Object Detection Using SURF Algorithm

Bismita S. , Aman S. , Osho A. , Manisha C.
School of Computing Sciences and Engineering, VIT Chennai, Tamilnadu, India 600127
Email: aman97ram@gmail.com

Abstract

This article presents a novel scale- and rotation-invariant detector and descriptor, coined SURF (Speeded-Up Robust Features). SURF approximates or even outperforms previously proposed schemes with respect to repeatability, distinctiveness, and robustness, yet can be computed and compared much faster.

This is achieved by relying on integral images for image convolutions; by building on the strengths of the leading existing detectors and descriptors (specifically, using a Hessian matrix-based measure for the detector, and a distribution-based descriptor); and by simplifying these methods to the essential. This leads to a combination of novel detection, description, and matching steps.

1. Introduction

Object detection is a computer technology related to computer vision and image processing. It deals with detecting instances of semantic objects in digital images and videos. The features extracted from an image are converted into gray image and matched with exciting image features and finally detecting the object using SURF algorithm. It is also used to find the object in various orientations and angles.

Speeded Up Robust Features (SURF) is a local feature detector and descriptor that can be used for tasks such as object recognition or registration or classification or 3D reconstruction. It is partly inspired by the scale-invariant feature transform (SIFT) descriptor.This technique ensures that the points of interest are scale invariant.

2. Methodology

First, 'interest points' are selected at distinctive locations in the image, such as corners, blobs, and T-junctions. The most valuable property of an interest point detector is whether it reliably finds the same interest points under different viewing conditions. Next, the neighbourhood of every interest point is represented by a feature vector. This descriptor has to be distinctive and, at the same time, robust to noise, detection errors, and geometric and photometric deformations. Finally, the descriptor vectors are matched between different images. The matching is based on a distance between the vectors, e.g. the Euclidean distance. The dimension of the descriptor has a direct impact on the time this takes, and a lower number of dimensions is therefore desirable.

3. Algorithm

Step 1: Read Images

Read the reference image containing the object of interest and target image containing a cluttered scene and convert to grayscale.

```
sceneImage1 = imread(str1);

sceneImage  = rgb2gray(sceneImage1);
```

Step 2: Detect Feature Points

Detect feature points in both images.

```
boxPoints = detectSURFFeatures(boxImage);

scenePoints =
detectSURFFeatures(sceneImage);
```

Visualize the strongest feature points found in the reference image.

```
imshow(boxImage);

title('100 Strongest Feature Points from Box Image');

hold on;

plot(selectStrongest(boxPoints, 100));
```

Step 3: Extract Feature Descriptors

Extract feature descriptors at the interest points in both images.

```
[boxFeatures, boxPoints] = extractFeatures(boxImage, boxPoints);

[sceneFeatures, scenePoints] = extractFeatures(sceneImage, scenePoints);
```

Step 4: Find Putative Point Matches

Match the features using their descriptors.

```
boxPairs = matchFeatures(boxFeatures, sceneFeatures);
```

Display putatively matched features.

```
matchedBoxPoints = boxPoints(boxPairs(:, 1), :);

matchedScenePoints = scenePoints(boxPairs(:, 2), :);

showMatchedFeatures(boxImage, sceneImage, matchedBoxPoints, ... matchedScenePoints, 'montage');

title('Putatively Matched Points (Including Outliers)');
```

Step 5: Locate the Object in the Scene Using Putative Matches

estimateGeometricTransform calculates the transformation relating the matched points, while eliminating outliers. This transformation allows us to localize the object in the scene.

```
[tform, inlierBoxPoints, inlierScenePoints] = ...
estimateGeometricTransform(matchedBoxPoints, matchedScenePoints, 'affine');
```

Display the matching point pairs with the outliers removed

```
showMatchedFeatures(boxImage, sceneImage, inlierBoxPoints, ...inlierScenePoints, 'montage');

title('Matched Points (Inliers Only)');
```

Step 6: Get the bounding polygon of the reference image.

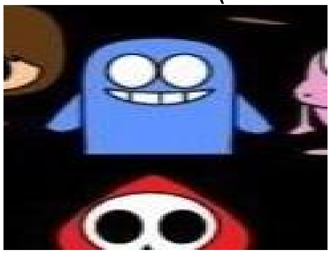
```
boxPolygon = [1, 1;...                    % top-left

size(boxImage, 2), 1;...              % top-right

size(boxImage, 2), size(boxImage, 1);... % bottom-right

 1, size(boxImage, 1);...              % bottom-left
1, 1];                  % top-left again to close the polygon
```

Step 7: Transform the polygon into the coordinate system of the target image. The transformed polygon indicates the location of the object in the scene.

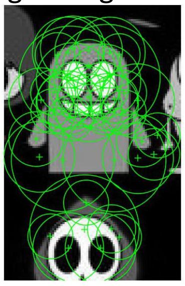
```
newBoxPolygon = transformPointsForward(tform, boxPolygon);

imshow(sceneImage);

hold on;

line(newBoxPolygon(:, 1), newBoxPolygon(:, 2), 'Color', 'y');
```

3. Data and Discussion

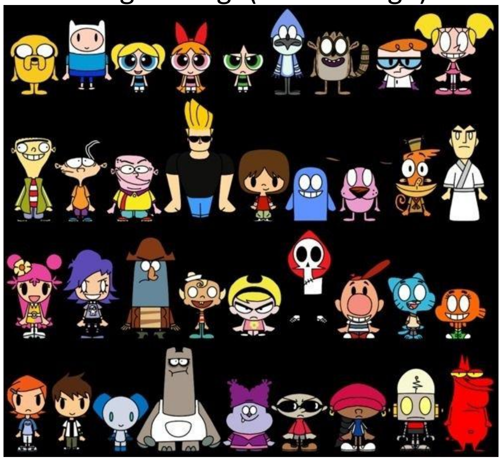
Object to be detected (source image)



100 strongest feature points extracted from source image using SURF algorithm



Target image(scene image)



300 strongest feature points extracted from scene image using SURF algorithm

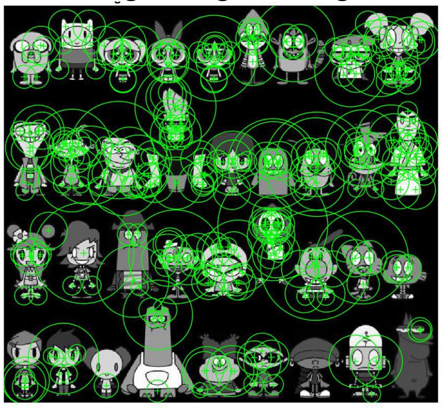
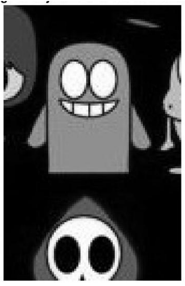


Image of source to be identified(grayscale)



Putatively matched points(including outliers) for source image

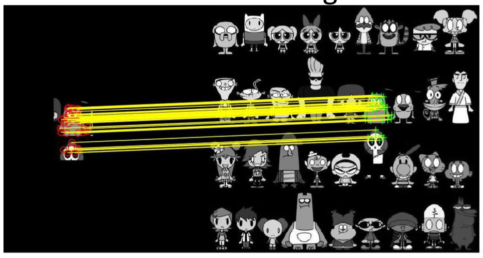
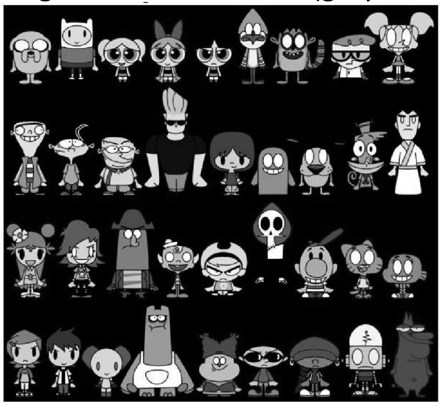
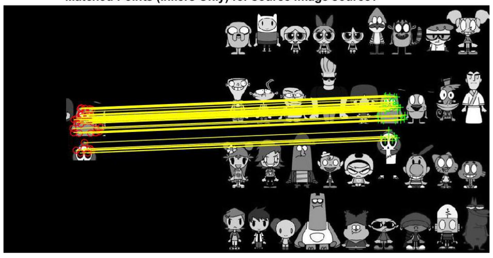


Image of cluttered scene(grayscale)

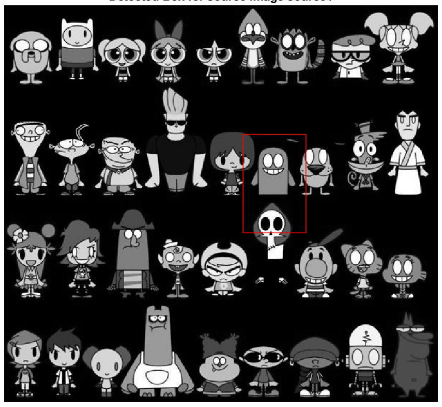


Matched Points containing only inliers



Detected object is in red coloured polygon boundary(scene image)

Detected Box for source image source1



4. Conclusion

From the output images we can see that the algorithm is successful in detecting any kind of object present in scene image based on source image irrespective of its orientation.

Algorithm of this algorithm gave very good results. Accuracy was about 99%.
We can use this algorithm to detect any kind of object and to detect many objects present in a image.

5. References

1. https://in.mathworks.com
2. Object Detection Using SURF and Superpixels; By Miriam Lopez-de-la-Calleja, Takayuki Nagai, Muhammad Attamimi, Mariko Nakano-Miyatake, Hector Perez-Meana
3. An Analysis of the SURF Method; By Edouard Oyallon, Julien Rabin
4. https://en.wikipedia.org