Scene-Consistent Detection of Feature Points in Video Sequences

Ariel Tankus and Yehezkel Yeshurun

School of Computer Science

Tel-Aviv University

Tel-Aviv 69978, Israel

{arielt,hezy}@post.tau.ac.il

Abstract

Detection of feature points in images is an important preprocessing stage for many algorithms in Computer Vision. We address the problem of detection of feature points in video sequences of 3D scenes, which could be mainly used for obtaining scene correspondence. The main feature we use is the zero crossing of the intensity gradient argument. We analytically show that this local feature corresponds to specific constraints on the local 3D geometry of the scene, thus ensuring that the detected points are based on real 3D features. We present a robust algorithm that tracks the detected points along a video sequence, and suggest some criteria for quantitative evaluation of such algorithms. These criteria serve in a comparison of the suggested operator with three other feature trackers. The suggested criteria are generic and could serve other researchers as well for performance evaluation of stable point detectors.

Index Terms: Feature point detection, scene-consistent detection, stable point tracking, tracking evaluation.

1 Introduction

Context-free detection of specific image points ("features") is being addressed in Computer Vision for a long time, as it is the basis of many higher level algorithms of visual information processing.

The large amount of work invested in detection of feature points yielded various approaches to the definition of its goal. Works in this area divide up into two classes: The first refers to the task as an "Attentional" task in the sense that the detected points should attract computational resources; this is apparently the case in biological systems. This definition of the task is sometimes referred to as detection of "Interest Points" or "Regions of Interest" (ROI). The second approach defines the task as consistent selection of a subset of image pixels, regardless of their "attentional" value. Different names for this approach are: "Anchor Points" or "Stable Point" detection.

The main goal of this paper is **robust detection of scene-consistent feature points in video sequence**s, and as such it takes the second approach. By robust we mean that the algorithm should consistently detect points in noisy images, and by scene-consistent (or: stable), that the algorithm should consistently detect the same 3D point over multiple video frames, regardless of illumination changes, pose variations, camera motion or parallax. This implies that the detection should depend merely on the local geometry of scene objects.

The literature of feature detection has a wide variety of detectors which base on edges. Edges in many cases are not intrinsic to one subject, but rather delineate the boundary between a subject and the background. Therefore, edge-based features in many cases depend on the background and viewing direction (e.g., silhouette of a person in upright position vs. profile). These inherent flaws of edge-based features raises a need for non-edge-based feature detectors.

This paper presents an operator for non-edge-based feature detection. The features are extrema of the intensity function, but only in smooth image domains. Pay attention, that an edge (e.g., a black line on a white paper) would *not* yield a strong response of the suggested operator, because the intensity function is discontinuous there (i.e., not a smooth domain). The uniqueness of the suggested approach lies in its ability to detect local extrema of smooth image domains in a reliable and robust way, while avoiding local

extrema created by edges. As the paper would present, this robustness has a solid theoretic basis; a large number of images would demonstrate that the robustness is attained in real-life scenes as well.

The contribution of this paper can be divided into two categories:

1. The theoretic point of view: The paper presents novel theorems characterizing the domains where the operator under consideration has maximal response. We map the image domains which the operator detects, according to their differential-geometric features. Having this mapping in hand, we characterize the 3D scene locations which yield these image domains, and thus yield maximal response of the operator. These theorems explain why the detection of the operator is robust, and why it consistently follows 3D points in the scene.

2. The practical viewpoint: Our paper focuses on the *detection* of features in video sequences. However, as is evident (and is discussed in depth below), when detected feature points are used for tracking, it is highly nontrivial to decouple the detection from the tracking. In order to evaluate the detected features and their fitness to real-life tasks, we examine the detection–tracking system as a whole. [1] takes a similar approach when they base the definition of the feature detector on the method they use for tracking. In order to compare our method with existing ones, we define measures of evaluation of the tracking results of a detection–tracking system. We have compared the detection–tracking system as a whole, when applicable. For detection methods that do not include tracking in the original work, we have used a common tracker. The advantage of this approach lies in its ability to evaluate how appropriate a selected group of features is for higher levels of processing. Using these measures, we show that our method favorably compares with three other methods (Kanade-Lucas-Tomasi [1], Junction Detection [2] and ImpHarris [3]).

The paper is structured as follows: Section 2 surveys the literature of feature detectors and trackers as well as that of evaluation criteria. Section 3 sketches the operator for detection of RoI in static images, that was suggested in [4]. Section 4 then shows analytically that this method detects only specific features of the image intensity function I(x,y), and proves that these image-space features correspond to the local 3D geometry of objects in the scene. These theorems are a mathematical characterization of the domains of strong

(i.e., infinite) response of the operator. This section establishes the theoretical basis explaining why the operator is very robust. Section 5 presents a simple algorithm, based on Kalman filter, that robustly tracks these features in video sequences. The usage of video sequences confronts the operator with new effects which cannot be dealt with when using static images: parallax, camera motion and 3D object transformations. The operator copes well with these effects, because it responds to intrinsic properties of 3D objects (as was proved in Sect. 4). In Sect. 6, we rigorously define two measures for evaluating detection–tracking systems: completeness (with respect to correct tracking of 3D points) and stability. These measures are generic and could be of use for other researchers as well. The measures serve in a comparison between the suggested tracker and three other trackers (Sect. 7). Following concluding remarks in Section 9, Appendices A and B supply the complete proofs of the theorems of Sect. 4.

2 Literature Survey

This section briefly surveys the recent work in the fields of feature point detection, point tracking and evaluation of feature points. We do not present a complete survey of these fields mainly because such surveys were recently published: [3] studies in detail both state-of-the-art interest point detectors and recent works on evaluation of feature detectors. [5] surveys works with attentional approach to interest point detection.

2.1 Feature Detection

Typical examples for the "Attentional" approach to feature detection can be found in [6], [7], [5]. Methods which do not attempt to generally focus attention, usually define their task as locating points in the image in a manner consistent with the 3D scene. The points could be either used for object recognition [8], or as correspondence points for recovering 3D characteristics of the scene.

[2] suggests automatic scale selection which associates a Region of Interest with each detected junction; level curve curvature is employed there for junction detection. [9] selects stable points at maxima and minima of a Difference of Gaussian function applied in scale space; the resulting features are invariant to image scaling, translation and rotation and robust to illumination changes and affine or 3D projection.

Corners are considered Anchor Points: [10] defines a "cornerness" measure in conjunction with edge detection to obtain accurate corner localization. [11] proposes a differential procedure for subvoxel localization of 3D point landmarks and the size of a RoI around them. [12] suggests corner detection in textured color images, and takes advantage of color signatures and Earth Mover's Distance to detect corners in situations where the adjacent color regions are not constant. [13] suggests a variant of the morphological closing operator for corner detection; it is called asymmetrical closing, and is based on brightness comparisons.

Of particular interest for our discussion is the Harris corner detector (also known as the Plessey feature point detector) [14], as some of the comparisons to be shown later refer to the improved version of this operator. The Harris operator is based on the local auto-correlation function computed using the first order image derivatives. The variation of the auto-correlation over different orientations is found by calculating functions related to the principle curvatures of the local auto-correlation. [15] used the Harris operator, and employed Kalman filter for tracking the features. [16] uses the Harris corner detector with an extended Kalman filter tracker to compute 3D structure. [3] uses an improved version of the Harris operator (marked: ImpHarris) in an extensive comparison of feature detectors. The improvement is the replacement of the method of differentiation from mask-based to Gaussian-based.

2.2 Feature Tracking

The literature of feature tracking is very rich, so only a few of them can be described in this paper.

[17] presents a point correspondence algorithm based on automatic egomotion compensation. They use methods of subpixel accuracy for feature matching and and tracking.

[18] presents a view-based image representation, the qualitative multi-scale feature hierarchy. This representation is used to improve performance of feature trackers by defining search regions in which lost features can be detected again.

[19] presents an original approach to motion estimation in color image sequences based on color region matching. The method takes into account the variations in segmentation by the extension of the matching model to multi matches. Color regions that cannot be matched at the feature level are matched on the pixel level based on correlation.

[20] integrates the information from the motion of 2D features with 1D boundaries in order to track independently moving objects. The 2D features are found using either the SUSAN corner detector [21] or the Harris corner detector [14]. Tracking is performed by a Kalman filter with motion model of constant velocity or constant acceleration, depending on the application.

[22] tracks vehicles, and takes a special care to allow occluded vehicles. For this they use the [23] corner detector (windowed 2nd moment matrix). When a corner is detected, a template of the graylevel image around this corner is extracted and used for correlation in the tracking module. The tracking module contains a Kalman filter. The filter prediction and the measurement are correlated. Association of a measure to a track is rejected if the correlation is too low.

[24] presents a feature tracker for long image sequences based on simultaneously estimating the motions and deformations of a collection of adjacent image patches. The patches achieve greater stability by sharing common corners. Their method models full bilinear deformations.

The KLT (Kanade-Lucas-Tomasi) tracking algorithm [25] is based on a model of affine image change. Features are selected to maximize tracking quality. A feature is present if the eigenvalues of the auto-correlation matrix are significant. Monitoring tracking quality is based on a measure of dissimilarity that uses the affine motion as the underlying image change model.

The original KLT was later improved by [26]. They employ an outlier rejection rule and proves that its theoretical assumptions are satisfied in the feature tracking scenario.

2.3 Evaluation of Feature Detectors and Trackers

[17] tracks the video sequence, and takes the points tracked at the last frame as initial points for back-tracking the video sequence from the last frame to the first. Ideally, they expect the back-tracking to result in the original points from which the regular order tracking began. The distance between original and back-tracked points serves for the evaluation (zero being the ideal). The problem with this method is that it does not take into account the whole tracking process: if the central part of a track skips between several 3D scene points, but its beginning and end follow the same 3D point, the track may appear perfect according to this measure. Another flaw is that for long sequences the

set of points being tracked at the last part of the sequence might be totally different from that at the beginning, even from objective reasons such as disappearance of the initial scene points in the video sequence.

[27] tracks human body motion. They obtain the ground-truth by markers worn by subjects. Obtaining the ground truth this way can be done only for lab video sequences, or when one deals with a limited aspect of the tracking problem as is the case here (they refer merely to human motion). Tracking points in generic video sequences cannot assume the existence of markers for accurate ground truth analysis. A certain level of noise of the tracked points should be allowed, as determination of the ground truth can by itself add noise to the analysis.

[28] evaluates feature tracking by three merits. The measures are the number of perfect tracks (i.e., when the point is correctly tracked along the whole track) divided by the total number of points under consideration, and a relaxed version of this which allows local deviations from the ideal trajectory if the last point is connected to the correct initial point. The drawback of these merits is that they do not quantify the *accuracy* of a trajectory: At the relaxed version, it might be the case that 99% of the trajectory is incorrect, but still it is counted as a correct one, as its first and last points are consistent. At the strict merit, it might be the case that only 0.1% of the track is incorrect, but still the whole track is considered incorrect. Another merit suggested in that paper is the link-based criterion: the total number of correct links divided by the total number of links. (A correct link is a vector that connects the same two points of two consecutive frames as the ideal trajectory). This measure does not quantify the correctness of the links on a per frame or per trajectory basis, but only total over all trajectories and all frames. This again makes it difficult to evaluate how stable is the tracking process, and how useful can tracks be for higher vision tasks.

[29] evaluates corner trackers with different motion models. The evaluation measure they use is the Probability of Correct Association (PCA): the probability at any given step that the tracking system will make a correct data association in the presence of clutter. They use trackers based on motion models and derive expressions for the PCA of the trackers under consideration. In addition, their evaluation assumes that any selected feature is present in every frame (they verify this manually). Such an assumption limits the use of the evaluation method in other sequences. The evaluation measures we are

about to present do not assume any motion model.

[30] suggests the self-consistency of the outputs of the algorithm as a means for estimating the accuracy and reliability of point correspondences algorithms. The method allows comparing different algorithms, comparing different scoring functions, comparing window sizes, and detecting change over time. The basic observation in that paper is that two hypotheses are consistent if they both refer to the same object in the world and the difference in their estimated attributes is small relative to their accuracies, or if they do not refer to the same object. They assume that the projection matrices and associated covariances are known of all images.

[3] introduces two evaluation criteria for interest points: repeatability rate and information content. Repeatability rate evaluates the geometric stability under different transformations. The repeatability rate is the percentage of the total observed points that are detected in both images. Information content measures the distinctiveness of features. Distinctiveness is based on the likelihood of a local grayvalue descriptor computed at the point within the population of all observed interest point descriptors. The entropy of these descriptors measures the information content of a set of interest points. Based on these criteria, six interest point detectors are compared, finding the improved version of the Harris operator to be superior to the others. The descriptors for evaluation of the information content criterion are differential rotation invariants up to second order.

3 Operator for Feature Detection

In order to accomplish scene-consistent detection of feature points in video sequences, we first present an operator that has been suggested [4] for detecting points in static images.

3.1 Intuition

The suggested operator detects local extrema of the intensity function, but only in domains where the intensity function is smooth. Intuitively, one may think of the detected domains as hilltops (or equivalently, bottoms of valleys) of the intensity function.

A property of local extrema of a smooth function is that the gradient of the function on local, closed curves containing the extremum, points outward *along the whole closed*

curve. However, the operator does *not* look for these closed curves *explicitly*, but rather, it takes advantage of the discontinuity of the 2D arctan function for fast and robust detection of such domains, as the next subsection would show.

3.2 Definition

The gradient map of an image in Cartesian coordinates is estimated by:

$$\begin{aligned} \nabla \mathrm{I}(x,y) &= (\frac{\partial}{\partial x}\mathrm{I}(x,y),\ \frac{\partial}{\partial y}\mathrm{I}(x,y)) \\ &\approx ([D_\sigma(x)G_\sigma(y)] * \mathrm{I}(x,y),\ [G_\sigma(x)D_\sigma(y)] * \mathrm{I}(x,y)) \end{aligned}$$

where $G_\sigma(t)$ is the 1D Gaussian with mean 0, and standard deviation σ, and $D_\sigma(t) = \frac{d}{dt}G_\sigma(t)$.

We turn to polar coordinates and compute the gradient argument:

$$\theta(x,y) = \arg(\nabla \mathrm{I}(x,y)) = \arctan(\frac{\partial}{\partial y}\mathrm{I}(x,y),\ \frac{\partial}{\partial x}\mathrm{I}(x,y))$$

where the 2D arctan function is defined by:

$$\arctan(y,x) = \begin{cases} \arctan(\frac{y}{x}) & \text{if } x > 0 \\ \arctan(\frac{y}{x}) + \pi & \text{if } x < 0,\ y \geq 0 \\ \arctan(\frac{y}{x}) - \pi & \text{if } x < 0,\ y < 0 \\ 0 & \text{if } x = 0,\ y = 0 \\ \frac{\pi}{2} & \text{if } x = 0,\ y > 0 \\ -\frac{\pi}{2} & \text{if } x = 0,\ y < 0 \end{cases} \tag{1}$$

and notice the well known discontinuity at the negative part of the x-axis (Fig. 1(Left)), which is the basis for our method.

3.3 Detecting the Presence of a Certain Range of Directions of the Intensity Normal

The discontinuity of the 2D arctan occurs at the negative x-axis, so it corresponds to angles of π or $-\pi$ radians. Because $\theta(x,y)$ is the azimuth of the normal to the intensity function, the presence of such a discontinuity implies that the azimuth of the normal there is π or $-\pi$ radians. Therefore, the presence of the discontinuity implies that part of our goal, intensity normal which points outward along the whole closed curve, has been

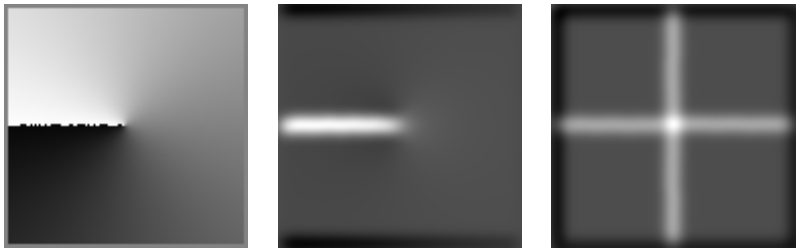


Figure 1: ***Left:*** *The 2D arctan. Pay attention to the discontinuity at the negative part of the x-axis, which is the basis of the method.* ***Middle:*** *The y-derivative of the 2D arctan.* ***Right:*** *Rotate the 2D arctan by 0°, 90°, 180°, and 270°, differentiate along the y-direction, rotate back, and sum the responses. (An isotropic operator.) Note that the center has strongest response, as it is a point surrounded by local closed curves where the intensity normal points outward in* all *directions.*

accomplished: we know of the presence of a certain range of directions of the intensity normal (near π or $-\pi$ radians).

But how can we detect the discontinuity of the 2D arctan? In order to do so, we define an operator:

$$\mathbf{Y}_{arg} \stackrel{def}{=} \frac{\partial}{\partial y}\theta(x,y) \approx [G_\sigma(x)D_\sigma(y)] * \theta(x,y) \tag{2}$$

When one differentiates $\theta(x,y)$ with respect to y, at the discontinuity ray the derivative approaches infinity ($\frac{\partial}{\partial y}\theta(x,y) \to \infty$). In practice, this appears as a very strong response at the discontinuity ray, as can be clearly seen in Fig. 1(Middle). Obviously, it is easy to isolate such a response, e.g. by thresholding. In other words, because of the strong (theoretically infinite) response of the derivative of $\theta(x,y)$, we can isolate the discontinuity of $\theta(x,y)$, which implies the presence of a certain range of values (near π or $-\pi$) of the angle of the intensity normal.

3.4 Detecting Locations Surrounded by Intensity Normals in All Orientations

In order to attain our objective and detect the presence of an outward normal along the whole closed curve, we have to define an operator whose strongest response occurs when the closed curve contains an outward normal in every possible orientation (an isotropic operator). Ideally, in order to define this operator based on Y_{arg}, one has to rotate the original image in all possible angles α, operate Y_{arg}, rotate the results back to their original

pose, and integrate the responses over all α.

In practice, we define an isotropic version of the operator, called: D_{arg}, to be the result of rotating the original image by 0, 90, 180 and 270 degrees, operating Y_{arg} on the rotated images, rotating back to the original pose, and summing the four responses (Fig. 1(Right)).

This infinite response of Y_{arg} is also the reason why only four angles are enough for the isotropic operator D_{arg}: Differentiating $\theta(x, y)$ in any direction which is *not* exactly parallel to the x-axis yields an infinite response at the negative x-axis due to the discontinuity of the 2D arctan. As a result, even differentiation in only two perpendicular directions yields an infinite response at the discontinuity ray for *any* orientation. Experiments demonstrating the robustness of D_{arg} to orientation changes were reported in [4].

All in all, the operator reveals extrema of smooth intensity patches by first isolating a certain range of values of the gradient argument $\theta(x, y)$, and then applying the method to the rotated images. The way to detect the specific range of angles of the intensity normal is via the discontinuity ray of the 2D arctan, whose presence indicate angles near π or $-\pi$ radians of the gradient argument. To detect the discontinuity ray, we differentiate $\theta(x, y)$ with respect to y, and have infinite responses at locations where the discontinuity ray is crossed. An example of the domains where strong D^2_{arg} responses occur appears in Fig. 2.

Since we are looking for a qualitative shape description, the Y_{arg} operator is very robust, in contrast with classic methods of shape-from-shading (e.g. [31]; see a survey at [32]). Many intensive tests of the operator robustness were shown in [4]: robustness in illumination strength changes, and in variations of orientation or scale. The robustness of the operator in dominating textures has lead to its usage for camouflage breaking [33], thus relaxing the original demand of constant albedo of the subject.

4 Response of Y_{arg} to the Intensity Surface and Scene Geometry

This section presents the mathematical basis of our claim that the response of Y_{arg} is stable. The complete proofs of the theorems are included in Appendices A and B. By definition, the theorems hold for D_{arg}, too.

a. Original image.

b. D^2_{arg}.

c. 70% threshold of D^2_{arg}.

d. Thresholded D^2_{arg} marked on original.

Figure 2: *An image with domains of maximal D^2_{arg} marked. Domains rich in edges or domains of flat objects have low response of D^2_{arg} and are not detected. All four edge-sparse 3D objects are detected.*

4.1 Response to the Intensity Surface

First, we qualitatively characterize the behavior of Y_{arg} in continuous ("well-behaving") domains. We analyze the case where the original intensity function $I(x, y)$ is twice continuously differentiable. Let (x_0, y_0) denote an image point where $\frac{\partial}{\partial y}\theta(x, y)$ approaches infinity. In other words, (x_0, y_0) is a point of high Y_{arg} response (recall the strong response ray in Fig. 1 (Middle)). Our basic observation is that at such a point (x_0, y_0), $\theta(x, y)$ has a jump-discontinuity (with respect to the y-direction):

1. Because $I(x, y)$, $\frac{\partial I(x,y)}{\partial x}$ and $\frac{\partial I(x,y)}{\partial y}$ are differentiable and continuous, and for all points (x_0, y_0) the left- and right-hand limits: $\lim_{y \to y_{0\pm}} \arctan(y, x_0)$ exist, it follows that

$\theta(x, y)$ has left- and right-hand limits in the y-direction, anywhere.

2. If at point (x_0, y_0) the left- and right-hand side limits are equal, $\theta(x_0, y)$ is continuous or has a removable singularity.

   (a) If $\theta(x, y)$ is continuous: Because $\frac{\partial I(x,y)}{\partial x}$ and $\frac{\partial I(x,y)}{\partial y}$ are assumed differentiable anywhere, point (x_0, y_0) must be a point where $\arctan(y, x_0)$ is continuous. Since $\arctan(y, x_0)$ is differentiable at all points where it is continuous, it follows that $\theta(x, y)$ is also differentiable.

   (b) If $\theta(x, y)$ has a removable singularity: The estimation of Y_{arg} is achieved using a convolution (Eq. 2). By definition, the convolution is an integral. The integral of a function with a removable singularity is identical to that of the fixed function (i.e., if one sets the value of the function at the singular point to the value of the left- and right-hand side limits of the function at that point). Therefore, this case is similar to that of a continuous $\theta(x, y)$, and $\frac{\partial}{\partial y}\theta(x, y)$ does not approaches infinity in this case, too.

3. If the left- and right-hand limits are different, the derivative would approach infinity. This is the jump-discontinuity case.

We are interested in domains where Y_{arg} approaches infinity; they are the stable points of the response. Formally,

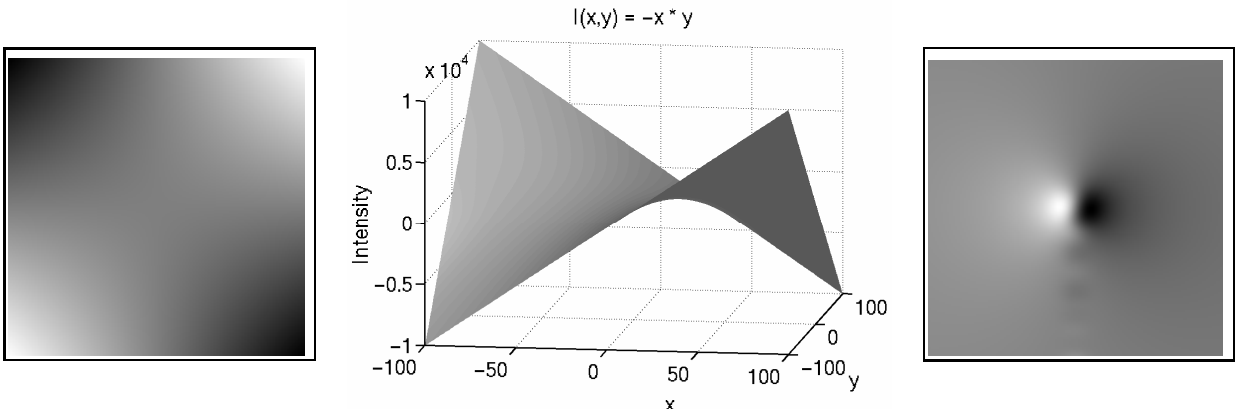
Theorem 1 *Let $I : R \times R \longmapsto R \in C^2$ (i.e., $I(x, y)$ is twice continuously differentiable w.r.t both x and y) be an intensity function. If (x_0, y_0) is a point where: $\lim_{y \to y_0} \frac{\partial}{\partial y}\theta(x, y)|_{x=x_0} = \pm\infty$ (i.e., a point where $Y_{arg} \to \pm\infty$), then there exists $\varepsilon > 0$ so that for all y, for which $| y - y_0 | < \varepsilon$, one of the following cases holds:*

1. $\frac{\partial I(x,y)}{\partial y}|_{x=x_0} = 0$ *for all* y *and*
   $\forall y < y_0$, $\frac{\partial I(x,y)}{\partial x}|_{x=x_0} \geq 0$, *and* $\forall y > y_0$, $\frac{\partial I(x,y)}{\partial x}|_{x=x_0} < 0$. ***

2. $\frac{\partial I(x,y)}{\partial y}|_{x=x_0} > 0$ *for* $y < y_0$ *and* $\frac{\partial I(x,y)}{\partial y}|_{x=x_0} = 0$ *for* $y > y_0$, *and **

   (a) $\forall y > y_0$, $\frac{\partial I(x,y)}{\partial x}|_{x=x_0} = 0$. *or:*

   (b) $\forall y < y_0$, $\frac{\partial I(x,y)}{\partial x}|_{x=x_0} = 0$. *or:*



Feature as image. Feature as intensity surface. Y_{arg}.

Figure 3: *Demonstration of Case 1 of Theorem 1:* $I(x,y) = -x \cdot y$. *The feature is at* $(0,0)$, *and indeed* Y_{arg} *has a very strong response there.*

(c) $\forall y < y_0$, $\frac{\partial I(x,y)}{\partial x}|_{x=x_0} > 0$, *and* $\forall y > y_0$, $\frac{\partial I(x,y)}{\partial x}|_{x=x_0} < 0$. ***

3. $\frac{\partial I(x,y)}{\partial y}|_{x=x_0} < 0$ *for* $y < y_0$ *and* $\frac{\partial I(x,y)}{\partial y}|_{x=x_0} = 0$ *for* $y > y_0$, ***
   except for the case where $\forall y : y \neq y_0$, $\frac{\partial I(x,y)}{\partial x}|_{x=x_0} > 0$.

4. (x_0, y_0) *is a local extremum of* $I(x_0, y)$,
   except for the case where $\forall y : y \neq y_0$, $\frac{\partial I(x,y)}{\partial x}|_{x=x_0} > 0$.

** The case where the conditions for* $y < y_0$ *are swapped with those for* $y > y_0$ *is also valid; it is an equivalent case and was therefore omitted.*

The complete proof of Theorem 1 appears in Appendix A. The rest of this subsection illustrates the response of Y_{arg} to the different features described by Theorem 1.

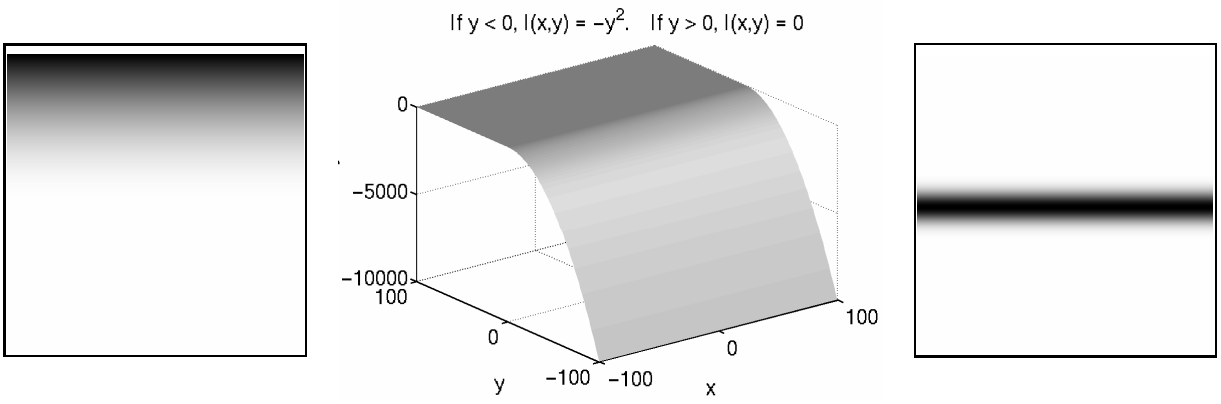
Figure 3 uses the intensity function: $I(x,y) = -x \cdot y$ as an example of case 1. It follows that: $\frac{\partial I(x,y)}{\partial x} = -y$, $\frac{\partial I(x,y)}{\partial y} = -x$. The feature point is $(x_0, y_0) = (0,0)$. To see that indeed this intensity surface belongs to case 1: $\frac{\partial I(x,y)}{\partial y}|_{x=x_0} = x_0 = 0$. $\forall y < y_0 = 0$, $\frac{\partial I(x,y)}{\partial x} = -y > 0$, and $\forall y > y_0 = 0$, $\frac{\partial I(x,y)}{\partial x} = -y < 0$. We can see the strong response of Y_{arg} at $(0,0)$ in this case (Fig. 3 (right)).

Figure 4 exhibits case 2 using the intensity function:

$$I(x,y) = \begin{cases} -y^2, & \text{if } y < 0 \\ 0, & \text{if } y \geq 0 \end{cases}$$

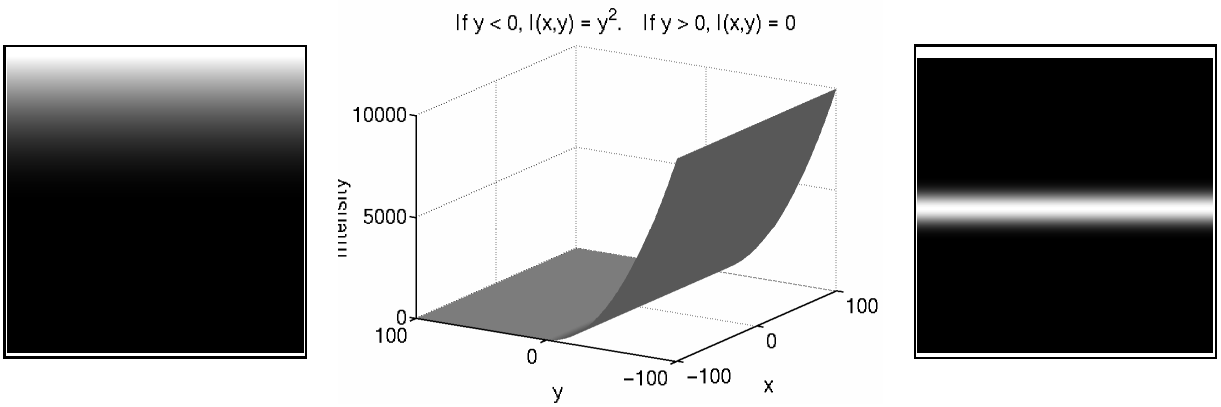
whose derivatives are:

$$\frac{\partial}{\partial y} I(x,y) = \begin{cases} -2y, & \text{if } y < 0 \\ 0, & \text{if } y \geq 0 \end{cases}$$



Feature as image. Feature as intensity surface. Y_{arg}.

Figure 4: *Demonstration of Case 2 of Theorem 1: for* $y > 0$ $I(x, y) = -y^2$, *for* $y \leq 0$, $I(x, y) = 0$. *The features are at* $y = 0$. Y_{arg} *has a very strong negative response there - the black stripe.*



Feature as image. Feature as intensity surface. Y_{arg}.

Figure 5: *Demonstration of Case 3 of Theorem 1: for* $y > 0$ $I(x, y) = y^2$, *for* $y \leq 0$, $I(x, y) = 0$. *The features are at* $y = 0$. Y_{arg} *has a very strong positive response there.*

and $\frac{\partial}{\partial x} I(x, y) = 0$ for all x. This is an example of cases (2)(a) or (2)(b) (it is appropriate for both) where the feature points are $y = 0$. The strong negative response (black strip) can be seen in Fig. 4 (right).

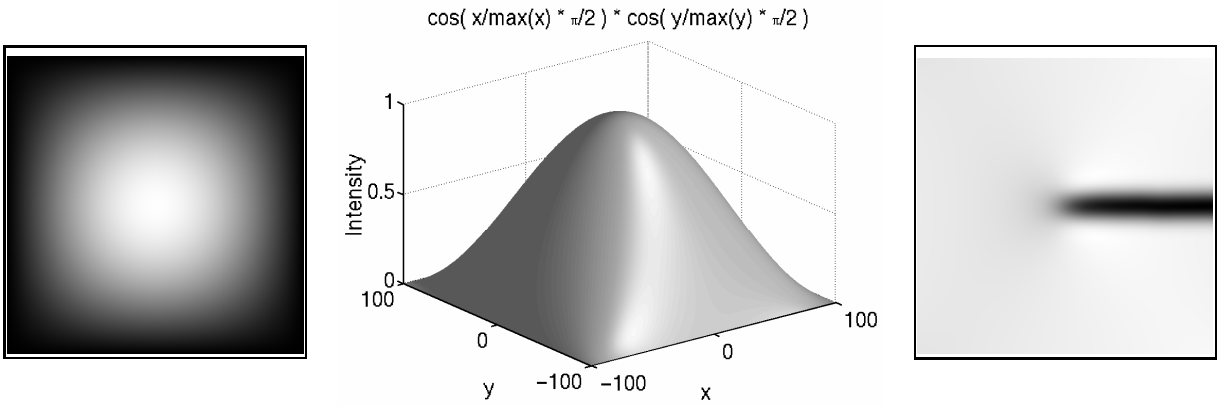
Similarly, in Fig. 5 the intensity function:

$$I(x, y) = \begin{cases} y^2, & \text{if } y < 0 \\ 0, & \text{if } y \geq 0 \end{cases}$$

demonstrates case (3). The response of Y_{arg} in this case is positive.

Figure 6 demonstrates case 4 with a local maximum of the intensity function in the y-direction. The intensity function in the example is:

$$I(x, y) = \cos(\frac{\pi x}{2 \max(x)}) \cos(\frac{\pi y}{2 \max(y)})$$



Feature as image. Feature as intensity surface. Y_{arg}.

Figure 6: *Demonstration of Case 4 of Theorem 1:* $I(x,y) = \cos(\frac{\pi x}{2\max(x)})\cos(\frac{\pi y}{2\max(y)})$. *The features are at the positive part of the* x*-axis, and indeed* Y_{arg} *responds to them. The negative part of the* x*-axis is the case excluded from case (4).*

whose maximum w.r.t y is at the x-axis. A strong negative response of Y_{arg} (Y_{arg} approaches $-\infty$) at the positive part of the x-axis. There is no strong response at the negative part of the x-axis, because there $I(x,y)$ is monotonically increasing as a function of x, and this is exactly the case excluded from case (4) ($\frac{\partial I(x,y)}{\partial x}|_{x=x_0} > 0$). As this example shows, when an intensity function has a local extremum (in the strong sense), the typical response of Y_{arg} would be to one side of the x-axis, either the negative or the positive, and to the center (i.e., the local extremum itself). By using the isotropic operator D_{arg}, one receives a strong response in all axes, but the strongest response is at the extremum itself (as any orientation of the image contributes to the center, but not to all other parts of the axes). This enables the isolation of the point of extremum of the intensity function.

4.2 Response to Local 3D Scene Structure

So far, the analysis referred merely to the connection between Y_{arg} and the intensity function. This section would establish a connection between some of the domains where Y_{arg} approaches infinity and the three dimensional scene object.

4.2.1 Basic Assumptions

Let us assume an ideal Lambertian surface with constant albedo is illuminated by a point light source at infinity and photographed by a projective camera. We assume a camera model where the radiance of the surface (L) in the direction of the camera is proportional

to the irradiance of the image (E) at the corresponding patch (i.e., $E \propto L$). Such a system is described at [34] (a lens system where the off-axis decay is compensated).

4.2.2 The Detected Domains — Case (4)

In Theorem 1 we saw that one of the cases (case 4) referred to detection of local extrema of the intensity surface with respect to the y-direction, and that D_{arg} may be used to isolate extrema of the intensity function in twice continuously differentiable domains. This subsection characterizes the geometry of the scene for which these extrema of the intensity function occur.

Assuming a Lambertian surface and an orthographic projection, the intensity function would be:

$$I_{orth}(x,y) = \rho \overrightarrow{N}(x,y) \cdot \overrightarrow{L} = \rho \cos(\alpha(x,y)) \tag{3}$$

where ρ is the constant albedo; $\overrightarrow{N}(x,y)$, the normal to the 3D surface $z(x,y)$; $\overrightarrow{L}$, the light source direction (this direction is constant); and $\alpha(x,y) = \angle(\overrightarrow{N}(x,y), \overrightarrow{L})$, the angle between the light source direction and the normal to the scene surface. Obviously, $\alpha(x,y) \in [0, \frac{\pi}{2}]$. In this domain, the cosine decreases monotonically. This means that if $\alpha(x,y)$ has a maximum (minimum) at point (x_0, y_0), then $\cos(\alpha(x,y))$ has a minimum (maximum) there, and vice versa. It follows that if $I_{orth}(x,y)$ attains a local extrema at (x_0, y_0), then the angle $\alpha(x,y)$ attains a local extrema there (of the opposite type).

To summarize this observation, local extrema of the intensity function I_{orth} are local extrema (of the opposite type) of the angle $\alpha(x,y)$ between the light source direction $\overrightarrow{L}$ and the normal $\overrightarrow{N}(x,y)$ to the 3D surface $z(x,y)$. Appendix B proves that this remains valid under a perspective projection, too.

The meaning of the result is that the extrema of the intensity function which Y_{arg} detects, correspond to extrema (of the opposite type) of the angle between the light source direction and the 3D surface. That is, as long as the light source direction remains constant, the detection of Y_{arg} (case 4) depends merely on the geometry of the scene: the angle between the normal of the 3D surface and a constant direction.

This last result explains (to a certain extent) why one may reduce the demand of constant albedo to a demand of *locally* constant albedo: If angle $\alpha(x,y)$ attains an extremum at a certain location (x_0, y_0) on the 3D surface $z(x,y)$, then a small neighborhood around (x_0, y_0) having a constant albedo is enough to make Y_{arg} approach infinity. In addition,

the operator does not detect edges, so the changes of albedo do not add more feature points.

Level sets provide another point of view showing a relation between the detected extrema points and scene geometry. Several level set methods for Shape from Shading [34], [35], [36] use the intensity extrema (assuming a smooth image) as singular points from which the SfS process begins. For the SfS problem with a vertical light source ($\overrightarrow{L} = (0, 0, 1)$), the local extrema of a smooth domain are local extrema of $z(x, y)$ (the depth of the 3D object). For the oblique case, [37] shows how to change the image coordinate system to the light source coordinate system (assuming an orthographic projection). They reduce the oblique light source problem to the vertical light source case, up to a change of the irradiance equation PDE, and show how to solve the modified PDE. This relates the detected domains of D_{arg} with the geometry of the 3D object.

To summarize case (4): In twice continuously differentiable domains, the operator detects extrema of the intensity function. These extrema relate to extrema of the angle $\alpha(x, y)$. Assuming a constant light source direction (as is common in SfS, for example), the detected points relate to scene geometry. What distinguishes Y_{arg} or D_{arg} from other methods of isolating extrema of the intensity function is its high robustness, and the consistency with which it detects its features.

4.2.3 The Detected Domains — Cases (1)–(3)

To characterize cases (1)–(3), we first define the term: *semi-weak change of sign of a derivative at point* (x_0, y_0). This term denotes that a derivative may vanish at one side of (x_0, y_0), but not at both sides. The common property of cases (1)–(3) in Theorem 1 is a semi-weak change of sign of a first order derivative (either $\frac{\partial}{\partial x} I(x, y)$ or $\frac{\partial}{\partial y} I(x, y)$). Therefore, cases (1)–(3) represent semi-weak local extrema (i.e., local extrema where the derivative undergoes a semi-weak change of sign) at either the x or the y directions.

A constant albedo Lambertian plane illuminated by a point light source at infinity produces a constant intensity function (i.e., image derivatives equal zero), because its normal is fixed (Eq. 3). If such a 3D planar surface changes smoothly into a convex (or concave) 3D surface, then the points where the change begins project into image points where a semi-weak change of sign of the derivatives occurs (as cases (1)–(3) require). As we see, the characterization of these points relies on the local geometry of the 3D scene

surface.

The complete mathematical analysis of the behavior of algorithms including extreme cases is highly important. However, when one performs such an analysis, one must take into account the distribution of inputs of the algorithm, that is, the distribution of images one is likely to see in a natural environment (i.e., non-laboratory images). Even the human vision system may fail on rare images. Therefore, as outliers may exist for cases (1)–(3), the frequency of images of such outliers should be examined. Outliers may exist if the images of a planar object and a convex object are adjacent, and exhibit the properties described in Theorem 1. However, due to the requirement of smooth transition between the intensity functions of the plane and the convex object, the probability of such outliers is relatively low. Higher level processing (e.g., tracking) can be used to further decrease the level of noise.

4.2.4 The Stability of the Detected Intensity Extrema

Isolation of extrema of the intensity function is certainly not new: [38], for example, mentions them as critical points (there, for 3D data). Local extrema divide to stable and unstable points according to the determinant of the Hessian matrix: critical points whose Hessian has a nonvanishing determinant are generic (see [38] for further details). When examining the cases detected by Y_{arg}, one may find out that some cases (e.g., case(1)) detects merely stable image locations, but some cases might include unstable extrema. The reason is that due to the differentiation of $\theta(x, y)$ at the y-direction, the characterization of Y_{arg} demands a change of sign (maybe semi-weak) of $\frac{\partial I}{\partial x}$ or $\frac{\partial I}{\partial y}$ only at the y-direction for a point to be detected. However, when examining D_{arg}, due to the rotation of the image, this requirement becomes a change of sign (again, maybe semi-weak) in any directional derivative at the detected points, which makes the points detected by D_{arg} stable.

Furthermore, the relation of intensity extrema to 3D as described above explains why these points are indeed stable: If a small perturbation of the intensity function changes its type, then the small perturbation changes also the type of the function $\alpha(x, y)$, which represents a change in the real world. Two kinds of perturbations may occur: to lighting direction and to the normal of the intensity function. If the perturbation comes from the lighting direction, this means one has arrived a critical lighting direction (cf. [39], there

discussing stability of viewpoints; similar analysis may be applied to lighting directions). In such a case, perturbations of the intensity are inevitable. If, on the other hand, the origin of the perturbation is a change of the normal to scene object, three cases should be considered: First, the object may rotate. In this case, again, a large perturbation implies that the object reached a critical angle with respect to the lighting direction, so the perturbation in intensity is once again a must. Second, the object may deform; clearly a change to the surface may cause intensity change, and maybe also change the set of detected points. Third, the object may scale. This is just another instance of object deformation. When considering camera zooming, due to the Lambertian model, the intensity function remains unchanged at points which previously occurred at the image, up to a change of the constant ρ (which affects the whole image, so an extrema would remain an extrema). However, zooming might reveal new details in the image (due to change in sample size), in which case again, there is a change in the details of the world which one perceives. A human vision system may also change its detection in such cases.

Therefore, the stability of the intensity extrema directly relates to the stability of the 3D scene object. Instability of the intensity extrema implies an inherent instability of scene: either a critical lighting direction or an object deformation.

This section shows that Y_{arg} responds to certain properties of the normal of a Lambertian surface illuminated by a point light source at infinity. This establishes a connection between Y_{arg} and *geometric* features of the 3D object, leading to stability of the detected points. The discussion is incomplete without referring to specular reflection: Specular reflection indeed distracts Y_{arg}, being [to a certain extent] a virtual image of the light source.

5 The Algorithm

As stated in the introduction, the focus of our method is feature *detection*, but in order to evaluate the fitness of the detected points to higher level tasks we evaluate the detection-tracking system as a whole. In this section we present the detection–tracking system. As we try and evaluate the detection part of the system, we use a simple tracker. Intuitively, if a simple tracker is enough to track the points produced by a certain detector, then these points must be more stable then if they were the input to a more sophisticated tracker

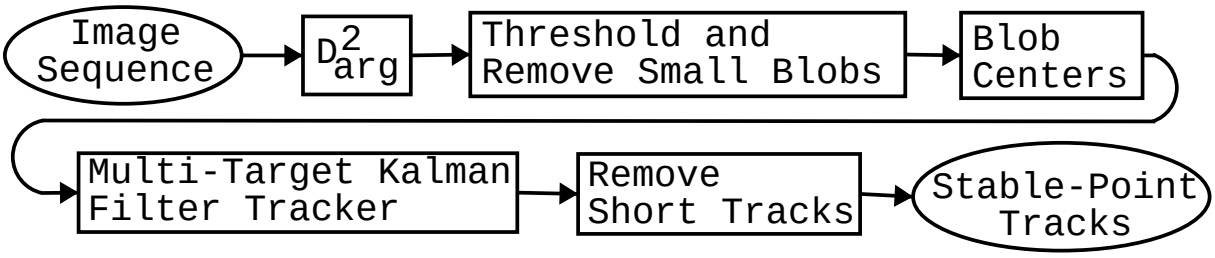


Figure 7: *Block diagram of the stable points detection and tracking algorithm.* ***Upper row blocks****: The stable-points detector.* ***Lower row blocks****: The tracking facility. The input to the tracking facility consists of point locations only (blob centers). The tracking facility has no other knowledge of the image.*

which could compensate for the instability of the detector.

The **stable-point detector** is based on the D_{arg} operator. As explained earlier, the points tracked by D_{arg} relate to geometric properties of the 3D scene object, and consistently follow the 3D object. As the input is discrete and bounded, the algorithm actually looks for the maximum of the dynamic range of D^2_{arg} (by thresholding). The stable points are the centers of gravity of the blobs produced by thresholding D^2_{arg}. These stable points are the only input to the point tracker: the point tracker has no knowledge about the mechanism producing its input points, and it obtains no other knowledge of the image.

The **point tracker** is a classic multi-target 2D Kalman filter tracker. The use of a Kalman filter to track points and assess their utility is not new and has been done, for example, in [15], [16], [40]; we employ it for the evaluation of the complete detection–tracking system. To estimate target position, the tracker assumes a simple motion model of constant velocity. Of course, this model is inaccurate, as camera motion is not limited to constant velocity. Section 5.1 introduces examples where the camera motion is complex: the camera rotates, accelerates, changes motion direction, vibrates (hand-held camera) and zooms. We compensate for the lack of a-priori knowledge of the real motion model, by setting the position components of the state vector of Kalman filter to the measurement itself (i.e., the point produced by the stable-point detector), each time a point is associated with a track. The velocity components remain unchanged. This reinforces the claim that stability is due to the stable-point detector, rather than the filtering process. We remove tracks which are too short, for further stabilization of the tracking.

5.1 Demonstration by Video Sequences

Let us demonstrate the stability of the algorithm using three of the video sequences we used to test the algorithm. In all sequences, constant parameters were maintained (i.e., no manual fit of parameters to the sequences), except for two cases: the maximal idle time (i.e., time when only Kalman-based approximation is maintained, without any new measurement associated to that track) and minimal track length. These parameters depend on the variability and length of the video sequence, respectively.

In the following examples of video sequences (Figs. 8, 9, 10), only the interior of the marked black frame participates in the tracking (to avoid boundary conditions). The tracks resulting from the stable-point extraction algorithm are marked on the images. The exact feature point is the center of each square (or: center of X mark). A label to the right of each mark holds its track number.

Figure 8 ("toys") contains frames from a video sequence taken in the laboratory. The sequence demonstrates a notable change in viewing angle. The significant camera motion can be observed from the high variation of the angle of the shadow of the pen tracked by Track 5. Most of the detected points are stable, in spite of the camera motion. Track 5, for example, has a short erroneous detection at the beginning of the sequence (for 4 frames), but for most of the sequence (37 frames out of 46) it tracks the 3D object (the pen) correctly.

Figure 9 ("parking-lot") shows frames from a video sequence of a parking lot, taken by a hand-held camera. In frames #25-#125 we demonstrate that in spite of the considerable zoom effect, all the points are faithfully tracked. In frames #175-#250, notice that tracks 9 and 10 correctly follow the background building, while track 11 consistently detects a seat inside a parking car. The parallax depicted in the relative motion of these tracks could be easily used for 3D scene correspondence.

Figure 10 ("traffic") is a sight of a highway from a nearby hill. The scene is extremely difficult to track, as it combines both a fast pan motion of the camera, motion of objects in the scene (cars), and 1:6 zoom out effect. On top of this, the camera is hand-held (as is the case in the toys and parking-lot sequences). Camera stability is significant especially in this case, because the initial part of the sequence has a significant zoom-in (as the zoom-out which comes at the end indicates), so any hand vibration translates into a significant change in viewing angle. As expected, the results of this sequence are worse

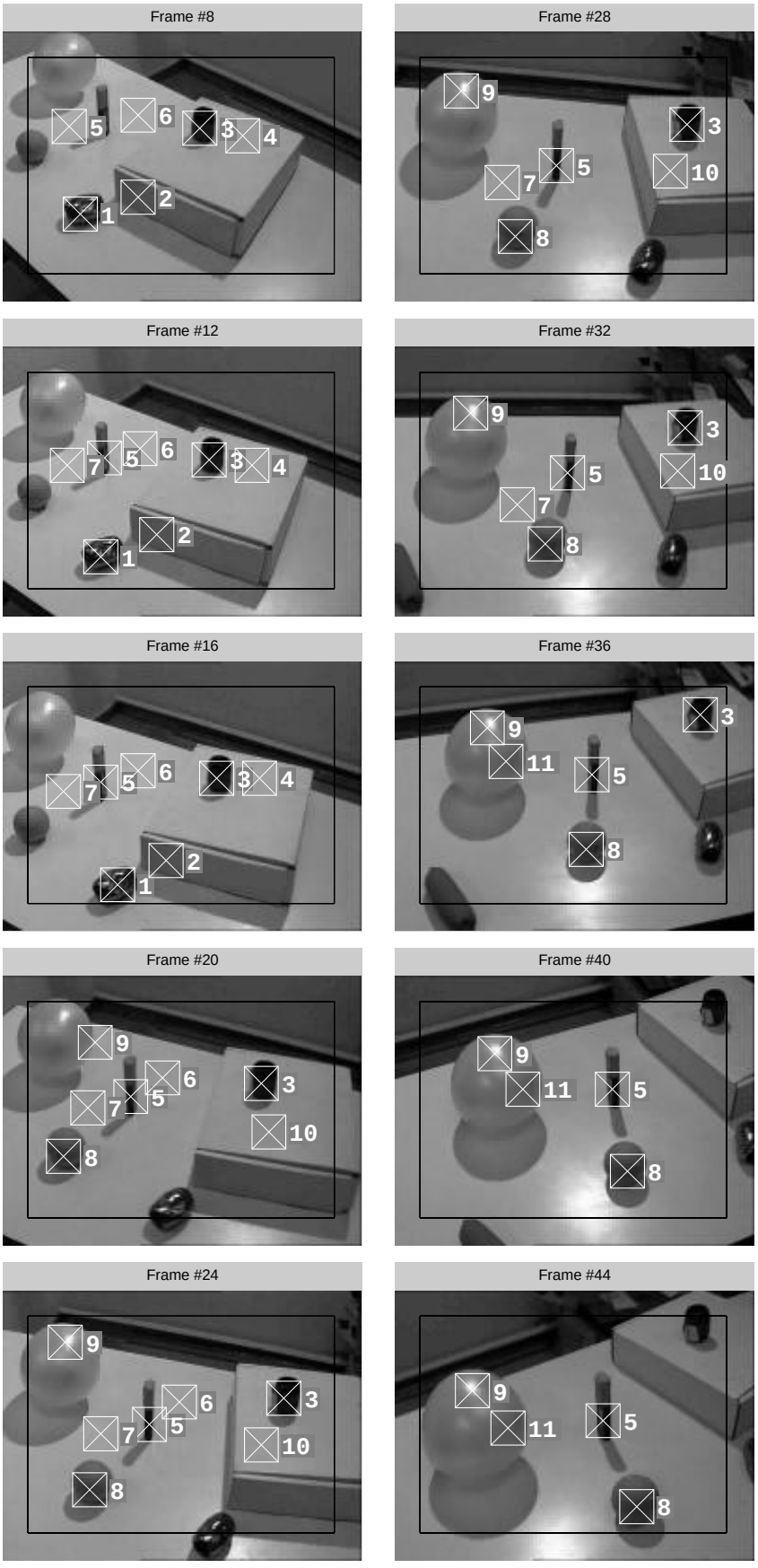


Figure 8: *Toys: Video sequence of objects in the laboratory. Note, for example, track 3 which follows the same object as long as it is in the frame, or track 8 which consistently detects the tennis ball.*

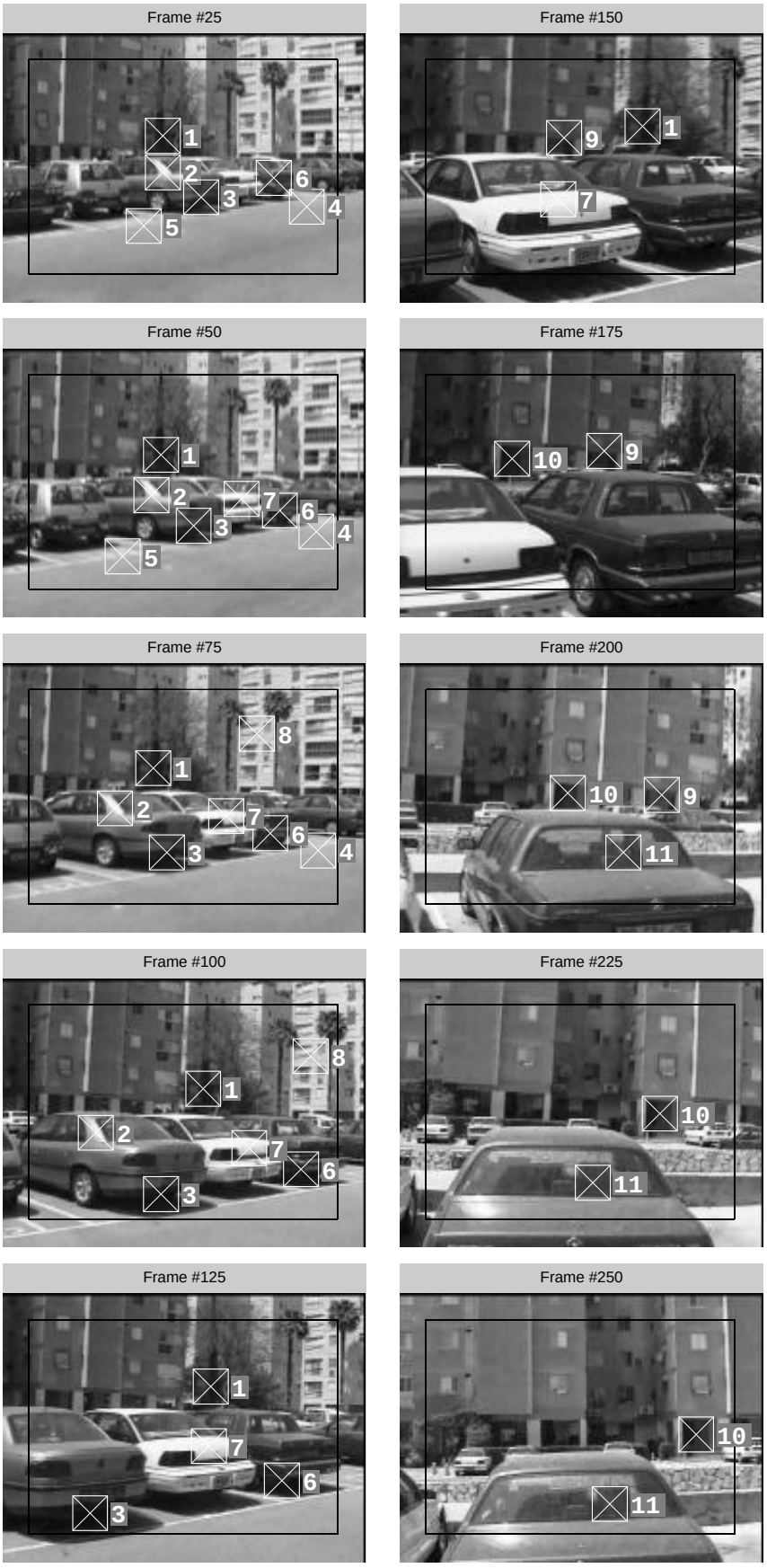


Figure 9: *Parking-lot: An outdoors video sequence of a parking lot. Despite the variability and parallax in the scene, tracking is correctly maintained in the vast majority of the cases. Track 1, for example, correctly tracks the tree in frames #1-#170. Pay attention to the change of scales of that tree in frames #25 and #150 (top row).*

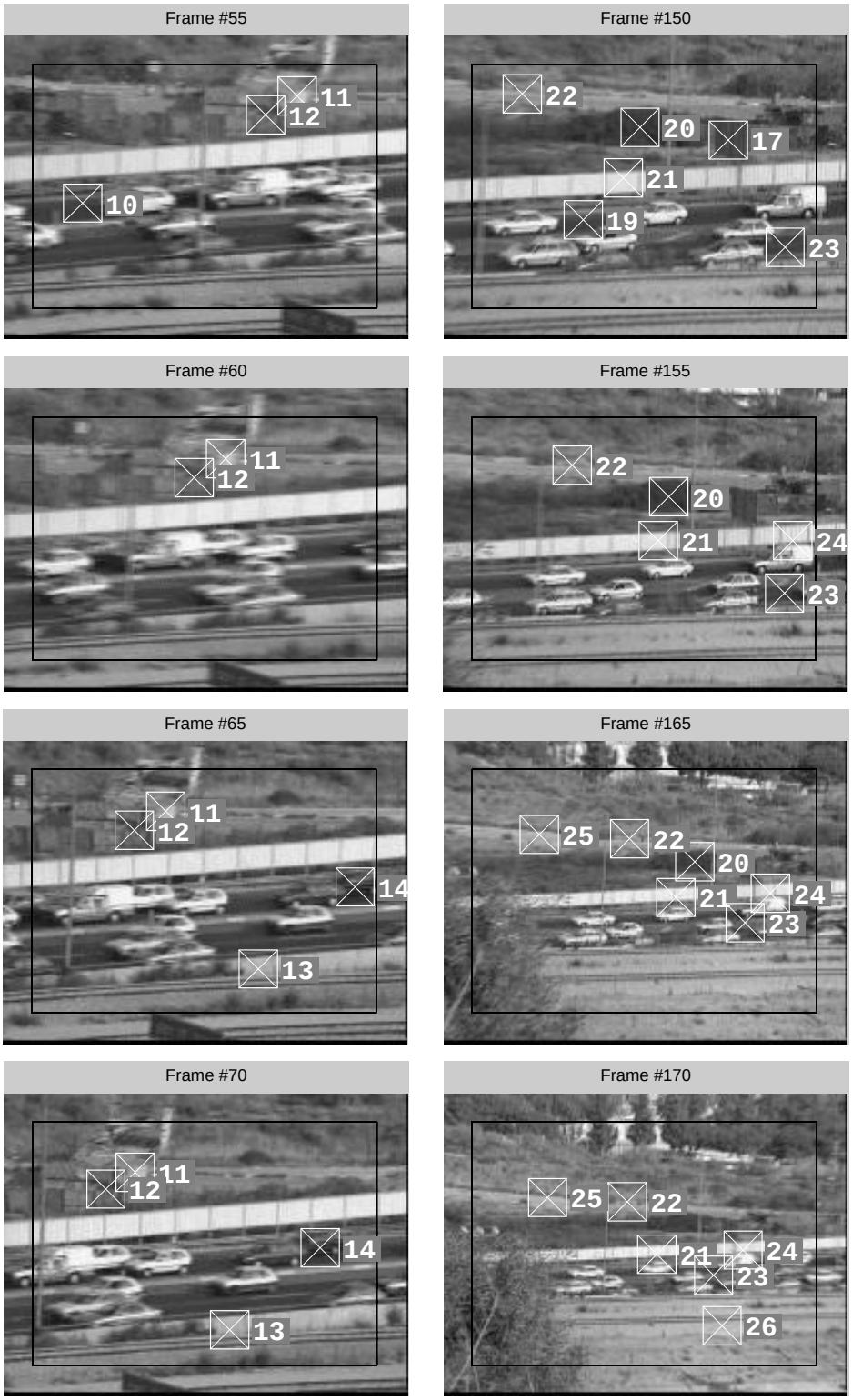


Figure 10: *Traffic: A sight of a highway from a nearby hill. The scene is very dynamic, combining several effects: camera motion, scene objects motion and zooming. Results are worse then in the toys or parking-lot sequences, but are still usable for algorithms requiring correspondence between successive frames. Note the correct tracking of track 22 (right column) despite strong zooming; Tracks 11 and 12 (left column) demonstrate fast camera motion.*

then of the other two examples (toys and parking-lot), but still, most of the tracks are correct for most of the time even for this challenging video, and can thus serve as an input to algorithms requiring correspondence of points between successive frames.

6 Evaluating the Performance of the Algorithm

An important issue in stable point tracking is the method of evaluation of the algorithm. Various evaluation techniques appear in the literature, but most of them are either not generic enough or do not take into account the 3D scene. Sometimes, methods who do take the 3D scene into account require the exact projection matrices and 3D positions, and thus are only appropriate for a lab test.

In the following subsections we define two different measures that would answer the flaws described above: one is more relevant when the goal is maximal-time point tracking; the other, when correspondence of points in successive frames is sought. These measures can serve for evaluation of 3D point tracking algorithms in general. The most important difference between them and existing evaluation measures is that they refer to the 3D scene points being tracked.

6.1 Terminology and Notation

In order to rigorously define the measures, we first introduce the basic terminology.

6.1.1 Definition of a Track

Let us define the *pixel space* of a video sequence of k frames where each frame has $n \times m$ pixels as: $\Omega = N_n \times N_m \times N_k$ where $N_i = \{1, 2, \ldots, i\}$. The *set of detected points* is the set of all pixels which a tracker selected; we denote it by D. $D \subseteq \Omega = N_n \times N_m \times N_k$.

The *track ID function* defines the track ID, as determined by the tracker; its notation is: $t : D \longmapsto N$. The track ID function must satisfy the constraint that in a certain frame, only one pixel may have a certain track ID. Formally,

$$\forall (a,b,c) \in D \;\; \forall (a',b',c') \in D \setminus \{\, (a,b,c) \,\} : \;\; t(a',b',c') = t(a,b,c) \;\Rightarrow\; c' \neq c \qquad (4)$$

Let $Q = \{q_1, ..., q_\alpha\} \subseteq D$ be a subset of detected image points. We say that Q *has track ID* T iff the track ID function of each point in Q is T: $t(q_1) = ... = t(q_\alpha) = T$. Let Q

denote the maximal point set which has track ID T; i.e., $\forall p \in D \setminus Q,\ t(p) \neq T$.

The *set of all pixels at distance ε pixels from track T* is the set of all image points whose distance to a point with track ID T in their frame is less than or equal to ε. Formally, let $P_\varepsilon^T = \{p_1, ..., p_\beta\} \subseteq \Omega$ denote the set of all pixels at distance ε pixels from track T.

$$\forall p_i = (u_1, v_1, f) \in P_\varepsilon^T \ \exists q_j = (u_2, v_2, f) \in Q : \parallel p_i - q_j \parallel \leq \varepsilon$$

where $i = 1, ..., \beta,\ j = 1, ..., \alpha$.

6.1.2 Selecting The Correct Scene Point of a Track

For the formal definition, let us assume we have an ideal function $\Gamma : P_\varepsilon^T \longmapsto R^3$ which maps pixels in the video sequence to the 3D scene point from which it originated. That is, given a point $p_i = (u_1, u_2, f) \in P_\varepsilon^T$ $(1 \leq i \leq \beta)$, $\Gamma(p_i)$ is the 3D scene point whose projection on the image plane of frame f is p_i.

An automatic tracker may sometimes fail to correctly track the same scene point, resulting in inconsistent tracking. Namely, a track may follow two different 3D points, each of them being tracked in a different part of the sequence. These parts may be interleaved, or may involve more than two scene points. In order to quantify the level of deviation from ideal tracking, one has to define which points in a track belong to the correct 3D scene point, and which do not. We say that point Γ_1 is the *correct scene point of track T* if, considering all scene points of pixels at distance ε from track T, the point Γ_1 obtains the maximal track time:

$$\parallel \{p_i \in P_\varepsilon^T \mid \Gamma(p_i) = \Gamma_1\} \parallel > \parallel \{p_j \in P_\varepsilon^T \mid \Gamma(p_j) = \Gamma_2\} \parallel \quad \forall \Gamma_2 \in R^3$$

If more than one 3D scene point, say Γ_1 and Γ_2, attain maximal track time, then we say that Γ_1 is the correct scene point in track T, provided that its tracking in track T began earlier (i.e., at an earlier frame) than tracking Γ_2 in track T. Formally, let Γ_1 be a 3D point with maximal tracking time (in the weak sense):

$$\parallel \{p_i \in P_\varepsilon^T \mid \Gamma(p_i) = \Gamma_1\} \parallel \geq \parallel \{p_j \in P_\varepsilon^T \mid \Gamma(p_j) = \Gamma_3\} \parallel \quad \forall \Gamma_3 \in R^3$$

and let $\Gamma_2 \in R^3$ be another point which attains the maximal tracking time:

$$\parallel \{p_i \in P_\varepsilon^T \mid \Gamma(p_i) = \Gamma_1\} \parallel = \parallel \{p_j \in P_\varepsilon^T \mid \Gamma(p_j) = \Gamma_2\} \parallel$$

In order to formally describe the case where Γ_1 begins to be tracked earlier than Γ_2, we distinguish two cases:

1. The projections of Γ_1 and Γ_2 (i.e., points p_i, p_j) are different (=their distance $> \varepsilon$):

$$\exists f_i,\ f_j:\ \ \exists p_i = (u_1, v_1, f_i) \in P_\varepsilon^T,\ \Gamma(p_i) = \Gamma_1,\ \ \exists p_j = (u_2, v_2, f_j) \in P_\varepsilon^T,\ \Gamma(p_j) = \Gamma_2:$$

$$\| p_i - proj_{f_i}(\Gamma(p_j)) \| > \varepsilon$$

where $proj_f : R^3 \longmapsto R^2$ is the projection transformation of a 3D scene point onto the plane of frame f. Under this assumption, we say that Γ_1 is the correct point iff:

$$\min\{f_i \,|\, p_i = (u_i, v_i, f_i) \in P_\varepsilon^T \text{ and } \Gamma(p_i) = \Gamma_1\} <$$
$$\min\{f_j \,|\, p_j = (u_j, v_j, f_j) \in P_\varepsilon^T \text{ and } \Gamma(p_j) = \Gamma_2\}$$

This definite minimum is assured to exist because in every frame, track T has at most one detected point (see Eq. 4).

2. The projections of Γ_1 and Γ_2 (i.e., points p_i, p_j) are less than ε pixels distant:

$$\forall f_i\, \forall f_j\ \ \exists p_i = (u_1, v_1, f_i) \in P_\varepsilon^T,\ \ \Gamma(p_i) = \Gamma_1,\ \ \exists p_j = (u_2, v_2, f_j) \in P_\varepsilon^T,\ \ \Gamma(p_j) = \Gamma_2:$$

$$\| p_i - proj_{f_i}(\Gamma(p_j)) \| < \varepsilon$$

In this case, one may arbitrarily select whether Γ_1 or Γ_2 is the correct point (as their projections are close enough).

6.2 Defining Completeness

One way to evaluate the performance of the algorithm is to evaluate its completeness. Intuitively, a track is *complete*, if the same 3D scene point is being tracked, up to a certain level of noise, in every frame where that 3D point appears.

The *completeness measure of track* T is the percent of frames where the correct point Γ_1 has been tracked with track ID T from the set of all frames where Γ_1 appears (i.e., the potential maximal track time):

$$\begin{aligned} completeness_T &= 100 \times \frac{\text{Actual Correct Track Time}}{\text{Potential Track Time}} \\ &= 100 \times \frac{\| \{f_i \,|\, \exists p_i = (u_i, v_i, f_i),\ p_i \in P_\varepsilon^T \text{ and } \Gamma(p_i) = \Gamma_1\} \|}{\| \{p \in \Omega \,|\, \Gamma(p) = \Gamma_1\} \|} \end{aligned}$$

The *completeness measure of a tracker for a video sequence* is the average completeness measure over all the tracks it detected for the specific video sequence.

Existing evaluation criteria (cf. Sect. 2.3) do not attempt to evaluate how persistent a feature tracker is, thus ignoring an important aspect of feature tracking.

To correctly evaluate the position in the following image, the 3D position Γ and the projection matrices $proj_f$ need to be known, but we do not need to have these: As is the case for almost any measuring scheme, we assume that a ground truth is given in order to be compared with. In this paper, the real point correlation has been carried out manually. Notice that the comparative norms we propose are aimed at comparing the general ranking of methods, and thus should be used on an accepted set of well defined sequences, for which a ground truth could be calculated.

In order to reduce the effect of subjective judgment, two measures were taken: First, a rigorous mathematical definition is presented, leaving only Γ and $proj_f$ for manual extraction. Second, a permissive noise level $\varepsilon = 3$ is allowed, compensating possible discrepancies between different manual calculations.

6.3 Stability of Tracking

For many practical purposes (e.g., the correspondence problem), a full tracking of 3D points, or even allocation of a single track number to a single scene point are not a must. In such applications, we look for a reliable association of several anchor points in one frame with the points in the successive frame, which are the projection of the same scene point. When associating points in the successive frame with points in the frame following it, the set of scene points the association refers to, might change. This leads to the stability criterion.

Let us examine a pair of successive frames: f_i, f_{i+1}. W.L.O.G, let $1, ..., r$ denote all track IDs which are common to both frames. Let $p_1^i, ..., p_r^i$ and $p_1^{i+1}, ..., p_r^{i+1}$ be the detected points for the corresponding tracks and frames. Let $proj_f : R^3 \longmapsto R^2$ be the projection transformation of a 3D scene point onto the plane of frame f. The *stability measure of frames f_i and f_{i+1} with allowed noise of ε pixels* is:

$$stability_\varepsilon(i,\, i+1) = 100 \times \frac{\mid \{j \in \{1, \ldots, r\} \,:\, \| \, proj_{i+1}(\Gamma(p_j^i)) - p_j^{i+1} \, \| \leq \varepsilon\} \mid}{r}$$

This measure resembles in some aspects the "repeatability" measure of [3]. The main difference, however, lies in the fact that the stability measure takes into account the tracking of an automatic tracker, thus examining the detection–tracking system as a whole.

The repeatability measure, on the other hand, aims merely at the detection part, thus implicitly assuming the tracking part (i.e., associating the detected points in successive frames) is always correct. Although such an assumption separates the evaluation of the detection part, being the goal of evaluation, from the tracking part, it may also introduce error in evaluation of the true stability of the features under consideration. An example for this may be a feature detector which selects all points in a large image region (maybe even the whole image) in two successive frames. Although the same 3D points are detected in both frames, such a detection may not be appropriate for tracking, because a tracker may not be able to correctly associate the feature points in the different frames due to their large number and proximity. Thus, an evaluation of a detection facility along with a tracker is more apt to real life tasks.

Another difference from the repeatability criterion is the group of points that one expects to be repeated: [3] examines the parts of the scene projected on both frames under consideration, and takes the lower number of interest points detected in either of these two images as the denominator for the repeatability measure.

In contrast, the suggested stability measure takes the denominator as the number of tracks which followed points in both these two frames (r). This means that if a track ends at a certain frame (i.e., there is no track with an identical track ID at the successive frame), this track is *not* considered in the group of points expected to be detected, even if the 3D point it refers to still appears in the successive frame. The reason is that for tasks such as correspondence, one may ignore such a track and use only tracks which did appear in both frames. The fact that the track have ended does not reduce the stability of the detection or tracking mechanisms (Bear in mind, that in order to evaluate how long a 3D point has been tracked we use the completeness measure; the stability measure should only evaluate how consistent the detection of a 3D point is, not how long). This difference is due to the fact that the stability measure considers *tracks* which have points in both frames, instead of the existence of a scene point in a frame, as the repeatability measure does.

7 Experimental Results

Various feature trackers have been suggested in the literature (see Sect. 2). In order evaluate the performance of our tracker, we compare our D_{arg}-based tracking algorithm with three other algorithms: Junction Detection, ImpHarris and KLT. Three of the algorithms under study: Junction Detection, ImpHarris and D_{arg}, use an identical tracker based on Kalman filter (the one described in Sect. 5). They all use exactly the same code for the tracker and share identical parameters. The KLT algorithm is itself a combined detector–tracker, and as such cannot be split.

Junction Detection is based on [2]: Junctions are detected according to the curvature of the level curves of the intensity function, multiplied by the gradient magnitude raised to the power of three. Scale is automatically selected by normalizing the derivatives. Features are tracked using a Kalman filter tracker.

The improved version of the Harris operator (marked: ImpHarris) was suggested in [3] (See Sect. 2.1). [3] uses 1% threshold of the maximum observed interest point strength. As the comparison would show (see Figs. 12, 13 for the parking-lot sequence only), this kind of threshold is too low for tracking: it results in a large number of adjacent feature points, many times in neighboring pixels. These feature points in many cases form connected edges rather than isolated points, as one usually expects from feature points. Fig. 11 demonstrates the problem. This problem decreases the amount of correct associations of features in successive frames by the tracker. To reduce this problem, we use a threshold of 20%. It can be seen from Figs. 12, 13 (for the parking-lot sequence only) that the increase of threshold improves the results of tracking the response of ImpHarris. Therefore, our comparison would refer to the 20% threshold version of ImpHarris. Tracking is, again, by Kalman filter.

The KLT (Kanade-Lucas-Tomasi) tracking algorithm[1] [25], [1] is the only tracker in our comparison which does not use a Kalman filter. This raises the question of whether the results are due to stable feature detection, or to the tracker. We address this issue once we present the experimental results.

In all trackers, tracks shorter than a certain percent of the video sequence are ignored. The percents are: 25% for the toys and parking-lot sequences, and 10% for the traffic

[1]Implementation by: Stan Birchfield, Stanford University, version: 1.1.5, 7 Oct. 1998.

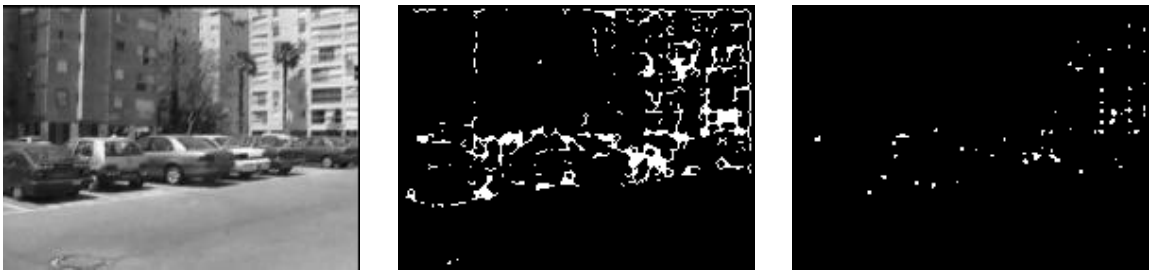


Figure 11: ***Left:*** *Frame #10 of the parking-lot sequence.* ***Middle:*** *ImpHarris with 1% threshold.* ***Right:*** *ImpHarris with 20% threshold. The increased threshold improves tracker ability to distinguish between feature points, leading to a more stable detection–tracking system.*

sequence. The threshold is lower in the later case, as the traffic video sequence has higher variability. The same thresholds were applied to all trackers.

7.1 Completeness Comparison

Figure 12 shows graphs of the completeness measure for the toys, parking-lot and traffic sequences. Each axes system shows the completeness measure of each of the four algorithms: D_{arg}, Junction Detection, KLT and ImpHarris (20% threshold version). The parking-lot sequence was also tested with the 1% version of ImpHarris (the original version).

In order to follow the development in time of the completeness measure, a sliding window over frames in the video sequence is employed. The calculation of the completeness measure refers to the frames of the window as if they were the whole video sequence. The window length is: 30 frames, and it shifts by 5 frames each time.

The allowed noise level in all sequences is: $\varepsilon = 3$ pixels.

The graphs show that the completeness of D_{arg} is at least comparable to that of Junction Detection and KLT, and usually better than ImpHarris. *On the traffic sequence*: Results for D_{arg}, Junction Detection and KLT are similar for this sequence, being a very difficult one. *On the toys sequence*: Junction Detection performs better than KLT. In part of the sequence, D_{arg} attains higher completeness than KLT, Junction Detection and ImpHarris. *On the parking-lot sequence*: The performance of D_{arg} is significantly better than that of Junction Detection, KLT or ImpHarris (in both versions). The most significant difference between Junction Detection or KLT and D_{arg} is at the last part of the sequence, when D_{arg} reaches very high completeness values (even 100%), while the completeness of Junction Detection and KLT drops. The reason is the high parallax at

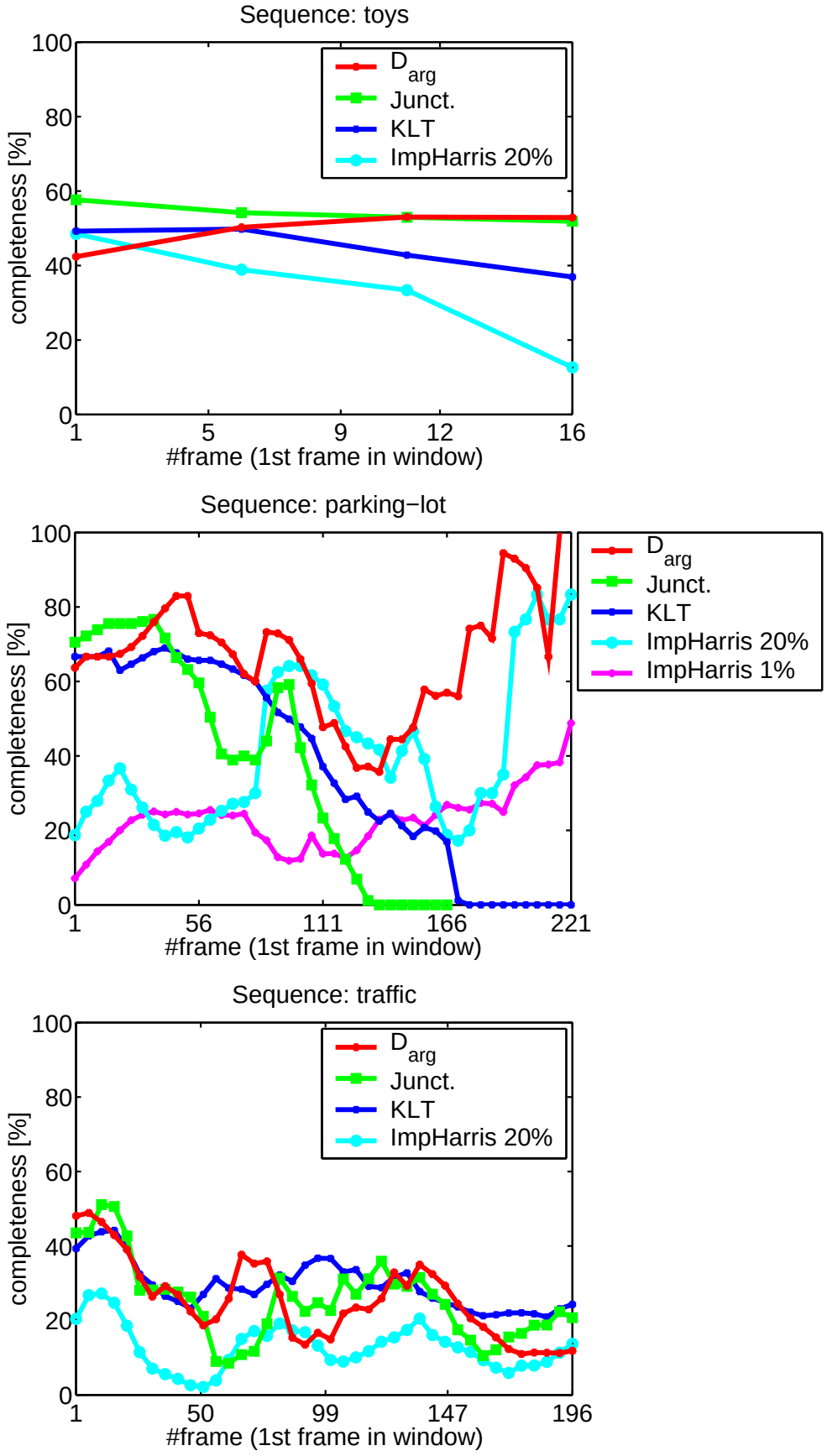


Figure 12: *Completeness comparison.* ***On the toys and traffic sequences:*** *D_{arg} performs as well as Junction Detection and KLT and most of the time better than ImpHarris.* ***On the parking-lot sequence:*** *D_{arg} attains better completeness than the other four trackers.*

the end of the sequence, when a car is very close to the camera, while the background buildings are quite far. D_{arg} copes well with parallax, as the feature it tracks is intrinsic to the 3D object. ImpHarris (20%) copes well with the last part of this sequence, but to a lower extent than D_{arg}.

7.2 Stability Comparison

Figure 13 introduces the stability criterion: $stability_{\varepsilon}(i,\, i+1)$, for the traffic, toys, and parking-lot sequences. The graphs show the sliding average stability over windows of 30 frames, shifted by 5 frames each time. The allowed noise level in all sequences is: $\varepsilon = 3$ pixels. As the graphs show, the stability of D_{arg} is higher than that of the other three trackers for the toys and parking-lot sequences. In parts of the parking-lot sequence, KLT equates with D_{arg}. For the traffic sequence we can order the trackers by their stability (descending order): KLT, D_{arg}, Junction Detection, and ImpHarris. In parts of this sequence D_{arg} equates with KLT.

7.3 No-Tracking Comparison

Our last criterion for tracker comparison would be the *no-track time*: the total time a tracker failed to track *any* point at all. We compare the total no-track time over all three video sequences together, for each of the three trackers. The total length of the three video sequences is: $46 + 252 + 227 = 525$ frames.

D_{arg} achieves the minimum no-track time: merely 4 frames without any tracking in all three video sequences. This no-track time is significantly less then that of the other three methods (ImpHarris: 24 frames; KLT: 81 frames; Junction Detection: 121 frames).

7.4 Results Summary

We see that D_{arg} is more stable than Junction Detection or ImpHarris, and sometimes (toys seq.) also more than KLT; Sometimes (parking-lot and traffic seq.) D_{arg} and KLT equate. One should also take into account the fact that the performance of D_{arg} in terms of stability is not at the expense of completeness, as D_{arg} maintains its level of completeness of tracking at least comparable to the Junction Detection and KLT trackers (sometimes even a better completeness), and better than the ImpHarris detector.

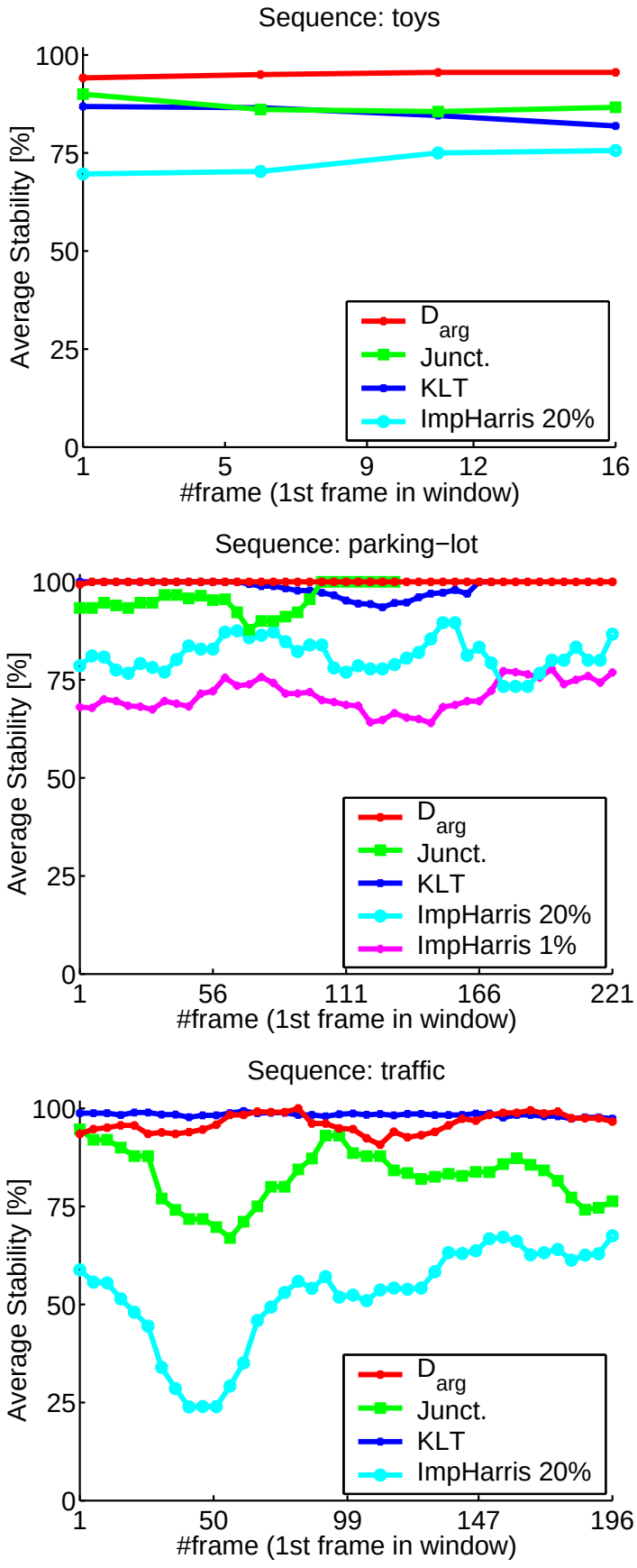


Figure 13: *Stability comparison. Tracking by D_{arg} is more stable than Junction Detection or ImpHarris. For the toys sequence, D_{arg} is more stable than KLT. For the parking-lot and traffic sequences, D_{arg} and KLT equate.*

D_{arg} has the lowest no-track time than the three other methods.

7.5 Interpreting The Results

When one interprets the results displayed here, one must refrain from over-generalization of the graphs introduced here. The reason is that the graphs rely merely on three video sequences. In order to perform a comparison from which a "best" detector can be declared, a much more extensive comparison should be applied, say on thousands of video sequences. Due to the large amount of manual work to be done in analyzing the results, this is beyond the scope of this paper.

The issue of test sequences does not refer only to the amount of sequences, but also to their distribution: some sequences may be edge-intensive, while others may be edge-sparse. Using ImpHarris, for example, in edge-intensive sequences may give better results than D_{arg}, while D_{arg} may yield better results in edge-sparse situations, because its detected domains would be more reliable in such cases (as shown by this paper). Different techniques are better suited for different scenes.

The conclusion from these results should be that a combination of scene-consistent detectors of various types should be employed in the general case, when the characteristic of scene is apriori unknown. One such combination of the ImpHarris detector at edge-intensive image domains and detector of extrema of the intensity at smooth domains is reported in [41]. The robustness of D_{arg} may improve the stability of the maxima detector there, and is a subject for future research.

When comparing various methods aimed at a common goal, there is always the question of using the most appropriate comparison measure. The goal of the algorithms tested here is to provide accurate sequences of corresponding image points for higher level tasks. To evaluate the suitability of algorithms for these tasks, we test the system as a whole, rather than its separated components. While it is rather clear that such a comparison of the three algorithms that use the same Kalman filtering is "fair" and differences in results are due to differences in detected features, the comparison with KLT that uses a built-in tracker may raise some issues. However, we have chosen to include KLT in our comparison since we consider it a commonly used method in this context, and as such it serves as a basis for comparison. It should also be noted that KLT tracking is considered to be more advanced than a mere Kalman filtering.

8 Task-Oriented Measures

In Sect. 6, we have introduced two generic measures for evaluation of algorithms for stable-point tracking: completeness and stability. The goal of these measures is to quantify the consistency of the tracks with the three dimensional scene being tracked. As was mentioned there, each of the two measures is more appropriate for a different task: completeness - for maximal time tracking, stability - for point correspondence in successive frames. This implies that evaluation of algorithms is task-dependent: different tasks require different evaluation schemes in order to reflect the quality of the algorithms at the relevant aspects of the problem.

Because evaluation schemes are task dependent, quantifying the consistency of tracks with the 3D scene points may not be enough for certain tasks. This may happen when the selected features are being tracked consistently with respect to the scene points, but the selected features themselves are not appropriate for the task. An example of inappropriate features is that of collinear features for 3D reconstruction of a complex (i.e., non-linear) scene: even when tracking is perfect with respect to the 3D scene points being tracked, a correct 3D reconstruction cannot be achieved based on these points.

Figure 14 ("Flower Garden") shows an example where the consistency of tracks with the 3D scene may not be appropriate for certain tasks. This video sequence contains a tree close to the camera, a flower garden farther away, and houses on the farther side of the garden. When the KLT algorithm is used to track points in the sequence, its tracking is more consistent with the 3D scene than D_{arg}-based tracking. This fact can be quantified using both completeness and stability measures, as Fig. 15 shows. However, what the measures do not show is that KLT tracks merely background objects like flowers and houses, but not the tree. D_{arg}, on the other hand, has some tracks which follow the tree, and others which follow the background. Applications which attempt to recover depth or segment the image based on motion may find the tracking suggested by D_{arg} more useful (the motion of the tree with respect to the camera is significantly different than that of the background, due to notable parallax).

In order to cope with the task-dependency of evaluation schemes, we suggest that the performance analysis of stable-point tracking algorithms is done in light of the higher level task which would employ the output of the tracking algorithm. More specifically,

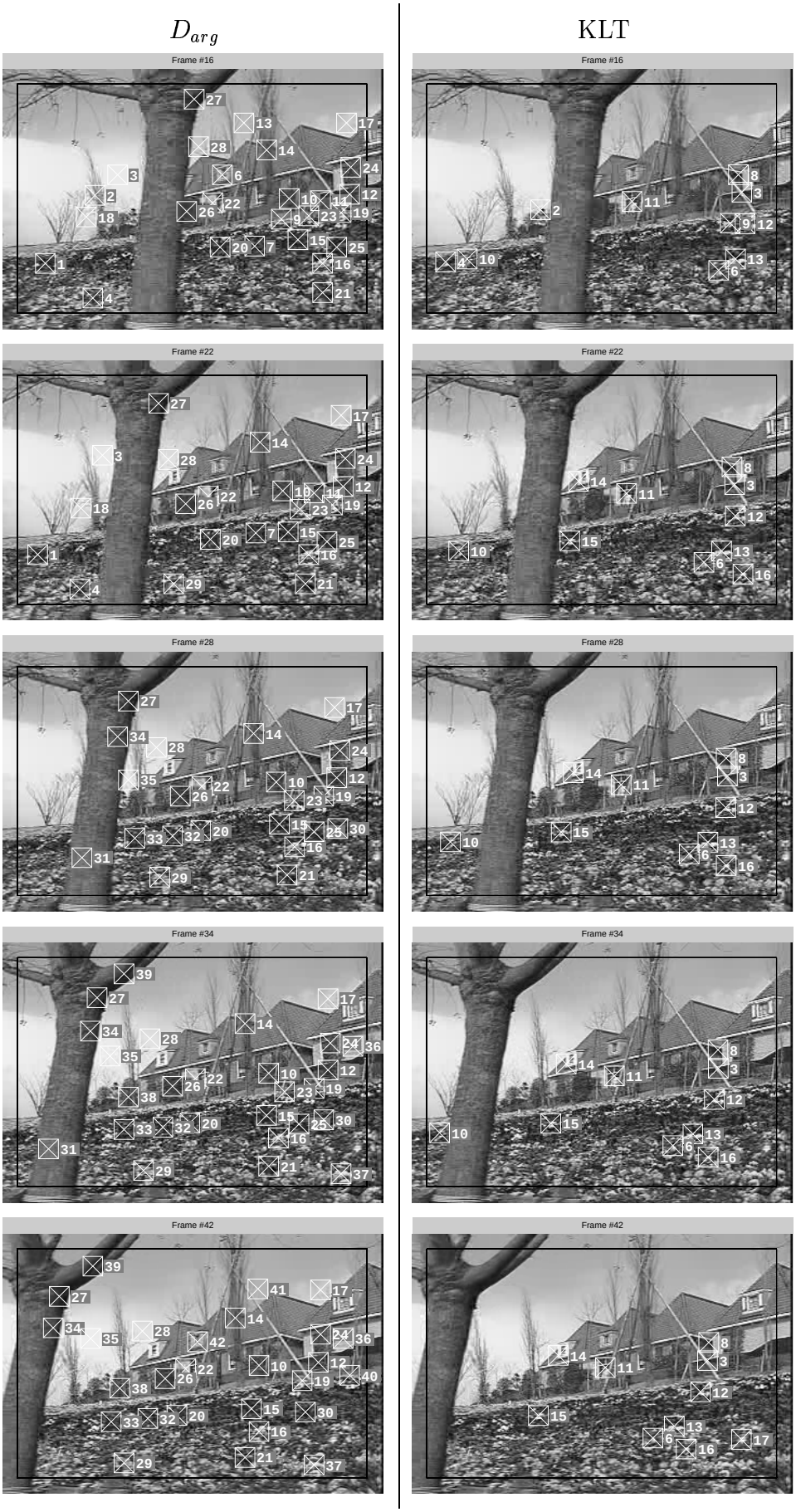


Figure 14: *Tracking the "Flower Garden" video sequence.* ***Left:*** *Tracking by D_{arg}. Both the tree and the background are being tracked. Pay attention in particular to tracks 27 and 34 which detect the tree.* ***Right:*** *Tracking by KLT tracks only background objects.*

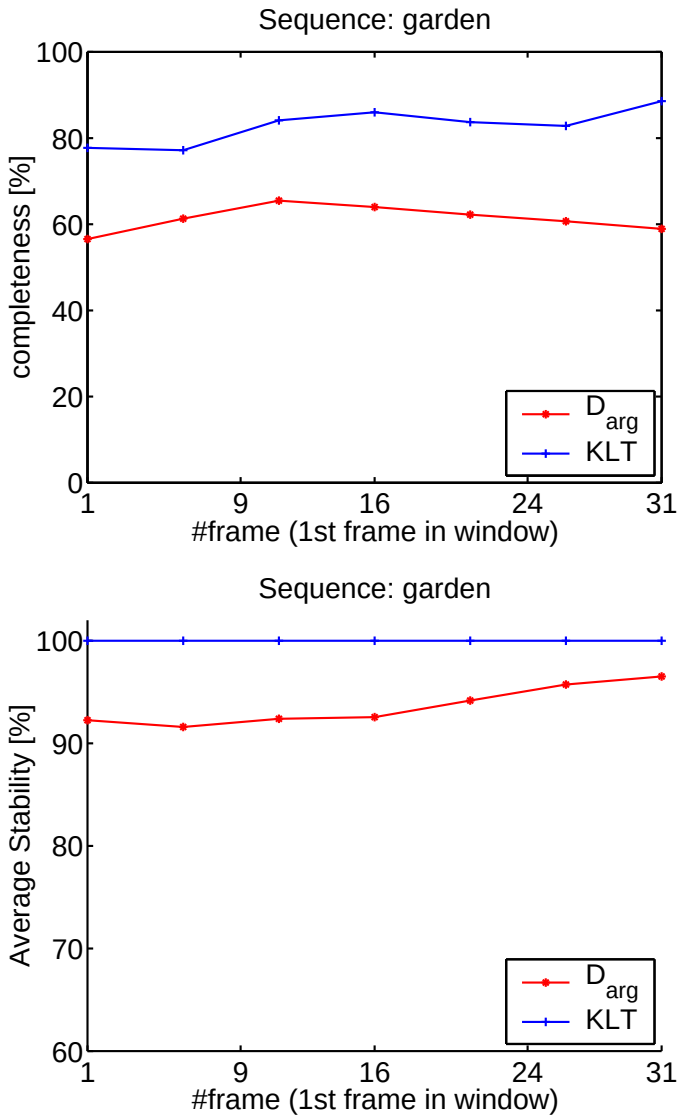


Figure 15: *Completeness and stability comparison for the "Flower Garden" video sequence. Both measures show that KLT performs better than D_{arg}. The measures cannot reflect the fact that D_{arg} (but not KLT) tracks the tree, because it is a smooth 3D edge-sparse object. Applications in which detection of the tree is crucial, may use a-priori weighting to give the tree a higher weight; such a weighting may change the preferred algorithm. In our case, such a task-oriented weighting would favor D_{arg}.*

one should apply a weighting scheme to the video sequence under consideration; different parts of the scene would gain different a-priori weights according to the higher level task of the tracked points. A track which follows a certain scene point would receive the a-priori weight of that scene point when calculating the completeness and stability measures. The overall analysis would thus fit better for the required task. This solution is similar, in a sense, to the a-priori distribution of viewpoints employed in [39] when determining stable views of objects.

In the "Flower Garden" example, some applications may need tracking of the tree in addition to the background. In such applications, one would allocate higher a-priori weights to the tree than to the background. Because only D_{arg} tracks the tree, this weighting would give rise only to the completeness and stability of D_{arg}, but not to that of KLT. The weighted measures may then show that D_{arg} is superior to KLT. In the case of an application where the tree ought to be tracked, indeed D_{arg} is to be preferred to KLT, and this fact is reflected only by the weighted measures. This demonstrates that the use of task-oriented a-priori weights renders the suggested measures more effective in view of the task for which tracking was originally initiated.

9 Conclusions

We have presented an operator that detects local extrema of the intensity function in twice continuously differentiable image domains. The operator is highly efficient and robust, with weak response to edges. Observing that the zero crossing of the gradient argument is a highly prominent feature, we analytically show that this zero crossing relates to specific features of the intensity surface, which, in turn, relates to specific local features of the 3D scene geometry. Based on this operator, a commonly used algorithm for stable point tracking (using a 2D multi-target Kalman filter tracker) is described. Several video sequences demonstrate the high robustness maintained by the algorithm.

Two measures, completeness and stability, are introduced in order to evaluate performance of algorithms for feature point tracking as well as correspondence establishing tasks. These measures overcome various flaws in existing evaluation measures of feature point trackers. The *completeness* measure is aimed at maximizing the tracking time of a 3D scene point. The goal of the *stability* measure is to keep consistent tracking of 3D

scene points between successive frames (but the set of tracked scene points may change between frames). Applying task-oriented a-priori weighting to these measures was shown to improve the suitability of the suggested measures to the original tasks for which tracking was initiated. The suggested measures are generic, and are suggested as a basis for comparison of 3D point tracking algorithms in general.

We have used the suggested measures in a comparison of our tracker with three other detection–tracking methods. The main conclusion from the comparison is that a combination of detectors should be employed: edge-based detectors (for example, corner detectors) should be employed in edge-intensive image domains, while robust non-edge-based detectors (e.g., D_{arg}) should be employed in edge-sparse image domains.

Acknowledgements

We thank Dr. Ronny Kimmel for his helpful remarks on level sets.

This research was supported by grants from the Minerva Minkowski center for geometry, the Excellence Center for Geometric Computing of the Israel Academy of Science, and Israel Ministry of Science.

References

[1] C. Tomasi and T. Kanade. Shape and motion from image streams: a factorization method—part 3: Detection and tracking of point features. Technical Report CMU-CS-91-132, Carnegie Mellon University, Pittsburg, PA, USA, April 1991.

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

[3] C. Schmid, R. Mohr, and C. Bauckhage. Evaluation of interest point detectors. *International Journal of Computer Vision*, 37(2):151–172, 2000.

[4] A. Tankus and Y. Yeshurun. Convexity-based visual camouflage breaking. *Computer Vision and Image Understanding*, 82(3):208–237, June 2001.

[5] Y. Yeshurun. Attentional mechanisms in computer vision. In V. Cantoni, S. Levialdi, and V. Roberto, editors, *Artificial Vision*, chapter 2, pages 43–52. Academic Press Inc., 1997.

[6] T. Lindeberg. Detecting salient blob-like image structures and their scales with a scale-space primal sketch: A method for focus-of-attention. *International Journal of Computer Vision*, 11(3):283–318, 1993.

[7] O. Stasse, Y. Kuniyoshi, and G. Cheng. Development of a biologically inspired real-time visual attention system. In Seong-Whan Lee, Heinrich H. Bülthoff, and Tomaso Poggio, editors, *Biologically Motivated Computer Vision*, number 1811 in LLNCS, pages 150–159, Seoul, Korea, May 2000. Springer.

[8] H. J. Wolfson. Model based object recognition by 'Geometric Hashing'. In *Proc. of the 1^{st} European Conference on Computer Vision*, pages 526–536, Antibes, France, April 1990. Springer Verlag.

[9] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157, Kerkyra, Greece, Sept 1999.

[10] R. Deriche and G. Giraudon. Accurate corner detection : An analytical study. In *Proceedings of the 3rd International Conference on Computer Vision*, pages 66–70, Osaka, Japan, 1990.

[11] S. Frantz, K. Rohr, and H. Siegfried Stiehl. Multi-step procedures for the localization of 2d and 3d point landmarks and automatic roi size selection. In Hans Burkhardt and Bernd Neumann, editors, *Proceedings of the 5^{th} European Conference on Computer Vision*, volume 1, pages 687–703, Freiburg, Germany, June 1998.

[12] M. A. Ruzon and C. Tomasi. Corner detection in textured color images. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1039–1045, Kerkyra, Greece, Sept 1999.

[13] R. Laganière. Morphological corner detection. In *Proceedings of the Sixth International Conference on Computer Vision*, pages 280–285, Bombay, India, January 1998.

[14] C. Harris and M. Stephans. A combined corner and edge detector. In *The 4th Alvey Vision Conference*, pages 147–151, 1988.

[15] C. Harris. The DROID 3D vision system. Technical Report 72/88/N488U, Plessey Research, Roke Manor, 1988.

[16] P. Beardsley, P. H. S. Torr, and A. Zisserman. 3D model acquisition from extended image sequences. In B. Buxton and Cipolla R., editors, *Proceedings of the 4th European Conference on Computer Vision*, volume 2, pages 683–695. Springer–Verlag, 1996.

[17] Q. Zheng and R. Chellappa. Automatic feature point extraction and tracking in image sequences for arbitrary camera motion. *International Journal of Computer Vision*, 15:31–76, 1995.

[18] L. Bretzner and T. Lindeberg. Qualitative multi-scale feature hierarchies for object tracking. In *Second International Conference on Scale-Space Theories in Computer Vision*, pages 117–128, Kerkyra, Greece, Sept 1999.

[19] V. Rehrmann. Object-oriented motion estimation in color image sequences. In Hans Burkhardt and Bernd Neumann, editors, *Proceedings of the 5^{th} European Conference on Computer Vision*, volume 1, pages 704–719, Freiburg, Germany, June 1998.

[20] S. M. Smith and J. M. Brady. ASSET-2: Real-time motion segmentation and shape tracking. *Transactions on Pattern Analysis and Machine Intelligence*, 17(8):814–820, 1995.

[21] S. M. Smith and J. M. Brady. SUSAN - A new approach to low level image processing. *International Journal of Computer Vision*, 23(1):45–78, May 1997.

[22] D. Beymer, P. McLauchlan, B. Coifman, and J. Malik. A real-time computer vision system for measuring traffic parameters. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, pages 495–501, San Juan, PR, June 1997.

[23] W. Förstner and E. Gülch. A fast operator for detection and precise location of distinct points, corners and centres of circular features. In *Proceedings of the Inter-*

commission Conference on Fast Processing of Photogrammetric Data, pages 281–305, Interlaken, Switzerland, 1987.

[24] S. B. Kang, R. Szeliski, and H.-Y. Shum. A parallel feature tracker for extended image sequences. *Computer Vision and Image Understanding*, 67(3):296–310, September 1997.

[25] J. Shi and C. Tomasi. Good features to track. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 593–600, Seattle, June 1994.

[26] T. Tommasini, A. Fusiello, E. Trucco, and V. Roberto. Making good features track better. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, Santa Barbara, CA, USA, June 1998.

[27] Y. Song, L. Goncalves, E. Di Bernardo, and P. Perona. Monocular perception of biological motion - detection and labeling. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 805–812, Kerkyra, Greece, Sept 1999.

[28] J. Verestóy and D. Chetverikov. Experimental comparative evaluation of feature point tracking algorithms. In *Evaluation and Validation of Computer Vision Algorithms*, Kluwer Series in Computational Imaging and Vision, pages 183–194, 2000.

[29] P. Tissainayagam and D. Suter. Performance analysis of a point feature tracker based on different motion models. *Computer Vision and Image Understanding*, 2000. submitted.

[30] Y. G. Leclerc, Q.-T. Luong, and P. Fua. Measuring the self-consistency of stereo algorithms. In *Proceedings of the 6^{th} European Conference on Computer Vision*, volume 1, pages 282–298, Dublin, Ireland, June/July 2000.

[31] A. P. Pentland. Local shading analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(2):170–187, March 1984.

[32] R. Zhang, P.-S. Tsai, J. E. Cryer, and M. Shah. Shape from shading: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8):690–705, August 1999.

[33] A. Tankus and Y. Yeshurun. Convexity-based camouflage breaking. In *Proceedings of the 15th International Conference on Pattern Recognition*, volume 1, pages 454–457, Barcelona, Spain, September 2000.

[34] B. K. P. Horn. *Robot Vision*. The MIT Press/McGraw-Hill Book Company, 1986.

[35] A. M. Bruckstein. On shape from shading. *Computer Vision, Graphics and Image Processing*, 44:139–154, 1988.

[36] R. Kimmel and A. M. Bruckstein. Global shape from shading. *Computer Vision and Image Understanding*, 62(3):360–369, Nov. 1995.

[37] R. Kimmel and J. A. Sethian. Optimal algorithm for shape from shading and path planning. *Journal of Mathematical Imaging and Vision*, 14(3):237–244, 2001.

[38] P. T. Sander and S. W. Zucker. Singularities of principal direction fields from 3-d images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(3):309–317, March 1992.

[39] D. Weinshall and M. Werman. On view likelihood and stability. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):97–108, February 1997.

[40] L. S. Shapiro. *Affine Analysis of Image Sequences*. Cambridge University Press, Cambridge, England, 1995.

[41] T. Tuytelaars and L. Van Gool. Matching widely separated views based on affinely invariant neighbourhoods. *International Journal on Computer Vision*, July 2001. submitted for publication.

A Response to the Intensity Surface - Proof

Theorem 1 *Let $I : R \times R \longmapsto R \in C^2$ (i.e., $I(x,y)$ is twice continuously differentiable w.r.t both x and y) be an intensity function. If (x_0, y_0) is a point where: $\lim_{y \to y_0} \frac{\partial}{\partial y}\theta(x,y)|_{x=x_0} = \pm\infty$ (i.e., a point where $Y_{arg} \to \pm\infty$), then there exists $\varepsilon > 0$ so that for all y, for which $| y - y_0 | < \varepsilon$, one of the following cases holds:*

1. $\frac{\partial I(x,y)}{\partial y}|_{x=x_0} = 0$ *for all* y *and*
   $\forall y < y_0$, $\frac{\partial I(x,y)}{\partial x}|_{x=x_0} \geq 0$, *and* $\forall y > y_0$, $\frac{\partial I(x,y)}{\partial x}|_{x=x_0} < 0$. *

2. $\frac{\partial I(x,y)}{\partial y}|_{x=x_0} > 0$ *for* $y < y_0$ *and* $\frac{\partial I(x,y)}{\partial y}|_{x=x_0} = 0$ *for* $y > y_0$, *and* *

   (a) $\forall y > y_0$, $\frac{\partial I(x,y)}{\partial x}|_{x=x_0} = 0$. *or:*

   (b) $\forall y < y_0$, $\frac{\partial I(x,y)}{\partial x}|_{x=x_0} = 0$. *or:*

   (c) $\forall y < y_0$, $\frac{\partial I(x,y)}{\partial x}|_{x=x_0} > 0$, *and* $\forall y > y_0$, $\frac{\partial I(x,y)}{\partial x}|_{x=x_0} < 0$. *

3. $\frac{\partial I(x,y)}{\partial y}|_{x=x_0} < 0$ *for* $y < y_0$ *and* $\frac{\partial I(x,y)}{\partial y}|_{x=x_0} = 0$ *for* $y > y_0$, *
   except for the case where* $\forall y : y \neq y_0$, $\frac{\partial I(x,y)}{\partial x}|_{x=x_0} > 0$.

4. (x_0, y_0) *is a local extremum of* $I(x_0, y)$,
   except for the case where $\forall y : y \neq y_0$, $\frac{\partial I(x,y)}{\partial x}|_{x=x_0} > 0$.

** The case where the conditions for $y < y_0$ are swapped with those for $y > y_0$ is also valid; it is an equivalent case and was therefore omitted.*

Proof:

Let:

$$\arctan(y,x) = \begin{cases} \arctan(\frac{y}{x}) & \text{if } x > 0 \\ \arctan(\frac{y}{x}) + \pi & \text{if } x < 0,\ y \geq 0 \\ \arctan(\frac{y}{x}) - \pi & \text{if } x < 0,\ y < 0 \\ 0 & \text{if } x = 0,\ y = 0 \\ \frac{\pi}{2} & \text{if } x = 0,\ y > 0 \\ -\frac{\pi}{2} & \text{if } x = 0,\ y < 0 \end{cases}$$

Let us define the domain: $D \stackrel{def}{=} \{(x,y) \in \mathrm{R}^2 \mid y \neq 0 \text{ or } x > 0\}$, and let: $\arctan|_D : D \longmapsto \mathrm{R}$ denote the 2D arctan function reduced to domain D. For each $(x,y) \in D$,

arctan $|_D$ is differentiable. $I(x,y)$ is twice continuously differentiable, so $\frac{\partial I(x,y)}{\partial x}$ and $\frac{\partial I(x,y)}{\partial y}$ are differentiable. By the chain rule, the compound function:

$$\theta|_E(x,y) \stackrel{def}{=} \arctan|_D \left(\frac{\partial I(x,y)}{\partial y} \, , \, \frac{\partial I(x,y)}{\partial x} \right)$$

where $E \stackrel{def}{=} \{(x,y) \in \mathrm{R}^2 \mid \frac{\partial I(x,y)}{\partial y} \neq 0 \text{ or } \frac{\partial I(x,y)}{\partial x} > 0\}$, is also differentiable. Therefore, if $(x,y) \in E$, then $\mid \frac{\partial}{\partial y}\theta(x,y) \mid = \mid \frac{\partial}{\partial y}\theta|_E(x,y) \mid < \infty$.

It follows, that if $\frac{\partial}{\partial y}\theta(x,y) \to \pm\infty$, then $(x,y) \notin E$. In other words, if $\frac{\partial}{\partial y}\theta(x,y) \to \pm\infty$, then $(x,y) \in \neg E = \{(x,y) \in \mathrm{R}^2 \mid \frac{\partial I(x,y)}{\partial y} = 0 \text{ and } \frac{\partial I(x,y)}{\partial x} \leq 0\}$.

Let $(x_0,y_0) \in \neg E$ be a point where: $\lim_{y \to y_0} \frac{\partial}{\partial y}\theta(x,y)|_{x=x_0} = \pm\infty$. From the above, necessarily: $\frac{\partial I(x,y)}{\partial y}|_{x=x_0,\, y=y_0} = 0$.

Because of the continuous differentiability of $\frac{\partial I(x,y)}{\partial y}$ (and similarly $\frac{\partial I(x,y)}{\partial x}$), if one examines a small enough neighborhood of (x_0,y_0), then at each side of y_0, $\frac{\partial I(x,y)}{\partial y}$ has a constant sign at that side, i.e., either $\forall y \in (y_0 - \varepsilon,\, y_0) : \frac{\partial I(x,y)}{\partial y} > 0$ or: $\forall y \in (y_0 - \varepsilon,\, y_0) : \frac{\partial I(x,y)}{\partial y} = 0$ or $\forall y \in (y_0 - \varepsilon,\, y_0) : \frac{\partial I(x,y)}{\partial y} < 0$, and similarly for $y \in (y_0,\, y_0 + \varepsilon)$. From here on, we assume all points (x,y) are in that ε y-neighborhood, and the x-rate is $x = x_0$. We next examine the signs of $\frac{\partial I(x,y)}{\partial y}$ and $\frac{\partial I(x,y)}{\partial x}$ at points (x,y) of a sufficiently small neighborhood of (x_0,y_0):

1. (x_0,y_0) is a point of inflection of $I(x_0,y)$, i.e., $\frac{\partial I(x,y)}{\partial y}$ has a constant sign (except for the point of inflection (x_0,y_0) itself, where it is 0):

   (a) $\forall y : y \neq y_0, \;\; \frac{\partial I(x,y)}{\partial y} > 0.$

   (b) $\forall y : y \neq y_0, \;\; \frac{\partial I(x,y)}{\partial y} < 0.$

   In either of these cases, $\theta(x,y)$ has a removable singularity, because the 2D arctan in quadrants I, II: $\arctan_{|\{y:y>0\}}(y,x)$, is continuous, and so is $\arctan_{|\{y:y<0\}}(y,x)$ (quadrants III, IV). Therefore, there is *no* jump discontinuity.

2. $\forall y : y \neq y_0, \;\; \frac{\partial I(x,y)}{\partial y} = 0$: Let us examine $\frac{\partial I(x,y)}{\partial x}$:

   (a) $\forall y : y \neq y_0, \;\; \frac{\partial I(x,y)}{\partial x} < 0.$

   Similar to case (1)(a), $\theta(x_0,y) = \pi$ for all $y \neq y_0$. A removable discontinuity contradicts the assumption that $\frac{\partial}{\partial y}\theta(x,y) \to \pm\infty$.

   (b) $\forall y : y < y_0, \;\; \frac{\partial I(x,y)}{\partial x} \geq 0$, and $\forall y : y > y_0, \;\; \frac{\partial I(x,y)}{\partial x} = 0.$ (or vice versa; the other case is similar). For all y, $\theta(x,y) = 0$, so no jump discontinuity occurs.

(c) $\forall y : y < y_0$, $\frac{\partial I(x,y)}{\partial x} \geq 0$, and $\forall y : y > y_0$, $\frac{\partial I(x,y)}{\partial x} < 0$. (or vice versa; the other case is similar).

$$\theta(x,y) = \begin{cases} 0, & \text{if } y < y_0 \\ \pi, & \text{if } y > y_0 \end{cases}$$

Jump discontinuity occurs in this case, and $\frac{\partial}{\partial y}\theta(x,y) \to +\infty$.

3. $\forall y : y < y_0$, $\frac{\partial I(x,y)}{\partial y} > 0$ and $\forall y : y > y_0$, $\frac{\partial I(x,y)}{\partial y} = 0$ (or vice versa). The possible cases are summarized in the following table. The left column describes the case under consideration. The middle column shows the behavior of $\theta(x,y)$ under that constraint. The right column indicates whether or not a jump discontinuity of $\theta(x,y)$ may occur in this case.

Condition:	$\theta(x,y) =$	$\theta(x,y)$ has:
$y < y_0$, $\frac{\partial I(x,y)}{\partial x} > 0$ $y > y_0$, $\frac{\partial I(x,y)}{\partial x} < 0$	$\begin{cases} \arctan(\frac{\partial I(x,y)}{\partial y} / \frac{\partial I(x,y)}{\partial x}) + \pi, & \text{if } y \leq y_0 \\ \pi, & \text{if } y > y_0 \end{cases}$	Continuity.
$y > y_0$, $\frac{\partial I(x,y)}{\partial x} = 0$	$\begin{cases} \theta(x,y) > 0, & \text{if } y < y_0 \\ \theta(x,y) = 0, & \text{if } y > y_0 \end{cases}$	Jump discontinuity
$y < y_0$, $\frac{\partial I(x,y)}{\partial x} = 0$	$\begin{cases} \frac{\pi}{2}, & \text{if } y < y_0 \\ 0 \text{ or } \pi, & \text{if } y > y_0 \end{cases}$	Jump discontinuity
$y < y_0$, $\frac{\partial I(x,y)}{\partial x} > 0$ $y > y_0$, $\frac{\partial I(x,y)}{\partial x} < 0$	$\begin{cases} \arctan(\frac{\partial I(x,y)}{\partial y} / \frac{\partial I(x,y)}{\partial x}) < \frac{\pi}{2}, & \text{if } y < y_0 \\ \pi, & \text{if } y > y_0 \end{cases}$	Jump discontinuity
$y < y_0$, $\frac{\partial I(x,y)}{\partial x} < 0$ $y > y_0$, $\frac{\partial I(x,y)}{\partial x} > 0$	$\begin{cases} \arctan(\frac{\partial I(x,y)}{\partial y} / \frac{\partial I(x,y)}{\partial x}) + \pi > \frac{\pi}{2}, & \text{if } y < y_0 \\ 0, & \text{if } y > y_0 \end{cases}$	Jump discontinuity

4. $\forall y : y < y_0$, $\frac{\partial I(x,y)}{\partial y} < 0$ and $\forall y : y > y_0$, $\frac{\partial I(x,y)}{\partial y} = 0$ (or vice versa).

Condition:	$\theta(x,y) =$	$\theta(x,y)$ has:
$y > y_0$, $\frac{\partial I(x,y)}{\partial x} \leq 0$	$\begin{cases} \theta(x,y) < 0, & \text{if } y < y_0 \\ \theta(x,y) = 0 \text{ or } \pi, & \text{if } y > y_0 \end{cases}$	Jump discontinuity
$y < y_0$, $\frac{\partial I(x,y)}{\partial x} \leq 0$	$\begin{cases} \arctan(\frac{\partial I(x,y)}{\partial y} / \frac{\partial I(x,y)}{\partial x}) - \pi \leq -\frac{\pi}{2}, & \text{if } y < y_0 \\ 0 \text{ or } \pi, & \text{if } y > y_0 \end{cases}$	Jump discontinuity

5. (x_0, y_0) is a local extremum of $I(x_0, y)$: $\frac{\partial I(x,y)}{\partial y}$ has different signs for $y > y_0$ and for $y < y_0$. For a minimum point (x_0, y_0), i.e., $\forall y : y < y_0$, $\frac{\partial I(x,y)}{\partial y} < 0$ and $\forall y : y > y_0$, $\frac{\partial I(x,y)}{\partial y} > 0$ (the case of a maximum point is equivalent):

Condition:	$\theta(x,y) =$	$\theta(x,y)$ has:
$y < y_0, \quad \frac{\partial I(x,y)}{\partial x} \leq 0$	$\begin{cases} \arctan(\frac{\partial I(x,y)}{\partial y} / \frac{\partial I(x,y)}{\partial x}) - \pi \leq -\frac{\pi}{2}, & \text{if } y < y_0 \\ \theta(x,y) > 0, & \text{if } y > y_0 \end{cases}$	Jump discontinuity
$y > y_0, \quad \frac{\partial I(x,y)}{\partial x} \leq 0$	$\begin{cases} \theta(x,y) < 0, & \text{if } y < y_0 \\ \arctan(\frac{\partial I(x,y)}{\partial y} / \frac{\partial I(x,y)}{\partial x}) + \pi \geq \frac{\pi}{2}, & \text{if } y > y_0 \end{cases}$	Jump discontinuity

6. $\forall y : y \neq y_0, \quad \frac{\partial I(x,y)}{\partial x} > 0.$

Condition:	$\theta(x,y) =$	$\theta(x,y)$ has:
$y \neq y_0$	$\arctan(\frac{\partial I(x,y)}{\partial y} / \frac{\partial I(x,y)}{\partial x})$	Continuity

□

B Intensity Extrema Under Perspective Projection

Theorem 2 *Let $I_{orth}(x,y)$ be the intensity function produced by orthographic projection of a Lambertian surface $z(x,y)$ with constant albedo illuminated by a point light source (as described in Sect. 4.2.1):*

$$I_{orth}(x,y) = \rho \overrightarrow{N}(x,y) \cdot \overrightarrow{L} = \rho \cos(\alpha(x,y))$$

where ρ is the constant albedo; $\overrightarrow{N}(x,y)$, a normal to the 3D surface $z(x,y)$, $\overrightarrow{L}$, the light source direction (assumed constant); and $\alpha(x,y) = \angle(\overrightarrow{N}(x,y), \overrightarrow{L})$. Obviously, $\alpha(x,y) \in [0, \frac{\pi}{2}]$.

Let $I_{proj}(u,v)$ be a perspective projection of $z(x,y)$. This means that:

$$I_{pers}(u,v) = I_{orth}(x,y) = \rho \cos(\alpha(x,y))$$

If (x_0, y_0) is a point of local maximum of $I_{orth}(x,y)$, then (x_0, y_0) is also a local maximum of $I_{pers}(x,y)$.

Before we prove this theorem, we first introduce a lemma regarding the connection between (x,y) and (u,v).

Lemma 1 *Let (x,y,z) be a Cartesian coordinate system, and let (u,v) be the Cartesian coordinate system which results from the projection of a 3D surface $z(x,y)$ on an image plane $I_{pers}(x,y)$:*

$$u = -\frac{f\,x}{z(x,y)}$$

$$v = -\frac{f\,y}{z(x,y)}$$

where f is the focal length.

If there exists δ for which $\forall x, y : \mid x - x_0 \mid < \delta$ and $\mid y - y_0 \mid < \delta$, then necessarily there exists δ_2 for which $\forall u, v : \mid u - u_0 \mid < \delta_2$ and $\mid v - v_0 \mid < \delta_2$.

Proof:

To prove the lemma, we need to consider the bounds on $z(x,y)$ as well as on x and y themselves. $z(x,y)$, the 3D surface, has only positive values in the strong sense (because $z(x,y) = 0$ means that the surface touches the center of perspectivity). Therefore, $\exists M > 0$: $\forall x, y\ z(x,y) > M > 0$.

Because the image domain is finite and therefore bounded: $\exists A > 0$: $-A \leq x \leq A$, and $\exists B > 0$: $-B \leq y \leq B$.

Another property we employ is the continuity of $z(x,y)$ at point (x_0, y_0), or formally:

$$\forall \epsilon_1 > 0 \ \ \exists \delta_1 > 0 : \ \ \forall x, y : \mid x - x_0 \mid < \delta_1, \ \ \mid y - y_0 \mid < \delta_1 :$$

$$\mid z(x,y) - z(x_0, y_0) \mid < \epsilon_1$$

Equipped with these properties, we next show the required constraint on u. We denote: $z_0 = z(x_0, y_0)$.

$$\begin{aligned} u - u_0 &= \frac{x_0 f}{z_0} - \frac{x f}{z(x,y)} < f\frac{x_0(z_0 + \epsilon_1) - x z_0}{z_0 z(x,y)} \\ &= f\frac{(x_0 - x)z_0 + \epsilon_1 x_0}{z_0 z(x,y)} \leq f(\frac{\mid x_0 - x \mid}{z(x,y)} + \frac{x_0}{z_0 z(x,y)}\epsilon_1) \\ &< f(\frac{\mid x_0 - x \mid}{M} + \frac{x_0}{M^2}\epsilon_1) < f(\frac{\delta_1}{M} + \frac{A}{M^2}\epsilon_1) \end{aligned}$$

We denote: $\delta_2 = f(\frac{\delta_1}{M} + \frac{A}{M^2}\epsilon_1)$. We have: $u - u_0 < \delta_2$. The proof that $u - u_0 > -\delta_2$ is similar in nature and is therefore omitted. The proof that $\mid v - v_0 \mid < \delta_2$ is equivalent to the above one, and is also omitted.

□

Based on Lemma 1, we now prove Theorem 2:

Proof:

By definition of a local maximum at point (x_0, y_0), the orthographic projection image $I_{orth}(x,y)$ satisfies:

$$\exists \delta > 0 : \ \ \forall x, y : \ \mid x - x_0 \mid < \delta, \ \ \mid y - y_0 \mid < \delta : \ \ I_{proj}(x_0, y_0) > I_{proj}(x, y)$$

Now, according to Lemma 1, the last equation implies:

$$\exists \delta_2 > 0 : \quad \forall u, v : \ \mid u - u_0 \mid < \delta_2, \quad \mid v - v_0 \mid < \delta_2 : \quad I_{proj}(x_0, y_0) > I_{proj}(x, y)$$

However, by definition of the perspective projection $I_{pers}(u, v) = I_{orth}(x, y)$. Therefore,

$$\exists \delta_2 > 0 : \quad \forall u, v : \ \mid u - u_0 \mid < \delta_2, \quad \mid v - v_0 \mid < \delta_2 : \quad I_{pers}(u_0, v_0) > I_{proj}(u, v)$$

where (u_0, v_0) is the perspective projection of (u, v). This completes the proof.

□

Corollary 1 *Let $I_{pers}(u, v)$ be the intensity function produced by perspective projection of a constant albedo Lambertian surface $z(x, y)$ illuminated by a point light source at infinity (as described in Sect. 4.2.1):*

$$I_{pers}(u, v) = \rho \overrightarrow{N}(x, y) \cdot \overrightarrow{L} = \rho \cos(\alpha(x, y))$$

where ρ is the constant albedo; $\overrightarrow{N}(x, y)$, a normal to the 3D surface $z(x, y)$; $\overrightarrow{L}$, the light source direction (assumed constant); and $\alpha(x, y) = \angle(\overrightarrow{N}(x, y), \overrightarrow{L})$. $\alpha(x, y) \in [0, \frac{\pi}{2}]$.

If at point (x_0, y_0) the angle $\alpha(x, y)$ has a local minimum, then the intensity function $I_{pers}(u, v)$ has a local maximum at (x_0, y_0).

The proof of the corollary follows directly from the decreasing monotonicity of the cosine at $[0, \frac{\pi}{2}]$, and Theorem 2, which establishes the relation of the two coordinate systems (x, y) and (u, v).

Similar theorems also apply for the case of a local minimum of the intensity function.

The importance of these theorems and corollaries is that they relate the intensity features detected by Y_{arg} with the geometric features of the 3D surface $z(x, y)$ at the detected locations. The fact that Y_{arg} detects certain geometric features of the 3D surface explains its robustness.