Real-time Uncharacteristic-part Tracking based on Points Tracking

Norimichi Ukita, Akira Makino, and Masatsugu Kidode
Nara Institute of Science and Technology

Abstract

In this research, we focus on how to track a target region that lies next to similar regions (e.g., a forearm and an upper arm) in zoom-in images. In our method, a group of feature points in a target region is extracted and tracked as the model of the target. Small differences between the neighboring regions can be verified by focusing only on the feature points. In addition, (1) the stability of tracking is improved using particle filtering and (2) tracking robust to occlusions is realized by removing unreliable points using random sampling.

1 Introduction

High-resolution images of a target object can improve the performance of various existing image-analysis algorithms. In particular, high-resolution images of human parts (e.g., a face, a hand, a forearm, and a tiny part of them) are useful in order to observe facial expression, hand/finger motion, and other minute motions and features. Continuous high-resolution imaging of a moving target can be realized by using a pan-tilt-zoom (PTZ) camera. For zoom-in observation with the PTZ camera, the following two techniques are required:

Tracking Continuously extract the region of the target in observed images.

Camera-control Continuously control the PTZ camera for capturing the target within the images.

If tracking is successful, a previously proposed camera-control algorithm (e.g., [1, 2]) is applicable to continuous high-resolution imaging regardless of the type of the target. We, therefore, focus on how to track a partially observed region of a moving object in real time.

Template matching is widely used for tracking (see [3], for example). Although it is applicable to any object tracking, the similarity can be high not only in the correct region of a target but also in its neighboring regions with textures similar to those of the correct region. Tracking using SNAKE[4], which searches for the boundary line of a target based on edge lines and their smoothness, is also inapplicable to tracking body regions with similar textures between which there is no edge line. Mean-shift[5] is also one of the famous real-time tracking algorithms. While this method is robust against deformation of a target shape, mistracking between similar neighboring regions is not avoidable because this method employs only a color histogram.

For zoom-in tracking, in this research, a set of feature points in a target region is regarded as a tracking target. The difference between similar neighboring regions can be expressed well by focusing only on characteristic feature points. Point tracking is one of major problems (e.g., stereo vision and shape from motion) in Computer Vision. In [6], for example, the Lucas-Kanade algorithm is analysed in detail in terms of computational complexity and stability. Tracking robust to specular highlights and lighting changes[7] is also important. Our method proposed in this paper is peculiar in terms of the following two characteristics:

Accept small drifts: Unlike point tracking algorithms for 3D reconstruction, a small *drift* is acceptable in our objective. This is because points might be included in the target region even when small drifts occur. In this research, therefore, the stability of tracking should be improved in exchange for the decline of precision. We employ particle filtering[8] for robustly tracking feature points with updating template images.

Explicitly remove outlier points: As with our approach, particle filtering is used to track a set of points in [9]. In [9], however, all points are evaluated equivalently for calculating the likelihood of each particle. This results in tracking failure when one or more of feature points are invisible due to occlusions. This problem can be solved by removing these occluded points from each particle.

2 Tracking with Particle Filtering

In particle filtering, a target region is represented as a state vector. A set of particles, each of which corresponds to a state vector, is distributed in order to find the target region. Let $\{s_t^{(1)}, \cdots, s_t^{(N)}\}$ and $\{\pi_t^{(1)}, \cdots, \pi_t^{(N)}\}$ be the set of the particles at time t and their likelihoods, respectively. With them, the target region at t is determined and the particles at $t+1$ are generated as follows:

1. **Likelihood calculation:** Likelihood $\pi_t^{(i)}$ of $s_t^{(i)}$ in an image observed at time t is calculated.
2. **State estimation:** The region of the target in the observed image is estimated from the state vectors of the particles and their likelihoods (e.g., the weighted mean the state vectors).
3. **Sampling:** New particles $s_{t+1}^{(i)}$ are generated based on the likelihoods of the particles at t.
4. **Drift:** $s_{t+1}^{(i)}$ are shifted based on the motion dynamics of the target.

3 Feature Points Tracking with Particle Filtering

The basic scheme of our method is based on tracking with particle filtering[8] described in Section 2. The

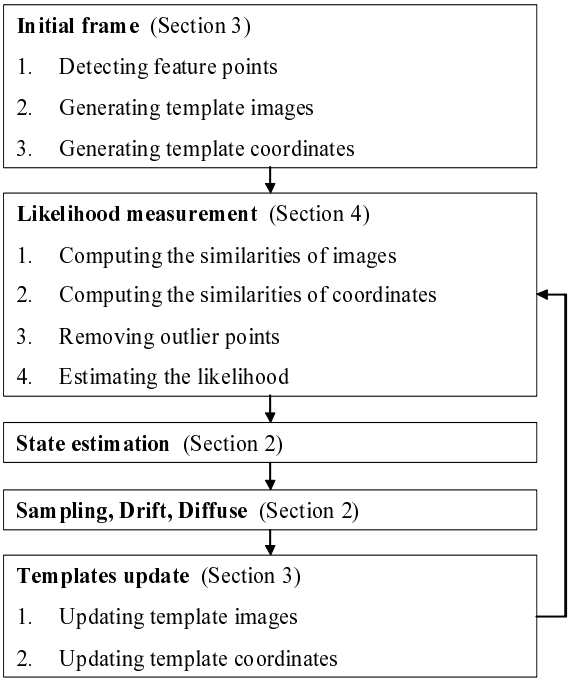


Figure 1: Process flow of feature points tracking.

characteristic issue of our method is how to calculate the likelihood of each particle as follows: Its process flow is shown in Figure 1. The distinctive feature of our method is outlier elimination.

Assume that the region of a target is given at an initial frame. At this frame, several feature points in the initial region are extracted by using [10]. A set of these points is regarded as a target being tracked. The state vector S of each particle is expressed with image coordinates of all the points, denoted by $(x_1, y_1), (x_2, y_2), \cdots$, and their average velocity, (v_x, v_y), as follows:

$$S = [\ x_1 \quad y_1 \quad x_2 \quad y_2 \quad \cdots \quad v_x \quad v_y\ ] \tag{1}$$

Assume that the points have the following natures:

- The change in the local image around a feature point is small at a small interval.
- The change in the geometric configuration of feature points is small at a small interval.

With these natures, two templates below are used:

Template image Local image around each point.

Template coordinates xy coordinates of each point.

These templates are obtained at an initial frame and they are dynamically updated. The likelihood of each particle is calculated by integrating the following two values (described in detail in Sec. 4):

Image similarity Similarity between the local image around each point in a particle and the template image of the point.

Coordinates similarity Similarity between the geometric configurations of the points in the template coordinates and each particle.

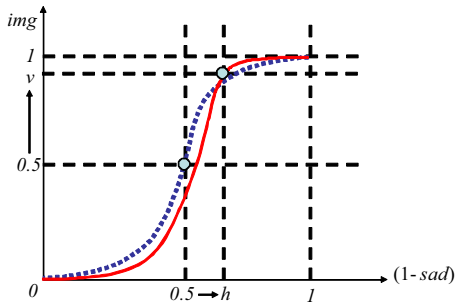


Figure 2: Weighted similarity function (dotted line: sin curve, solid line: a weighted similarity).

As with the previous tracking algorithm using particle filtering[8] described in Section 2, the weighted mean of the particles is regarded as the state of the target (i.e., the estimated points). Sampling and drift of our algorithm are also similar to those of [8]. The estimated velocity of the target (i.e., v_x and v_y in a particle) is used to shift the particles in the current frame.

Although the appearance of the target object in observed images change as it moves, careless update of the templates incurs mistracking. In our method, therefore, the image templates are updated not only using the updated template but also using the initial template as proposed in [3]. The template image is updated if both of the following two conditions are satisfied[3]: (1) the similarity between the current template image and the local image around the estimated point is higher than a threshold and (2) the maximum similarity between the local image around the estimated point and the images that are obtained by rotating, translating, expanding, and shrinking the initial template image is higher than a threshold. The template coordinates are also updated in the same way with updating the template image.

4 Likelihood

This section describes how to compute the likelihood of each particle with two similarities in Sec. 4.1 and 4.2 and outlier removal that overcomes small deformation of points and occlusion in Sec 4.3.

4.1 Similarity of Feature Points

The image similarity is evaluated by the SAD (Sum of Absolute Difference). The computed SAD is normalised (i.e., from 0 to 1) by the template size and the maximum value of the intensity range.

Note that even the position of each feature point in the best particle might be a little different from the true position. Since even the small difference results in the significant decrease of the image similarity (i.e., SAD), the similarity of the best point might be accidentally lower than that of another point. To cope with this problem, the similarity is determined so that the similarity decreases drastically at some distance from the true position as illustrated in Figure 2. The practical image similarity of point p in particle i, $I_p(i)$, is

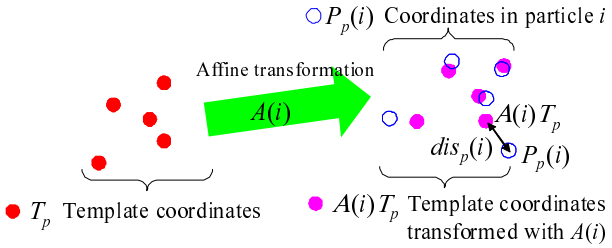


Figure 3: Affine transformation of points.

calculated by the following equation:

$$I_p(i) = 2v \sin\left(\frac{\pi}{2}\frac{0.5}{h}c_p(i)\right) \quad \text{if } c_p(i) \le h \tag{2}$$

$$I_p(i) = 2(1-v)\sin\left(\frac{\pi}{2}(c_p(i) - (2h-1))\right) + (2v-1) \quad \text{if } c_p(i) > h \tag{3}$$

where $c_p(i) = 1-$ "the normalised SAD", and h and v are constants[1] indicated in Figure 2.

4.2 Similarity of Geometric Configurations

The similarity of geometric configurations is calculated by comparing the template coordinates and the coordinates of points in each particle. As illustrated in Figure 3, let $P_p(i)$ and T_p be the coordinates of point p in particle i and the coordinates of point p in the template coordinates, respectively. Note that the correspondence of points between $P_p(i)$ and T_p is known. First, the affine parameters $A(i)$ between $P_p(i)$ and T_p is estimated, and then T_p is transformed using $A(i)$ as illustrated in the right-hand of Figure 3. The distance $dis_p(i)$ between the transformed coordinates $A(i)T_p$ and $P_p(i)$ is obtained by $dis_p(i) = |A(i)T_p - P_p(i)|$. With $dis_p(i)$, the coordinates similarity $P_p(i)$ is expressed by the following formula:

$$P_p(i) = \frac{1}{dis_p(i)+1} \tag{4}$$

4.3 Removing Outlier Points

Using the image and coordinates similarities $I_p(i)$ and $P_p(i)$, the likelihood of particle i is estimated by $\left(\sum_{p=1}^{N} e_p(i)\right)/N$, where N is the number of the feature points in each particle and $e_p(i) = I_p(i) \cdot P_p(i)$.

$e_p(i)$ is very low in a position far from its true position and a position occluded by another object. We call such a point an *outlier point*. In general, the former problem (i.e., a wrong position) can be easily resolved by distributing a large number of particles in order to prepare the one in which all points are located in their correct positions. As the particles increase, however, the computational speed declines. Furthermore, the latter problem (i.e., an occluded point) cannot be solved by boosting the particles. In our method, therefore, the likelihood of a particle is evaluated without points, each of which has an extremely low value.

[1]In our experiments, $h = 0.8$ and $v = 0.9$.

Affine parameters, required for computing the coordinates similarity, can be estimated from at least three corresponding points. If one or more outlier points are included in the corresponding points, the estimated affine parameters produce the transformed points, all of which are far from the points in the particle. If no outlier point is included, $e_p(i)$ may be low only in the outlier point(s). Outlier removal is, therefore, implemented by robust estimation using random sampling as follows:

Step 1 Select random three points in a particle and estimate the affine parameters between them and the template coordinates.

Step 2 Transform the template coordinates using the affine parameters.

Step 3 Calculate $e_1(i), \cdots, e_N(i)$ and select their median value, $e_{med}(i)$.

Step 4 Repeat Steps 1, 2, and 3 k times and consider the maximum $e_{med}(i)$ to be the one that corresponds to the optimal affine parameters.

Steps 1, 2, and 3 should be repeated so that no outlier point is included in at least one combination of the selected points. The probability that at least one combination without outliers is selected (denoted by P)[2] is determined[11] by $P = 1 - \{1 - (1-\epsilon)^F\}^k$, where ϵ and F denote the ratio of outlier points in all the feature points and the number of selected samples (i.e., three in our case), respectively.

Finally, outlier points are removed using the optimal affine parameters as follows:

Step 1 Calculate $e_1(i), \cdots, e_N(i)$ using the optimal affine parameters

Step 2 Compute the standard deviation of $e_1(i), \cdots, e_N(i)$ (denoted by σ).

Step 3 Remove points, each of whose $e_p(i)$ is at a distance of over 2.5σ from the average, from the particle i and then evaluate its likelihood.

5 Experiments

We conducted experiments using a PC (Pentium4 3.0GHz) with 1024×768 pixels images captured at 30 fps. In all the experiments, the number of particles, the number of points in a particle, and the size of a template image were 1000, 7, and 15×15, respectively. With these conditions, our method worked at 25 fps.

For comparative experiments, the following four methods were examined:

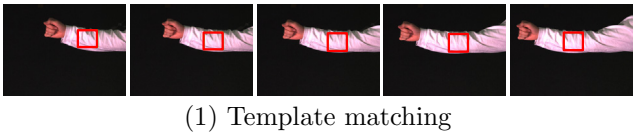
Method1 Template matching with SAD evaluation.

Method2 KLT tracker[6].

Method3 Points tracking using particle filtering WITHOUT removing outlier points.

Method4 Proposed method: Points tracking using particle filtering WITH removing outlier points.

[2]In our experiments, P was 0.95.



(1) Template matching

(2) KLT tracker

(3) Proposed method without outlier removal

(4) Proposed method

Figure 4: Uncharacteristic part tracking.

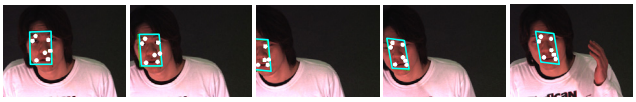


Figure 5: Partially-occluded part tracking.

One of the advantages of our method is to be able to track a target region neighboring similar regions as shown in Figure 4. The target region is enclosed by a rectangle. The results of Method1 stayed at the same area in the observed images while the arm translated. Method2 and Method3 lost track of the target region in midstream. Our proposed method could track the target region until the last frame.

The second advantage of our method is to be able to track a region partially occluded. As shown in Figure 5, our method could track the face region without even when/after the face partially got out of the image and the hand partially obscured the face.

We conducted experiments for quantitative comparison (Table 1). For three targets (a white shirt, a textured shirt, a face with partial occlusions), five sequences consisting of 50 frames were prepared. True regions of the target were given by hand and compared with the tracking results; a result overlapping over 80% with the true region is regarded as a successful result. While the results of textured pattern tracking were equivalent, our method was superior to the others in tracking the white shirt and the face with occlusions.

Table 1: Quantitative comparison: success rate of part tracking.

	1. white	2. textured	3. face
Method1	64.5%	71.9%	84.2%
Method2	73.5%	92.5%	88.8%
Method3	84.3%	94.1%	89.8%
Method4	87.7%	94.2%	96.3%

6 Concluding Remarks

We focused on how to track an uncharacteristic region and proposed a points tracking method using particle filtering with removing outlier points. Future work includes the following aspects:

Speed-up In an algorithm with particle filtering, speed performance is critical because the robustness improves as the number of particles increases. The random sampling algorithm should be improved because its computational time makes up a larger percentage of the total.

Template update Our method employs the template update algorithm proposed in [3]. This algorithm compares not only a current template but also an initial template with a current observed image in order to avoid *drift* in selecting a new template. Employing the initial template, however, prevents matching with a significantly deformed image. A more sophisticated way is, therefore, required for robust and high-speed tracking.

Grouping To track points distributed in multiple limbs, the points must be grouped based on their motions, as with the rigid-motion based grouping algorithm[12].

References

[1] P. Y. Oh and P. K. Allen, "Design of a partitioned visual feedback controller," in Proc. of *IEEE International Conference on Robotics and Automation*, pp.1360–1365, 1998.

[2] H. Wu, Q. Chen, H. Oike, C. Hua, T. Wada, and T. Kato, "High Performance Object Tracking System Using Active Cameras," in Demo Proc. of *ICCV*, 2005.

[3] I. Matthews, T. Ishikawa, and S. Baker, "The Template Update Problem," *PAMI*, Vol.26, No.6, pp.810–815, 2004.

[4] N. Peterfreund "The Velocity Snake: Deformable Contour for Tracking in Spatio-Velocity Space," *CVIU*, Vol.73, No.3, pp.346–356, 1999.

[5] D. Comaniciu, et al, "Kearnel-Based Object Tracking," *PAMI*, Vol.25, No.5, 2003.

[6] S. Baker and I. Matthews, "Lucas-Kanade 20 years on : A Unifying Framework," *IJCV*, Vol.56, No.3, pp.221-255, 2004.

[7] M. Gouiffes, C. Collewet, C. Fernandez-Maloigne, and A. Tremeau, "Feature points tracking : robustness to specular highlights and lighting changes," in Proc. of *9th European Conference on Computer Vision*, Vol.4, pp.82–93, 2006.

[8] M. Isard and A. Blake, "Condensation - conditional density propagation for visual tracking," *IJCV*, Vol.28, No.1, pp.5–28, 1998.

[9] K. Kawamoto, "Guided Importance Sampling based Particle Filtering for Visual Tracking," in Proc. of *IEEE Pacific-Rim Symposium on Image and Video Technology*, pp.158–167, 2006.

[10] J. Shi and C. Tomasi, "Good Features to Track," in Proc. of CVPR, pp.593–600, 1994.

[11] P. Rousseeuw and A. Leroy, "Robust Regression & Outlier Detection," *John Wiley & Sons*, 1987.

[12] K. Inoue and K. Urahama, "Separation of multiple objects in motion images by clustering," in Proc. of ICCV, Vol.1, pp.219–224, 2001.