KAKARLA SAI KEERTHY (12071A1023)

PINNAKA RAVI TEJA (1211064293)

MOTION ESTIMATION USING SURF & ORB

INTRODUCTION:

Keypoints are the points in an image that are interesting or stands out in an image which are likely to be recognized in another image. Keypoints/interest points constitute to edges, corners, blobs or ridges based on the type of application. For different images of the same scene, key points should have repeatability. A descriptor is a vector of values, which describes the image patch around an keypoint. Descriptors are the way to compare keypoints. An interest point and its descriptor together can be used for many computer vision tasks, such as Image representation, image classification and retrieval, object recognition and matching, 3D scene reconstruction, motion tracking, texture classification, robot localization and biometrics systems.

The first step of any matching/recognition system is to detect interest locations in the images and describe them. Once the descriptors are computed, they can be compared to find a relationship between images for performing matching/recognition tasks. Feature descriptors extracted from the image can be based on second-order statistics, parametric models, coefficients obtained from an image transform, or even a combination of these measures.

Two types of image features can be extracted from image content representation namely global features and local features.

Global features (e.g., color and texture) aim to describe an image and can be interpreted as a property of the image involving all pixels. Global representation suffers from limitations - they

are not invariant to significant transformations and sensitive to clutter and occlusion. Global features can not distinguish foreground from background of an image, and mix information from both parts together.

Local features aim to detect keypoints or interest regions in an image and describe them. The main goal of local feature representation is to distinctively represent the image based on some salient regions while remaining invariant to viewpoint and illumination changes. Local descriptors are fast to compute, fast to match, memory efficient, and yet exhibiting good accuracy.

There are different algorithms for finding out keypoints and their descriptors. Some of them are SIFT, SURF, ORB, BRISK. In the next part of the report, we give an overview of SURF, ORB algorithms, compare them and mention their advantages and disadvantages.

DESCRIPTION OF ADOPTED APPROACHES:

SURF:

SURF(Speeded Up Robust Features) is a Computer Vision algorithm for keypoint detection and description. It is rotation invariant and was designed by Bay, H., Tuytelaars, T. and Van Gool, L to overcome the lethargic features of SIFT(Scale Invariant Feature Transform). SURF is way more faster than SIFT and outperforms the latter with respect to distinctiveness, repeatability and robustness.

The SURF Algorithm has 3 main parts

1) **Interest Point Detection:**

The Interest Point Detection theory uses Hessian-matrix approximation, which uses the concept

of integral images. Fast computation of box type convolution filters is allowed. The sum of pixels in the image within a rectangular region, bounded by origin and point x is given by

$$I_{\Sigma}(\mathbf{x}) = \sum_{i=0}^{i \leq x} \sum_{j=0}^{j \leq y} I(i, j)$$

To find the points of interest, SURF uses Hessian Matrix. It is used in detecting blob like structures at locations of maximum determinant. The Hessian matrix in x=(x,y) at a scale σ is defined as

$$H(p, \sigma) = \begin{pmatrix} L_{xx}(p, \sigma) & L_{xy}(p, \sigma) \\ L_{yx}(p, \sigma) & L_{yy}(p, \sigma) \end{pmatrix}$$

Where Lxx(x, σ) is the convolution of Gaussian Second order derivative with Image I(x,y) at point x.

Scale Space representation

Scale spaces are implemented in SURF using box filters of varying sizes. The least level is got from output of 9x9 filters. Rather than decreasing the image size, scale space is obtained by up-scaling the filter size. Gradually, the masks(filters) get bigger 9x9,15x15,21x21…, considering the integral image discrete nature and filter structure. The maximum of the determinant of Hessian matrix is interpolated. This is useful as the difference in scale between first two layers is large.

2) **Descriptor**

The descriptor describes the distribution of intensity content of pixels, within the neighbourhood of the point of interest. The description is obtained for all the point of interests in the current

frame, corresponding to the previous one. Thus, this is iterated over the entire frames in a video or live camera and the descriptors are obtained. The distribution is built on first order Haar wavelet responses in x and y direction, and use 64 dimensions only. This helps in reduction of time for computation and matching.

The initial step is fixing a reproducible orientation with respect to data in a circular region around the area of point of interest. A square region aligned to the selected orientation is constructed and the SURF descriptor is obtained from it.

Orientation assignment

In order to achieve rotational invariance, the orientation of the point of interest needs to be found. The Haar wavelet responses in both x- and y-directions within a circular neighbourhood of radius 6s around the point of interest are computed, where 's' is the scale at which the point of interest was detected. The obtained responses are weighted by a Gaussian function centered at the point of interest, then plotted as points in a two-dimensional space, with the horizontal response in the abscissa and the vertical response in the ordinate. The dominant orientation is estimated by calculating the sum of all responses within a sliding orientation window of size pi/3. The horizontal and vertical responses within the window are summed. The two summed responses then yield a local orientation vector. The longest such vector overall defines the orientation of the point of interest. The size of the sliding window is a parameter that has to be chosen carefully to achieve a desired balance between robustness and angular resolution.

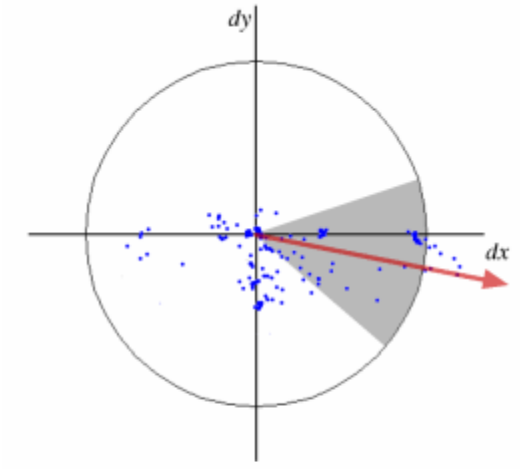


Fig. 10. Orientation assignment: A sliding orientation window of size $\frac{\pi}{3}$ detects the dominant orientation of the Gaussian weighted Haar wavelet responses at every sample point within a circular neighbourhood around the interest point.

(figure from Wikipedia)

Descriptor based on the sum of Haar wavelet responses

A square region is extracted from the area around the point, centered on the interest point. The window size is 20s. The specific region of interest is split into smaller 4x4 sub-regions and the Haar wavelet responses are extracted for each of these at 5x5 regularly spaced points. These are weighted with a Gaussian to smoothen and for robust nature.

Fig. 11. Detail of the Graffiti scene showing the size of the oriented descriptor window at different scales.

Matching

Matching pairs can be found in all the frames, by comparing the descriptors. Feature descriptors matching can be done using several algorithms such as brute force matching method, flann based matching method etc. In our program, we use Flann Based Matching method .

Advantages

1. The key advantage of SURF is that the method is fast, reliable and robust in nature. The speed gain is due to the use of integral images, which reduces the number of operations for convolutions.
2. The high repeatability is useful for camera self calibration, which affects the accuracy of camera.
3. Object detection and motion tracking is scale and rotation invariant.
4. Surf doesn't need long and tedious training like in case of cascaded haar classifier based detection.

Disadvantages

1. SURF is less accurate compared to the SIFT algorithm.
2. Though it is robust and fast, the cost comes with reduction in efficiency.

Applications

1. Since the SURF algorithm is rotation invariant, its widely used in mobile robots where it needs to recognize the objects in various orientations.
2. SURF is used in the real time motion tracking, image stitching, object recognition and in stereo correspondence algorithms. The optical flow in live camera or videos can be computed using SURF and this is used in various Computer Vision applications.

ORB

Oriented FAST and rotated BRIEF (ORB) is a fast robust feature detector, designed as an efficient alternative to SIFT and SURF. It is a fusion of FAST keypoint detector and BRIEF descriptor. The algorithm applies Harris corner measure, after using FAST to find keypoints, to get top N points. It uses pyramid to produce multi-scale features.

Orientation Compensation

ORB computes the intensity weighted centroid as a measure of corner orientation. The moments of a patch are calculated as

$$m_{pq} = \sum_{x,y} x^p y^q I(x,y)$$

The centroid is calculated by

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right)$$

A vector is constructed from corner's center to centroid OC. The orientation of patch is

$$\theta = \text{atan2}(m_{01}, m_{10})$$

Then it is rotated to a canonical rotation and then the descriptor is computed, obtaining rotational invariance. The following steps are followed in the algorithm:

Each test is run against all the training patches. Tests are ordered by a distance forming a vector

V. Take the first test into result vector S and remove it from V. The next test is taken from V and is compared against all in S. If the correlation is greater than a threshold, leave it; else, add it to S. Repeat iteratively until there are 256 tests in S.

Advantages:

1. ORB descriptor works much faster and better than SIFT and SURF.
2. ORB is a better choice in low power consumption devices like mobile phones, where it can be used in different computer vision applications.

Disadvantages:

1. ORB is less scale invariant compared to SIFT and SURF feature algorithms.

Applications:

1. ORB is used in object detection and patch tracking in smart phone.
2. ORB is used in all mobile applications which need lower computational power like in drones, traffic monitoring, etc..

Flann Based Matching:

FLANN (Fast Library for Automatic Nearest Neighbours) is a library for performing fast approximate nearest neighbor searches in high dimensional spaces.

FLANN is written in C++ and contains bindings for the following languages: C, MATLAB and Python.

The hierarchical k-means tree or multiple randomized kd-trees are the algorithms that are generally used to achieve this. The original kd-tree algorithm splits the data in half at each level of the tree on the dimension for which the data exhibits the greatest variance. The hierarchical k-means tree is constructed by splitting the data points at each level into K distinct regions using a

k-means clustering, and then applying the same method recursively to the points in each region. We stop the recursion when the number of points in a region is smaller than K. We used the Flann matcher class in OpenCV to match the feature descriptors. This matcher trains flann::index on a train descriptor collection and calls its nearest search methods to find the best matches. So, this matcher may be faster when matching a large train collection than the brute force matcher.

Comparision of SURF and ORB:

REAL VALUED	**DETECTOR**	**DESCRIPTOR**
SURF	**Box filter pyramid**	**Haar wavelets**

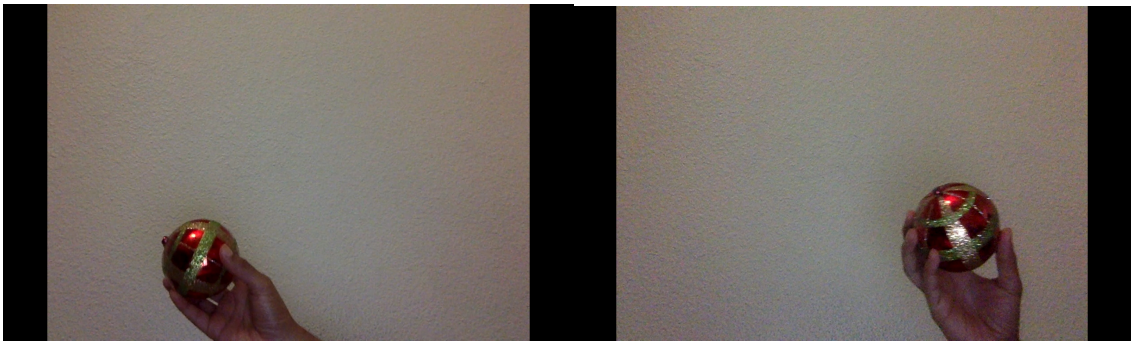
BINARY	**DETECTOR**	**ORIENTATION**
ORB	**oFAST**	**Moments**

Although **ORB** uses image pyramids for scale invariance and intensity centroid for rotation invariance, it is not as robust as **surf**. However **ORB** may be an efficient alternative to **surf** if performance is important because it works on binary descriptors and is freely available (**Surf** is patent protected). Precision and Accuracy depends on the set of images which we take. We cannot say surf gives more accuracy than orb or vice versa. ORB is faster when compared to that of SURF Algorithm in obtaining feature descriptors. In our application, for a fixed number of keypoints, orb is faster than surf. Orb is also faster compared to surf even if image scale increases. Repeatability is more for ORB compared to SURF.

Results :

The optical flow is the pattern of motion between frames in a video or live camera, when there is any movement of object or camera. A 2D vector field obtained, where each vector is displacement vector showing the movement. The keypoints have been calculated for a video using the SURF and ORB local feature detectors. Motion estimation is done using the pyramid implementation of the Lucas-Kanade or KLT algorithm.

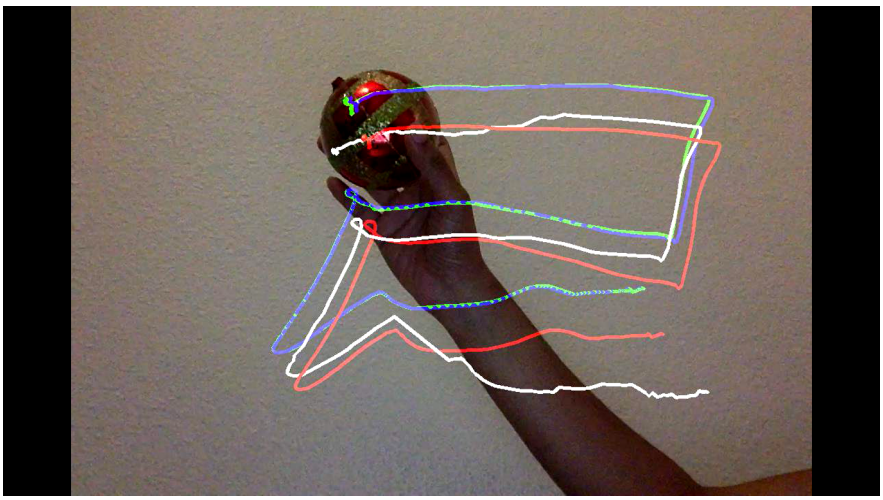
Input :



Output :

SURF keypoints with CalcOpticalFlowPyrLK():

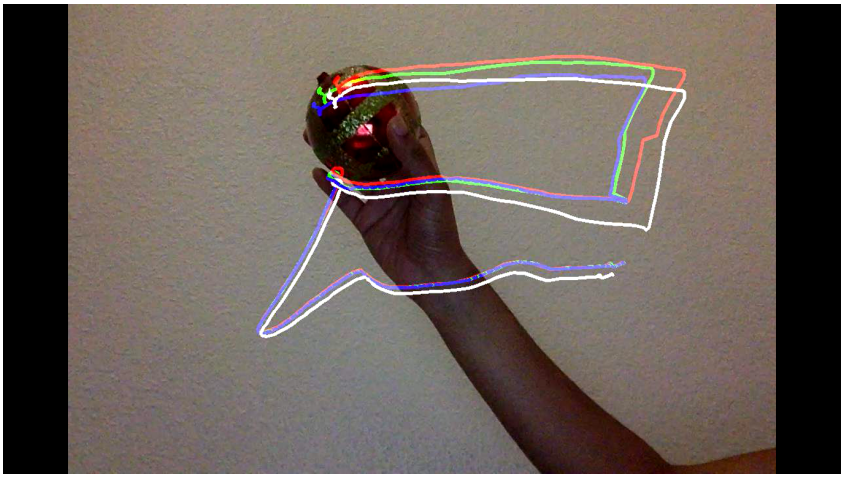
The different keypoints have been tracked using different colours. 4 feature key points have been detected by the algorithm using SURF and the resulting optical flow is obtained as follows:



We can change the number of keypoints obtained by changing the HessainThreshold value. If we increase the value of HessainThreshhold, the number of keypoints decrease and vice versa. Here we have taken hessainThreshold = 7000 which gives us four keypoints. If we take a different input video or input from a camera, we will have to change the hessainThreshold value accordingly.

ORB keypoints with CalcOpticalFlowPyrLK():

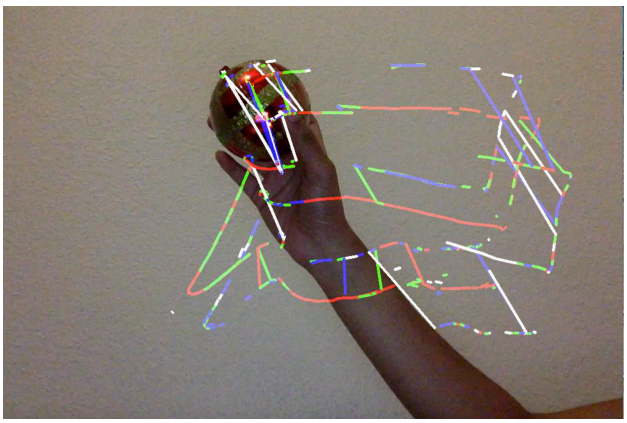
The different keypoints have been tracked using different colours. 4 feature key points have been detected by the algorithm using ORB and the resulting optical flow is obtained as follows:



We can change the number of keypoints obtained by changing the nFeatures value. nFeatures is proportional to number of keypoints. Here we have taken nFeatures= 4 which gives us four keypoints.

SURF descriptors with Flann Matching:

4 feature key points have been detected by the algorithm using SURF and the resulting optical flow is obtained as follows:



We can change the number of keypoints obtained by changing the HessainThreshold value. If we increase the value of HessainThreshhold, the number of keypoints decrease and vice versa. Here we have taken hessainThreshold = 7000 which gives us four keypoints. If we take a different input video or input from a camera, we will have to change the hessainThreshold value accordingly.

ORB descriptors with Flann Matching:

4 feature key points have been detected by the algorithm using ORB and the resulting optical flow is obtained as follows:

We can change the number of keypoints obtained by changing the nFeatures value. nFeatures is proportional to number of keypoints. Here we have taken nFeatures= 4 which gives us four keypoints.

Analysis of output :

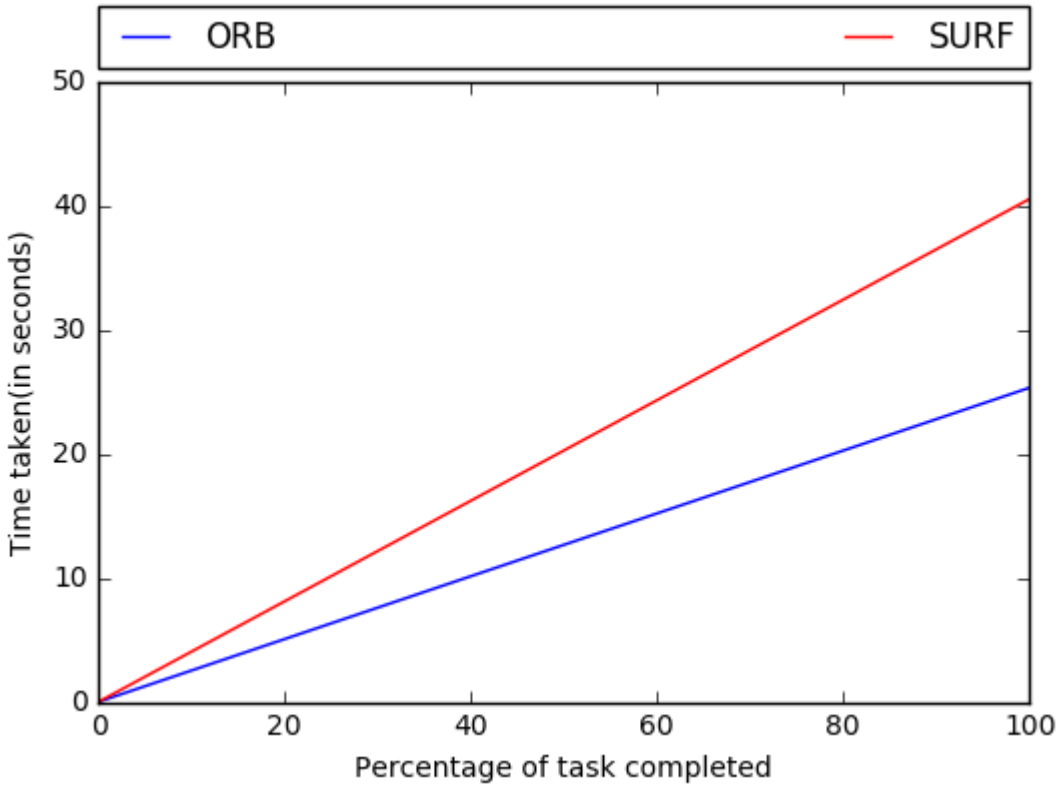
	Processing time
SURF with CalcOpticalFlowPyrLK()	**40.47**
ORB with CalcOpticalFlowPyrLK()	**25.25**
Surf descriptors with Flann Matching	**40.45**
ORB descriptors with Flann Matching	**24.98**

Table: For number of keypoints =4 , the above are the processing times taken by each algorithm.

The time taken for the implementations has been computed under constrained conditions. The experiment was performed using a Mac 8GB, i5,1.6GHz

As clearly observed, the time taken for ORB feature descriptors is very less compared to the rest followed by SURF feature descriptors.

If we compare SURF with CalcOpticalFlow() and ORB with CalcOpticalFlow() ,the time taken for the percentage of task completed has been plotted as follows:



Accuracy:

For the given input, from the output videos in the folder and the images from result, we can say that the accuracy for ORB feature detector is slightly higher compared to the SURF detector. The output is comparatively smoother when using ORB .

When using descriptors, the object has been more accurately tracked using ORB. Also, the time taken for ORB to generate the optical flow has been lower too. So, we can infer that ORB is a better algorithm compared to SURF for environments like that of the input video. In terms of tracking, we can say that tracking is done well using ORB descriptors.

Conclusion :

There are many unsolved problem in the research community of Computer vision and image processing related to object tracking and motion estimation. Some of them arise due to :

1. The camera which is used in recording might not be stable. This will cause blur and lower the quality of the video recorded, which can't be filtered out.
2. The lighting in the scenes may be inadequate. For example, the objects in the foreground must have sufficient contrast compared to the background, which will be used in identifying good keypoints and descriptors.
3. Object tracking and estimation algorithms might not work in too busy environments i.e if there are too many objects moving in the frame, like in a busy street, the model might not filter out the unwanted keypoints and huge computation needs to be done in tracking all the objects. Hence, the number of objects to be tracked should be minimal.

How to overcome some of the problems ?

1. The cameras should be in a still position and then used in tracking.
2. With the advent of GPUs, computation power is now comparitively cheaply available. Hence, it is necessary to see that there is sufficient CPU or GPU power to harness the needs of the chosen Computer vision algorithm.
3. Motion mask need to be setup appropriately when used in a busy area. If there is any consistent and repeating motion in the tracking video like a waving flag or some signal, it can be removed using the proper settings and environment. Else, remove the localized area from tracking. Also, the image quality needs to be reduced for a faster performance.

There are many other unaddressed problems in this area like automatic driving vehicles, radar surveillances, drone monitoring systems. Research is being done to work on efficient algorithms

for anomaly detection, identifying thefts and monitoring the traffic. More work needs to be done in the field of unmanned vehicles. The present motion tracking systems is not able to detect and decide the event. The SIFT, SURF and ORB features are not able to work on large number of moving objects. They're not able to track and identify objects efficiently. With more Machine learning models and better training, we can expect these models to be more robust and efficient in the future.

References:

Some of the images have been taken from some of the following research papers.

1. "Distinctive Image Features from Scale-Invariant Keypoints," International Journal of Computer Vision, vol. 60, issue 2, pages 91-110, Nov. 2004.
2. H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "SURF: Speeded Up Robust Features," Elsevier Computer Vision and Image Understanding, vol. 110, no. 3, pp. 346-359, June 2008
3. E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: an efficient alternative to SIFT or SURF," International Conference on Computer Vision (ICCV), pp. 2564-2571, Nov. 2011
4. B. D. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," International Joint Conference on Artificial Intelligence, pp. 674-679, 1981
5. C. Tomasi and T. Kanade, "Detection and Tracking of Point Features," Carnegie Mellon University Technical Report CMU-CS-91-132, April 1991.
6. J. Shi and C. Tomasi, "Good Features to Track," IEEE Conference on Computer Vision and Pattern Recognition, pp. 593-600, 1994

Online Resources

http://docs.opencv.org/3.0-beta/doc/tutorials/features2d/feature_detection/feature_detection.html

http://stackoverflow.com/questions/27533203/how-do-i-use-sift-in-opencv-3-0-with-c

http://stackoverflow.com/questions/13423884/how-to-use-brisk-in-opencv

http://docs.opencv.org/trunk/d1/d89/tutorial_py_orb.html

http://docs.opencv.org/modules/video/doc/motion_analysis_and_object_tracking.html

http://www.ces.clemson.edu/~stb/klt/ http://robots.stanford.edu/cs223b04/algo_tracking.pdf