Implementation of High Performance Speeded Up Robust features Detection

J. Naga Surekha
M.Tech Student, Embedded Systems,
Gudlavalleru Engineering College,
Gudlavalleru , India.
e-mail: jonnalasurekha666@gmail.com

Y. Sri Chakrapani
Assoc Prof., Dept. of E.C.E,
Gudlavalleru Engineering College,
Gudlavalleru , India.
e-mail: srichakrapani@gmail.com

Dr. M.Kamaraju
Professor, Dept. of E.C.E ,
Gudlavalleru Engineering College
Gudlavalleru , India.
e-mail: madduraju@yahoo.com

***Abstract* -** In this paper, the interest points are detected by using SURF algorithm. Different Image frames consisting of different resolutions will be given as input for the proposed system to perform SURF detector algorithm. This algorithm performs the following contents: integral image, zero padding, determinant of Hessian matrix (consists of second order Gaussian derivatives for an input image), local maxima technique. Integral image is the cumulative sum of pixel values and zero padding adds zeroes to rows and columns of an image. Hessian matrix consists of Gaussian second order partial derivatives calculated from different sizes of box filters (Octave scales) for an input image. The local maxima technique is used for detecting interest points based on the position of the pixels from the image obtained by calculating the determinant of Hessian matrix. After detecting the interest points, a comparison between threshold and interest points is performed for different octave scales.

***Keywords* –** SURF;Integral Image, Hessian Matrix,Zero Padding.

1 INTRODUCTION

Feature detection is a low-level image processing operation [4]. It is usually performed as the first operation on an image, and examines every pixel to see if there is a feature present at that pixel. If this is a part of larger algorithm, then the algorithm will typically examine the image in the region of the features. As a built-in pre-requisite to feature detection, the input image is usually smoothed by a Gaussian kernel in a scale space representation and one or several feature images are computed, often expressed in terms of local image derivatives operations.

Occasionally, when feature detection is computationally expensive and there are time constraints, a higher level algorithm may be used to guide the feature detection stage, so that only certain parts of the image are searched for features.

Many computer vision algorithms use feature detection as the initial step, so as a result, a very large number of feature detectors have been developed. These vary widely in the kinds of feature detected, pedestrian detection [11] the computational complexity and the repeatability.

2 LOCAL INVARIANT FEATURES

A local feature is an image pattern which differs from its immediate neighborhood. It is usually associated in [9] with a change of an image property or several properties simultaneously, although it is not necessarily localized exactly on this change. The image properties commonly considered are intensity, color, and texture.

Local invariant features are a powerful tool that has been applied successfully in a wide range of systems and applications. In the following, we distinguish three broad categories of feature detectors based on their possible usage. It is not exhaustive or the only way of categorizing the detectors but it emphasizes different properties required by the usage scenarios.

First, one might be interested in a specific type of local features in [12], as they may have a specific semantic interpretation in the limited context of a certain application. For instance, edges detected in aerial images often correspond to roads; blob detection can be used to identify impurities in some inspection task; etc. These were the first applications for which local feature detectors have been proposed. Second, one might be interested in local features since they provide a limited set of well localized and individually identifiable anchor points.

3 CHARACTERISTICS OF FEATURE DETECTORS

D.G. Lowe et al defined a local feature as "*an image pattern which differs from its immediate neighborhood*" [8]. The purpose of local invariant features is to provide a representation that efficiently matches local structures between images. That is, a sparse set of local measurements will be obtained that capture the essence of the underlying input images and encode their interesting structures. To meet this goal, the feature detectors and extractors must have certain properties keeping in mind that the importance of these properties depends on the actual application settings and compromises need to be made. The following properties are important for utilizing a feature detector in computer vision applications:

- Robustness, the feature detection algorithm [7] should be able to detect the same feature locations independent of scaling, rotation, shifting, photometric deformations, compression artifacts, and noise.
- Repeatability, the feature detection algorithm should be able to detect the same features of the same scene or object repeatedly under variety of viewing conditions.
- Accuracy, the feature detection algorithm should accurately localize the image features (same pixel locations), especially for image matching tasks, where precise correspondences are needed to estimate the epi-polar geometry.
- Generality, the feature detection algorithm should be able to detect features that can be used in different applications.
- Efficiency, the feature detection algorithm should be able to detect features in new images quickly to support real-time applications.
- Quantity, the feature detection algorithm in [8] should be able to detect all or most of the features in the image. Where, the density of detected features should reflect the information content of the image for providing a compact image representation.

4 IMPLEMENTATION OF SURF ALGORITHM

4.1 SURF Algorithm

In this , we use SURF(Speeded-up Robust Features) algorithm to detect features because of it should provide better results, faster than SIFT(Scale-invariant feature transformation) algorithm. SURF uses the hessian matrix to find interest points. The determinant of the hessian matrix expresses the extent of the response and is an expression of a local change around the area. SURF was published after SIFT and it was intended to overcome the computational cost derived from using this latter and also the amount of time consumed by the algorithm. The SURF implementation used in this study was developed by Christopher Evans in 2008 and has been continuously improved and revised up to May 2010. He also wrote the paper "Notes on the Open SURF Library" where is explained in detail the analysis of the Speeded-Up Robust Features computer vision algorithm along with a breakdown of the Open-SURF implementation. It also contains useful information on machine vision and image processing in general (Evans, 2008). This library is available in two versions: C++ and C. The C++ version comes with the image matching component whereas the C only has the feature detection component. Both the C++ and implementations are used in this study.

4.2 SURF Feature Detection

The Speed-Up Robust Feature detector (SURF) was conceived to ensure high speed in three of the feature detection steps: detection, description and matching (Bay et al., 2006). Unlike PCA-SIFT, SURF speeded up the SIFT detection process without scarifying the quality of the detected points. SURF was published after SIFT and it was intended to overcome the computational cost derived from using this latter and also the amount of time consumed by the algorithm. The SURF implementation used in this study was developed by Christopher Evans in 2008 and has been continuously improved and revised up to May 2010. He also wrote the paper "Notes on the Open SURF Library" where is explained in detail the analysis of the Speeded-Up Robust Features computer vision algorithm along with a breakdown of the Open-SURF implementation is shown in Fig.1. It also contains useful information on machine vision and image processing in general (Evans, 2008). This library is available in two versions: C++ and C#. The C++ version comes with the image matching component whereas the C# only has the feature detection component. Both the C++ and C# implementations are used in this study.

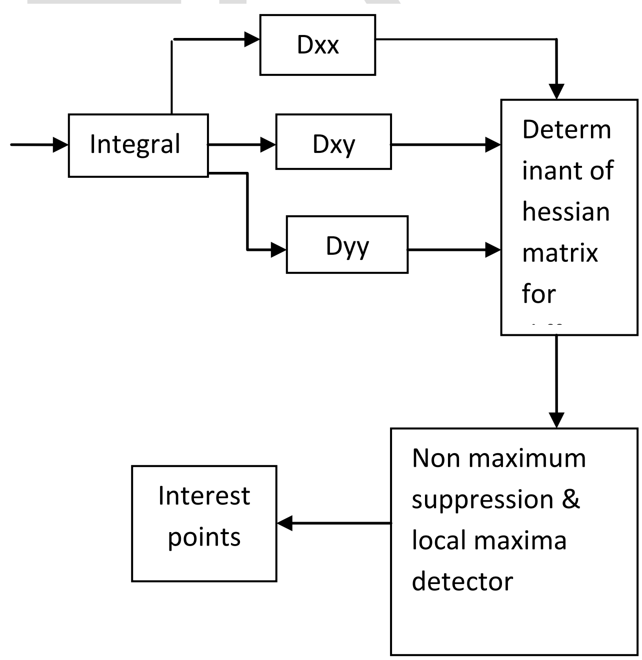

Fig. 1 Block Diagram of SURF Detector

4.3 Second Order Gaussian Derivatives

Dxx:

Dxx is an integral image convolution with the box filter approximation of 2nd X- Derivative of Gaussian. For the calculation of Dxx, provide the zero-padded integral image as the first argument. The filter in this algorithm is designed to work only if the size is multiple of 3, therefore, only if the filter size is odd.

Dyy:

Dyy is an integral image convolution with the box filter approximation of 2nd Y- Derivative of Gaussian. For the calculation of Dyy, provide the zero-padded integral image as the first argument. The filter in this algorithm is designed to work only if the size is multiple of 3, therefore, only if the filter size is odd.

Dxy:

Dxy is an integral image convolution with the box filter approximation of 2nd x-y Derivative of Gaussian. For the calculation of Dxy, provide the zero-padded integral image as the first argument. The filter in this algorithm is designed to work only if the size is multiple of 3, therefore, only if the filter size is odd.

Table 1 Comparison table of Threshold and number of interest points detected for various octave scale

Threshold value	Number of interest points in octave scale1	Number of interest points in octave scale 2	Number of interest points in octave scale3
	(filter size: 9, 15, 21,27)	(filter size: 15, 27, 39,51)	(filter size: 7, 51,75,99)
0.2	1071	483	292
0.4	919	443	275
0.6	832	421	256
0.8	767	409	251
1	705	403	249
1.2	654	393	243
1.4	606	386	239
1.6	578	379	235
1.8	551	370	230
2	529	364	226

From the above table 1 it is clear that the increase in the threshold value will decrease the number of interest points detected. In this table comparison of threshold and interest points is shown for three octave scales.

Table 2 Comparision Table of Various Octaves with Different Images

Image Size	No of Octaves=3	No of Octaves=4
128x128	Pts:185	Pts:218
640x480	Pts:1392	Pts:1661
800x600	Pts:1615	Pts:1986
1024x768	Pts:1686	Pts:2166

From the above table 2 it is clear that the increase in the image size will increase the number of interest points detected with different octaves. In this table comparison of different images and interest points is shown for three and four octave scales.

5 CONCLUSION AND FUTURE SCOPE

Conclusion

In this the SURF algorithm is implemented for detecting interest points in a multiple objects image. The main components of SURF algorithm are integral image generator, Hessian detector and the local maxima finder. In this implementation, the comparison of the threshold and the interest points is performed. The interest points are detected irrespective of size of the image, in which the number of interest points decrease with the increase in the threshold value.

Future scope

In this, detection of interest points is done using SURF Algorithm, which can be further implemented in FPGA, which give better results. After detecting interest points, Feature point orientation and description can be performed, this gives greater accuracy for object recognition.

ACKNOWLEDGMENT

J.Naga Surekha is M.Tech student in Gudlavalleru Engineering College, Andhra Pradesh. She has complete B.Tech from NOVA College of Engineering and technology. Interesting areas embedded systems and VLSI Email id:- jonnalasurekha666@gmail.com

Mr.Y.Sri Chakrapani is currently working as Associate professor in the Department of Electronics and Communications, Gudlavalleru Engineering College, Gudlavalleru India Emailid: srichakrapani@gmail.com. Interesting areas VLSI and embedded systems.

Dr. M.Kamaraju Professor &Head, Department of E.C.E Gudlavalleru Engineering College, Gudlavalleru, India. He has about 23 years of teaching experience and presented papers in several international and national conferences and published papers in international journals.

REFERENCES

[1] Jin Zhao, Xinming Huang, and Yehia Massoud, ''An Efficient Real-Time FPGA Implementation for Object Detection,'' 2014 IEEE Conference pp. 313-316.

[2] J. Zhao, S. Zhu, and X. Huang, "Real-time traffic sign detection using surf features on fpga ," in High Performance Extreme Computing Conference (HPEC), 2013 IEEE Conferrence , pp. 1–6.

[3] Alahi, R. Ortiz, and P. Vandergheynst, "Freak: Fast retina keypoint," in Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. IEEE, 2012, pp. 510–517.

[4] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in Computer Vision (ICCV), 2011 IEEE International Conference on. IEEE, 2011, pp. 2548–2555.

[5] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," Computer vision and image understanding, vol. 110, no. 3, pp. 346–359, 2010.

[6] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," International journal of computer vision, vol. 60, no. 2, pp. 91–110, 2009.

[7] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recog nition using shape contexts," Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 24, no. 4, pp. 509–522, 2008.

[8] T. Gevers and A. W. Smeulders, "Color-based object recognition," Pattern recognition, vol. 32, no. 3, pp. 453–464, 2007.

[9] K. Brkic, "An overview of traffic sign detection methods," Department of Electronics, Microelectronics, Computer and Intelligent Systems Faculty of Electrical Engineering and Computing Unska, vol. 3, p. 10000.

[10] D. M. Gavrila, "Pedestrian detection from a moving vehicle," in Computer Vision, ECCV 2006. Springer, 2006, pp. 37–49.

[11] C. Bahlmann, Y. Zhu, V. Ramesh, M. Pellkofer, and T. Koehler, "A system for traffic sign detection, tracking, and recognition using color, shape, and motion information," in Intelligent Vehicles Symposium, 2005. Proceedings. IEEE. IEEE, 2005, pp. 255–260.

[12] Broggi, C. Caraffi, R. Fedriga, and P. Grisleri, "Obstacle detection with stereo vision for off-road vehicle navigation," in Computer Vision and Pattern Recognition - Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on, 2005, pp. 65–65.