Object Tracking in Video Using Flip-Invariant Scale Invariant Feature Transform

Sasmit M. Gokhale[1], Tanmay K. Deshmukh[2], Vinod V. Dalavi[3], Prof. Uttara Gogate[4]

Department of Computer Engineering, Shivajirao S. Jondhale College of Engineering, Dombivli Maharashtra, India.

---***---

Abstract - *The Scale Invariant Feature Transform (SIFT) is invariant to rotation, lighting and scale changes in images. Due to this property of SIFT, it is very popular keypoint descriptor. However, it is not invariant to flips, and this is one of the biggest drawbacks of SIFT. This new descriptor, named as flip-invariant SIFT preserves the original properties of SIFT along with being invariant to flips. Flip Invariant SIFT starts by estimating the dominant curl of a local patch and then performs flipping before computation of SIFT. We have implemented F-SIFT for object detection in a video.*

Key Words: Flip-Invariant Scale Invariant Feature Transform (F-SIFT), Scale Invariant Feature Transform (SIFT), Object Detection, Object Tracking, Frames.

1. INTRODUCTION

There are many applications developed recently that use the Scale Invariant Feature Transform make object classifiers. The main advantages of SIFT [1] it is invariant to rotation, lighting and scaling. It is also invariant to displacement of pixels in a local region.
SIFT is computed over a local salient region. The region is rotated to its dominant orientation and is located using multi-scale detection. Thus, the descriptor is invariant to both scaling and rotation. Also SIFT [1] is invariant to color and lighting because of spatial partitioning.
SIFT has the above listed merits, but however the main disadvantage is that SIFT is invariant to flip, which means that if the image is flipped, the result of SIFT [1] changes too. Flip is something that can be encountered a lot of times. Flip can occur because of change in point of views. Also, it can be seen of different video platforms like YouTube that people perform horizontal flipping on their videos or images to hide copyright violations. The figure 1 (a) and 1 (b) demonstrate flipping.

1.1 Object detection and its applications

Object detection is a technology that is related to image processing. It is the process of finding objects in any image. Object detection has a lot of scope these days in different fields like forensic to find evidence in photograph of crime scenes and other technologies like fingerprint detection and face detection.

Some of the interesting applications of object detection are:-

1) Face detection- This is one of the most widely used applications of object detection. Its use is not just in mobile phones and other electronic devices but also on the internet, for example, when you upload any image on social networking websites it can also detect faces in the image.

2) Vehicle detection- Object detection with tracking can help us determine the speed with which vehicle is travelling, thus it can be used in departments such as traffic police to catch people who exceed the speed limit.

3) People counting- Object detection can be used to count people from a certain image. It is used to count amount of people visiting a store or a certain area.

4) Forensics- It can be used to find any objects on the crime scene that could be used as evidence.

Fig -1(a): Original Image **Fig -1(b)**: Flipped Image

2. EXISTING SYSTEM

The existing detection systems store the video features in the form of codewords. A simple representation of video features makes the system more efficient.
Existing system uses Scale Invariant Feature Transform. SIFT [1] is derived from directionally sensitive gradient fields,

and is not flip invariant. However, flip-like transformations can be observed in images due to flipping performed using editor applications or change in capturing viewpoint.

3. PROPOSED SYSTEM

We have implemented the keypoint descriptor Flip-invariant SIFT, that preserves the original properties of SIFT while being tolerant to flips. F-SIFT [5] estimates the dominant curl of a local patch and then goes on to geometrically normalize the patch by performing flipping operation before the computation of SIFT. We implement F-SIFT for object detection in a video.

Due to the computation of dominant curl followed explicit flipping of local region, F-SIFT becomes slower than SIFT. However, it can be observed that there is improvement in detection effectiveness.

4. LITERATURE SURVEY

There are many different flip invariant descriptors like RIFT [2], SPIN [2], FIND [4] and MI-SIFT [3]. The main difference between these descriptors is the partitioning scheme of local region. SIFT [1] works by dividing a region into 4 × 4 blocks. SIFT [1] describes each grid with an 8 directional gradient histogram. Then the features are generated by concatenation of histograms in row major order from left to right and the histogram bins in clockwise manner. Flip transformation of the region will disorder the blocks and bins placement which could result in a different descriptor. The potential solutions for dealing with this problem can be partitioning scheme alteration or scanning order and feature transformation.

The partitioning scheme that is adopted by RIFT [2] is different as compared to SIFT [1]. It divides a given region along the log-polar direction. Then, the computation of 8-directional histograms is carried out for each division. These are then concatenated to form a descriptor. Since the partitioning scheme itself is flip as well as rotation invariant, RIFT [2] is not seen to be sensitive to order of scanning. GLOH can be viewed as an integrated version of SIFT. GLOH and RIFT [2] provide finer partitioning. However, the invariance property no longer exists after the spatial constraint is strengthened. SPIN [2] preserves flip invariance property while enforcing spatial information by encoding a region as a 2D histogram of pixel intensity and distance from region center. Despite the improvement, nevertheless, the empirical evaluation in reported that SPIN [2] as well as RIFT [2] and GLOH are outperformed by SIFT.

FIND [4] is another descriptor which scans the 8-directional gradient histograms and allows overlapped partitioning and the descriptors that are produced before and after flipping operation are also flipped versions of each other. The descriptors that are generated as a result of flip operation can be recovered by scanning the histograms in reverse order. FIND [4] uses parameter thresholding to estimates whether a region is left or right pointing. FIND [4] explicitly makes the descriptor flip invariant. When we compare two descriptors pointing left and right, we rearrange the descriptor components in proper order of feature matching. The pointing direction is is estimated based on the parameter setting. More importantly, incorrect estimation could lead to incorrect matching result. In addition, similar to RIFT [2], the partitioning scheme does not produce descriptor as distinctive as SIFT [1]. MI-SIFT [3] operates directly on SIFT while making it a flip invariant descriptor. It is done by identifying feature component groups that are placed disorderly due to flip operation and the identification is done explicitly.

5. FLIP-INVARIANT SIFT

Flip can happen along any arbitrary axis. However, this can become too complicated to understand. Hence, to make things simple we can assume that the flip occurs at an axis that is predefined and then followed by certain degree of rotation.

Thus to make a descriptor flip invariant we can normalize a given local region and the feature extraction can be performed by rotating the region to a certain axis and then performing flip operation about the axis. One of the important things in making a descriptor flip invariant is taking the decision whether flipping is to be performed or not.

One of the answers to this question is computation of dominant curl. Given a vector field F(x, y, z) defined in R3 which is differentiable in a region, the curl of F is given by

$$\nabla\times F = \begin{bmatrix} i & j & k \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \\ F1 & F2 & F3 \end{bmatrix} \quad \dots (1)$$

Here, curl is defined in a 2D discrete vector field I [5]. The curl at a point is the cross product on the first order partial derivatives along x and first order partial derivatives along y directions respectively. The flow along the tangent direction can be defined by

$$C = \sum_{(x,y)\in I} \sqrt{\frac{\partial I(x,y)^2}{\partial x} + \frac{\partial I(x,y)^2}{\partial y}} \times \cos\theta$$

Where

$$\frac{\partial I(x,y)^2}{\partial x} = I(x-1, y) - I(x+1, y)$$

$$\frac{\partial I(x,y)^2}{\partial y} = I(x, y-1) - I(x, y+1) \quad \dots (2)$$

The computed value of C can have either positive or negetive sign. The sign only changes if the vector field is flipped. If we enforce that the sign of flow is clockwise, normalization is performed by explicitly flipping the reigons whose sign are anti-clockwise.

$$C = \sum_{(x,y)\in I} \sqrt{\frac{\partial I(x,y)^2}{\partial x} + \frac{\partial I(x,y)^2}{\partial y}} \times \cos\theta \times G(x,y,\sigma) \quad ....(3)$$

For a region, rotated to its dominant orientation, the equation (3) is calculated to get the value and the direction of C. F-SIFT [7] achieves flip invariance by enforcing that the directions of all the regions should be the same as obtained by computing C. If the signs of any region is opposite, then explicit flipping is performed to normalize the regions.

Then in the next step, the SIFT [1] descriptors are extracted from these regions.
Thus, it can be said that F-SIFT [7] directly works on top of SIFT, thereby preserving all the properties of SIFT.

In our application, the input is a video in .yuv format. This video is first converted to avi file. Then the video is broken down into frames that are selected after a fixed interval of time. Then the object to be detected is selected, following which the algorithm computes and performs object detection.

6. RESULT

1) GUI of the application

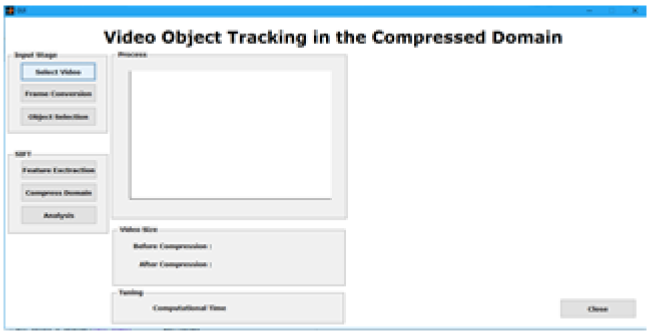


Fig- 2: Graphical User Interface

2) Video selection

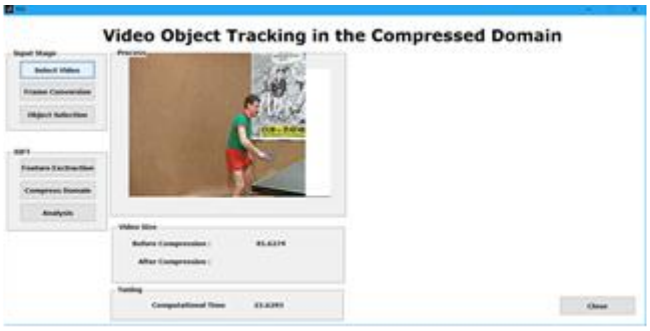


Fig- 3: Selection of the video

3) Conversion of video to frames

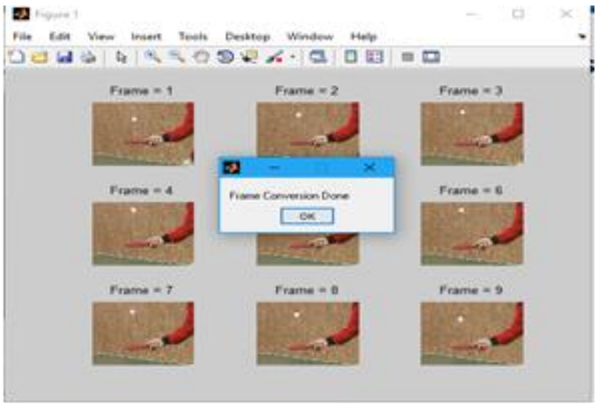


Fig- 4: Converting to frames

4) Object selection

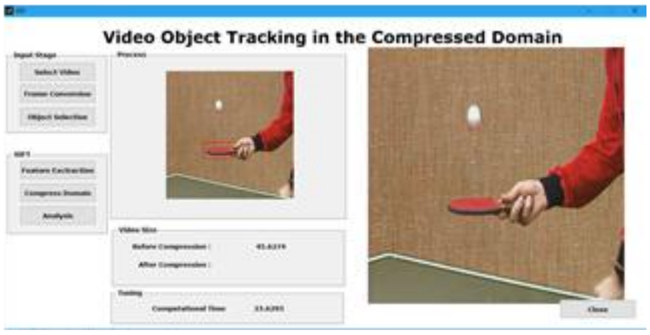


Fig- 5: Selecting object to detect

5) Detection of object in various frames (normal)

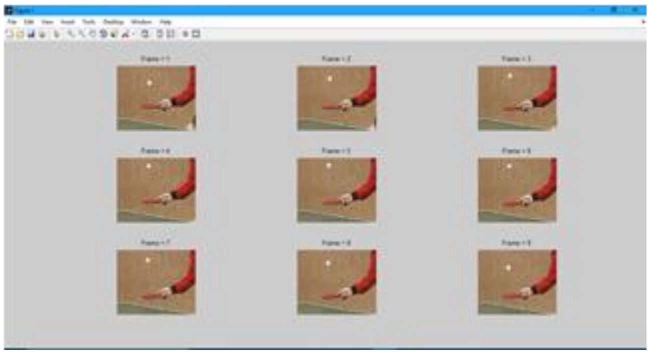
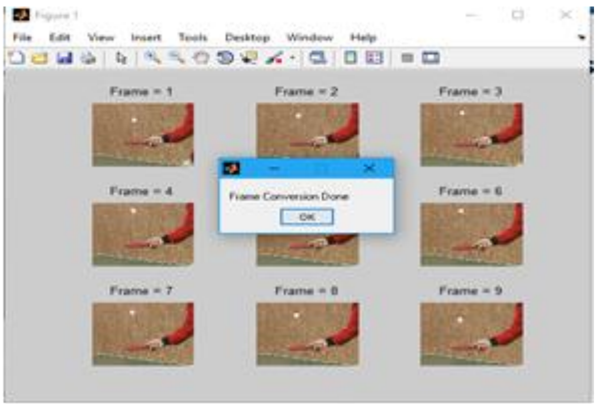


Fig- 6: Object detection

6) Detection of object in flipped frames



Frame Conversion Done

OK

Fig- 7: Flip object detection

We have tested the application for various objects based on their size and mobility, and we have derived the results as follows:-

- For objects that are large in size and moving quickly, the descriptor is accurate for about 50% of the times.
- For objects that are small and moving quickly, the descriptor is not much accurate in its detection.
- For objects that are large in size and moving slowly, the descriptor detects them most of the times.
- For static objects, the descriptor detects it almost in every frame.
- For small objects that move slowly, the descriptor detects them in around 60% of the frames.

7. CONCLUSION

We have implemented the application for detection of Objects in videos. We have implemented Flip- invariant SIFT which preserves the properties of Scale Invariant Feature Transform, while also making it invariant to flips.

8. REFERENCES

[1] D. Lowe, "Distinctive image features from scale-invariant keypoints,"Int. J. Comput. Vis., vol. 60, no. 2, pp. 91–110, 2004.

[2] S. Lazebnik, C. Schmid, and J. Ponce, "A sparse texture representation using local affine regions," IEEE Trans. Pattern Anal. Mach. Intell., vol. 27, no. 8, pp. 1265–1278, Aug. 2005.

[3] R. Ma, J. Chen, and Z. Su, "MI-SIFT: Mirror and inversion invariant generalization for SIFT descriptor," in Proc. Int. Conf. Image Video Retr., 2010, pp. 228–236.

[4] X. Guo and X. Cao, "FIND: A neat flip invariant descriptor," in Proc. Int. Conf. Pattern Recognit., Aug. 2010, pp. 515–518

.

[5] Wan-Lei Zhao and Chong-Wah Ngo, Flip-Invariant SIFT for Copy and Object Detection, IEEE Transactions On Image Processing, VOL. 22, NO. 3, MARCH 2013