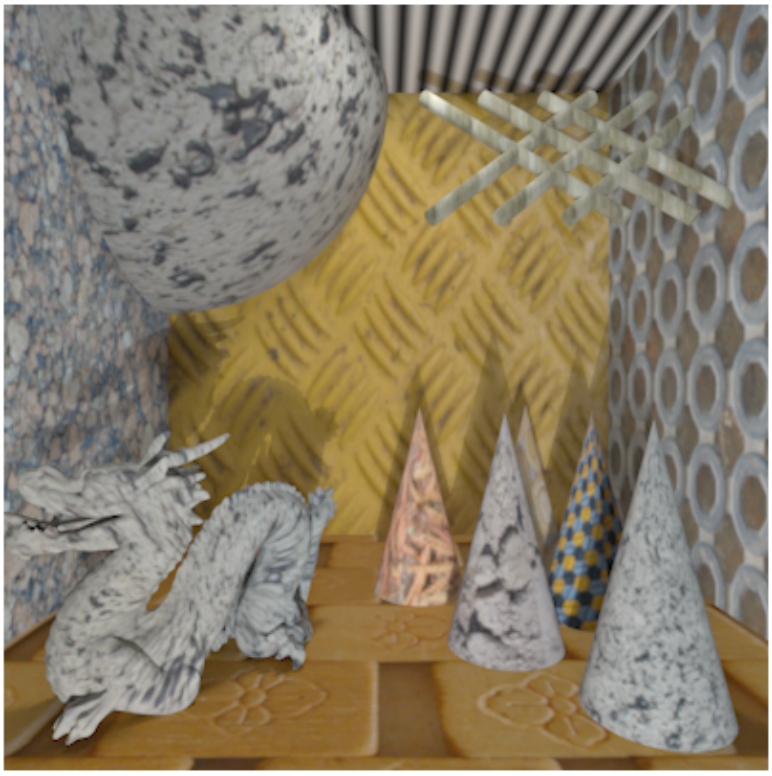
Detect the featrure points using SURF feature detection

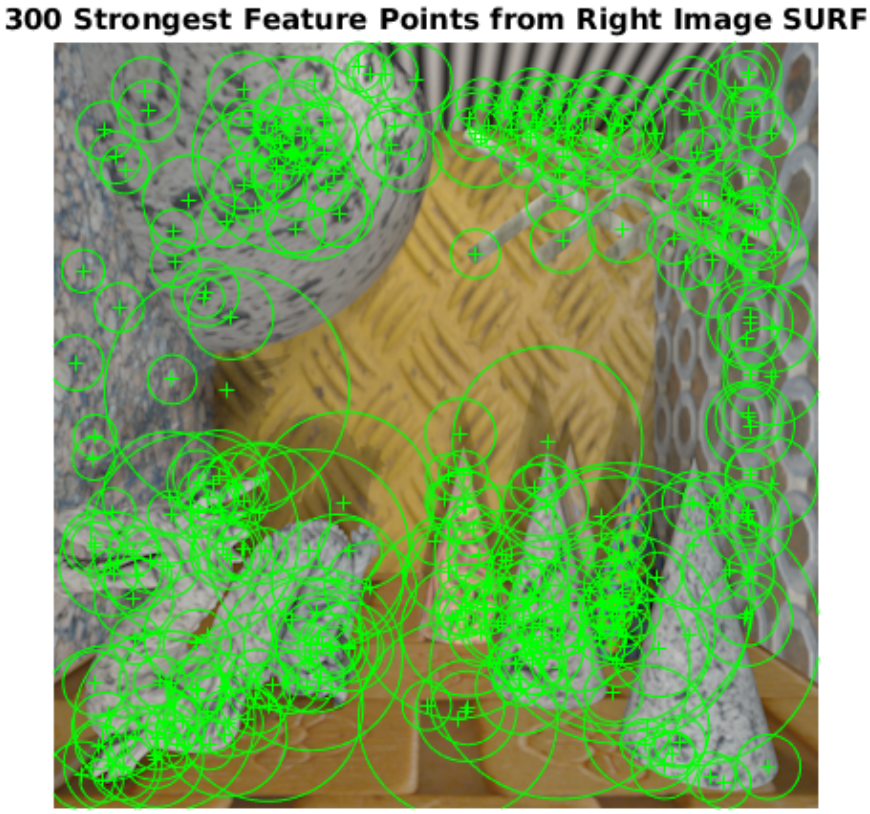
```
LeftPoints = detectSURFFeatures(LeftImageGray);
RightPoints = detectSURFFeatures(RightImageGray);
```

```
% Visualizing the strongest points in the object image
figure;
imshow(LeftImage);
title('300 Strongest SURF Feature Points from Left Image');
hold on;
plot(selectStrongest(LeftPoints, 300));
```

```
% Visualizing the strongest points in the target scene image
figure;
imshow(RightImage);
title('300 Strongest Feature Points from Right Image SURF');
hold on;
plot(selectStrongest(RightPoints, 300));
```

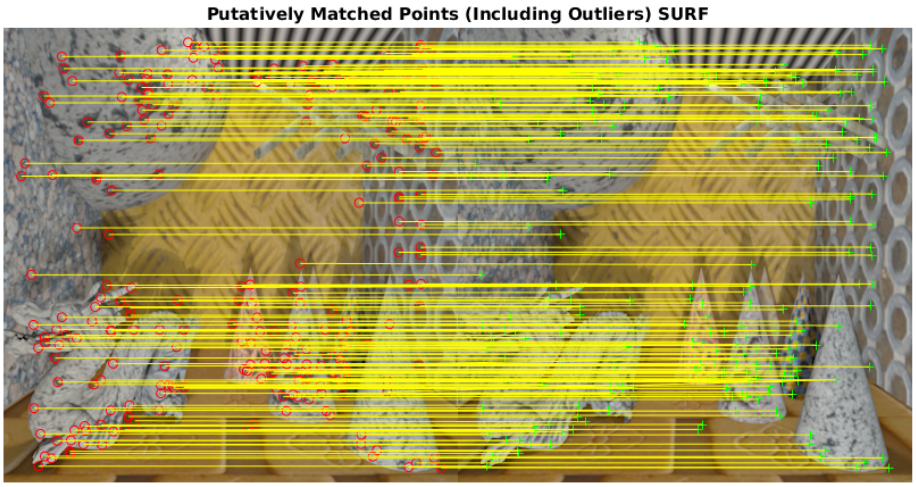






Extracting feature descriptors at interest points in both images

```
[LeftFeatures, LeftPoints] = extractFeatures(LeftImageGray,
 LeftPoints);
[RighFeatures, RightPoints] = extractFeatures(RightImageGray,
 RightPoints);

% Matching the features using the descriptors
Pairs = matchFeatures(LeftFeatures,
 RighFeatures, 'Metric', 'SAD', 'matchThreshold', 5);

% Display the matched features
matchedLeftPoints = LeftPoints(Pairs(:, 1), :);
matchedRightPoints = RightPoints(Pairs(:, 2), :);
figure;
showMatchedFeatures(LeftImage, RightImage, matchedLeftPoints, ...
    matchedRightPoints, 'montage');
title('Putatively Matched Points (Including Outliers) SURF');
hold on
```

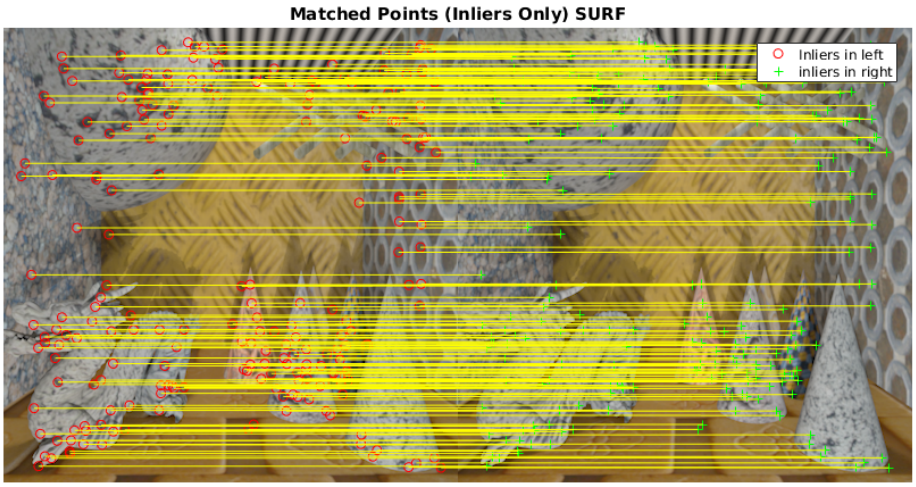


Generating the fundamental matrix

```
[fMatrix inliers, status] = ...
    estimateFundamentalMatrix(matchedLeftPoints,
 matchedRightPoints, 'Method','RANSAC','NumTrials',10000,'DistanceThreshold',0.1,'

if status ~= 0 || isEpipoleInImage(fMatrix, size(LeftImage)) ...
  || isEpipoleInImage(fMatrix', size(RightImage))
  error(['Either not enough matching points were found or '...
         'the epipoles are inside the images. You may need to '...
         'inspect and improve the quality of detected features ',...
         'and/or improve the quality of your images.']);
end



inlierleft = matchedLeftPoints(inliers,:);
inlierright = matchedRightPoints(inliers,:);
```

Displaying the matched points after removing the outliers using RANSAC

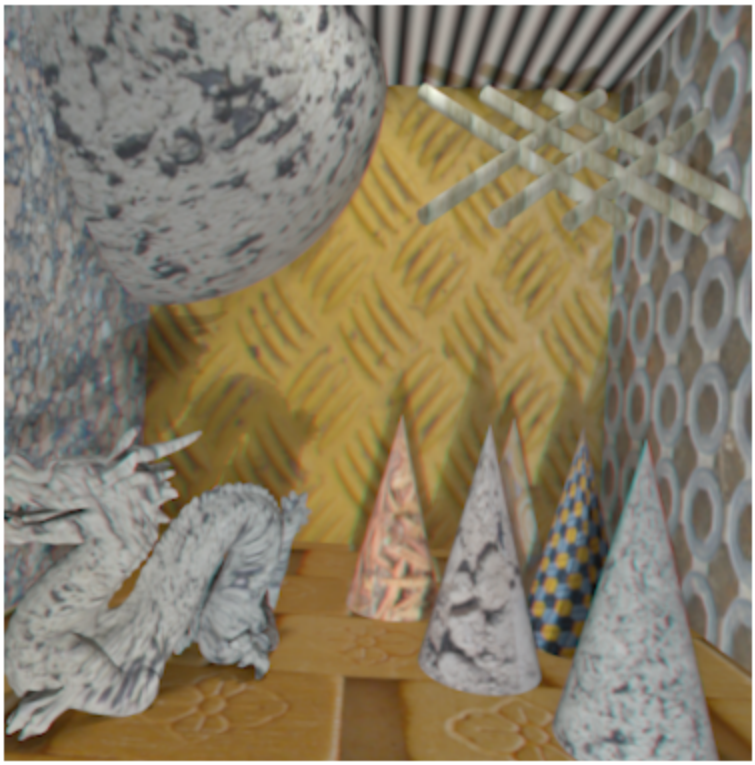
```
figure;
showMatchedFeatures(LeftImage, RightImage, inlierleft, ...
    inlierright, 'montage');
title('Matched Points (Inliers Only) SURF');
legend('Inliers in left','inliers in right' );
```



Rectifying the images

```
% Computing the rectification paramaters

[t1, t2] = estimateUncalibratedRectification(fMatrix, ...
  inlierleft.Location, inlierright.Location, size(RightImage));
tform1 = projective2d(t1);
tform2 = projective2d(t2);

% Rectify the images and display them as a stereo anaglyph
[LeftRect, RightRect] = rectifyStereoImages(LeftImage, RightImage,
 tform1, tform2);
figure;
imshow(stereoAnaglyph(LeftRect, RightRect));
title('Rectified Stereo Images (Red - Left Image, Cyan - Right
 Image)');
```

Rectified Stereo Images (Red - Left Image, Cyan - Right Image)



Published with MATLAB® R2016a