Hybrid Detect-Track and Detect-Describe-Associate Tracker

BoofCV Tech Report 2012-0001
Author: Peter Abeles
Document Version: 1
Date: 2012 December 6

Abstract

The following tech report describes the combined tracker provided in BoofCV. The combined tracker is a hybrid of Detect-Describe-Associate (DDA) and Detect-Track (DT) tracking approaches. DDA trackers are robust to abrupt changes in point of view, but does not take advantage of strong temporal correlation in video sequences. DT is stable and fast for gradual changes, but fails during abrupt transitions. In BoofCV, any feature detector or descriptor can be combined with KLT to create a new tracker which exhibits better speed and stability in general. Performance is evaluated using real-world data from different environments, camera motions, illumination, and image compression.

1 Introduction

Image point feature tracking in video sequences is a key component in many computer vision problems, such as structure from motion, and object detection/recognition. Specific applications include: scene recognition for loop-closure, face recognition, video stabilization, visual odometry, and augmented reality. The most popular feature trackers can be categorized as employing detect-describe-associate (DDA) or detect-track[1] (DT) paradigms.

DDA-based trackers operate by 1) detecting feature locations and characteristics, 2) describing each feature with local image information, and 3) associating features between images. Numerous feature detectors (Harris [4], Shi-Tomasi [13], FAST [12], MSER [8]) and local feature descriptors (SIFT [6], SURF [1], BRIEF [3]) are available for DDA trackers. Association is done by selecting pairs of feature descriptors between two images that minimize an error metric. DDA trackers are robust to changes in view, but are often computationally expensive and complex to implement.

DT operates by first detecting features, then updating the location of each track as new images arrive. Kanade-Lucas-Tomasi (KLT) [7, 13] tracker and its variants are all DT tracker. Unlike DDA trackers, DT trackers update each feature's location by performing a local search in image space. However, if the scene changes abruptly, a DT tracker will fail.

The primary contributions of this paper are a new hybrid tracker and the formal analysis of several trackers against a set of video sequences. The hybrid tracker combines the best features of DDA and DT tracking. It works by nominally using a DT tracker, but when an individual track experiences a fault, a DDA-type approach is used to recover the track. A modular design is employed that allows any DDA or DT tracker to be used without modification. As new DDA and DT trackers are developed, their improved performance will automatically improve the hybrid tracker.

The performance of two specific implementations of the hybrid tracker are considered, FH-SURF-KLT and FH-BRIEF-KLT. Track stability and runtime speed are evaluated across a set of real-world image sequences with known homographies. In addition, the performance of KLT, FAST-BRIEF, FH-SURF, FH-BRIEF are also considered[2].

2 Related Work and Background

The first step for both DDA and DT trackers is feature detection. Corner (e.g. Harris and Shi-Tomasi) and blob (e.g. Hessian Determinant [5]) detectors are commonly used to identify salient features within images for tracking purposes. Unlike corners, blobs exhibit a strong response in scale-space and are commonly used in more recently proposed DDA trackers for scale invariance.

After a feature has been detected, a description of that feature is extracted from its local region. The simplest descriptors are composed of pixel intensity values. KLT is a DT tracker and uses raw intensity values for track descriptions. DDA requires a more robust solution to handle greater changes in appearance. Recently proposed DDA descriptors work by encoding the magnitude and direction of changes in intensity as a smooth function of location.

The major philosophical difference between the two approaches involves how tracks are updated. DDA perform an expensive detection process across the entire image, computes descriptors for each found feature, and associate features. DT perform a local update for each track independently, which minimizes the difference between a track's description and the image.

Feature association is an expensive operation for DDA trackers. The simplest solution has a complexity of $O(N^2)$, although

[1] More commonly referred to as *"detect then track"*.

[2] Note that specific implementations of the hybrid tracker are named using a three word pattern, (DETECTOR)-(DESCRIPTOR)-(DT). DDA trackers are named using two words (DETECTOR)-(DESCRIPTOR). Each word signifies an internal algorithm.

faster, but more complex, alternatives are available (e.g. k-d trees [2]). An optimal solution is defined as the two features which minimize a distance metric, often the L_2 (Euclidean) norm:

$$a_i = \min_j ||F_i^0 - F_j^1|| \quad (1)$$

where F^0 and F^1 are sets of feature descriptors from two different images.

Association runtime performance can be improved by using specialized architecture specific SIMD instructions. An alternative way to reduce the complexity is to only consider local associations, like DT trackers do. However, only considering local associations negates a primary advantage of DDA trackers, which is that they can recover from large motions that DT cannot.

DT trackers updates tracks using a local search. Specifically, KLT updates track locations using a translational motion model and minimizes the difference between the track's description and the image using the following cost function:

$$\epsilon = \int_W \left[I(x-d) - J(x)\right]^2 w \cdot dx \quad (2)$$

where W is a local region around location x, w is an optional weight, d is the translation parameter being optimized, and I and J are the first and second images in the sequence. The largest displacement that can be estimated is determined by the size of W. Larger displacements are typically handled using a pyramidal approach.

Unlike DDA trackers, KLT's runtime speed for track update is dependent on the number of tracks and not image size. Track stability is often improved by updating the description after each frame. The downside to updating the descriptor is that image noise will cause tracks to perform a random walk.

KLT is capable of automatically detecting many types of track faults. A fault is declared when any of the following conditions are met: a) when the residual pixel error is too large, b) an impossibly large motion is found, or c) the track moves outside the image.

Several hybrid trackers that employ both KLT and a DDA tracker have been proposed in the past. In Uemura and Mikolajczyk [14] diverged KLT tracks are detected using a SIFT descriptor when detecting human actions. In Pilet and Saito [10] robustness is added to a region based normalized-cross-correlation (NCC) tracker by switching to KLT when association fails to find a match. To reduce image processing overhead for visual loop closing in SLAM, Pradeep et. al [11] proposed to attach SIFT descriptions to KLT tracks.

The hybrid tracker works by nominally using a DT tracker, but when an individual track experiences a fault, a DDA type approach is used to recover the track. Pradeep's approach is the most similar to the one discussed in this paper. How and when tracks are respawned is the primary difference between the discussed tracker and Pradeep's approach. In the is work, tracks are respawned on an individual basis, while Pradeep respawns all tracks when a new keyframe is set.

Tracking stability is not examined in any of the cited works. In the just mentioned papers, their hybrid trackers are mentioned in passing since their focus is on other vision problems. In this work, a detailed analysis of stability and runtime performance is provided.

3 Algorithm Description

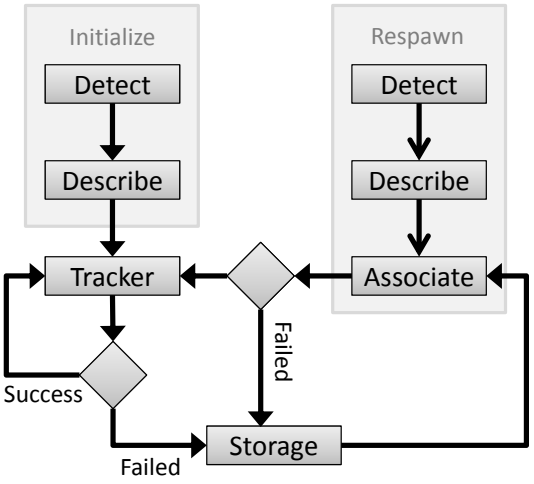

Figure 1: Flow chart showing track life cycle. Each image feature is tracked as long as possible using DT tracker. Dropped DT tracks are periodically respawned by detecting new features inside an image.

The hybrid DDA-DT tracker has a modular design and specific implementations are provided algorithms for feature detection, describing features, and tracking features. Detectors return a list of feature locations and associated characteristics (e.g. scale and orientation). For each detected feature, the descriptor returns a DDA description. The DT tracker updates feature locations as new images in the sequence arrive.

A feature track is defined by a 2D location and has two types of descriptions, namely, DDA description and DT description. The DDA description is immutable and the DT description is updated after each image is processed. Even after a track has been respawned the DDA description is not changed.

Tracking has three phases: 1) initialization, 2) tracking, and 3) respawning. During initialization, new tracks are created from detected features and assigned their descriptions. Track locations are updated using the provided DT tracker. If a track fault is detected, the track is dropped and its description placed in storage. After too many tracks have been dropped, an attempt is made to respawn the tracks by associating them to newly detect features in the image. A flow diagram of the track life cycle is shown in Figure 1.

During respawn or when spawning new tracks, all active tracks and tracks in storage are associated with detected features. After association is finished, tracks in storage that have been successfully associated are respawned. Associations to active tracks are ignored, but by considering them the false positive rate is significantly reduced. New tracks can be spawned from unassociated detected features.

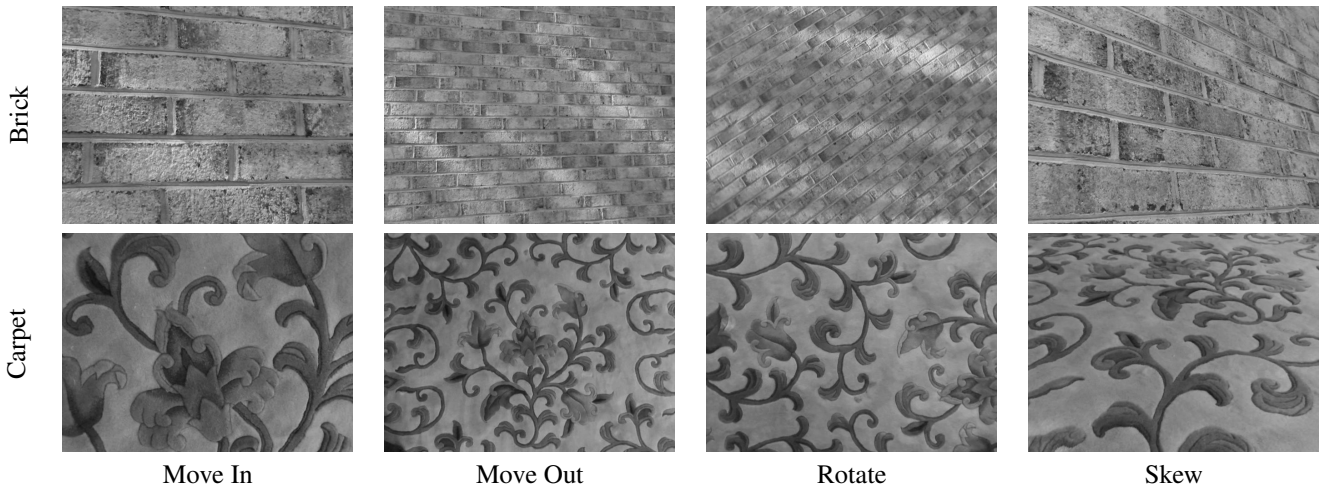

Figure 2: Four different 6-DOF camera motions are evaluated for scenes of planar objects. *Bricks* is more uniform in appearance and level of texture, while *carpet* has clearly defined objects with less interior texture

Figure 3: Changes in ambient light are evaluated in *illumination*. In *panoramic* the camera is approximately rotated about its focal point in an urban environment. For *compressed*, the bricks skewed sequence is highly compressed using JPEG. Compression artifacts are difficult to see in this figure due the reduced image size.

4 Design Issues

When selecting algorithms to use inside the hybrid tracker, there are several issues that need to be considered. Can the DT tracker track features found by the detector? Can the DT tracker detect track faults?

Texture is used to by KLT to track features. Unlike corner features, blob features contain all their texture information along the blob's outside edges, with little information inside. In theory, if a large blob was detected, then the lower pyramid layers (high resolution) would fail due to the textureless inner region. In practice, this was found not to be an issue with real-world data due to sufficient texture across scales.

In the hybrid algorithm, a feature only switches to using the DDA approach when the DT tracker detects a fault. KLT trackers typically employ several methods for detecting faults, but are susceptible to gradual drift. As will be shown below, it is possible for a slowly changing video sequence to produce worse performance than one with abrupt changes.

5 Experimental Setup

Performance is evaluated using video sequences for which a homography describes the relationship between each video frame. This approach is similar in spirit to approach taken by Mikolajczyk et. al [9], where sequences of still images are provided along with corresponding homographies. Videos were collected using a consumer grade handheld point-and-shoot camera. Specifically, a Sony Cyber-Shot DSC-Hx5V camera capturing 640x480 MP4 video.

A homography provides a unique mapping between pixels in two image, $x' \propto H \cdot x$ where $H \in \Re^{3\times 3}$ is the homography and x is a homogeneous 2D pixel coordinate. A homographic relationship comes about when the scene is planar, or the camera's motion is purely rotational. Thus, the evaluated video sequences are either of planar objects, panoramic, or taken with a stationary camera.

An automated algorithm is used to reconstruct the "true" homography between the first image and each subsequent image. Procedure: 1) Remove lens distortion. 2) Track point features. 3) Estimate homography using points. 4) Non-linear refinement using inlier point set. 5) Non-linear refinement that minimizes average squared difference of pixel intensity.

Track stability is measured using F-measure Eq. 3, which is a function of precision and recall.

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{3}$$

Precision is defined as the number of true positive tracks divided by the total number of active tracks. Recall is defined as

the number of true positives divided by the initial number of tracks. A true positive track is defined as a track that is within tolerance of the true feature location. The choice of tolerance for defining true positive tracks is some what arbitrary, and a value 5 pixels was selected.

Four scenes are used to evaluate performance (see figures 2 and 3). Two scenes of planar objects with four different 6-DOF camera motions, one still camera with of a bookshelf with variable illumination, and one panoramic of an urban environment. The two 6-DOF scenes, *bricks* and *carpet*, are views of planar objects with different textures intended to stress trackers differently. The effect of frame rate/object speed on tracker performance is examined by reprocessing a sequence with axial camera rotation at different frame rates.

Table 1: Summary of Evaluated Trackers

	Type	Detector	Descriptor
KLT	DT	Shi-Tomasi	5x5 Pyramid
FAST-BRIEF	DDA	FAST	BRIEF-64
FH-BRIEF	DDA	FH	BRIEF-64
FH-SURF	DDA	FH	SURF-64
FH-BRIEF-KLT	**Hybrid**	FH	Multiple
FH-SURF-KLT	**Hybrid**	FH	Multiple

Table 2: Detector Capabilities

	Invariance				Info		Type
	Translation	Scale	Orientation	Illumination	Scale	Orientation	
Shi-Tomasi	X		X	X			Corner
FAST	X		X				Corner
FH	X	X	X	X	X	X	Blob

A summary of feature detectors considered is in Table 1 and evaluated trackers in Table 2. The two implementations of the hybrid tracker, FH-SURF-KLT and FH-BRIEF-KLT, are highlighted in Table 1. Inside the detectors table, *invariance* refers to detection invariance and *info* refers to extracted feature characteristics. SURF is a popular state-of-the-art descriptor designed for speed an stability. BRIEF is a more recently proposed descriptor that uses binary encoding. Fast Hessian (FH) is the scale-space blob detector proposed with SURF. Both Shi-Tomasi and FAST are corner detectors, with the former using the image gradient and the latter using pixel intensity values. FAST is popular in applications with constrained computation resources due to its speed, but less stable than other detectors.

FH-BRIEF-KLT and FH-SURF-KLT are two variants of the hybrid tracker. To maximize speed, the FAST-BRIEF tracker uses the FAST detector. Both FH-BRIEF and FH-SURF trackers use the FH detector while using BRIEF and SURF descriptors respectively. All trackers use the first frame as their keyframe and never spawn new tracks.

If provided, all descriptors, detectors, and trackers use recommended parameters from the original papers. Algorithms are tuned for stability across across all scenarios. Runtime performance is evaluated on an Intel Quad Core 2 Q6600 at 2.4 GHz running Ubuntu Linux with kernel 2.6.35. Algorithms are provided by BoofCV[3], see project home page for a discussion on correctness and performance.

6 Results

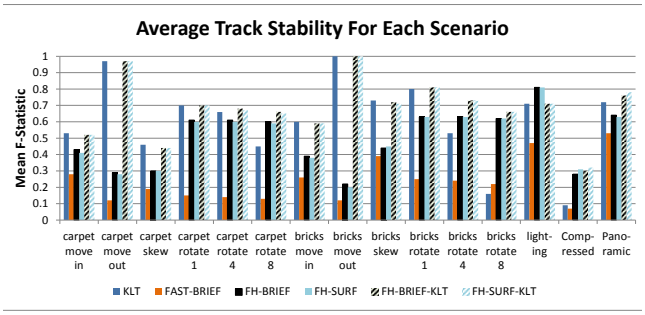

Figure 4: Average F-statistic across each scenario, higher is better. Hybrid trackers are top performer in almost every scenario. Other trackers exhibit greater variability, with poor performance in one or more scenarios.

Stability as an average across each sequence is shown in Figure 4. The complementary nature of KLT and the DDA trackers is clearly evident. During move out scenarios KLT excels while DDA trackers perform poorly and the reverse is true for rotation scenarios. Note that the hybrid trackers perform well in all scenarios.

The rate at which KLT tracks drift is a function of image noise and how fast features move. Track drift is troublesome for hybrid trackers since they rely on KLT to detect its own failures. This issue is illustrated by evaluating stability after skipping N frames for axial camera rotation (Figure 6). When every frame is processed (N=1) KLT's tracks drift but are never dropped. As the number of dropped frames increase, KLT starts to detect the failures and drop tracks. DDA tracker performance is unaffected by dropped frames. The hybrid trackers performance can actually improve as more frames are dropped because KLT can then detect failed tracks.

Stability results as a function of video frame is shown for the other scenarios in Figure 7. For move in, move out, and skew KLT has the best performance, with the exception of illumination and compressed, closely followed by the hybrid trackers. Hybrid trackers performed slightly worse than KLT in the just mentioned scenarios because they considered associations with features no longer visible. For the compressed scenario, KLT performance rapidly decreases while hybrid and DDA trackers experience a more gradual decay. Hybrid trackers performed best for the panoramic sequence.

Changes in scale are difficult for feature detectors to handle because of the camera's finite resolution. KLT does not need to handle large changes in scale explicitly because it updates the track's appearance model after each frame is processed. KLT

[3] http://boofcv.org

experiences feature drift in the illumination scenario, which in turn affects the hybrid trackers. FAST detector is not invariant to illumination, causing FAST-BRIEF to perform relatively poor.

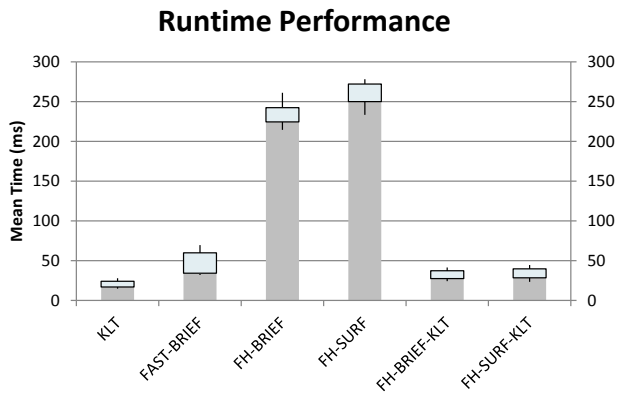

Figure 5: Runtime performance box and whisker plot for all scenarios, lower is better. Y-axis is the average time to process a frame in milliseconds. Statistics shown are minimum, 25%, median, 75%, and maximum. Hybrid trackers are about 7.5 times faster than comparable DDA trackers and 1.5 times slower than KLT on average.

Runtime performance is shown in Figure 5. KLT is the fastest tracker, followed by the hybrid trackers. BRIEF is the fastest DDA tracker, running several times faster than the two FH based DDA trackers.

References

[1] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. *Computer Vision and Image Understanding (CVIU)*, 110:356–359, 2008.

[2] J. L. Bentley. Multidimensional divide-and-conquer. *Communications of the ACM*, 23(4):214–229, Apr. 1980.

[3] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, and P. Fua. Brief: Computing a local binary descriptor very fast. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34:1281–1298, 2012.

[4] C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference*, 1988.

[5] T. Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30(2):79–116, 1998.

[6] D. Lowe. Distinctive image features from scale-invariant keypoints, cascade filtering approach. *International Journal of Computer Vision (IJCV)*, 60:91–110, January 2004.

[7] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *International Joint Conference on Artificial Intelligence*, 1981.

[8] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761–767, 2004.

[9] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27:1615–1630, October 2005.

[10] J. Pilet and H. Saito. Virtual augmenting hundreds of real pictures: An approach based on learning, retrieval, and tracking. In *IEEE Virtual Reality 2010*, 2010.

[11] V. Pradeep, G. Medioni, and J. Weiland. Visual loop closing using multi-resolution sift grids in metric-topological slam. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[12] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer Vision*, volume 1, pages 430–443, May 2006.

[13] J. Shi and C. Tomasi. Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, 1994.

[14] H. Uemura and S. I. K. Mikolajczyk. Feature tracking and motion compensation for action recognition. In *British Machine Vision Conference (BMVC)*, 2008.

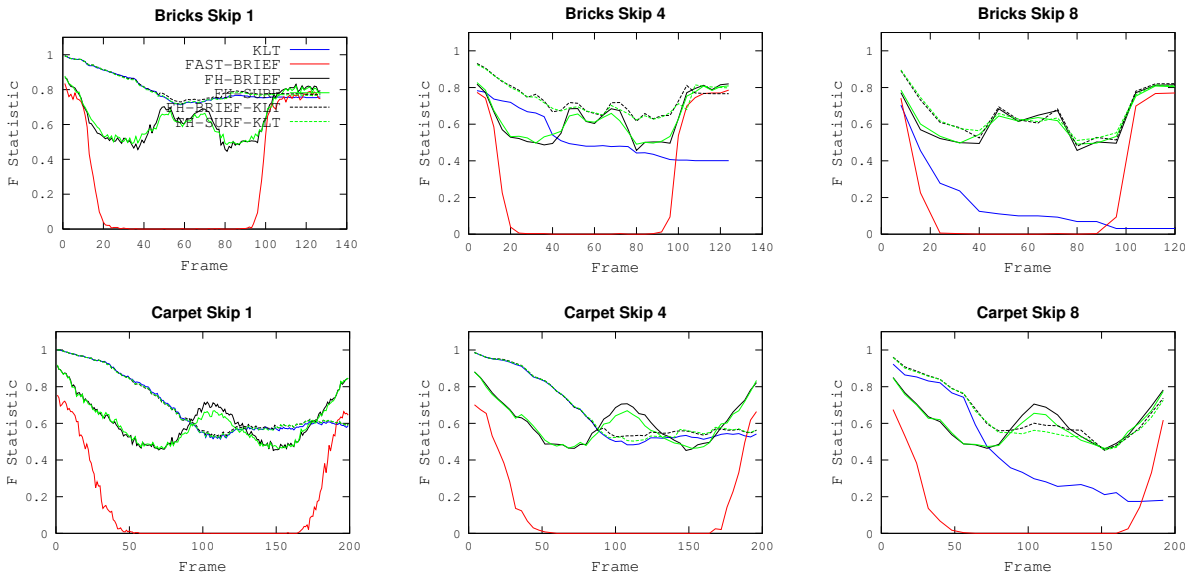

Figure 6: Track stability by video frame for axial rotation when N frames are skipped, higher is better. Performance of DDA trackers is independent of the number of skipped frames, while KLT is highly dependent. Hybrid trackers avoided failure condition and maintain good tracking performance. Note that the hybrid trackers often performed better when KLT experienced catastrophic failure.

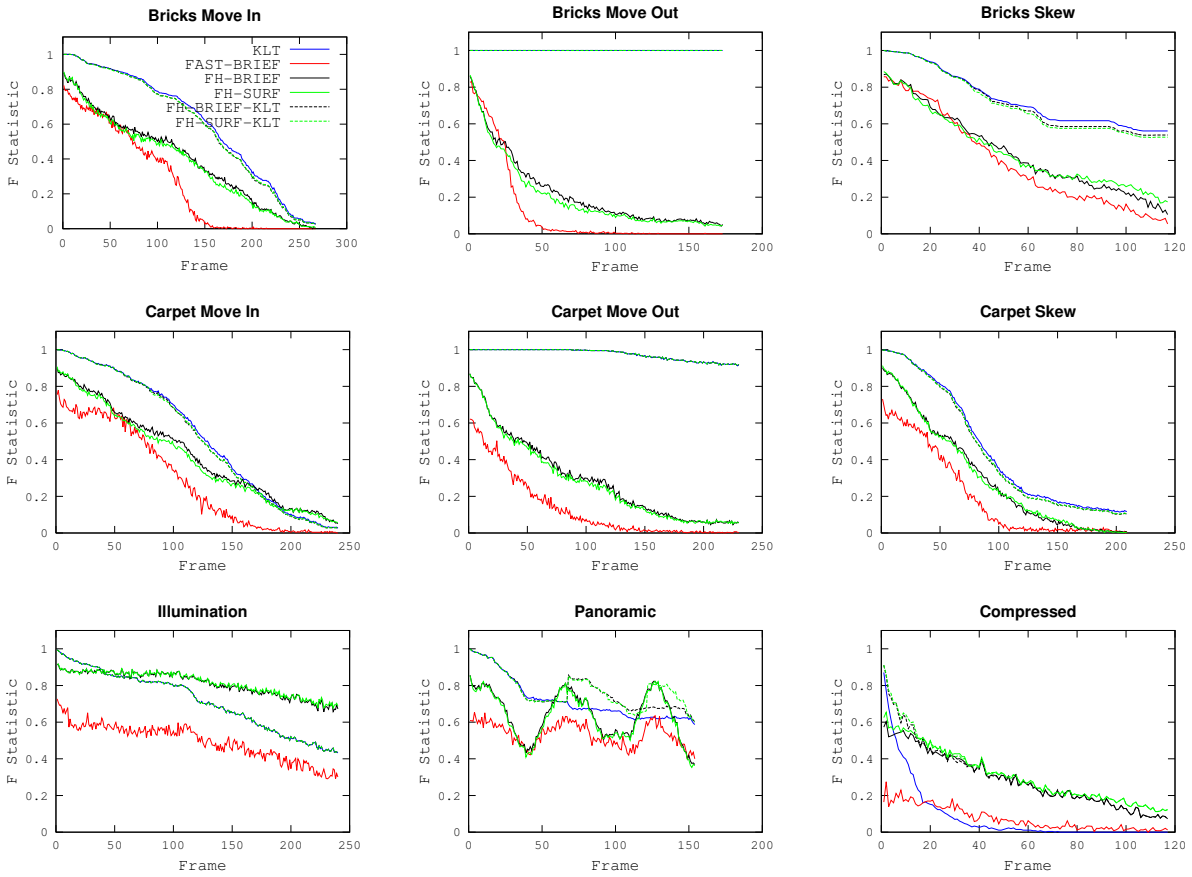

Figure 7: Track stability by video frame, higher is better. DDA style trackers experienced failures due to the change in scale and perspective exceeding their detection capability. KLT and the hybrid trackers are able to maintain good track quality for changes in scale due to the track description being continuously updated. KLT and hybrid trackers experienced noticeable track drift when a sequence changes too fast, but not fast enough to cause tracks to drop.