Feature Point Tracking Algorithm Evaluation for Augmented Reality in Handheld Devices

Amila Perera, Akila Pemasiri, Sameera Wijayarathna, Chameera Wijebandara and Chandana Gamage

Department of Computer Science and Engineering, University of Moratuwa, Sri Lanka

Abstract. In augmented reality applications for handheld devices, accuracy and speed of the tracking algorithm are two of the most critical parameters to achieve realism. This paper presents a comprehensive framework to evaluate feature tracking algorithms on these two parameters. While there is a substantial body of knowledge on these aspects, a novel feature introduced in this paper is the use of error associated with the estimated directional movement in performance measurements to improve the evaluation framework. The work described in this paper is a comparative evaluation of nine widely used feature point tracking algorithms using the developed measurement framework and the results are interpreted based on the characteristics of the algorithms as well as the characteristics of test image sequences.

1 Introduction

In the field of computer vision, feature point tracking is the standard method most often used to extract motion information in an image sequence. There are many applications of feature point tracking in robotics, human computer interaction, surveillance and activity monitoring. In this study we have focused on evaluation of feature point tracking algorithms, which can be used to estimate the 3D pose of a calibrated 2D camera in augmented reality applications. This is important, for example, when we want to project a 3D object in a scene captured on a 2D camera. In this evaluation we have considered algorithms error rate, speed, robustness to unexpected changes of image properties and suitability for augmented reality applications.

The well-known feature point tracking algorithms require the image sequence to be of good quality. Algorithms use varying sets of assumptions (for example brightness consistency assumption in Kanade Lucas Tomasi (KLT feature tracker). The processing power consumed by different algorithms vary widely. In all of these algorithms, occurrence of a non-admissible event in an image sequence adds an element of uncertainty that affects the robustness of the output.

In this study, nine feature point tracking algorithms have been evaluated and compared. The objective of this study is to understand the performance and robustness of the algorithms under the criteria of their capability to operate on handheld mobile platforms with built-in cameras. The images captured using such devices may possess a lower image quality and may be subjected to involuntary device movements.

This paper is organized as follows. Section 2 includes the details of the algorithms used and section 3 describes the data set used for the evaluation. Section 4 focuses on the evaluation methods and section 5 describes the results.

2 Tracking Algorithms

Feature trackers can be categorized to two main types as Trackers using Optical Flow Algorithms and Trackers using Feature Descriptor Matching Algorithms.

For both of these types, feature points are required to be identified as an initial step. Typically, feature points allow them to be distinctively identified among other points.

Feature descriptors that store locality information of points are used in feature descriptor matching algorithms. Speeded Up Robust Features (SURF) [10], Scale Invariant Feature Transform(SIFT) [5], Shi-Tomasi Corner Detector [12], Binary Robust Invariant Scalable Keypoints (BRISK) [13], and Oriented FAST and Rotated BRIEF (ORB) [7] are feature detectors that are used in the implementation of algorithms that are compared in this work. Also, the feature detectors SURF, SIFT, BRISK and ORB have feature description capabilities in addition to detection. Furthermore, Fast Retina Keypoint (FREAK) [1] feature descriptor has also been used in this comparison. Trackers using optical flow algorithms and trackers using feature descriptor matching algorithms with different types of feature detectors and descriptors have been evaluated. Descriptions of different types of algorithms along with the names that are being referred in this paper are shown in Table 1.

Table 1. Description of trackers.

Tracker	**Type of tracker**	**Feature detector**	**Feature descriptor**
BRISK	Feature matching	BRISK	BRISK
FARNEB	Dense optical flow, using Gunnar Farnebacks algorithm	Shi-Tomasi	-
FREAK	Feature matching	SURF	FREAK
KLT	Lukas Kanade Optical flow	Shi-Tomasi	-
ORB	Feature matching	ORB	ORB
SIFT	Feature matching	SIFT	SIFT
SIFTSURF	Feature matching	SIFT	SURF
SURFSIFT	Feature matching	SURF	SIFT
SURF	Feature matching	SURF	SURF

It should be noted that when tracking feature points using a combination of a detector and a descriptor, certain combinations (for example, SIFT and FREAK) are not used if an improved version of an algorithm is used in another combination (for example, SURF and FREAK where SURF is an improved version of SIFT [10]). Furthermore, we have selected to evaluate only the currently known best performing algorithm in a family of algorithms (for example, Harris Corner detector [9] is not evaluated in favour of Shi-Tomasi Corner detector).

3 Evaluated Dataset

Table 2. Description of dataset.

Video	**Main challenging factors**	**Number of Frames**
Resolution : 1280*720		
1	Illumination variation, Scale variation	540
2	Autofocus, Motion blur, Very slow camera movement	469
3	Motion blur, High speed movements in all direction	290
4	Camera rotation, Shaky camera	273
5	Camera rotation, Autofocus, Motion blur	436
6	Camera rotation and translation, Autofocus, Motion blur	428
7	Camera rotation and translation, Ellipse shape camera path	700
8	Autofocus, Movement in all directions, Change in background texture	656
9	Illumination change, Autofocus	488
Resolution : 1920*1080		
10	Illumination change, Autofocus, Motion blur, Specular reflection	142
11	Background texture change	526
12	Specular reflection, Very slow camera movements, Camera rotation	393
13	Scale variation	548
14	Camera rotation, translation, Specular reflection	604
15	Camera rotation and translation, Scale variation, Shaky camera movements	526
16	Camera rotation and translation, Scale variation, Shaky camera movements	450
Resolution : 640*480		
17	Camera rotation and translation, Scale variation, Shaky camera movements	502

In order to select the best tracking algorithm for augmented reality applications in handheld devices, it was decided to use a dataset that include all the challenging factors in the expected context. The challenges considered were motion-blur, illumination variation, scale variation, camera rotation, occlusion, shaky camera and changes of the focal point. The occurrence of these features in the dataset is elaborated in the Table 2.

All the videos used as the dataset was captured using several mobile devices to ensure that the quality and the characteristics mimic live data which is expected to be received by an operational system.

4 Evaluation Method

The evaluation requires a set of feature points in the image sequence with known positions. This is called the ground truth. To obtain the ground truth few points were tracked throughout the image sequence manually. These points were selected after carefully observing the whole image sequence for features that can be clearly distinguished by the human eye (for example, the marked point in Fig. 1). The manual tracking of selected points was done by more than one tester and the averages of the tracked points were taken as the ground truth in order to reduce the human error.

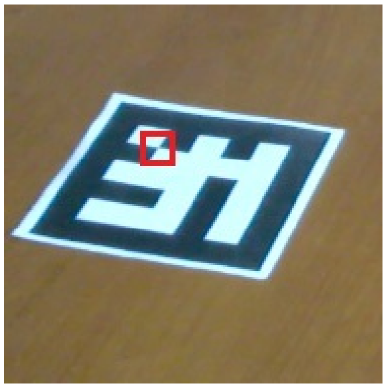

Fig. 1. Example of a manually tracked point.

The execution of implemented tracking algorithms with prepared video input gives a sequence of homography matrices as output. From these, for each frame in each image sequence, a perspective transformation is applied by using the first frame of the image sequence as the reference. Let ground truth points of the reference frame be $P^{'}_{g0}$, $P^{'}_{g1}$, $P^{'}_{g2}$ and $P^{'}_{g3}$. Then, using the homography matrix corresponding to a frame, the approximate positions of P_{g0}, P_{g1}, P_{g2} and P_{g3} in that frame is computed as shown in equation 1.

$$< P^{'}_{g0}, P^{'}_{g1}, P^{'}_{g2}, P^{'}_{g3} >= H < P_{g0}, P_{g1}, P_{g2}, P_{g3} > \quad (1)$$

$P^{'}_{g0}$, $P^{'}_{g1}$, $P^{'}_{g2}$ and $P^{'}_{g3}$ are the estimated points in n$^{\text{th}}$ frames which are corresponding to the ground truth points of that frame and H is the homography matrix.

Accuracy and performance are the two aspects which are considered in this paper. In order to measure the accuracy, the estimates taken are (1) the distance between projected points and the ground truth, which is the distance error and (2) the optical flow deviation angle, which is the directional error.

4.1 The distance error

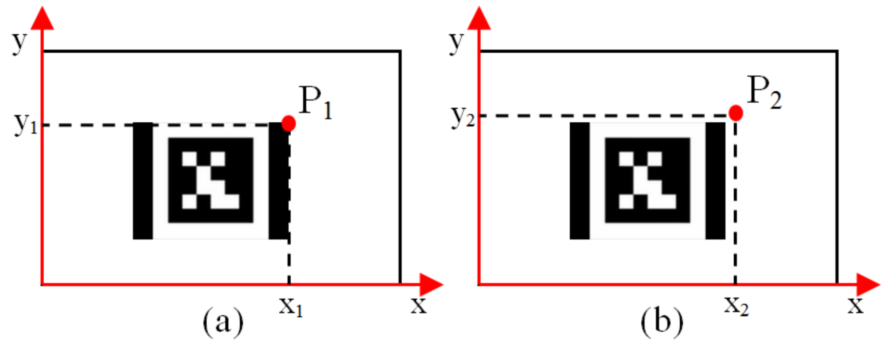

Fig. 2. Distance between the tracking points.

Taking the point P_1 in Fig. 2 as a point of the ground truth and P_2 in (b) of Fig. 2 as the corresponding tracked point by the algorithm the Euclidean distance between P_1 and P_2 was measured using equation 2. There d stands for the Euclidean distance between P_1 and P_2, x_1, x_2, y_1 and y_2 are the corresponding x, y coordinates of the points.

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{2}$$

4.2 The directional error

In Fig. 3 (a) P_1 is a ground truth point in the n^{th} frame and in Fig. 3 (c), P_2 is a point that is being tracked by the algorithm, corresponding to P_1 in Fig. 3 (a). $P^{'}_1$ and $P^{'}_2$ are the points in the $(n+1)^{th}$ frame and those are corresponding to P_1 and P_2 in Fig. 3 (a) and (c). Taking the vector between P_1 and $P^{'}_1$ as $U^{'}$ and the vector between P_2 and $P^{'}_2$ as $V^{'}$ where

$$V^{'} = (v_1, v_2) \tag{3}$$

$$U^{'} = (u_1, u_2) \tag{4}$$

The cosine of the angle between $U^{'}$ and $V^{'}$ can be obtained by equation 5.

$$cos(\alpha) = \frac{u_1.v_1 + u_2.v_2}{\sqrt{(u_1^2+u_2^2)*(v_1^2+v_2^2)}} \tag{5}$$

This angle is an indication of the directional movements.

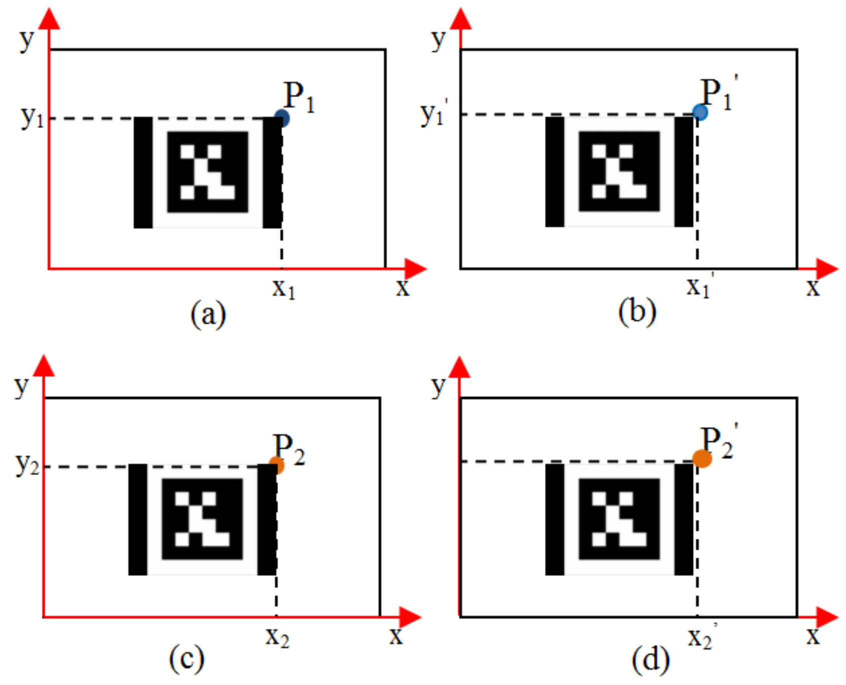

Fig. 3. The transition between n^{th} frame and $(n+1)^{th}$ frame.

4.3 Speed

In order to obtain an estimation of the performance of each algorithm the number of frames processed for a single unit of time was taken, using the same operating platform conditions.

4.4 Aggregated score

For a complete comparison all three measurements need to be aggregated into one score that will give the final ranking. The approach used in this work is to standardize the scores of each criterion and doing a weighted summation (equation 6).

$$FinalScore_T = \frac{\beta * Z_{S,T} + \frac{\gamma}{Z_{d,T}} + \frac{\delta}{Z_{\alpha,T}}}{\beta + \gamma + \delta} \tag{6}$$

In equation 6 β, γ and δ are the weights used according to the required priority for the speed, the distance error and for the directional error.

$$S = \frac{\sum_{i=1}^{v} \sum_{j=1}^{f_i} \frac{1}{t_{i,j}}}{\sum_{i=1}^{v} \sum_{j=1}^{f_i} 1} \tag{7}$$

$$E_d = \frac{\sum_{i=1}^{v} \sum_{j=1}^{f_i} \sum_{k=1}^{p_{i,j}} d_{i,j,k}}{\sum_{i=1}^{v} \sum_{j=1}^{f_i} \sum_{k=1}^{p_{i,j}} 1} \tag{8}$$

$$E_\alpha = \frac{\sum_{i=1}^{v} \sum_{j=1}^{f_i} \sum_{k=1}^{p_{i,j}} \alpha_{i,j,k}}{\sum_{i=1}^{v} \sum_{j=1}^{f_i} \sum_{k=1}^{p_{i,j}} 1} \tag{9}$$

Where v is the size of dataset, f is the number of frames in i^{th} video and $t_{i,j}$ is the time taken to process the j^{th} frame of i^{th} video and $p_{i,j}$ is the number of points marked as groundtruth in j^{th} frame of i^{th} video. d is the distance error (equation 2) and α is the directional error (equation 5).

In equation 6 $Z_{S,T}$ is the standardized value of S in equation 7 for the selected tracker T, $Z_{d,T}$ is the standardized value of E_d in equation 8 and $Z_{\alpha,T}$ is the standardized value of E_α in equation 9.

5 RESULTS

In analyzing the results, it was necessary to identify the characteristics of the image sequences. For this purpose, the amount of approximated blur presented in a video and the average contrast and color depth of each image in a video were taken in to account.

To obtain approximated blur measurement, the reciprocal of number of edges in the scene was considered. The number of edges was obtained using Canny Edge Detector [4].

To identify the illumination change, the three channel RGB image sequences were converted in to grey scale. For each image the average of pixel values were taken as it reflects the average intensity of the image.

For each image the standard deviation of pixel values were taken as the measurement of color depth. The results of the analysis are summarized below. In the figures related to interpretation of results the legend which is followed is depicted in Fig. 4. The area shown by a red rectangle is the focused area in each graph.

5.1 Tracker BRISK

This tracker exhibited poor performance in the presence of blur caused by motion as well as in the presence of blur caused by autofocus. This is due to BRISK descriptors storing different values for the same feature in the presence of blur. This behavior of the tracker on Video 2 is illustrated by Fig. 5.

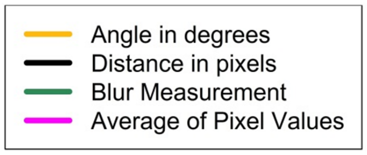

Fig. 4. Legend used in graphs.

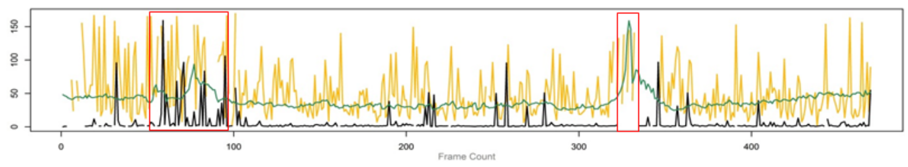

Fig. 5. Distance, angle and blur measure of Video 2 —Tracker BRISK.

5.2 Tracker FARNEB

This tracker indicates a gradual increase in error with time (Fig. 6) (Fig. 7).

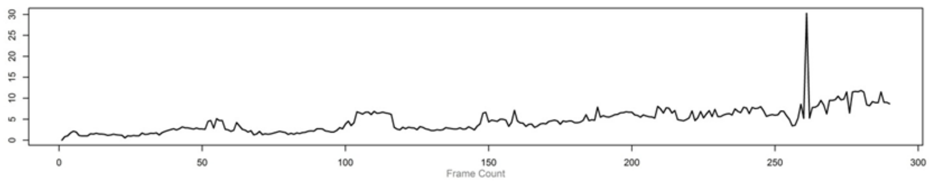

Fig. 6. Distance measure of Video 3 — Tracker FARNEB.

Sudden differences in image intensities are not handled well (Fig. 8). As this is an optical flow algorithm this tracker cannot handle fast rotations of the camera. Fig. 9 illustrates four frames from the image sequence of Video 12 which contains camera rotation along with the tracked points.

It can be observed that moving closer to a tracked feature is handled well if the feature gives the ability to identify the magnitude of flow in two orthogonal directions. Features points that resemble corners are most suited for this scenario. Fig. 10 illustrates three frames from the image sequence of Video 14 taken at difference distances along with the tracked point. From that it can be identified that corners are the best tracked points. Poorly tracked points are marked with colored rectangles.

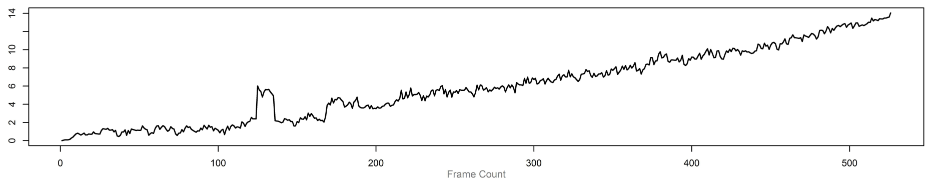

Fig. 7. Distance measure of Video 11 — Tracker FARNEB.

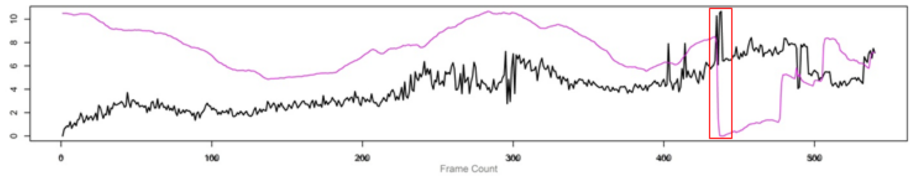

Fig. 8. Distance measure and average of pixel values of Video 1 — Tracker FARNEB.

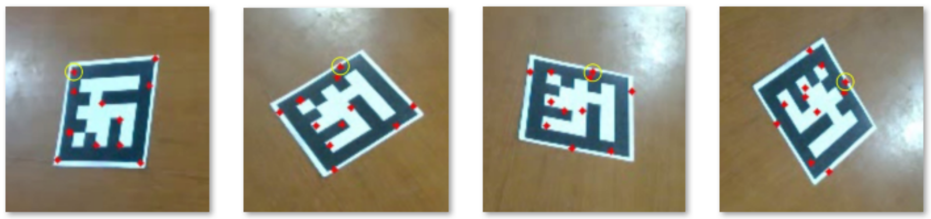

Fig. 9. Camera rotation in Video 12.

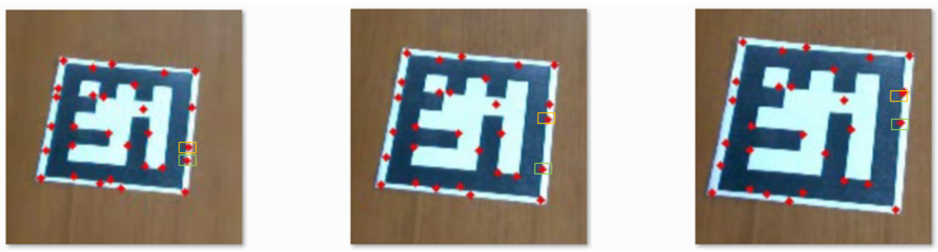

Fig. 10. Points tracked in Video 12 –Tracker FARNEB.

5.3 Tracker FREAK

This tracker is not capable of handling blur caused by autofocus or fast motion of the camera (Fig. 11). As observed, orientation changes and scale variations of the features are handled well by tracker FREAK.

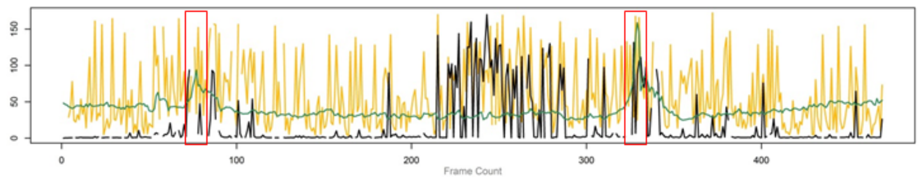

Fig. 11. Distance, angle and blur measure of Video 2 – Tracker FREAK.

Fig. 12 illustrates the trajectories of ground truth points in Video 13. Starting points are indicated in red and end points are indicated in blue. It can be seen that the video has been subjected to scale variation. The distance error related to that is illustrated in Fig. 13.

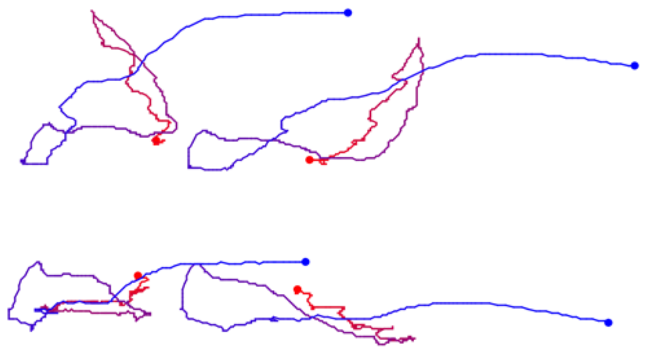

Fig. 12. Trajectories of ground truth points in Video 13.

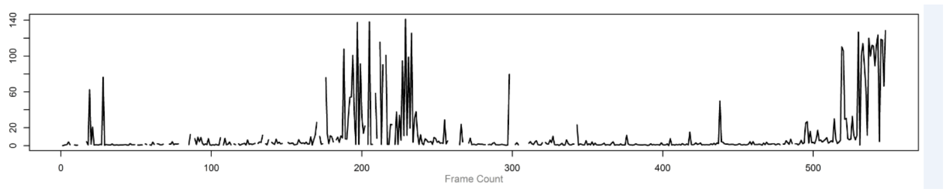

Fig. 13. Distance measure of Video 13 — Tracker FREAK.

5.4 Tracker KLT

This tracker depicts a gradual increasing error proportionate to the time (Fig. 14). In Tracker KLT, sudden difference in image intensity of image sequence disrupts the tracking process (Fig. 15). Blurring caused by autofocus or fast motion does not have an effect on this tracker (Fig. 16). This tracker as an optical flow algorithm which incorporates a simple motion model cannot handle rotation.

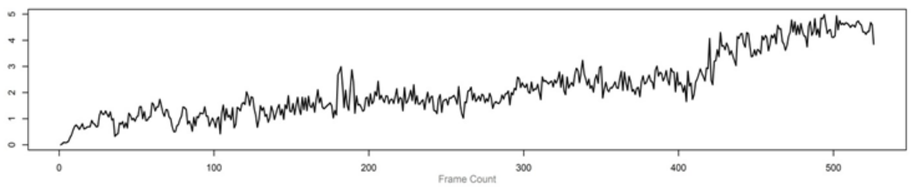

Fig. 14. Distance measure of Video 11 – Tracker KLT.

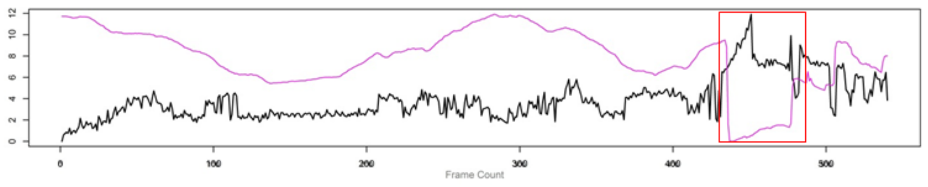

Fig. 15. Distance measure and average of pixel values of Video 1 – Tracker KLT.

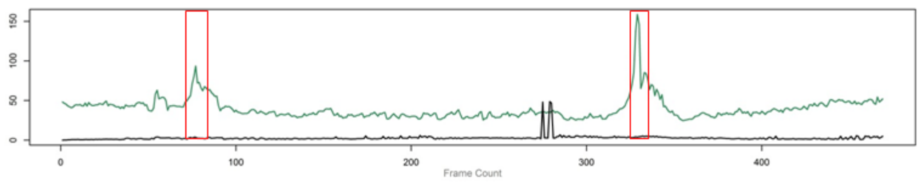

Fig. 16. Distance measure and blur measurement of Video 2 – Tracker KLT.

5.5 Tracker ORB

Tracker ORB has the capability to handle sudden intensity changes (Fig. 17). Accuracy of tracking decreases with scaling of the features being tracked. Motions blur and blur caused by autofocus does not affect the tracking performance

(Fig. 18) and this tracker exhibits a considerable accuracy even at the time where the orientation changes of the features exist.

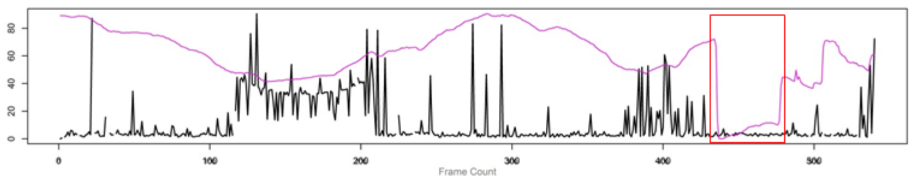

Fig. 17. Distance measure and average of pixel values of Video 1 – Tracker ORB.

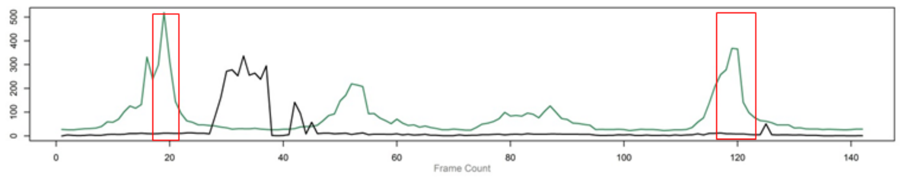

Fig. 18. Distance measure and blur measurement of Video 10 – Tracker ORB.

5.6 Tracker SIFT, Tracker SIFTSURF, Tracker SURFSIFT, Tracker SURF

These trackers use different combinations of SIFT and SURF feature detector and descriptor. The performance of trackers that use SURF as the descriptor show poor performance relative to the trackers that use SIFT descriptor.

6 CONCLUSION

The overall evaluations of the algorithm′s consider the algorithms accuracy and speed. For accuracy two measures are used which are the distance error and the directional error. For speed the number of frames processed per second is used.

By considering only the criterion of error distance it can be concluded that Lucas Kanade optical flow algorithm (Tracker KLT) demonstrates the best performance consistently. But it should be noted that this algorithm does not handle rotation accurately due to the motion model utilized in the algorithm. Both optical flow algorithms have shown low distance error in comparison to feature matching algorithms. Among the feature matching algorithms Tracker ORB algorithm and the Tracker SURFSIFT algorithm shows consistent performance which is in the range of the optical flow algorithms. It can be concluded that optical flow algorithms can process translation and give consistent tracking but fails on rotation. Feature matching algorithms can handle rotation but fails on motion blur and give inconsistent tracking output.

When the evaluated algorithms are listed in ascending order according to the number of frames processed per second, it can be concluded that Tracker KLT is the fastest algorithm. It can be observed that the margin between Tracker KLT (Tracker 4) and the rest of the algorithms is very high. Most of the feature matching algorithms have very low speed scores except for the algorithm which uses BRISK detector and descriptor. The dense optical flow algorithm is also in the range of feature matching algorithms. This is due to the fact that KLT is a sparse optical flow algorithm and only considers a few number of feature points compared to the feature matching algorithms and dense optical flow algorithm.

According to the directional error, it can be concluded that optical flow algorithms which takes into account the motion details found within two consecutive images to estimate the succeeding position of a point will give more accurate output in contrast to feature matching algorithms which only use the locality information stored in the feature vectors. The dense optical flow algorithm has the best score for this criterion.

Modern mobile devices has the ability to capture high resolution images typically having camera specifications in range 4MP to 20MP or more. But two devices with the same camera specification can give different qualities in the same conditions. It is clear that the higher resolutions give accurate results but at the cost of speed. This is due to the fact that mobile platforms have restrictions on the number of calculations per second and memory available. Some of the interesting issues found was due to software features such as auto focusing and auto white balancing. Auto focusing created blur in the image sequence while AWB created intensity changes which were undetectable to normal viewing but disrupted Tracker performance.

Normally a user using a mobile device for AR will be restricted to certain types of movements. Movements that include translation, circular motion around an estimated center (example: Walking around an augmented object) and scaling (Moving towards and backwards). Pure rotations can be expected to be less frequent as it is not a normal viewing movement as it does not change the perspective of viewing. Thus it would be much profitable to have a fast and accurate tracker for the more prevailing movements and have another algorithm to handle rotations supporting the main algorithm in use. According to the results it can be concluded that sparse optical flow algorithms are better equipped for handling translation at high speeds whereas feature matching algorithms take much more time to process a frame but better at handling rotations (orientation changes). This is due to the fact that sparse optical flow algorithms consider the information within a constrained area around selected keypoints and use a simplified motion model which allows for fast processing. Feature matching uses the information from the complete frame and takes much more time for processing as the complexity of calculations are much higher but this allows to find correspondence between images regardless of the motion/scaling that occurred by not having restrictions on the motion model.

Our context of augmented reality requires both speed and accuracy for a satisfactory outcome. When the scores are calculated (equation 6) with the values $\beta = 0.3$, $\gamma = 0.4$ and $\delta = 0.3$ the final score of each tracker is in Table 3.

Table 3. Final score of trackers.

Tracker	**Final Score**
KLT	2.27411145
FARNEB	0.231517254
ORB	0.142172204
FREAK	-0.326357422
SURFSIFT	-0.371670024
SIFT	-0.388266766
BRISK	-0.461264852
SURF	-0.560889913
SIFTSURF	-0.672624851

7 FUTURE WORK

It must be noted that the implementation of these algorithms are basic.Using different optimization techniques the performance of these algorithms can be improved. Furthermore there are many combinations of descriptors, detectors and optical flow algorithms that have not been evaluated in this study. Evaluating these optimizations and new algorithms can be done using the same method proposed in this paper.

References

1. Alahi, A., Ortiz, R., Vandergheynst, P.: FREAK: Fast Retina Keypoint. IEEE Conference on Computer Vision and Pattern Recognition (2012)
2. Barron, J.L., Fleet, D.J., Beauchemin, S.S., Burkitt, T.A.: Performance of Optical Flow Techniques. Conference on Computer Vision and Pattern Recognition (1992) 236-242
3. Black, J., Ellis, T.J., Rosin, P.: A Novel Method for Video Tracking Performance Evaluationl Expansion. IEEE Performance Evaluation of Tracking and Surveillance Workshop (2003)
4. Canny, J.: A Computational Approach to Edge Detection. IEEE Trans. Pattern Analysis and Machine Intelligence **8** (1986) 679-698
5. David, G.L.: Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision **60** (2004) 91–110

6. Dubrofsky, E.: Homography Estimation. M.Sc. Canada: University Of British Columbi (2009)
7. Ethan, R., Vincent, R., Gary, R.B.: ORB: An Efficient alternative to SIFT or SURF. International Conference on Computer Vision (2011) 2564–2571
8. Farneback, G.: Two-Frame Motion Estimation Based On Polynomial Expansion. Scandinavian Conference on Image Analysis (2003)
9. Harris, C., Stephens, M.: A Combined Corner and Edge Detector. Proceedings of the 4th Alvey Vision Conference (1988) 147-151
10. Herbert, Bay., Tinne T., Luc V.G.: SURF: Speeded Up Robust Features. 9th European Conference on Computer Vision (2006)
11. Lucas, B.D., Kanade, T.: An Iterative Image Registration Technique with an Application to Stereo Vision. International Joint Conference on Artificial Intelligence **2** (1981) 674–679
12. Shi, J., Tomasi, C.: Good Features to Track. 9th IEEE Conference on Computer Vision and Pattern Recognition (1994)
13. Stefan, L., Margarita, C., Roland, S.: BRISK: Binary Robust Invariant Scalable Keypoints. International Conference on Computer Vision (2011) 2548–2555
14. Tommasini, T., Fusiello, J., Trucco, E, Roberto, V.,: Online Object Tracking: A Benchmark. Conference on Computer Vision and Pattern Recognition (1998) 178–183
15. Wu, Y., Lim, J., Yang, M.H.: Online Object Tracking: A Benchmark. Conference on Computer Vision and Pattern Recognition (2013) 2411 –2418