Feature Tracking with Automatic Selection of Spatial Scales

Lars Bretzner and Tony Lindeberg

Computational Vision and Active Perception Laboratory (CVAP),
Department of Numerical Analysis and Computing Science,
KTH, S-100 44 Stockholm, Sweden.
Email: bretzner@bion.kth.se, tony@bion.kth.se

Revised version to appear in
Computer Vision and Image Understanding. vol. 71, pp. 385–392, Sept. 1998.

Abstract

When observing a dynamic world, the size of image structures may vary over time. This article emphasizes the need for including explicit mechanisms for automatic scale selection in feature tracking algorithms in order to: (i) adapt the local scale of processing to the local image structure, and (ii) adapt to size variations that may occur over time.

The problems of corner detection and blob detection are treated in detail, and a combined framework for feature tracking is presented in which the image features at every time moment are detected at locally determined and automatically selected scales. The integrated tracking algorithm has the ability to adapt to size variations, and can in this way overcome some of the inherent limitations of exposing fixed-scale tracking methods to image sequences in which the size variations are large. It is also shown how the stability over time of the scale descriptor is used as a part of a multi-cue similarity measure for matching.

Experiments on real-world sequences are presented showing the performance of the algorithm when applied to (individual) tracking of corners and blobs.

1 Introduction

Being able to track image structures over time is a useful and sometimes necessary capability for vision systems intended to interact with a dynamic world. There are several computer vision algorithms in which tracking arises as an important subproblem. Some situations are:

- Fixation of a physical point/region in the world over time.
- Object recognition in a dynamically varying environment.
- Motion segmentation and structure from motion computations.

There is an extensive literature on tracking methods operating without specific *a priori* knowledge about the world, such as object models or highly restricted domains. The work in this direction can be classified into three main categories: correlation based tracking, optical flow based tracking and feature tracking, the latter is described next.

Over the years a large number of approaches have been developed for tracking image features such as edges and corners over time. Essentially, what characterizes a feature tracking method is that image features are first extracted in a bottom-up processing step and then these features are used as the main primitives for the tracking and matching procedures. Concerning corner tracking, Shapiro *et al.* [1] detect and track corners individually in an algorithm originally aimed at applications such as videoconferencing. Smith and Brady [2] track a large set of corners and use the results in a flow-based segmentation algorithm. Zheng and Chellappa [3] have studied feature tracking when compensating for camera motion, and Gee and Cipolla [4] track locally darkest points with applications to pose estimation. In contour tracking, snakes can be used to track moving, deforming image features [5, 6] and such an approach is applied to estimate time-to-contact in [7]. Koller *et al.* [8] track combined motion and grey-level boundaries in traffic surveillance. An overview of different approaches to edge tracking can be found in the recent book by Faugeras [9].

The subject of this article is to consider the domain of feature tracking and to complement previous works on this subject by addressing the problem of scale and scale selection in the spatial domain and by introducing new similarity measures in the matching step. In most previous works, the analysis is performed at a single predetermined scale. Here, we will emphasize and show by examples why it is useful to include an explicit mechanism for automatic scale selection to be able to handle situations in which the size variations are large. Besides avoiding explicit setting of scale levels for feature detection, and thus overcoming some of the fundamental limitations of processing image sequences at a single scale, it will be demonstrated how scale levels selected by a scale selection procedure can constitute a useful source of information when defining a similarity measure over time, as well as for adapting the window size for correlation to the local image structure.

Moreover, since the resulting matching algorithm we will arrive at is based on a similarity measure defined as the combination of different discriminative properties, and with small modifications can be applied to tracking of both corners and blobs, we will emphasize this multi-cue aspect as an important component for increasing the robustness of feature tracking algorithms.

The presentation is organized as follows: Section 2 illustrates the need for adaptive scale selection in feature tracking. It gives a hands-on demonstration of the improvement in performance that can be obtained by including a scale selection mechanism when tracking features in image sequences in which the size variations over time are large. Section 3 describes the feature detection step and reviews the basic components in a general principle for scale selection. Sections 4 and 5 explain how the scale information obtained from these processing modules can be used in the prediction step and

in the evaluation of matching candidates. Section 6 summarizes how these components can be combined with a classical feature tracking scheme with prediction followed by detection and matching. Section 7 shows the performance of the algorithm when applied to real-world data. Feature tracking using adaptive scales is compared to tracking at one, fixed scale. Comparisons are also made between single-cue and multi-cue similarity measures. Finally, we conclude in section 8 by summarizing the main properties of the method and by outlining natural extensions.

2 The need for automatic scale selection in feature tracking

In an image sequence, the size of image structures may change over time due to expansions or contractions. A typical example of the former is when the observer approaches an object as shown in figure 1. The left column in this figure shows a few snapshots from a tracker which follows a corner on the object over time using a standard feature tracking technique with a fixed scale for corner detection and a fixed window size for hypothesis evaluation by correlation. After a number of frames, the algorithm fails to detect the right feature and the corner is lost. The reason why this occurs, is simply the fact that the corner no longer exists at the predetermined scale. As a comparison, the right column shows the result of incorporating a mechanism for adaptation of the scale levels to the local image structure (details will be given in later sections). As can be seen, the corner is correctly tracked over the whole sequence. (The same initial scale was used in both experiments.)

Another motivation to this work originates from the fact that all feature detectors suffer from localization errors due to e.g noise and motion blur. When detecting rigid body motion or recovering 3D structure from feature point correspondences in an image sequence, it is important that the motion in the scene is large compared to the localization errors of the feature detector. If the inter-frame motion is small, we therefore have to track features over a large number of frames to obtain accurate results. This requirement constitutes a key motivation for including a scale selection mechanism in the feature tracker, to obtain longer trajectories of corresponding features as input to algorithms for motion estimation and recovery of 3D structure.

3 Feature detection with automatic scale selection

A natural framework to use when extracting features from image data is to define the image features from multi-scale differential invariants expressed in terms of Gaussian derivative operators [10, 11], or more specifically, as maxima or zero-crossings of such entities [12]. In this way, image features such as corners, blobs, edges and ridges can be computed at any level of scale.

A basic problem that arises for any such feature detector concerns how to determine at what scales the image features should be extracted, or if the feature detection is performed at several scales simultaneously, what image features should be regarded as significant. A framework addressing this problem has been developed in [13, 12]. In summary, one of the main results from this work is a general principle for scale selection, which states that scale levels for feature detection can be selected from the scales at which normalized differential invariants assume maxima over scales. In this section, we shall give a brief review of how this methodology applies to the detection of features such as blobs and corners. The image features so obtained, with their associated attributes resulting from the scale selection method, will then be used as basic primitives for the tracking procedure.

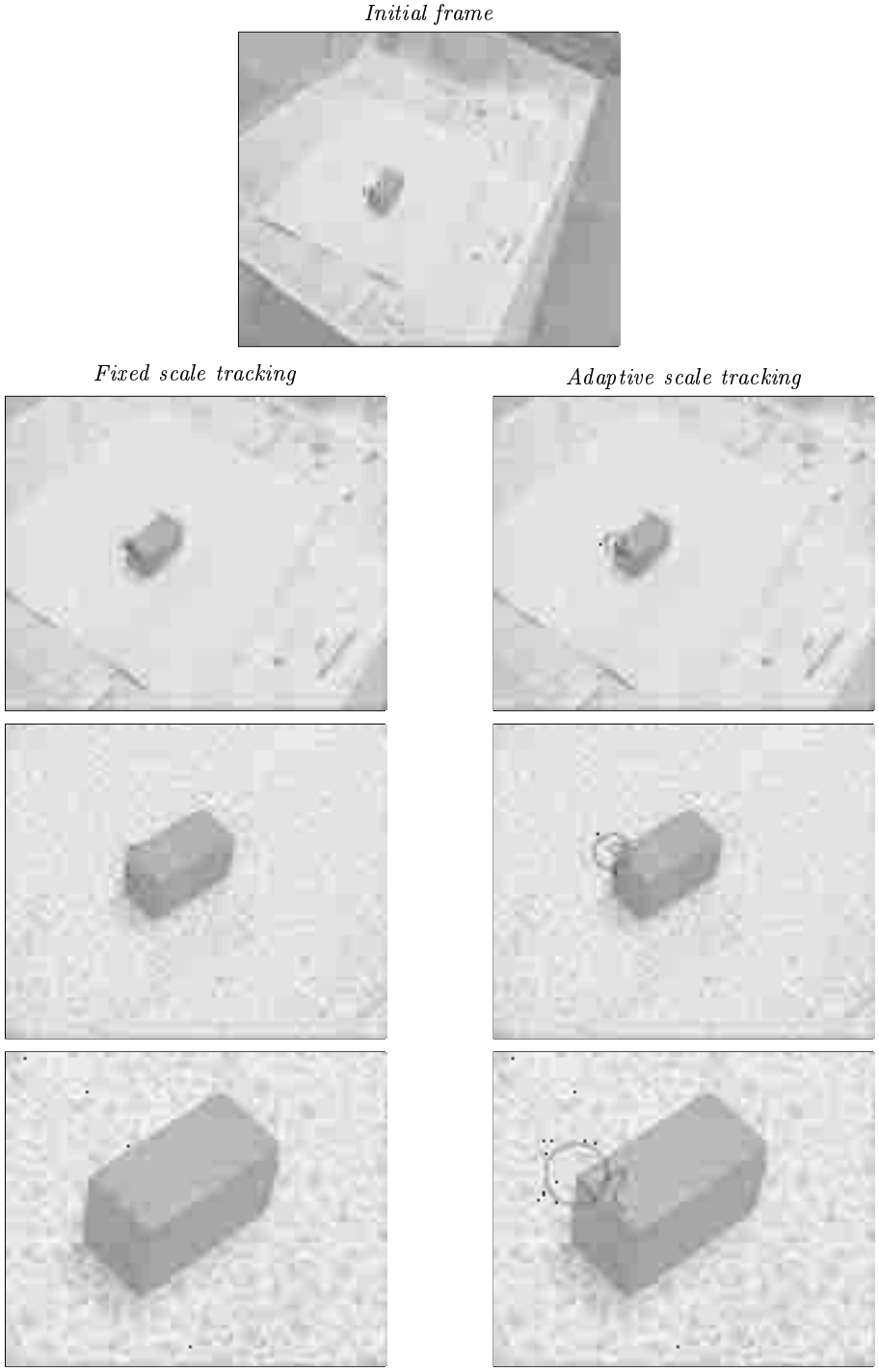

Figure 1: Illustration of the importance of automatic scale selection when tracking image structures over time. The corner is lost using detection at a fixed scale (left column), whereas it is correctly tracked using adaptive scale selection (right column). The size of the circles correspond to the detection scales of the corner features.

3.1 Normalized derivatives

The scale-space representation [14, 15] of a signal f is defined as the result of convolving f

$$L(\cdot;\ t) = g(\cdot;\ t) * f(\cdot) \tag{1}$$

with Gaussian kernels having different values of the scale parameter t

$$g(x;\ t) = \frac{1}{2\pi t} e^{-(x^2+y^2)/(2t)} \tag{2}$$

In this representation, γ-normalized derivatives [16] are defined by

$$\partial_\xi = t^{\gamma/2}\, \partial_x \tag{3}$$

where t is the variance of the Gaussian kernel. From this construction, a normalized differential invariant is then obtained by replacing all spatial derivatives by corresponding normalized derivatives according to (3).

3.2 Corner detection with automatic scale selection

A common way to define a corner in a grey-level image in differential geometric terms is as a point at which both the curvature of a level curve

$$\kappa = \frac{-\left(L_{yy} L_x^2 + L_{xx} L_y^2 - 2 L_x L_y L_{xy}\right)}{\left(L_x^2 + L_y^2\right)^{3/2}} \tag{4}$$

and the gradient magnitude

$$|\nabla L| = \sqrt{L_x^2 + L_y^2} \tag{5}$$

are high [17, 18, 19, 20]. If we consider the product of κ and the gradient magnitude raised to some power, and choose the power equal to three, we obtain the essentially affine invariant expression

$$\tilde{\kappa} = L_{yy} L_x^2 + L_{xx} L_y^2 - 2 L_x L_y L_{xy} \tag{6}$$

with its corresponding γ-normalized differential invariant

$$\tilde{\kappa}_{\gamma-norm} = t^{2\gamma} \tilde{\kappa} \tag{7}$$

In [21] it is shown how a junction detector with automatic scale selection can be formulated in terms of the detection of *scale-space maxima* of $\tilde{\kappa}^2_{\gamma-norm}$, *i.e.*, by detecting points in scale-space where $\tilde{\kappa}^2_{\gamma-norm}$ assumes maxima with respect to both scale and space. When detecting image features at coarse scales it turns out that the localization can be poor. Therefore, this detection step is complemented by a second localization stage, in which a modified Förstner operator [22], is used for iteratively computing new localization estimates using scale information from the initial detection step (see the references for details).

A useful property of this corner detection method is that it leads to selection of coarser scales for corners having large spatial extent.

3.3 Blob detection with automatic scale selection

As shown in the abovementioned references, a straightforward method for blob detection can be formulated in an analogous manner by detecting scale-space maxima of the square of the normalized Laplacian

$$\nabla^2_{norm} L = t\,(L_{xx} + L_{yy}) \tag{8}$$

This operator gives a strong response for blobs that are brighter or darker than their background, and in analogy with the corner detection method, the selected scale levels provide information about the characteristic size of the blob.

4 Tracking and prediction in a multi-scale context

When tracking features over time, both the position of the feature and the appearance of its surrounding grey-level pattern can be expected to change. To relate features over time, we shall throughout this work make use of the common assumption about small motions between successive frames.

There are several ways to predict the position of a feature in the next frame based on its positions in previous frames. Whereas the Kalman filtering methodology has been commonly used in the computer vision literature, this approach suffers from a fundamental limitation if the motion direction suddenly changes. If a feature moving in a certain direction has been tracked over a long period of time, then the built-in temporal smoothing of the feature trajectory in the Kalman filter, implies that the predictions will continue to be in essentially the same direction, although the actual direction of the motion changes. If the covariance matrices in the Kalman filter have been adapted to small oscillations around the previously smooth trajectory, it will hence be likely that the feature is lost at the discontinuity. For this reason, we shall make use of a simpler first-order prediction, which uses the motion between the previous two successive frames as a prediction to the next frame.

Within a neighbourhood of each predicted feature position, we detect new features using the corner (or blob) detection procedure with automatic scale selection. The support regions associated with the features serve as natural regions of interest when searching for new corresponding features in the next frame. In this way, we can avoid the problem of setting a global threshold on the distance between matching candidates. There is, of course, a certain scaling factor between the detection scale and the size of the support region. The important property of this method, however, is that it will automatically select smaller regions of interest for small-size image structures, and larger search regions for larger size structures. Here, we shall make use of this scale information for three main purposes:

- Setting the search region for possible matching candidates.
- Setting the window size for correlation matching.
- Using the stability of the detection scale as a matching condition.

We set the size of the search region to the spatial extent of the previous image feature, multiplied by a safety factor. Within this window, a certain number of candidate matches are selected. Then, an evaluation of these matching candidates is made based on a combined similarity measure to be defined in the next section.

5 Matching on multi-cue similarity

Based on the assumption of small inter-frame image motions, we use a multiple cue approach to the feature matching problem. Instead of evaluating the matching candidates using a correlation measure on a local grey-level patch only, as done in most feature tracking algorithms, we combine the correlation measure with significance stability, scale stability and proximity measures as defined below.

Patch similarity. This measure is a normalized Gaussian-weighted intensity cross-correlation between two image patches. Here, we compute this measure over a square centered at the feature and with its size set from the detection scale. The measure is derived from the cross-correlation of the image patches, see [23], computed using a Gaussian weight function centered at the feature. The motivation for using a Gaussian weight function is that image structures near the feature center should be regarded as more significant than peripheral structures. Given two brightness functions I_A and

I_B, and two image regions $D_A \subset \mathbb{R}$ and $D_B \subset \mathbb{R}$ of the same size $|D| = |D_A| = |D_B|$ centered at p_A and p_B respectively, the weighted cross-correlation between the patches is defined as:

$$C(A,B) = \frac{1}{|D|} \sum_{x \in D_A} e^{-(x-p_A)^2} I_A(x)\, I_B(x - p_A + p_B) - \frac{1}{|D|^2} \sum_{x_A \in D_A} e^{-(x-p_A)^2} I_A(x_A) \sum_{x_B \in D_B} e^{-(x-p_B)^2} I_B(x_B) \tag{9}$$

and the normalized weighted cross-correlation is

$$S_{patch}(A,B) = \frac{C(A,B)}{\sqrt{C(A,A)\, C(B,B)}} \tag{10}$$

where

$$C(A,A) = \frac{1}{|D|} \sum_{x \in D_A} (e^{-(x-p_A)^2} I_A(x))^2 - \frac{1}{|D|^2} \Big(\sum_{x \in D_A} e^{-(x-p_A)^2} I_A(x) \Big)^2 \tag{11}$$

and $C(B,B)$ is defined analogously. As is well-known, this similarity measure is invariant to superimposed linear illumination gradients. Hence, first-order effects of scene lightning do not affect this measure, and the measure only accounts for changes in the structure of the patches.

Significance stability. A straightforward significance measure of a feature detected according to the method described in section 3 is the normalized response at the local scale-space maximum. For corners, this measure is the normalized level curve curvature according to (7) and for blobs it is the normalized Laplacian according to (8). To compare significance values over time, we measure similarity by relative differences instead of absolute, and define this measure as

$$S_{sign} = |\log \frac{R_B}{R_A}| \tag{12}$$

where R_A and R_B are the significance measures of the corresponding features A and B.

Scale stability. Since the features are detected at different scales, the ratio between the detection scales of two features constitutes a measure of stability over scales. To measure relative scale variations, we use the absolute value of the logarithm of this ratio, defined as

$$S_{scale} = |\log \frac{t_B}{t_A}| \tag{13}$$

where t_A and t_B are the detection scales of A and B.

Proximity We measure how well the position x_A of feature A corresponds to the position x_{pred} predicted from feature B

$$S_{pos} = \frac{\|x_A - x_{pred}\|}{\sqrt{t_B}} \tag{14}$$

where t_B is the detection scale of feature B.

Combined similarity measure. In summary, the similarity measure we make use of a weighted sum of (10), (12) and (13),

$$S_{comb} = c_{patch} S_{patch} + c_{sign} S_{sign} + c_{scale} S_{scale} + c_{pos} S_{pos} \tag{15}$$

where c_{patch}, c_{sing}, c_{scale} and c_{pos} are tuning parameters to be determined.

6 Combined tracking algorithm

By combining the components described in the previous sections, we obtain a feature tracking scheme based on a traditional predict-detect-update loop. In addition, the following processing steps are added:

- *Quality measure.* Each feature is assigned a quality measure indicating how stable it is over time.
- *Bidirectional matching.* To provide additional information to later processing stages about the reliability of the matches, the matching can be done bidirectionally. Given a feature F_1 from the feature set, we first compute its winning matching candidate F_2 in the current image. If then F_1 is the winning candidate of F_2 in the backward matching direction, the match between F_1 and F_2 is registered as safe. This processing step is useful for signalling possible matching errors.

During the tracking procedure each feature is associated with the following attributes:

- its detection scale t_{det},
- its estimated size $D = k_{size} * \sqrt{t_{det}}$ bounded from below to D_{min},
- its position,
- its quality value.

An overview of the tracking algorithm is given in figure 2. At a more detailed level, each individual module operates as follows:

Prediction The prediction is performed as described in section 4. For each feature in the feature set, a linear prediction of the position in the current frame is computed based on the positions of the corresponding feature in the two previous frames. The size of the search window is computed as $k_{w1} * D$ (with the size D bounded from below). When a trajectory is initiated, there is no feature history to base the prediction on, so we use a larger search window of size $k_{w2} * D$ ($k_{w2} > k_{w1}$) and use the original feature position as the predicted position.

Detection In each frame, image features are detected as described in section 3. The window obtained from the prediction step is searched for the same kind of features over a locally adapted range of scales $[t_{min}, t_{max}]$, where $t_{max} = k_{range} * t_{det}$ and $t_{min} = t_{det}/k_{range}$. The number n of detected candidates depends on which feature extraction method we use in the detection step.

Matching The matching is based on the similarity measures described in section 5. The original feature is matched to the candidates obtained from the detection step and the winner is the feature having the highest combined similarity value above a fixed threshold T_{comb} and a patch correlation value above a threshold T_{patch}. These thresholds are necessary to suppress false matches when features disappear due to e.g occlusion.

If a feature is matched, the quality value is increased by dq_i and its position, its scale descriptor, its significance value and its grey-level patch are updated.

If no match is found, the feature is considered unmatched, its quality value is decreased by dq_d and its position is set to the predicted position.

Finally for each frame, the feature set is parsed to detect feature merges and to remove features having quality values below a threshold T_q. When two features merge, their trajectories are terminated and a new trajectory is initiated. In this way, we obtain more reliable feature trajectories for further processing.

Algorithm:

For each frame:

For each feature F in the feature set:

1. **Prediction**
   - 1.1 Predict the position of the feature F in the current frame based on information from the previous frames.
   - 1.2 Compute the search region in the current frame based on information from the previous frames and the scale of the feature.
2. **Detection**

   Detect n candidates C_k over a reduced set of scales in the region of interest in the current frame.
3. **Matching**
   - 3.1 Match every candidate C_k to the feature F and find the best match using the combined similarity measure.
   - 3.2 Optionally, perform bidirectional matching to register safe matches.
   - 3.3 Compare the similarity value to a predetermined threshold:
     If above: consider the feature as matched; update its position, its scale descriptor, its significance value, its grey-level patch and increase its quality value.
     If below: consider the feature as unmatched; update its position to the predicted position and decrease its quality value.

Parse the feature set to detect feature merges and remove features having quality values below a certain threshold.

Figure 2: Overview of the feature tracking algorithm.

7 Experimental results

7.1 Corner tracking

Let us first demonstrate the performance of the algorithm when applied to an image sequence consisting of 60 frames. In this sequence, the camera moves in a fairly complex way relative to a static scene. The objects of interest on which the features (here corners) are detected are a telephone and a package on a table. From the junctions detected in the initial frame, a subset of 14 features were selected manually. Figure 3 shows the initial frame and the situation after 30, 50 and 60 frames. In the illustrations, black segments on the trajectories indicate matched positions, while white segments show unmatched (predicted) positions. The matching is based on the combined similarity measure incorporating patch correlation, scale stability, significance stability and proximity. The detection scales of the features are illustrated by the size of the circles in the images, and we see how all corners are detected at fine scales in the initial frame. The left column shows the result when using automatic scale selection. As time evolves, the detection scales adapt to the size changes of the image structures; tracked sharp corners are still detected at fine scales while blunt corners are detected at coarser scales when the camera approaches the scene.

The right column of figure 3 shows the result of an attempt to track the same corners at fixed scales, using the automatically determined detection scales from the initial image. As can be seen, the sharpest corners are correctly tracked but the blunt corners are inevitably lost. This effect is similar to the initial illustration in section 2.

Figure 4 shows another example for a camera tracking a toy train on a table. In the initial frame, 29 corners were selected manually; 25 on the train and 4 on an object in the background. Some of these corners are enumerated and will be referred to when discussing the performance below.

Corner no	*Patch similarity only*	*Combined similarity measure*
1	lost in frame 29	lost in frame 29
2	mismatched in 18	mismatched in 18
3	mismatched in 16	mismatched in 16
4	lost in 83	—
5	mismatched in 63	—
6	lost in 81	lost in 75
7	lost in 33	—
8	lost in 46	lost in 46

Table 1: Table showing when eight of the enumerated corners in the train sequence are lost. Note that out of the corners which are lost when matching on patch similarity only, three corners are tracked during the whole sequence when using the combined similarity measure.

The left column of figure 4 shows the situation after 60 and 100 frames, using the combined similarity measure in the matching step. Noisy image data and motion blur will increase the number of matching failures. Corners no 2, 3, 6 and 8 are lost due to moving structures in the background causing accidental views. The importance of using the combined similarity measure in the matching step is illustrated in the right column, showing the result of matching on patch correlation only. We see that corners no 4, 5, and 7, which were all tracked using the combined similarity measure, now are lost. Table 1 shows, for both experiments, when the enumerated corners in the train sequence are lost.

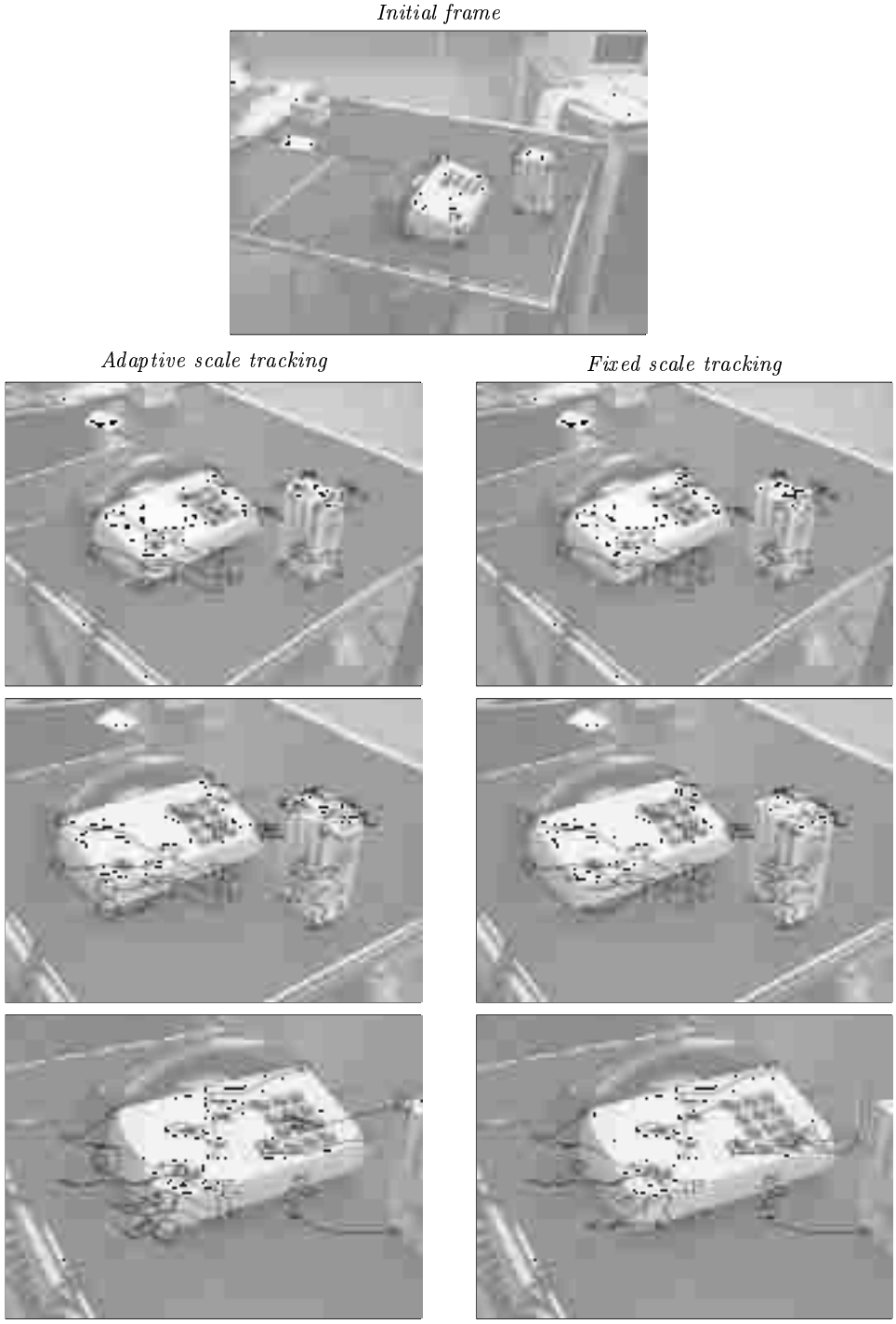

Figure 3: **Top:** The initial frame of the phone sequence with 14 detected corners. **Left:** Corner tracking using adaptive scale selection: the tracked corners after 30, 50 and 60 frames. All corners are correctly tracked. **Right:** Using fixed scales over time: note that the blunt corners are lost compared to the adaptive scale tracking in the left column.

7.2 Blob tracking

Let us now apply the same framework for blob tracking. In the train sequence, we manually selected 11 blobs on the train and 2 blobs in the background in the initial

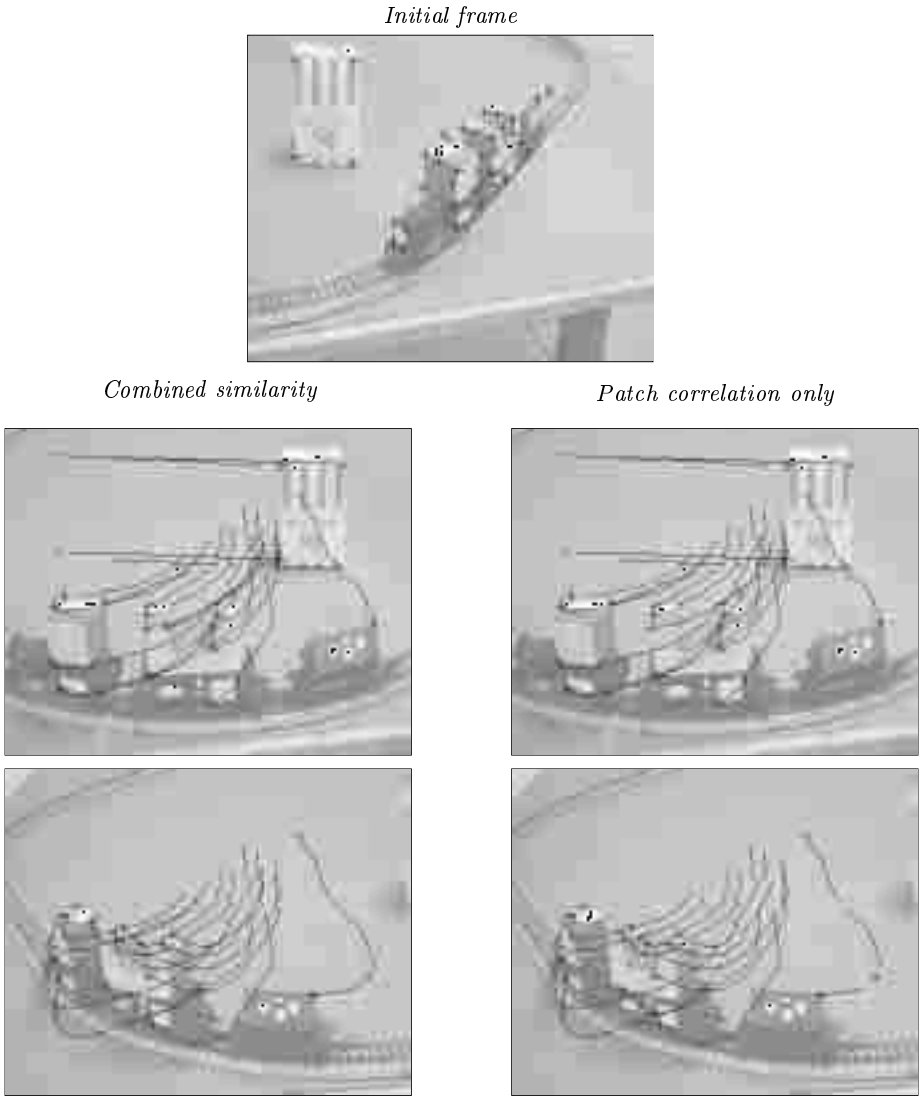

Figure 4: **Top:** The initial frame of the train sequence with 29 detected corners. **Left:** Corner tracking using adaptive scale selection and matching on combined similarity: the tracked corners after 60 and 100 frames. **Right:** Using adaptive scale selection and matching the candidates on patch correlation only: three more corners are lost as compared to the left column.

frame. Figure 5 shows the initial frame and the situation after 30, 90 and 150 frames. The size of the circles in the figures correspond to the detection scales of the blobs. Note in the left column how the detection scale adapts to the local image structure when the blobs undergo expansion followed by contraction. All visible blobs except one are tracked during the whole sequence.

Referring to the need for automatic scale selection in feature tracking, as advocated in section 2, it is illustrative to show the results of attempting blob tracking with feature detection at a fixed scale. The scale level for detecting each blob was automatically selected in the first frame and was then kept fixed throughout the sequence. The right column of figure 5 shows how the tracker has severe problems due to the expansion

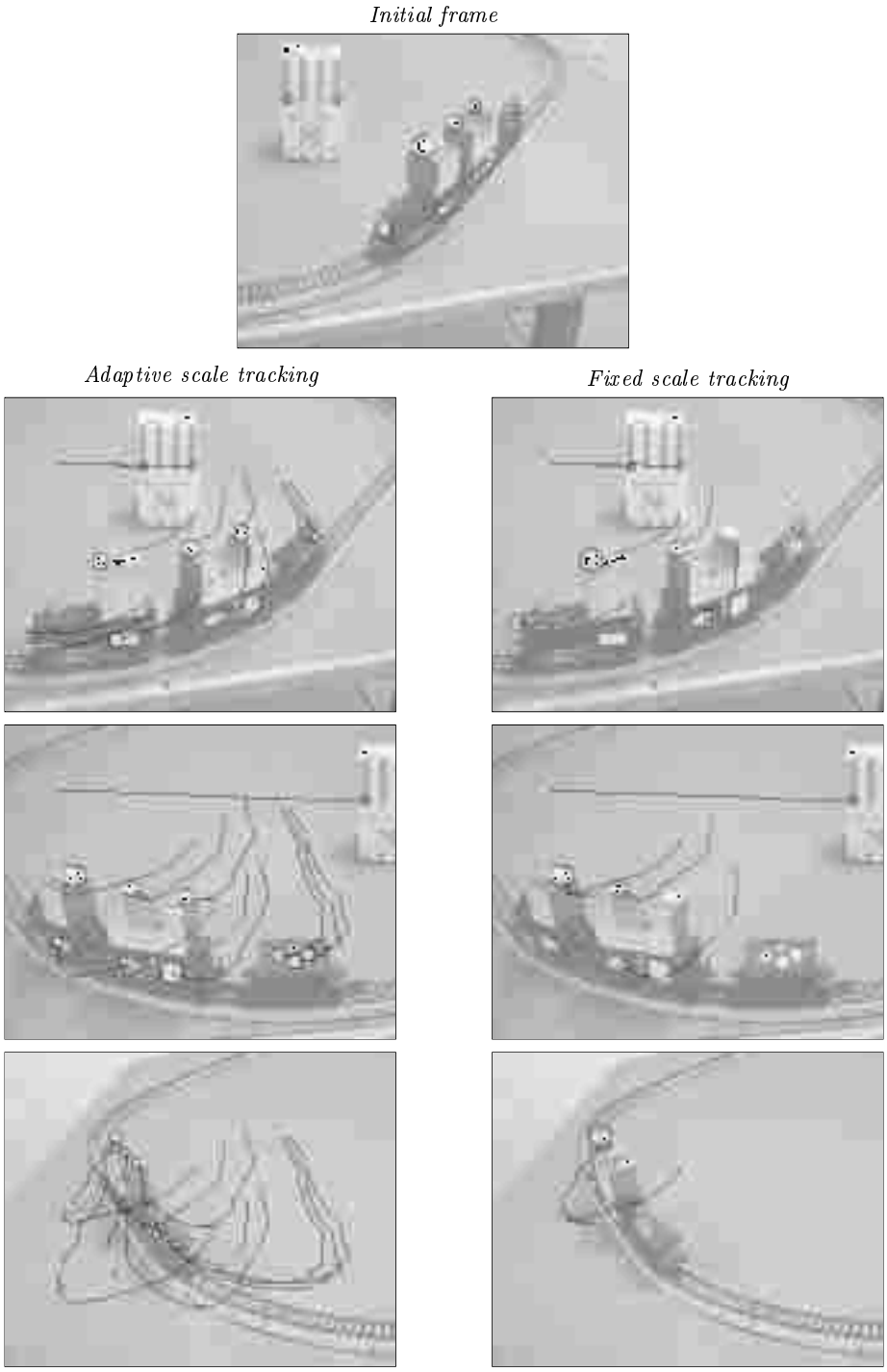

Figure 5: **Top:** The initial frame of the train sequence with 13 detected blobs. **Left:** Blob tracking using adaptive scale selection: the tracked blobs after 30, 90 and 150 frames. All blobs are correctly tracked. **Right:** Using fixed scales in the detection procedure: only one blob is correctly tracked over the whole sequence.

and contraction in the sequence.

As a further illustration of the capability of the algorithm to track blobs under

Figure 6: The initial frame of the shirt sequence with the 20 strongest blobs detected in a rectangular window and the situation after 25, 50 and 87 frames using automatic scale selection. Note how the scales, illustrated by the size of the circles, adapt to the size changes of the image structures.

large size changes we applied it to a sequence of 87 images where a person, dressed in a spotted shirt, approaches the camera. In a rectangular area in the initial frame, the

Figure 7: Tracking with fixed scales over time: most blobs are already lost after 25 frames because they no longer exist at the initially chosen scale.

20 most significant blobs were automatically detected. Figure 6 shows the initial frame and the results after 25, 50 and 87. We see that all blobs are correctly tracked over the entire sequence using automatic scale selection. When trying to track the blobs at a fixed scale, as shown in figure 7, most of the blobs are lost already after 25 frames. The last correctly tracked blob is lost after about 50 frames.

In summary, these experiments show that similar qualitative properties hold for blob tracking and for junction tracking: (i) By including the significance values and the selected scale levels in the matching criterion, we obtain a better performance than when matching on grey-level correlation only. (ii) The performance of tracking at adaptively determined scale levels is superior compared to similar tracking at a fixed scale.

Let us finally illustrate how feature tracking with automatic scale selection over a large number of frames is likely to give us trajectories which correspond to reliable and stable physical scene points or regions of interest on objects. By explicitly registering the features that are stable over time, we are able to suppress spurious feature responses due to noise, temporary occlusions etc. Figure 8 shows snapshots from a sequence in which the 10 most significant blobs in a region around the face of the subject have been detected and tracked. The subject first approaches the camera and then moves back to the initial position. The figure shows the situation after 20, 45 and 90 frames. We can see that after a while only four features remain in the feature set and these are the stable features corresponding to the nostrils and the eyes. This ability to register stable image structures over time is clearly a desirable quality in many computer vision applications. Notably, for general scenes with large expansions or contractions, a scale selection mechanism is essential to allow for such registrations.

8 Summary and Discussion

We have presented a framework for feature tracking in which a mechanism for automatic scale selection has been built into the feature detection stage and the additional attributes of the image features obtained from the scale selection module are used for guiding the other processing steps in the tracking procedure.

We have argued that such a mechanism is essential for any feature tracking procedure intended to operate in a complex environment, in order to adapt the scale of processing to the size variations that may occur in the image data as well as over time. If we attempt to track features by processing the image data at one single scale only, we can hardly expect to be able to follow the features over large size variations. This property is a basic consequence of the inherent multi-scale nature of image structures, which means that a given object may appear in different ways depending on the scale of observation.

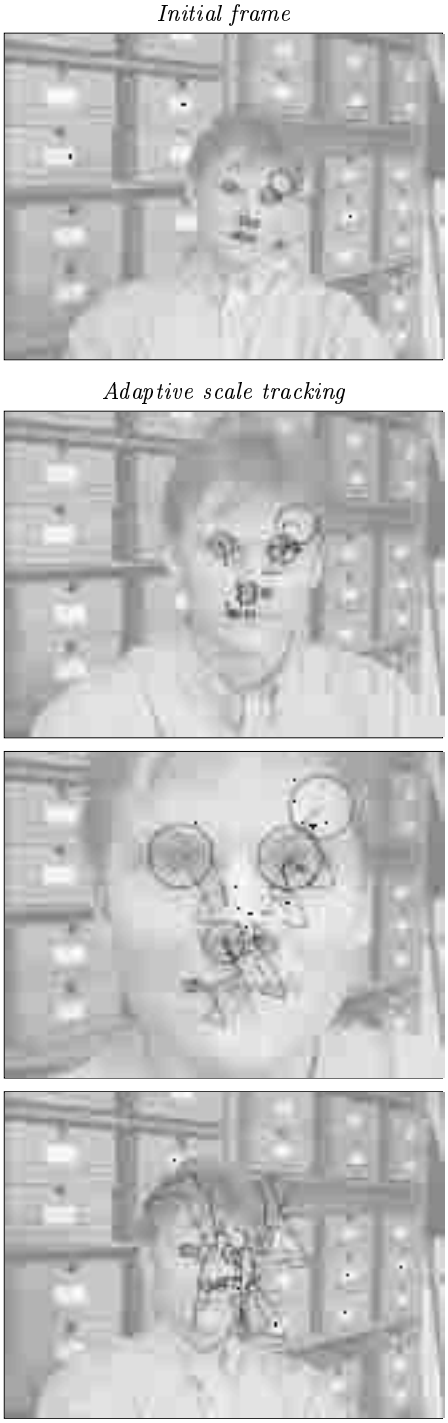

Figure 8: Blob tracking in the face sequence using automatic scale selection, the initial frame and the situation after 20, 45 and 90 frames. Note that blob responses that are unstable over time are suppressed.

Specifically, based on a previously developed feature detection framework with au-

tomatic scale selection, we have presented a scheme for tracking corners and blobs over time in which:

- the image features at any time moment are detected using a feature detection method with automatic scale selection, and
- this information is used for
  - guiding the detection and selection of new feature candidates,
  - providing context information for the matching procedure,
  - formulating a similarity measure for matching features over time.

Besides avoiding explicit selection of scale levels for feature detection, the feature detection procedure with automatic scale selection allows us to track image features over large size variations. As demonstrated in the introductory example in section 2, we can in this way obtain a substantial improvement in the performance relative to a fixed-scale feature tracker.

Since the scale levels obtained from the scale selection procedure reflect the spatial extent of the image structures, we can also use this context information for avoiding explicit settings of distance thresholds and predefined window sizes for matching. Moreover, by including the scale and significance information associated with the image features from the scale selection procedure into a multi-cue similarity measure, we showed how we in this way can improve the reliability of the low-level matching procedure.

Of course, there are inherent limitations in tracking each feature individually as done in this work, and as can be seen from the examples, there are a number of situations where the tracking algorithm fails. Typically, this occurs because of rapid changes in the local grey-level pattern around the corner, corresponding to violations of the assumption about small inter-frame motions.

A notable conclusion that can be made in this context, is that despite these limitations, we have shown by examples that the resulting tracking procedure is able to track most of the visible features that can be followed over time in the sequences presented in this article. By this we argue that the type of framework presented here provides an important step towards overcoming some of the limitations in previous feature tracking algorithms.

8.1 Spatial consistency and statistical evaluation.

In the scheme presented so far, each feature is tracked *individually*, without any explicit notion of coherently moving clusters. It is obvious that the performance of a tracking method can be improved if the latter notion can be introduced, and the overall motion of the clusters can be used for generating better predictions, as well as more refined evaluation criteria of matching candidates. To investigate if the motions of the tracked features possibly correspond to the same rigid body motion, we might compute descriptors such as affine 3-D coordinates. Interesting work in this direction have been presented in [24, 25, 26].

It is also natural to include a statistical evaluation of the reliability of matches as well as their possible agreement with different clusters, as done in [26]. Whereas such an approach has not been explored in this work, this should not be interpreted as implying that the scale selection method excludes the usefulness of a statistical evaluation. The main intention behind this work has been to explore how far it is possible to reach by using a bottom-up construction of feature trajectories and by including a mechanism for automatic scale selection in the feature detection step. Then, the intention is that these two approaches should be applied in a complementary manner, where the scale

selection method serves as a pre-conditioner for generating more reliable hypotheses with more reliable input data. The scale selection method can also provide context information over what domains statistical evaluations should be made.

8.2 Multi-cue tracking

A tracking method based on a single visual cue, like those reviewed in section 1 may have a rather good performance under certain conditions but may fail in more complex scenes. In this context, a multi-cue approach to the tracking problem is natural, i.e a system in which several types of algorithms operate simultaneously and the algorithm most suitable to a given situation dominates. This means that the vision system must have the ability to evaluate the reliability of the various tracking methods and to switch between them in an appropriate way.

Initial work in this direction, combining disparity cues with optical flow based object segmentation, has been performed by Uhlin *et al.* [27]. The approach developed here lends itself naturally to integration with such techniques, in which such cues can be used for evaluating candidate feature clusters, and the feature tracking module in turn can be used as a more refined processing mechanism for maintaining object hypotheses over time. Of course, this leads to basic problems of feature selection. One possible approach for addressing such problems has been presented by Shi and Tomasi [28].

8.3 Temporal consistency

As a final remark it is worth pointing out that in this work, the image features in each frame have been extracted *independently* from each other and without any other explicit use of temporal consistency than the heuristic condition that a feature hypothesis is allowed to survive over a few frames. To make more explicit use of temporal consistency, it is natural to incorporate the notion of a temporal scale-space representation [29] and to include scale selection over the temporal scale domain as well [30].

In this context, it is also natural to combine the feature tracking approach with a simultaneous calculation of optical flow estimates and to integrate these two approaches so as to make use of their relative advantages. These subjects, including the integration of multiple tracking techniques into a multi-cue framework, constitute major goals of our continued research.

References

[1] L. S. Shapiro, H. Wang, and J. M. Brady, "A matching and tracking strategy for independently moving objects," in *Proc. British Machine Vision Conference*, pp. 306–315, Springer Verlag, Berlin, 1992.

[2] S. M. Smith and J. M. Brady, "Asset-2: Real-time motion segmentation and shape tracking," *IEEE Trans. Pattern Analysis and Machine Intell.*, vol. 17, no. 8, pp. 814–820, 1995.

[3] Q. Zheng and R. Chellappa, "Automatic feature point extraction and tracking in image sequences for arbitrary camera motion," *International Journal of Computer Vision*, vol. 15, no. 1, pp. 31–76, 1995.

[4] A. Gee and R. Cipolla, "Fast visual tracking by temporal consensus," Tech. Rep. CUED/F-INFENG/TR207, Dept of Engineering, University of Cambridge, England, 1995.

[5] Blake *et al.*, "Affine-invariant contour tracking with automatic control of spatiotemporal scale," in *Proc. 4th International Conference on Computer Vision* (H.-H. Nagel, ed.), (Berlin, Germany), IEEE Computer Society Press, 1993.

[6] Curwen *et al.*, "Parallel implementation of Lagrangian dynamics for real-time snakes," in *Proc. British Machine Vision Conference*, Springer Verlag, Berlin, 1991.

[7] R. Cipolla and A. Blake, "Surface orientation and time to contact from image divergence and deformation," in *Proc. 2nd European Conference on Computer Vision* (G. Sandini, ed.), (Santa Margherita Ligure, Italy), pp. 187–202, Springer Verlag, Berlin, 1992.

[8] D. Koller, J. Weber, and J. Malik, "Robust multiple car tracking with occlusion reasoning," in *Proc. 3rd European Conference on Computer Vision* (J.-O. Eklundh, ed.), (Stockholm, Sweden), pp. 189–196, Springer Verlag, Berlin, 1994.

[9] O. Faugeras, *Three-dimensional computer vision*. Cambridge, Massachusetts: MIT Press, 1993.

[10] J. J. Koenderink and A. J. van Doorn, "Generic neighborhood operators," *IEEE Trans. Pattern Analysis and Machine Intell.*, vol. 14, pp. 597–605, Jun. 1992.

[11] L. M. J. Florack, B. M. ter Haar Romeny, J. J. Koenderink, and M. A. Viergever, "Scale and the differential structure of images," *Image and Vision Computing*, vol. 10, pp. 376–388, Jul. 1992.

[12] T. Lindeberg, *Scale-Space Theory in Computer Vision*. The Kluwer International Series in Engineering and Computer Science, Dordrecht, Netherlands: Kluwer Academic Publishers, 1994.

[13] T. Lindeberg, "On scale selection for differential operators," in *Proc. 8th Scandinavian Conf. on Image Analysis* (K. H. K. A. Høgdra, B. Braathen, ed.), (Tromsø, Norway), pp. 857–866, Norwegian Society for Image Processing and Pattern Recognition, May. 1993.

[14] A. P. Witkin, "Scale-space filtering," in *Proc. 8th Int. Joint Conf. Art. Intell.*, (Karlsruhe, Germany), pp. 1019–1022, Aug. 1983.

[15] J. J. Koenderink, "The structure of images," *Biological Cybernetics*, vol. 50, pp. 363–370, 1984.

[16] T. Lindeberg, "Edge detection and ridge detection with automatic scale selection," in *Proc. IEEE Comp. Soc. Conf. on Computer Vision and Pattern Recognition, 1996*, (San Francisco, California), pp. 465–470, IEEE Computer Society Press, June 1996.

[17] L. Kitchen and A. Rosenfeld, "Gray-level corner detection," *Pattern Recognition Letters*, vol. 1, no. 2, pp. 95–102, 1982.

[18] J. J. Koenderink and W. Richards, "Two-dimensional curvature operators," *J. of the Optical Society of America*, vol. 5:7, pp. 1136–1141, 1988.

[19] R. Deriche and G. Giraudon, "Accurate corner detection: An analytical study," in *Proc. 3rd Int. Conf. on Computer Vision*, (Osaka, Japan), pp. 66–70, 1990.

[20] J. Blom, *Topological and Geometrical Aspects of Image Structure*. PhD thesis, Dept. Med. Phys. Physics, Univ. Utrecht, NL-3508 Utrecht, Netherlands, 1992.

[21] T. Lindeberg, "Junction detection with automatic selection of detection scales and localization scales," in *Proc. 1st International Conference on Image Processing*, vol. I, (Austin, Texas), pp. 924–928, IEEE Computer Society Press, Nov. 1994.

[22] W. A. Förstner and E. Gülch, "A fast operator for detection and precise location of distinct points, corners and centers of circular features," in *Proc. Intercommission Workshop of the Int. Soc. for Photogrammetry and Remote Sensing*, (Interlaken, Switzerland), 1987.

[23] L. S. Shapiro, H. Wang, and J. M. Brady, "A corner matching and tracking strategy applied to videophony," Tech. Rep. OUEL 1933/92, Robotics Research Group, University of Oxford, 1992.

[24] I. D. Reid and D. W. Murray, "Tracking foveated corner clusters using affine structure," in *Proc. 4th International Conference on Computer Vision* (H.-H. Nagel, ed.), (Berlin, Germany), pp. 76–83, IEEE Computer Society Press, 1993.

[25] C. S. Wiles and M. Brady, "Closing the loop on multiple motions," in *Proc. 5th International Conference on Computer Vision*, pp. 308–313, IEEE Computer Society Press, 1995.

[26] L. S. Shapiro, *Affine analysis of image sequences.* Cambridge, England: Cambridge University Press, 1995.

[27] T. Uhlin, P. Nordlund, A. Maki, and J.-O. Eklundh, "Towards an active visual observer," in *Proc. 5th Int. Conf. on Computer Vision*, (Cambridge, MA), pp. 679–686, June 1995.

[28] J. Shi and C. Tomasi, "Good features to track," in *Proc. IEEE Comp. Soc. Conf. on Computer Vision and Pattern Recognition*, pp. 593–600, IEEE Computer Society Press, 1994.

[29] T. Lindeberg and D. Fagerström, "Scale-space with causal time direction," in *Proc. 4th European Conf. on Computer Vision*, vol. 1064, (Cambridge, UK), pp. 229–240, Springer Verlag, Berlin, Apr. 1996.

[30] T. Lindeberg, "On automatic selection of temporal scales in time-casual scale-space," in *Proc. AFPAC'97: Algebraic Frames for the Perception-Action Cycle* (G. Sommer and J. J. Koenderink, eds.), vol. 1315 of *Lecture Notes in Computer Science*, (Kiel, Germany), pp. 94–113, Springer Verlag, Berlin, Sept. 1997.

[31] T. Lindeberg, "Scale selection for differential operators," report, ISRN KTH/NA/P--94/03--SE, Dept. of Numerical Analysis and Computing Science, KTH, Stockholm, Sweden, Jan. 1994. Extended version in *Int. J. of Computer Vision*, vol 30, number 2, 1998. (In press).

[32] T. Lindeberg, "Feature detection with automatic scale selection," *Int. J. of Computer Vision*, vol. 30, no. 2, pp. 77–116, 1998.

[33] T. Lindeberg, "Edge detection and ridge detection with automatic scale selection," *Int. J. of Computer Vision*, vol. 30, no. 2, pp. 117–154, 1998.

A Algorithmic details

This appendix gives a detailed listing of the parameters that influence the algorithm as well as the parameter settings that have been used for generating the experiments.

A.1 Prediction

The parameters determining the size of the search window (see section 6) were

$$k_{size} = 5$$
$$k_{w1} = 1.5$$
$$k_{w2} = 2 * k_{w1}$$
$$D_{min} = 16$$

A.2 Feature detection

When detecting features with automatic scale selection, the following scale ranges were used in the initial frame:

Junction detection	*Blob detection*
$t_{min} = 4.0$	$t_{min} = 4.0$
$t_{max} = 256.0$	$t_{max} = 512.0$

and the parameter γ in the normalized derivative concept (see section 3) was set to:

Junction detection	*Blob detection*
$\gamma = 0.875$	$\gamma = 1$

When searching for new image features, the search for matching candidates to a feature detected at scale t_{det} was performed in the interval $[t_{det}/k_1, t_{det}k_1]$, where $k_{range} = 3$.

In all experiments, the sampling density in the scale direction was set to correspond to a minimum of 5 scale levels per octave. In all other aspects, the feature detection algorithms followed the default implementation of junction and blob detection with automatic scale selection described in [12, 31, 32, 33]. The maximum number of matching candidates evaluated for each feature was:

Junction detection	*Blob detection*
$n = 8$	$n = 20$

A.3 Matching

The following thresholds were used in the matching step

Junction detection	*Blob detection*
$T_{patch} = 0.75$	$T_{patch} = 0.6$
$T_{comb} = 0.65$	$T_{comb} = 0.5$

and the parameters for controlling the quality measure over time (see section 6)

$$dq_i = 0.2$$
$$dq_d = 0.1$$
$$T_q = 0$$

Similarity measures: Relative weights In the experiments presented here, the following relative weights (see section 5) were used in the combined significance measure (15):

Junction detection	*Blob detection*
$c_{patch} = 1.0$	$c_{patch} = 1.0$
$c_{sign} = -0.08$	$c_{sign} = -0.25$
$c_{scale} = -0.08$	$c_{scale} = -0.08$
$c_{pos} = -0.1$	$c_{pos} = -0.1$

To give a qualitative motivation for using these orders of magnitude for the relative weights, let us first estimate the ranges in which these descriptors will vary:

- For the cross-correlation measure, it trivially holds that $|S_{patch}| < 1$. By the thresholding operation on this value, $|T_{patch}| = 0.7$, the variation of this entity is confined to the interval $|S_{patch}| \in [0.7, 1.0]$. In practice, the relative variations are usually in the interval $|S_{patch}| \in [0.8, 1.0]$.
- Concerning the significance measure, the significance values of corners computed from an image with grey-level values in the range $[0, 255]$ typically vary in the interval $\log R < 25$. Empirically, the relative variations are usually of the order of $\Delta \log R < 3$. For blob features, the corresponding values are $\log R < 8$ and $\Delta \log R < 1$.
- Concerning the stability of the scale values, the restricted search range given by k_{range}, implies that the relative variation of this descriptor will always be less than $\Delta \log t \approx 1$.
- For the proximity measure the maximum value is $\sqrt{2} * 0.5 * k_{range} * k_{w1} \approx 5$. With smooth scene motions the value is normally considerably smaller.

Motivated by the fact that the relative variation in S_{patch} is about a factor of ten smaller than the other entities, the relative weights of the components in S_{comb} were set according to the table above.

Note that the correlation measure is the dominant component, and the relative influence of the other components corresponds to about half that variation.

The reason why c_{sign} is increased in blob detection, is that the dimension of the significance measures are different:

$$[\tilde{\kappa}^2_{\gamma-norm}] = [\text{brightness}]^6$$
$$[(\nabla^2_{norm} L)^2] = [\text{brightness}]^2$$

Hence, it is natural to increase the coefficient of $S_{sign} = |\log \frac{R_B}{R_A}|$ by a factor of three in blob detection compared to junction detection. As a general rule, we have not performed any fine-tuning of the parameters, and all parameter values have been the same in all experiments.