Benno Heigl and Dietrich Paulus and Heinrich Niemann

Tracking Points in Sequences of Color Images

Lehrstuhl für Mustererkennung, (Informatik 5)
Martensstr. 3, 91058 Erlangen
Universität Erlangen–Nürnberg

German Russian Workshop 1999
Herrsching, Germany
September 1999

Tracking Points in Sequences of Color Images

Benno Heigl and Dietrich Paulus and Heinrich Niemann

Lehrstuhl für Mustererkennung, (Informatik 5)
Martensstr. 3, 91058 Erlangen
Universität Erlangen–Nürnberg
{heigl,paulus,niemann}@@informatik.uni-erlangen.de
http://www5.informatik.uni-erlangen.de

Abstract Tomasi and Kanade presented a strategy for the selection and for subsequent tracking of points in sequences of gray–level images. The strategy is based on the notion of "interest" of a point and requires that the disparity of a point is small between any two images.
In this contribution we extend this principle to sequences of *color* images in different color spaces. We compare the results to those computed with the original algorithm. The goal is to track as many points as possible for as long as possible. Using the trajectories of the points and the camera parameters, 3–D information is reconstructed in our experimental setup. A pan/tilt unit mounted to a linear sledge is used. We describe how the zoom lens is calibrated by a simple algorithm.
Keywords: color, motion, tracking, 3–D, calibration

1 Introduction

Two basic questions must be answered when points are tracked in image sequences: how to select the features, and how to track them from frame to frame.

For the selection of point features, several methods are known from literature. Well–known examples are the "interest operator" in [3] and the "edge detector" from [6]. [8] gives an overview over several techniques. To track point features, many methods exist using the correlation between local windows. An overview can also be found in [8]. The solution described in [9] combines these two problems to one; it is based on [2], where a method for registering images for stereo matching is proposed.

The applications of point tracking are manifold. In the following we use this technique to recover depth from a static scene seen by a calibrated camera moving on a linear sledge. We also show its application for an automatic calibration of a multi–media camera.

This article is structured as follows. Sect. 2 explains the original approach. In Sect. 3 the algorithm is extended to color images and tested for two different color spaces. Sect. 4 shows how to restrict the search direction. Experiments are given in Sect. 5, followed by two applications shown in Sect. 6. The paper ends with the conclusion in Sect. 7.

2 Tracking Points in Gray Value Images

The basic idea in [9] is to select those points in the image sequence, which exhibit features for stable tracking. Thereby the two questions which were formerly posed independently, are now combined to one problem.

We formalize the problem using an image function $f(x, y, t)$ depending on the position $\boldsymbol{x} = (x, y)^{\mathrm{T}}$ and on the time t. If we look at a point at time t and at position $\boldsymbol{x}$ and at the same point at time $t + \tau$ and at position $\boldsymbol{x} + \boldsymbol{d}(x, y, t)$ (with $\boldsymbol{d}(x, y, t) = (d_1(x, y, t), d_2(x, y, t))^{\mathrm{T}}$), we assume that the residual error $\epsilon = (f(x - d_1(x, y, t), y - d_2(x, y, t), t) - f(x, y, t+\tau))^2$ is minimal. We define ϵ within a Window $\mathcal{W}(x, y)$ as

$$\epsilon = \iint\limits_{(\mu,\nu)^{\mathrm{T}} \in \mathcal{W}(x,y)} \left(f(\mu - d_1(x, y, t), \nu - d_2(x, y, t), t) - f(\mu, \nu, t + \tau)\right)^2 d\mu d\nu \,. \quad (1)$$

We denote the gradient at time t and position (μ, ν) by $\boldsymbol{g}(\mu, \nu, t) = \nabla_{\mu,\nu} f(\mu, \nu, t)$. Using the gradient we approximate the intensity function by a Taylor expansion:

$$f(\mu - d_1(x, y, t), \nu - d_2(x, y, t), t) \approx f(\mu, \nu, t) - \left(\boldsymbol{g}(\mu, \nu, t)\right)^{\mathrm{T}} \cdot \boldsymbol{d}(x, y, t) \,.$$

Using this approximation we can set the derivative to zero and can minimize the value of ϵ using Eq. (1) with respect to the displacement $\boldsymbol{d}(x, y, t)$. This yields the following equation:

$$\boldsymbol{G}(x, y, t)\boldsymbol{d}(x, y, t) = \boldsymbol{e}(x, y, t) = \iint\limits_{(\mu,\nu)^{\mathrm{T}} \in \mathcal{W}(x,y)} \left(f(\mu, \nu, t) - f(\mu, \nu, t + \tau)\right) \boldsymbol{g}(\mu, \nu, t)\, d\mu d\nu \quad (2)$$

with a symmetric 2×2–matrix $\boldsymbol{G}(x, y, t)$

$$\boldsymbol{G}(x, y, t) = \iint\limits_{(\mu,\nu)^{\mathrm{T}} \in \mathcal{W}(x,y)} \boldsymbol{g}(\mu, \nu, t)(\boldsymbol{g}(\mu, \nu, t))^{\mathrm{T}} d\mu d\nu = \begin{pmatrix} g_{11} & g_{12} \\ g_{12} & g_{22} \end{pmatrix} \quad .$$

This set of linear equations can be solved uniquely. The solution for the displacement $\boldsymbol{d}(x, y, t)$ is an approximation of the real displacement. If we resample the function $\boldsymbol{f}(x + d_1(x, y, t), y + d_2(x, y, t), t)$ inside of the window $\mathcal{W}(x, y)$ using bi–linear interpolation, we can iterate the algorithm and compute the displacement with sub–pixel accuracy. In [9], the minimal Eigenvalue of the matrix $\boldsymbol{G}(x, y, t)$ is used as measure for selecting points for tracking. It judges whether the texture of the object at the considered location is strong enough to be distinguished from signal noise, and whether the point can be tracked in any direction. This defines an operator which judges points for point tracking. We call it *tracing operator* in the following. Results of this operator can be seen in Figure 1

3 Tracking Points in Color Images

In [1] we extened the ideas presented in Sect. 2 to color images which are represented by vector–valued intensity functions $\boldsymbol{f}(x, y, t)$ and did experiments for RGB data. The derivatives and equations for the tracking can be derived in analogy to the scalar case. In the following we use we use arbitrary color spaces.

The residual value for a displacement vector $\boldsymbol{d}(x, y, t) = (d_1(x, y, t), d_2(x, y, t))^{\mathrm{T}}$ inside a window $\mathcal{W}(x, y)$ is defined for color vectors by:

$$\epsilon = \iint\limits_{(\mu,\nu)^{\mathrm{T}} \in \mathcal{W}(x,y)} \|\boldsymbol{f}(\mu - d_1(x, y, t), \nu - d_2(x, y, t), t) - \boldsymbol{f}(\mu, \nu, t + \tau)\|^2 \, d\mu d\nu \quad . \tag{3}$$

The Taylor approximation for the color image case uses the Jacobian $\boldsymbol{J}(x, y, t)$ of $\boldsymbol{f}(x, y, t)$ in the variables x and y.

$$\boldsymbol{f}(\mu - d_1(x, y, t), \nu - d_2(x, y, t), t) \approx \boldsymbol{f}(\mu, \nu, t) - (\boldsymbol{J}(\mu, \nu, t))^{\mathrm{T}} \boldsymbol{d}(x, y, t) \, . \tag{4}$$

Inserting (4) into (3) and defining $\boldsymbol{h}(\mu, \nu, t) := \boldsymbol{f}(\mu, \nu, t) - \boldsymbol{f}(\mu, \nu, t + \tau)$, we get:

$$\epsilon = \iint\limits_{(\mu,\nu)^{\mathrm{T}} \in \mathcal{W}(x,y)} \|\boldsymbol{h}(\mu, \nu, t) - (\boldsymbol{J}(\mu, \nu, t))^{\mathrm{T}} \boldsymbol{d}(x, y, t)\|^2 \, d\mu d\nu \, . \tag{5}$$

The optimal displacement $\boldsymbol{d}(x, y, t)$ which minimizes the error ϵ can be determined by setting the gradient of ϵ with respect to $\boldsymbol{d}(x, y, t)$ to zero:

$$\iint\limits_{(\mu,\nu)^{\mathrm{T}} \in \mathcal{W}(x,y)} \Big(2\boldsymbol{J}(\mu, \nu, t)(\boldsymbol{J}(\mu, \nu, t))^{\mathrm{T}} \boldsymbol{d}(x, y, t) - 2\boldsymbol{J}(\mu, \nu, t)\boldsymbol{h}(\mu, \nu, t) \Big) \, d\mu d\nu = \boldsymbol{0}$$

$$\underbrace{\iint\limits_{(\mu,\nu)^{\mathrm{T}} \in \mathcal{W}(x,y)} \boldsymbol{J}(\mu, \nu, t)(\boldsymbol{J}(\mu, \nu, t))^{\mathrm{T}} \, d\mu d\nu}_{\boldsymbol{G}(x, y, t)} \; \boldsymbol{d}(x, y, t) = \underbrace{\iint\limits_{(\mu,\nu)^{\mathrm{T}} \in \mathcal{W}(x,y)} \boldsymbol{J}(\mu, \nu, t)\boldsymbol{h}(\mu, \nu, t)) \, d\mu d\nu}_{\boldsymbol{e}(x, y, t)}$$

$$\boldsymbol{d}(x, y, t) = \tag{6}$$

As for scalar functions, this system of equations can be solved uniquely. Like in the scalar case, the minimal Eigenvalue for the matrix $\boldsymbol{G}(x, y, t)$ defines the tracing operator. An example is shown in Figure 1.

Even better results can be obtained, if the camera motion is known, as we show next.

4 Tracking in Known Direction

If the camera motion is known, we can use a prediction for tracking points. In the case of translational motion, we can use the epipolar constraints known from stereo vision. This further extends the ideas of [9]:

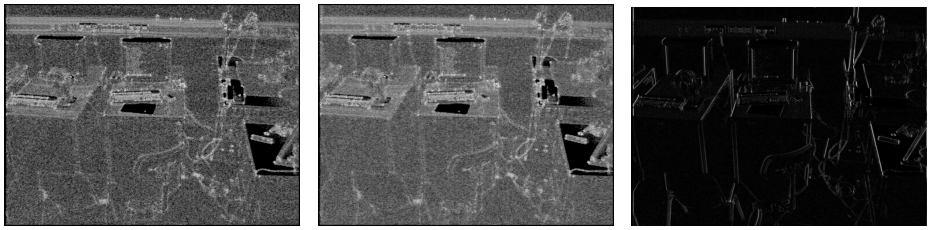

Figure1. Comparison of the results of the tracing operator for a gray value image (left) and a color image (middle); the interest operator for tracking color point features in horizontal direction (right)

We denote the known displacement direction of a point $\boldsymbol{x} = (x, y)^{\mathrm{T}}$ by $\boldsymbol{r}(x, y, t)$ and assume that $\|\boldsymbol{r}(x, y, t)\| = 1$. We now want to find the optimal displacement $\boldsymbol{d}(x, y, t) = u(x, y, t)\, \boldsymbol{r}(x, y, t)$ by simply determining $u(x, y, t)$. The computation is similar to the formulas given above; instead of the gradient we now use the directional derivative of the function $\boldsymbol{f}$ in direction $\boldsymbol{r}(x, y, t)$.
Using the Taylor approximation

$$\boldsymbol{f}(\boldsymbol{x} - u(x, y, t)\, \boldsymbol{r}(x, y, t)) \approx \boldsymbol{f}(\boldsymbol{x}) - u(x, y, t)\, (\boldsymbol{J}(\mu, \nu, t))^{\mathrm{T}} \boldsymbol{r}(x, y, t) \tag{7}$$

and the error estimation

$$\epsilon = \iint\limits_{(\mu,\nu)^{\mathrm{T}} \in \mathcal{W}(x,y)} \|\boldsymbol{h}(\mu, \nu) - u\, (\boldsymbol{J}(\mu, \nu, t))^{\mathrm{T}} \boldsymbol{r}(x, y, t)\|^2 \, d\mu d\nu \tag{8}$$

we obtain after differentiation with respect to u and setting to zero:

$$u(x, y, t) = \frac{\iint\limits_{(\mu,\nu)^{\mathrm{T}} \in \mathcal{W}(x,y)} (\boldsymbol{h}(\mu, \nu))^{\mathrm{T}} (\boldsymbol{J}(\mu, \nu, t))^{\mathrm{T}} \boldsymbol{r}(x, y, t)\, d\mu d\nu}{\iint\limits_{(\mu,\nu)^{\mathrm{T}} \in \mathcal{W}(x,y)} \|(\boldsymbol{J}(\mu, \nu, t))^{\mathrm{T}} \boldsymbol{r}(x, y, t)\|^2 \, d\mu d\nu} . \tag{9}$$

For this case, the tracing operator is defined as the value of the denominator. Its value is large for those points, which can be tracked well. Therefore, the case of the denominator being zero does not matter. The same method also is applicable to gray value images. Figure 1 shows, that with the restriction of motion direction, all inhomogeneities in motion direction are locations for features which can be tracked good.

5 Experiments

For the evaluation, we used an image sequence of an office environment. Figure 2 shows the tracked points for a sequence consisting of 32 single frames with 768 $\times$ 576 pixels each. The maximum displacement of corresponding points is 55 pixel between the first and the last frame. In this example, 1000 features have been tracked with window size 5×5 pixels.

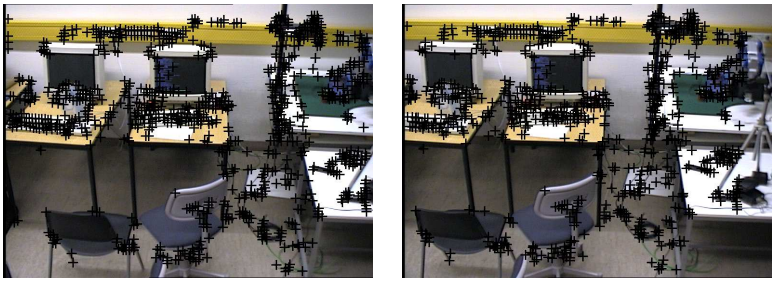

Figure2. The best 1000 features selected in the first frame of the color image sequence (left); tracked features in last frame (right)

To evaluate the correctness of the tracking process, one measure can be how many points get lost during tracking. The criterion for *loosing* a point is, that the mean pixel intensity difference within the window to the preceding frame is more than 8 gray–values (in the color case, the color value is converted to the luminance Y of XYZ color space). Figure 3 shows the number of lost points in dependence on the size of a square window for different methods and different color spaces. It can be seen, that the use of color decreases the number of lost points significantly. But the application of the (RG, BY, WB) color space is worse than the plain RGB space; this is surprising since the differences of color vectors in (3) use the Euclidian distance measure which does not correspond directly to perceptual differences in RGB. The decrease of lost points here is 37% in the case of a window size of 3×3. The following table shows the computing times on an SGI O2 for a window size of 5×5:

Computing times in seconds		
task	gray–level	color
computing gradients	1.2	3.6
feature selection	39.5	41.0
tracking points	0.4	1.0
total time (32 frames)	90.3	187.2

For the first frame, the gradients are computed and features are selected. For the other 31 frames, the gradients are computed and the points are tracked. It can be seen that a lot of time is required for computing gradients. Most time of feature selection is spent for finding maxima after applying the tracing operator, supposing a minimal distance of 10 pixels between features. It is remarkable, that the time for tracking color features is less than three times as long as for gray–value features. The times for feature selection and point tracking increases linearly with the number of pixels within the window. Therefore it is a proper decision to choose a small window size, color tracking, and the RGB color space to get reliable results and fast computation.

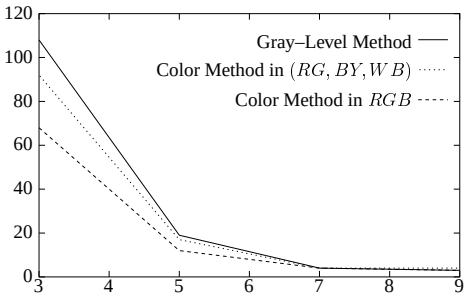

Figure3. Number of lost points for different methods; 1st: using the gray–level method; 2nd: using the color method in the (RG, BY, WB) color space; 3rd: using the color method in RGB.

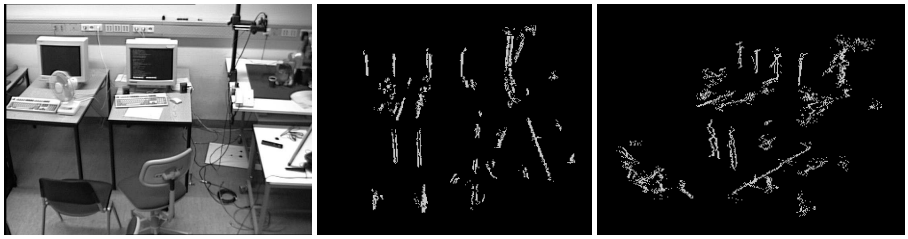

Figure4. First image of a test sequence with 32 single frames taken by a linear moving camera (left). The 2283 resulting 3–D points seen from two views (middle and right).

6 Application

In this section, we show two applications to point tracking. The first one is for reconstructing 3–D points from image sequences taken during linear motion. The second one shows how to calibrate a zoom lens by tracking points while changing the focal length.

In the first application, a pan/tilt camera is moved by a linear sledge to recover depth. The goal is, to determine the 3–D position of scene points. For each point, the world coordinates are denoted by the vector $(x_w, y_w, z_w)^T$. Only such points are selected to be tracked which are local maxima of the tracing criterion described in Sect. 3.

Supposing the image plane always to be parallel to the moving direction of the linear sledge, the problem is similar to stereo vision supposing parallel optical axes and intersecting image planes [5]. In contrast to stereo vision, in our case the correspondence problem is not to be solved explicitly. The point correspondences result from tracking during camera movement. The trajectories then can directly be used to determine the world coordinates.

We make the generalization of a camera with an arbitrary orientation, given by pan and tilt angle. This case is reduced to the above mentioned parallel one by projecting all viewing rays to the plane $z = 1$. The camera coordinate system is chosen such that the z–axis is orthogonal to and the x–axis is parallel to the moving direction of the camera. Then the camera coordinate system is chosen to be independent of the actual camera orientation. Every viewing ray and its corresponding scene point then is determined

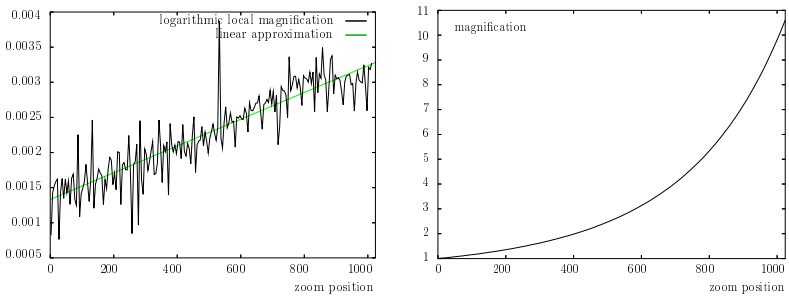

Figure5. Logarithmic local magnification and its linear approximation (left). Resulting function of magnification relative to the minimal zoom position (right).

by the vector $(x_p, y_p, 1)^T$ corresponding to homogeneous coordinates. Viewing rays in the direction of motion cannot be represented by this vector. This is no limitation to the generality of the method, as the projection of the corresponding scene points does not change during motion and therefore no depth information can be obtained at all. To calculate the normalized coordinates x_p and y_p, the projection parameters must be known. We use the calibration technique [10] to get them.

Because of epipolar constraints, the y_p coordinate is constant for all positions of the linear sledge. Without loss of generality, we suppose the world coordinate system to be equal to the camera coordinate system at the leftmost position of the linear sledge. The vector $(x_c, 0, 0)$ then describes the translation of the world into the camera coordinate system. The normalized image coordinate x_p depends linearly on the value x_c by $x_p(x_c) = (x_w - x_c)/z_w$. Using this equation, the world coordinates $(x_w, y_w, z_w)^T$ of the tracked scene point can be estimated by linear regression [7]. This approximation additionally gives a measure for the uncertainty of the trace. It can be used to eliminate points whose trajectories are not linear and therefore do not correspond to a single scene point.

Figure 4 shows the result for the same image sequence as in Sect. 5. The computing time for this sequence is 167 seconds on an SGI O2, including the processes of tracking and 3–D reconstruction.

Another application to point tracking is to determine the dependence of focal length on the setting of the motor of an automatic zoom lens. We call motor position of the zoom lens *zoom position*. The method [11] achieves this by calibrating the projection parameters for several zoom positions using a calibration pattern and interpolating functions for each parameter. The main practical problem of this approach is to segment the calibration pattern for the whole domain. In our application [4], we only needed the value of focal length. Therefore we developed a new simple method based on tracking points.

We record a sequence of images changing the zoom positions over the whole range. For this sequence we track points, substituting disappearing ones by new ones. The local magnification between two neighboring frames can be calculated by averaging the magnification of connection lines of randomly chosen pairs of points. To compute the magnification of a single zoom position relative to the minimal one, all local magnifi-

cations between them have to be multiplied. Experiments show, that this attempt is too sensitive to outliers of local magnification. To avoid this, the logarithms of the local magnifications are approximated linearly (see Figure 5, left). The function of the logarithmic magnification then can be calculated by integrating the approximated logarithmic local magnification. This trick reduces the influence of local errors. After applying the exponential function, we get the magnification relative to the minimal zoom position. (see Figure 5, right). If the focal length of the minimal zoom position is known, each focal length can be calculated by multiplying the corresponding relative magnification.

It would also be possible to use point tracking for calibrating the minimal focal length by analyzing point trajectories in image sequences taken from a slightly rotating camera.

7 Conclusion

This article shows, that the method [9] for tracking point features in gray–value images can be extended successfully for color images. It is verified by experiments, that color information increases the robustness of the tracking procedure. Considering the knowledge of motion direction, more features can be selected for tracking. The use for depth recovery and zoom lens calibration shows the application for different problems.

By applying resolution hierarchies, larger displacements of features between single frames will be manageable in future.

References

1. B. Heigl and D. Paulus. Punktverfolgung in Farbbildsequenzen. In D. Paulus and Th. Wagner, editors, *Dritter Workshop Farbbildverarbeitung*, pages 87–92 & 105, Stuttgart, 1997. IRB-Verlag.
2. B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7^{th} International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.
3. H. P. Moravec. Robot spatial perception by stereoscopic vision and 3D evidence grids. Technical Report CMU-RI-TR-96-34, The Robotics Institue, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213, September 1996.
4. D. Paulus, U. Ahlrichs, B. Heigl, and H. Niemann. Wissensbasierte aktive Szenenanalyse. In P. Levi, editor, *Mustererkennung 1998*, Heidelberg, September 1998. Springer. accepted.
5. Stefan Posch. *Automatische Tiefenbestimmung aus Grauwert-Stereobildern*. Dissertation, Deutscher Universitäts Verlag, Wiesbaden, 1990.
6. A. Rosenfeld and M. Thurston. Edge and curve detection for visual scene analysis. *IEEE Transactions on Computers*, 20:562–569, 1971.
7. G. Schmidt. Tiefe aus linearer Kamerabewegung (*Depth from linear camera motion*). Technical report, Student's thesis, Lehrstuhl für Mustererkennung (Informatik 5), Universität Erlangen–Nürnberg, Erlangen, 1995.
8. J. Shi and C. Tomasi. Good features to track. In *Proceedings of Computer Vision and Pattern Recognition*, pages 593–600, Seattle, Washington, June 1994. IEEE Computer Society Press.
9. C. Tomasi and T. Kanade. Detection and tracking of point features. Technical Report CMU-CS-91-132, Carnegie Mellon University, 1991.

10. R. Y. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *IEEE Journal of Robotics and Automation*, Ra-3(3):323–344, August 1987.
11. R.G. Willson. *Modeling and Calibration of Automated Zoom Lenses*. PhD thesis, Carnegie Mellon University, Pittsburgh, 1994.