ULTRA-FAST TRACKING BASED ON ZERO-SHIFT POINTS

Jan Dupač

RS Dynamics, s. r. o.
Starochodovská 1359/76, 149 00 Praha 4,
Czech Republic

Jiří Matas

Czech Technical University
Department of Cybernetics
Technicka 2, 166 27 Praha 6, Czech Republic

ABSTRACT

A novel tracker based on points where the intensity function is locally even is presented. Tracking of these so called zero-shift points (ZSPs) is very efficient, a single point is tracked on average in less than 10 microseconds on a standard notebook. We demonstrate experimentally the robustness of the tracker to image transformations and a relatively long lifetime of ZSPs in real videosequences.

Index Terms— tracking, zero-shift points.

1. INTRODUCTION

The term "tracking" covers a wide range of techniques which can be categorized according to a number of criteria: the character of the tracked object, e.g. a square image patch, a free-form region, an articulated structure or a dynamic texture, by the ability to adapt, learn and recover from failures, by the speed of tracking, by the choice of a predictor, and other properties.

In this paper, we focus on the problem of tracking image patches and propose an extremely fast local tracking method. The novelty and the strength of the method stems from an integrated solution of two design issues a tracking algorithm must address: (1) the choice of objects to track and (2) the method of establishing correspondence between the instance of the objects in consecutive frames.

Let us look how the issues are addressed in most commonly used low-level tracking algorithms. The KLT tracker [1] establishes correspondence by (iterative) gradient-based minimisation. Typically, the tracked objects are image patches with two high eigenvalues of the structure tensor [2], which are essentially the same as regions centered around Harris interest points [3]. These so called "good features to track" or Harris regions possess the property that they are different from all regions in their neighbourhood, a necessary condition for establishing reliable point-to-point correspondence. However, these points do not have, at least not by design, any property that would facilitate the minimisation step.

Other popular low-level tracking methods, e.g. [4, 5], rely on very fast detectors and establish correspondence by matching of detected points, which is feasible if only a small number of alternatives must be verified, ie. when the error in prediction and hence the search radius is small with respect to the density of points.

Inspired by the active appearance models of Cootes et al. [6], Jurie and Dhome [7] realised that in some image regions an approximately linear relationship exists between observations and displacements, allowing ultra-fast tracking by performing a few dot-products in a high-dimensional spaces; the method was later generalised and made more precise by applying a sequence of lower dimensional linear operations which make progressively more accurate predictions [8]. The significant weakness of the linear prediction methods is the need, before tracking starts, to perform both a time-consuming search for suitable regions and learning of the linear mapping.

The proposed method keeps the advantage of the linear predictor method - extremely fast tracking - but removes the search for suitable regions. We propose to track *zero-shift points* (ZSP), i.e. points where the intensity function is locally even. The concept of ZSP was inspired by the 1D iterative version of the SWD interest point detector [9]. The points have two important properties: (1) they are distinguished by the zero-shift and (2) in their neighborhoods, 2D shift vectors shifts "point" to them. Given the two properties, tracking becomes simple: follow 2D shifts until a point of zero-shift is reached. Calculating the 2D shifts requires 4 convolutions per pixel visited in process (a small fraction of all pixels), which only takes in the order of microseconds.

The rest of the paper is structured as follows. First, the concept of zero-shift points is introduced in Section 2. Next, we describe the low-level tracker (3.1) as the main contribution of our work and the basic building block of the other algorithms. Section 3.2 describes simple coarse-to-fine algorithm providing predictions. Section 4 treats the implementation issues. Finally, experiments are described in Section 5 and conclusions are given in Section 6.

2. THE CONCEPT OF ZERO SHIFT POINTS

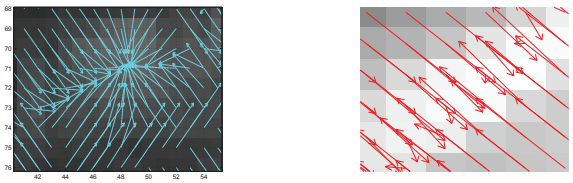
Since ZSPs are defined in terms of local shift vectors, we first explain their computation. The mapping $\Delta^+ = f^+(\mathbf{y})$ from image pixels to the shift vectors estimates the position of the maximum of the first harmonic wave of the window centered at the pixel $\mathbf{y} = [r_0, c_0]$. Similarly, Δ^- estimates the position of the minimum. The elements of the shift vectors $\Delta^+ = [\delta_h^+, \delta_v^+]$ and $\Delta^- = [\delta_h^-, \delta_v^-]$ are computed according to equation (1). The '.' in $\delta_.^+$, $\delta_.^-$, $a_.$, $b_.$ is either 'h' (horizontal) or 'v' (vertical) respectively:

$$\delta_.^+ = \begin{cases} T \arctan(a_./b_.)/(2\pi) & \text{if}(b < 0) \\ -T \operatorname{sign}(a_.)/4 & \text{else} \end{cases}$$
$$\delta_.^- = \begin{cases} T \arctan(a_./b_.)/(2\pi) & \text{if}(b > 0) \\ T \operatorname{sign}(a_.)/4 & \text{else} \end{cases} \quad (1)$$

where $a_.$ and $b_.$ stand for sine and cosine coefficients. They are computed according to Eq. (2) from rectangular windows of length T and width W centered at the point $\mathbf{y} = [r_0, c_0]$. Both T and W are odd integers. The best results were obtained when W was the nearest odd integer to $T/2$. The coefficients are defined as

$$\begin{aligned} a_h &= \textstyle\sum_{i=-w}^{w} \sum_{j=-t}^{t} I(r_0+i, c_0+j)\mathbf{S}(j+t), \\ b_h &= \textstyle\sum_{i=-w}^{w} \sum_{j=-t}^{t} I(r_0+i, c_0+j)\mathbf{C}(j+t), \\ a_v &= \textstyle\sum_{i=-t}^{t} \sum_{j=-w}^{w} I(r_0+i, c_0+j)\mathbf{S}(i+t), \\ b_v &= \textstyle\sum_{i=-t}^{t} \sum_{j=-w}^{w} I(r_0+i, c_0+j)\mathbf{C}(i+t), \\ w &= (W-1)/2, \quad t = (T-1)/2, \\ \mathbf{C}(i) &= \cos(\phi_i), \quad \mathbf{S}(i) = \sin(\phi_i), \\ \phi_i &= 2\pi(i+0.5)/T, \quad i = 0, 1, ...T-1. \end{aligned} \quad (2)$$

The point $\mathbf{z}$ where $\Delta^\cdot$ becomes (approximately) a zero vector is called a *zero shift point* (ZSP)There are two sets of ZSPs, ZSP$^+$ associated with maxima (and the Δ^+ field) and ZSP$^-$ with minima (and Δ^-) of the intensity function respectively. Since the processing of the two sets is identical and independent, we do drop the superscripts in the presentation below. Such points are subpixel entities, i.e. they rarely appear at pixel centers. Vectors Δ at pixels around $\mathbf{z}$ are pointing close to $\mathbf{z}$ as depicted in Fig 1.



(a) (b)

Fig. 1. Examples of typical shift fields (a) near a single ZSPs in a blob-like region and (b) near a ridge with multiple ZSPs.

Typical locations of ZSPs are centers of approximately elliptical "blobs", ie. symmetric areas with higher/lower intensity then their neighbourhood (Fig. 1a)), narrow edge features (Fig. 1b)) and flat areas which are bigger than the length T of the moving window. ZSPs on ridge features and flat areas are not suitable for tracking since they are unstable and can be filtered out by simple rules, see Section 4.

3. THE TRACKING ALGORITHM

3.1. Single scale ZSP tracker

The following simple iterative algorithms tracks a single ZSP from its position $\mathbf{y}_0$ in the previous frame (or from a prediction of its position in the current frame) to a subpixel position $\mathbf{x}$ in the current image. The parameters of the algorithm are: the zero tolerance z (default $0.05T$), the maximum tracking distance d_m (default $T/2$) and the maximum number of iterations n (default 8). The default values were chosen experimentally.

1. $i = 0$.
2. Compute shift vector Δ according Eq. 1, new position $\mathbf{x}_i = \mathbf{y}_i + \Delta$.
3. Test convergence. If $|\mathbf{x}_i - \mathbf{x}_{i-1}| < z$ finish, $\mathbf{x} = \mathbf{x}_i$.
4. Test divergence. If $|\mathbf{x}_i - \mathbf{y}_0| > d_m$ tracking fails.
5. $\mathbf{y}_{i+1} = \text{round}(\mathbf{x}_i)$, $i = i + 1$, if $i < n$ go to 2 else fail.

3.2. Multiscale tracking without a predictor

This section describe a simple multiscale tracking algorithm which does not assume any motion model. The only assumption is that inter-frame disparities are smaller than approximately $T_{max}/2$. The algorithm works in a coarse-to-fine manner. It can be implemented either by using an image pyramid (or the pyramid of integral images) and single period T or the original image (integral image) and variable period. The scaling step is an octave.

1. Set level L to the highest level (or period T to the longest period).
2. Track ZSPs at level L or period T: compute disparity for each point.
3. Detect outliers – points whose disparity differs from the disparity of neighbors more than δ_m.
4. Remove points that are outliers for more than k consecutive frames.
5. Correct outlier disparity – replace the disparity of outliers by median/mean disparity of their neighbors.
6. Correct outlier point position for next frame to initial position plus disparity.
7. Propagate disparity to lower level (shorter period) points – add disparity of the nearest point from higher level to the initial position of the tracked ZSP.
8. If not at the lowest level/shortest period go to step 2

4. IMPLEMENTATION ISSUES

Efficient computation of coefficients. The use of integral images significantly speeds up the tracking of a large number of points. Integral images are cumulative sums along image rows and columns:

$$J^h(r,0) = I(r,0), \quad J^h(r,c+1) = J^h(r,c) + I(r,c+1),$$
$$J^v(0,c) = I(0,c), \quad J^h(r+1,c) = J^h(r,c) + I(r+1,c). \tag{3}$$

The 2D convolutions for sine and cosine coefficient evaluation (see Eq. 2) can be efficiently computed if the intensity values are first summed along the width of the window to get vector $\mathbf{V}$ of the length T. Then, the coefficients are computed as the dot product of the base vector $\mathbf{S}$ or $\mathbf{C}$ with $\mathbf{V}$. By using integral images, the sum of W values for each element of $\mathbf{V}$ can be replaced by a single subtraction. Efficient computation of coefficients:

$$a_h = \mathbf{S}.\mathbf{V}^h, b_h = \mathbf{C}.\mathbf{V}^h, a_v = \mathbf{S}.\mathbf{V}^v, b_v = \mathbf{C}.\mathbf{V}^v,$$
$$\mathbf{V}^h(i+t) = \textstyle\sum_{j=-w}^{w} I(r_0+j, c_0+i) =$$
$$= J^v(r_0+w, c_0+i) - J^v(r_0-w-1, c_0+i),$$
$$\mathbf{V}^v(i+t) = \textstyle\sum_{j=-w}^{w} I(r_0+i, c_0+j) =$$
$$= J^h(r_0+i, c_0+w) - J^h(r_0+i, c_0-w-1),$$
$$w = (W-1)/2, \quad t = (T-1)/2, \quad i = -T/2 \ldots T/2. \tag{4}$$

The computational cost mainly depends on the three operations: (i) prepossessing, ie. integral image calculation, which requires just two integer additions per pixel, (ii) coefficient calculation requiring T integer subtractions and T integer multiply-accumulate instruction per coefficient; four coefficients are necessary per each iteration of single point, (iii) shift: 2 floating point division, 2 floating point atan operations and 2 floating point multiplication per each iteration of single point. The low level tracker needs about 4 iterations for each point.

The low level tracker spends less than 10us on each point per frame on HP6540b notebook. The implementation runs in single thread and some parts are is still far from being optimal.

Searching for ZSPs. It is not necessary to test each pixel for the zero-shift condition since ZSPs suitable for tracking lie inside a basin of attraction with size approximately equal to $T/2$ or bigger. Therefore it is enough to start Algorithm 1 from points on regular grid with a period of $\approx T/2$. Not all ZSPs found in such a way are suitable for tracking. It is valuable to remove ZSPs in flat areas (one of $b_{\cdot}$ is small) and on edges, i. e. points $[r,c]$ where $|\delta_v(r, c \pm t)| > kt$ or $|\delta_h(r \pm t, c)| > kt, t = \mathrm{ceil}(T/8)$.

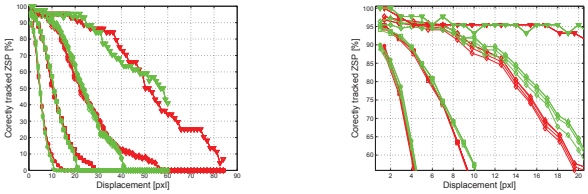
5. EXPERIMENTS

The experiments presented below focus mainly on the performance of the low level tracker and examine its properties necessary for successful integration into a higher level tracking system. The most important properties are the range of the tracker (maximum inter-frame disparity or predictor error), the ability of the tracked ZSPs to survive photometric and geometric changes and the precision of tracking.

The performance plots in Sections 5.1 and 5.2 were obtained using the beginning of the "Mouse pad" sequence [8], which is available online. Results on other public sequences [1] [8, 10, 11] are similar.

5.1. Simulated displacement

The experiment evaluates the range of displacements the ZSP tracker handles. First, the stable subset of ZSPs $\{\mathbf{x}_i\}$ was detected. A displacement was added to the detected positions in axis $([0,d],[0,-d],...)$ and diagonal $([d,d],[d,-d],...)$ directions to get a point $\mathbf{y}_i$ from which simulated tracking starts. Points $\mathbf{y}_i$ were tracked to $\mathbf{x}'_i$. The distance e between $\mathbf{x}_i$ and $\mathbf{x}'_i$ was considered as tracking error and the points with $e > e_m$, $e_m = kT/T_{min}$, $k \in \{0.5, 1, 1.5, 2\}$ were considered "not tracked".



(a) (b)

Fig. 2. **ZSP robustness** as a function of period T (T=9: stars, =19: squares, =39: diamonds, =79: triangles), displacement direction (green vertical and horizontal, red diagonal). The total number of correctly tracked ZSPs was 674 (T=9), 231 (T=19), 55 (T=39), 11(T=79). The ROC curves (same color and markers) are obtained by varying maximum error e_m.

As shown in Fig. 2, the maximum tracking distance is proportional to the period T used for tracking. The influence of the displacement direction is negligible.

5.2. Synthetic data

The second experiment evaluated sensibility of ZSPs to geometric transformations.

First, the set of filtered ZSPs $\{\mathbf{x}_i\}$ was detected in the reference image. Next, the image was warped by homography H and points $\mathbf{x}_i$ were projected to $\mathbf{x}'_i = H\mathbf{x}_i$. Finally, the tracker ran from $\mathbf{y}_i = \mathrm{round}(\mathbf{x}'_i)$ outputting $\mathbf{x}''_i$. The tracking error was defined as the distance e between $\mathbf{x}'_i$ and $\mathbf{x}''_i$. Point with error $e > e_m$, $e_m = kT/T_{min}$, $k \in \{0.5, 1, 2\}$ were considered tracking failures .

5.3. Real data

The low level tracker from section 3.1 was equipped with a simple coarse-to-fine predictor which uses tracking results from coarse scale to predict displacement for finer scale.

[1] ftp://cmp.felk.cvut.cz/pub/cmp/data/lintrack/index.html
http://info.ee.surrey.ac.uk/Personal/Z.Kalal/Publications/cvpr2010/data/
http://www.metaio.com/research

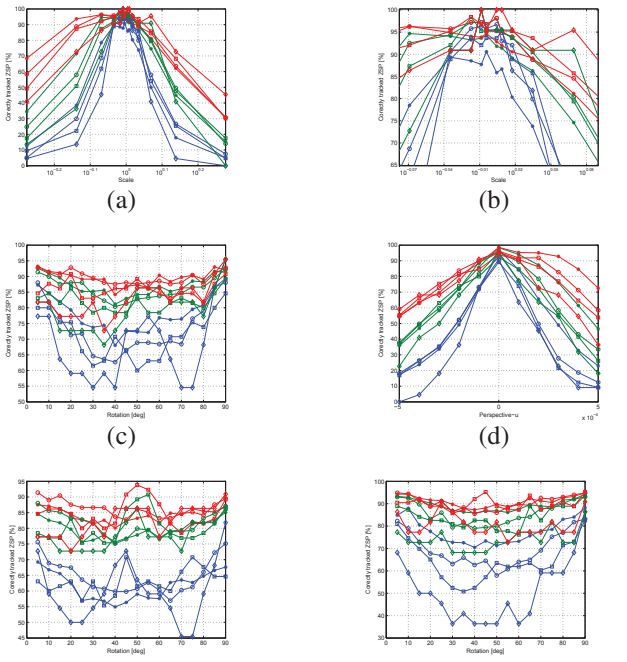


Fig. 3. Robustness to a), b) scale changes, c) pure rotation, d) perspective distortion $H_{3,1}$ and rotation plus 10% scale changes e) zoom out, f) zoom in, for different T (9 stars, 19 cycles, 39 squares, 79 diamonds) and different maximal error $e_m = kT/9, k \in \{0.5(blue), 1(green), 2(red)\}$

The experiment was performed on real sequence with known ground [8]. There is a significant scale change between frames 80 and 130 (approximately 1:2) causing the reduction of the number of successfully tracked points. Other parts of sequence contain small rotation and mild perspective and scale changes and translation.

The tracker was tested also on the other publicly available sequences - the tracked points survived several tens to hundreds of frames depending on the difficulty of the sequence. The low level tracker do not address occlusions at all, therefore it was the main source of failures together with big scale changes.

6. CONCLUSIONS

A tracker of zero-shift-points (ZSPs) was presented. The tracking of such points is trivially implemented by shifting the current estimate until an approximately zero shift is encountered. ZSPs are not so rare, typical image of 640x480 pixels contain about 1000 points. The tracker is very fast, a single point-to-point correspondence was found on average in less than 10 microseconds on a standard notebook.

Experiments demonstrated robustness of the tracker to image transformations and a relatively long lifetime of ZSPs in real videosequences.

Acknowledgement. The authors were supported by EC project FP7-ICT-247022 MASH and by Czech Science Foundation Project P103/10/1585. We thank Michal Perdoch for kindly providing the code for precise warping and support with tracker performance evaluation.

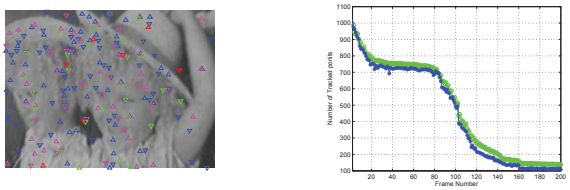


Fig. 4. a) ZSP in real image (detail from the first frame of "Mouse pad" sequence), $T = 9$ blue, $T = 19$ magenta, $T = 39$ green, $T = 79$ red. b) The results of multiscale tracking on real data with known ground truth, green circles – total number of point provided by tracker, blue stars – number of points consistent with ground truth (error better than 3 pixels).

7. REFERENCES

[1] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *IJCAI '81*, 1981, pp. 674–679.

[2] J Shi and C Tomasi, "Good features to track," in *CVPR'94*, 1994, pp. 593–600.

[3] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proc. of 4th Alvey Vision Conference*, 1988, pp. 147–151.

[4] R. O. Castle, G. Klein, and D. W. Murray, "Video-rate localization in multiple maps for wearable augmented reality," in *12th IEEE Int Symp on Wearable Computers*, 2008, pp. 15–22.

[5] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *ISMAR'07*, 2007.

[6] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active appearance models," *IEEE T PAMI*, pp. 681–685, 2001.

[7] F. Jurie and M. Dhome, "Real time robust template matching," in *BMVC2002*, 2002.

[8] K. Zimmermann, J. Matas, and T. Svoboda, "Tracking by an optimal sequence of linear predictors," *IEEE T PAMI*, pp. 677–692, 2009.

[9] J. Dupač and V. Hlaváč, "Stable wave detector of blobs in images," in *DAGM*. 2006, pp. 760–769,

[10] Z. Kalal, J. Matas, and K. Mikolajczyk, "P-N Learning: Bootstrapping Binary Classifiers by Structural Constraints," *CVPR*, 2010.

[11] S. Lieberknecht, S. Benhimane, P. Meier, and N. Navab, "A dataset and evaluation methodology for template-based tracking algorithms," in *ISMAR*, 2009.