

Analytical Dynamic Programming Tracker

Seiichi Uchida, Ikko Fujimura, Hiroki Kawano, Yaokai Feng

Kyushu University, Fukuoka, Japan

Abstract. Visual tracking is formulated as an optimization problem of the position of a target object on video frames. This paper proposes a new tracking method based on dynamic programming (DP). Conventional DP-based tracking methods have utilized DP as an efficient breadth-first search algorithm. Thus, their computational complexity becomes prohibitive if the search breadth becomes large according to the increase of the number of parameters to be optimized. In contrast, the proposed method can avoid this problem by utilizing DP as an analytical solver rather than the conventional breadth-first search algorithm. In addition to experimental evaluations, it will be revealed that the proposed method has a close relation to the well-known KLT tracker.

1 Introduction

Generally, visual tracking [1] is formulated as an optimization problem of the target object position (and other posture parameters) at each video frame. Thus, tracking accuracy as well as computational complexity depends on optimization strategy.

Dynamic programming (DP) [2] is a well-known optimization strategy. It has been utilized in visual tracking for over 20 years and is still applied in many recent tracking methods. Since DP guarantees the global optimality of its solution, it can provide the most reliable (i.e., accurate) tracking result for a given optimization problem. This ability also indicates robustness against various distortions, such as occlusion.

Surprisingly, all the conventional DP-based tracking methods have utilized DP in the same manner. That is, DP is always utilized as an efficient *breadth-first search* algorithm; many candidate trajectories representing partial tracking results are branched and unified at each frame. To the author's best knowledge, all of the conventional methods are commonly built on this breadth-first search algorithm, although they have their own originality at cost definition, computation reduction, etc.

Main contribution of this paper is to propose a novel DP-based tracking method, called *analytical DP tracker*. In the proposed method, we will utilize DP as an *analytical* solver of the visual tracking problem, not as a breadth-first search algorithm. It is very rare that DP is used as an analytic solver in past researches on computer vision and pattern recognition. (A smoothing method by Angel [3] and a contour matching method by Serra and Berthod [4] are the

only exceptions.) This forgotten aspect of DP, however, is very beneficial from the viewpoint of computational efficiency.

The key idea of developing the analytical DP tracker is a quadratic representation of the cost at each frame. Here, the cost means some cost for locating the target at a certain position on the frame. By the quadratic representation, the objective function of the tracking problem becomes differentiable and it is possible to apply analytical DP to find the optimal tracking result efficiently. The quadratic cost may be found naturally in some kinds of objective functions (as discussed in 4.3), or may be derived intentionally through some approximations. Although quadratic approximation seems to be so rough, there are several techniques (such as bi-directional strategy described in Section 7.2) to have sufficient results under the approximation.

The merits of the analytical DP tracker are summarized as follows:

- Given a quadratic cost at every frame, it can provide the optimal tracking result only by $O(T)$ computations, where T is the number of frames.
- It inherits good properties of the conventional DP-based trackers; that is, it can guarantee the global optimality of the tracking result, it is robust to occlusion, and it is capable of introducing user’s interaction.
- It has a strong relation to a reputable tracker called KLT tracker [5, 6].
- It can be extended for dealing with other deformations such as rotation and scaling with slight increase of computational complexity. Although the conventional DP-based trackers also can be extended, it causes drastic increase of computational complexity.

2 Related Work

Visual tracking methods can be classified into two types: online tracking and offline tracking. The majority belong to the former because it can realize real-time tracking. KLT tracker [5, 6] and particle filter are typical online tracking methods. Online tracking methods determine the target position using the current and past frames. The inevitable drawback of online tracking is that if it misses the target at a certain frame due to any distortion (e.g., occlusion), it is very difficult to recover correct tracking in succeeding frames.

Although offline tracking methods are minority, they have great robustness against distortions. This is because it optimizes its tracking result after all video frames are given; in other words, the target position of each frame is determined by using not only past frames but also future frames. Thus, offline tracking will be better than online tracking if real-time processing is not necessary. For example, motion and behavior analysis (such as ball tracking for understanding and annotating soccer games) will be a good application for offline tracking. Motion capture for modeling human motion will also be another application because it requires accuracy rather than real-time process.

DP is one of the most important tools for offline tracking. It is well-known that DP is the most popular tool for nonlinear matching (called DP matching [7] or dynamic time warping), curve detection [8], and Snakes [9–11]. DP has also

been employed in offline tracking methods [12–18] for over 20 years. Although there are many modern optimization strategies nowadays, such as graph cut and belief propagation (message passing), DP is still utilized in recent offline tracking methods. This may be because of its good properties, such as global optimality of its solution, numerical stability, and versatility in dealing with various cost functions and constraints.

One drawback of DP is its computational complexity. Since DP has been utilized as a breadth-first search algorithm in computer vision and pattern recognition, its computational complexity becomes large if search breadth becomes large. When we track a target object moving on an $M \times N$ image without rotation and scaling, search breadth becomes $O(MN)$. If we allow R -level rotation and S -level scaling, search breadth becomes $O(MNRS)$. In these cases, $O(M^2N^2T)$ and $O(M^2N^2R^2S^2T)$ computations are necessary in total, respectively.

In order to reduce large computations of the conventional DP-based tracking methods, various techniques have been employed. A classic technique is beam-search which “prunes” less hopeful candidate trajectories at each frame. More sophisticated techniques can be found in recent methods. In [17], k-d trees are utilized to select possible target positions efficiently at each frame. In [18], AdaBoost is utilized to “recognize” possible target positions. These recent methods are still similar to classic beam-search from the viewpoint that all these methods limit possible candidate trajectories by using some local criteria at each frame.

In order to avoid the struggle against huge search breadth, another aspect of DP is newly exploited in the proposed method as a radical remedy. As already noted in Section 1, DP can be used as a very efficient analytical solver for the tracking problem if the cost at each frame is defined as a quadratic function. The detail of the analytical DP tracker will be discussed in Section 4 after reviewing the conventional DP-based tracking method in Section 3.

3 Conventional DP Tracker

3.1 General formulation of tracking problem

Generally, visual tracking is formulated as the minimization problem of the following objective function:

$$F(\mathbf{w}_1, \dots, \mathbf{w}_T) = \lambda \sum_{i=1}^T d_i(\mathbf{w}_i) + \sum_{i=1}^{T-1} \|\mathbf{w}_{i+1} - \mathbf{w}_i\|^2, \quad (1)$$

where $\mathbf{w}_i = (x_i, y_i)^T$ is the object position in the i -th frame image with $M \times N$ pixels. The first term $d_i(\mathbf{w}_i)$ evaluates some cost for locating the target at \mathbf{w}_i in the i -th frame. It is predetermined for each \mathbf{w}_i as shown in Fig. 1 (a). The second term is used for smoothing target motion. The constant λ is a weight. The position $\bar{\mathbf{w}}_i = \mathbf{w}_i$ which minimizes F gives the tracking result.

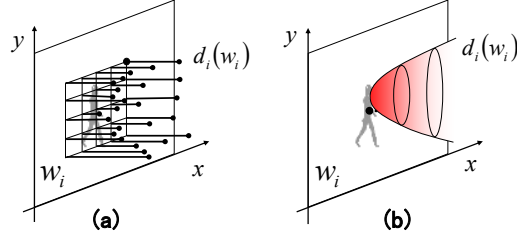


Fig. 1. Cost for conventional method (a) and the proposed method (b).

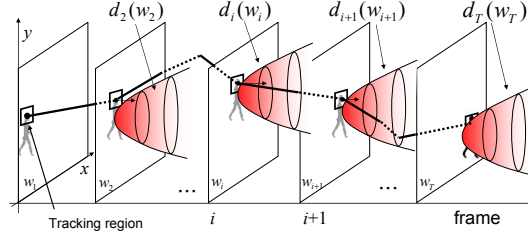


Fig. 2. Quadratic function $d_i(w_i)$ and optimization process of w_i .

3.2 Solution by conventional DP

DP has been employed to obtain the globally optimal solution of the above tracking problem. In order to derive the conventional DP algorithm, consider the following function:

$$f_i(\mathbf{w}_{i+1}) = \min_{\mathbf{w}_1, \dots, \mathbf{w}_i} \sum_{k=1}^i [\lambda d_k(\mathbf{w}_k) + \|\mathbf{w}_{k+1} - \mathbf{w}_k\|^2]. \quad (2)$$

Using f_i , the global minimum $\min F$ can be represented as

$$\min F = \min_{\mathbf{w}_T} [f_{T-1}(\mathbf{w}_T) + \lambda d_T(\mathbf{w}_T)]. \quad (3)$$

From the principle of optimality [2], Eq. (2) can be rewritten as a recursion equation,

$$f_i(\mathbf{w}_{i+1}) = \min_{\mathbf{w}_i} [f_{i-1}(\mathbf{w}_i) + \lambda d_i(\mathbf{w}_i) + \|\mathbf{w}_{i+1} - \mathbf{w}_i\|^2]. \quad (4)$$

At $i = 1$, if the initial target position is given as $\bar{\mathbf{w}}_1$, this equation becomes

$$f_1(\mathbf{w}_2) = \lambda d_1(\bar{\mathbf{w}}_1) + \|\mathbf{w}_2 - \bar{\mathbf{w}}_1\|^2 \quad (5)$$

The conventional DP algorithm is based on the calculation of $f_i(\mathbf{w}_{i+1})$ from $i = 1$ to $(T - 1)$ according to (5) and (4) at every *discretized* position $\mathbf{w}_{i+1} \in [1, 2, \dots, M] \times [1, 2, \dots, N]$. Then, using $f_{T-1}(\mathbf{w}_T)$ and (3), $\min F$ is obtained. Note that the number of possible positions, MN , is the search breadth of the

conventional DP algorithm. This is because every \mathbf{w}_{i+1} has its own candidate trajectory whose accumulated cost is $f_i(\mathbf{w}_{i+1})$.

The globally optimal tracking result $\overline{\mathbf{w}}_1, \dots, \overline{\mathbf{w}}_T$ is obtained through a backtracking process. Specifically, the optimal target position $\overline{\mathbf{w}}_T$ is first determined as \mathbf{w}_T which gives $\min F$. Then, from $i = T - 1$ to 1, $\overline{\mathbf{w}}_i$ is determined as \mathbf{w}_i which minimizes (4) with $\overline{\mathbf{w}}_{i+1} = \mathbf{w}_{i+1}$.

3.3 Computational complexity of conventional DP

Let \mathcal{W}_i denote the number of all possible \mathbf{w}_i , that is, the search breadth at the i -th frame. In this paper, we define $\mathbf{w}_i = (x_i, y_i)^T$ and thus $\mathcal{W}_i = MN$. Eq. (4) requires $O(\mathcal{W}_{i-1})$ computations for its minimum search. Since the above algorithm calculates (4) for $O(\mathcal{W}_i T)$ times, the total computational complexity becomes $O(\mathcal{W}_{i-1} \mathcal{W}_i T) = O(M^2 N^2 T)$.

If we allow rotation and scaling, \mathbf{w}_i becomes a 4-dimensional vector of x_i , y_i , rotation angle, and scaling factor. Clearly, by this extension, the conventional method suffers from huge computations due to the drastic increase of \mathcal{W}_i . Thus, limiting \mathcal{W}_i is a straightforward remedy. For example, setting an $L \times L$ window on each frame will reduce the computations ($M^2 N^2 \rightarrow L^4$). Unfortunately, this remedy still suffers from a dilemma that accuracy may be sacrificed.

4 The Analytical DP Tracker

4.1 Quadratic representation of cost

The key idea to derive the analytical solution of the tracking problem of Section 3.1 is a quadratic representation of the cost $d_i(\mathbf{w}_i)$, i.e.,

$$d_i(\mathbf{w}_i) = \mathbf{w}_i^T \mathbf{P}_i \mathbf{w}_i + \mathbf{q}_i^T \mathbf{w}_i + r_i, \quad (6)$$

where \mathbf{P}_i is a 2×2 matrix, \mathbf{q}_i is a two-dimensional vector, and r_i is a scalar. (Unless otherwise mentioned, we assume only x - y translation (i.e., $\mathbf{w}_i = (x_i, y_i)^T$). We assume these elements are predetermined in some way, whose example will be shown in Section 5. As noted before, this quadratic cost will be found naturally in some kinds of objective function. It will also be derived intentionally through some approximations. Section 4.3 will detail this point.

Figure 1 (b) illustrates a quadratic cost. Figure 2 illustrates a sequence of quadratic costs $d_1(\mathbf{w}_1), \dots, d_i(\mathbf{w}_i), \dots, d_T(\mathbf{w}_T)$ in addition to the sequence $\mathbf{w}_1, \dots, \mathbf{w}_i, \dots, \mathbf{w}_T$ to be optimized. It should be noted that \mathbf{w}_i can be considered as a continuous variable under the quadratic representation of (6). Again, although quadratic representation seems to be rough, it can be practically compensated by using iterative updating (which is also employed in KLT), or bi-directional strategy, or multiple quadratic functions, etc.

4.2 Solution by analytical DP

Now, the objective function (1) becomes differentiable with respect to $\mathbf{w}_i = (x_i, y_i)^T$. For the optimal tracking result, we may have a naive idea to solve the system of equations, $\{\partial F/\partial x_i = 0, \partial F/\partial y_i = 0, \forall i\}$. Unfortunately, its coefficient matrix becomes large (a $2T \times 2T$ matrix) and does not have a special structure (like a tridiagonal matrix) which enables fast solution.

Fortunately, DP provides a very efficient solution for the optimization problem. This new solution is different from the breadth-first search of Section 3.2; it is *analytical* solution with just $O(T)$ computations.

An important fact to derive the analytical solution by DP is that $f_i(\mathbf{w}_{i+1})$ becomes a quadratic function of \mathbf{w}_i , as well as $d_i(\mathbf{w}_i)$. This fact can be proven inductively by using the quadratic property of $d_i(\mathbf{w}_i)$ and the smoothness term. (Its proof is omitted here.) From this fact, $f_i(\mathbf{w}_{i+1})$ can also be represented as a quadratic function,

$$f_i(\mathbf{w}_{i+1}) = \mathbf{w}_{i+1}^T \mathbf{A}_i \mathbf{w}_{i+1} + \mathbf{b}_i^T \mathbf{w}_{i+1} + c_i, \quad (7)$$

where \mathbf{A}_i is a 2×2 symmetric matrix, \mathbf{b}_i is a two-dimensional vector, and c_i is a scalar. These elements are variables and to be determined during the optimization process. (In contrast, \mathbf{P}_i , \mathbf{q}_i , and r_i in (6) are predetermined constants.)

Substituting (7) into (4), we have

$$f_i(\mathbf{w}_{i+1}) = \min_{\mathbf{w}_i} \left[\mathbf{w}_i^T \mathbf{A}_{i-1} \mathbf{w}_i + \mathbf{b}_{i-1}^T \mathbf{w}_i + c_{i-1} + \lambda (\mathbf{w}_i^T \mathbf{P}_i \mathbf{w}_i + \mathbf{q}_i^T \mathbf{w}_i + r_i) + \|\mathbf{w}_{i+1} - \mathbf{w}_i\|^2 \right]. \quad (8)$$

The right-hand side of the above equation is a quadratic function of \mathbf{w}_i and therefore $\bar{\mathbf{w}}_i$ which attains its minimum is derived analytically as

$$\bar{\mathbf{w}}_i = [\mathbf{A}_{i-1} + \lambda \mathbf{P}_i + \mathbf{I}]^{-1} (\mathbf{w}_{i+1} - (\mathbf{b}_{i-1} + \lambda \mathbf{q}_i)/2). \quad (9)$$

This shows the (backtrack) procedure to determine the optimal target position $\bar{\mathbf{w}}_i$ for a given \mathbf{w}_{i+1} .

Substituting $\bar{\mathbf{w}}_i$ of (9) into (8) and comparing with (7), we have

$$\left. \begin{aligned} \mathbf{A}_i &= \mathbf{I} - [\mathbf{A}_{i-1} + \lambda \mathbf{P}_i + \mathbf{I}]^{-1} \\ \mathbf{b}_i &= (\mathbf{b}_{i-1} + \lambda \mathbf{q}_i)^T [\mathbf{I} - \mathbf{A}_i] \\ c_i &= -\frac{1}{4} \mathbf{b}_i (\mathbf{b}_{i-1} + \lambda \mathbf{q}_i) + c_{i-1} + \lambda r_i \end{aligned} \right\}. \quad (10)$$

This shows the recursive procedure to determine $\mathbf{A}_i, \mathbf{b}_i, c_i$ from $\mathbf{A}_{i-1}, \mathbf{b}_{i-1}, c_{i-1}$. Substituting (7) and (6) into (5), the initial condition of this recursive procedure, i.e., $\mathbf{A}_1, \mathbf{b}_1, c_1$, is given as follows:

$$\left. \begin{aligned} \mathbf{A}_1 &= \mathbf{I} \\ \mathbf{b}_1 &= -2\bar{\mathbf{w}}_1 \\ c_1 &= \lambda d_1(\bar{\mathbf{w}}_1) + \bar{\mathbf{w}}_1^T \bar{\mathbf{w}}_1 \end{aligned} \right\}. \quad (11)$$

Figure 3 summarizes the above procedure for the solution of the tracking problem based on analytical DP. After (11), $\{\mathbf{A}_i, \mathbf{b}_i, c_i | i = 1, \dots, T-1\}$ is determined by using (10) recursively. Then, $\bar{\mathbf{w}}_T$ is determined by substituting $\mathbf{A}_{T-1}, \mathbf{b}_{T-1}, c_{T-1}$ into the following equation, which is derived by the minimum condition of (3):

$$\bar{\mathbf{w}}_T = -\frac{1}{2}[\mathbf{A}_{T-1} + \lambda \mathbf{P}_T]^{-1}(\mathbf{b}_{T-1} + \lambda \mathbf{q}_T) \quad (12)$$

The minimum value $\min F$ is given by using $\bar{\mathbf{w}}_T$ and (3). Finally, the backtrack procedure (9) will provide the tracking result as $\bar{\mathbf{w}}_{T-1}, \dots, \bar{\mathbf{w}}_2$.

4.3 Global optimality of solution

The tracking result given by the analytical DP tracker is the globally optimal solution which minimizes (1) with the quadratic cost (6). Thus, if the optimization problem is originally defined with the quadratic cost, the analytical DP tracker will provide the most accurate solution.

As an example, let us consider an objective function $F = \prod_i P_1(\mathbf{w}_i)P_2(\mathbf{w}_{i+1} - \mathbf{w}_i)$ where $P_1(\mathbf{w}_i)$ is a likelihood and $P_2(\mathbf{w}_{i+1} - \mathbf{w}_i)$ is a smoothness prior and both of P_1 and P_2 are two-dimensional normal distributions. In this case, the logarithm of F results in the optimization problem of (1) with the quadratic cost (6). Thus, the solution by the proposed method will be the truly globally optimal solution of its original problem.

In contrast, for an optimization problem which is NOT originally defined with the quadratic cost, the analytical DP tracker will provide an approximation solution of the original problem. In this case, we must derive the quadratic cost by approximating the original cost. The tracking result $\bar{\mathbf{w}}_1, \dots, \bar{\mathbf{w}}_T$ is different from the globally optimal solution of the original problem. In Section 5, a method to derive the quadratic cost (Fig. 1(b)) from a discrete cost (Fig. 1(a)) is discussed.

4.4 Computational complexity

As shown from Fig. 3, the computational complexity of the proposed method is $O(T)$ and thus independent of the image size $M \times N$. This fact reveals the superiority of the proposed method over the conventional DP-based tracking method. This superiority will become more significant if we deal with rotation and scaling. By this extension, \mathbf{w}_i becomes a 4-dimensional vector. As noted in Section 3.3, the computational complexity of the conventional method will increase drastically by this extension. In contrast, the computational complexity of the proposed method will increase very slightly; the increase mainly comes from the fact that 2×2 matrices become 4×4 matrices. This shows a good property of the analytical DP tracker.

The above computational complexity of the analytical DP tracker does not include the computations for the quadratic representation of $d_i(\mathbf{w}_i)$, that is, the computations for predetermining \mathbf{P}_i , \mathbf{q}_i , and r_i . Thus, to retain the computational superiority, it is important to minimize the computations for the quadratic

<p>Input: Coefficients of cost functions $\{\mathbf{P}_i, \mathbf{q}_i, r_i \mid i = 1, \dots, T\}$, weight $\lambda \in \mathfrak{R}^+$, and initial position: $\bar{\mathbf{w}}_1 \in \mathfrak{R}^2$</p> <p>Output: Optimal location: $\bar{\mathbf{w}}_2, \dots, \bar{\mathbf{w}}_i, \dots, \bar{\mathbf{w}}_T$, and minimum distance: $\min F$</p> <p>[Step 1: Initial condition] Obtain $\mathbf{A}_1, \mathbf{b}_1, c_1$ by (11).</p> <p>[Step 2: DP recursion] For $i = 2$ to $T - 1$, obtain $\mathbf{A}_i, \mathbf{b}_i, c_i$ from $\mathbf{A}_{i-1}, \mathbf{b}_{i-1}, c_{i-1}$ by (10).</p> <p>[Step 3: Termination] Obtain $\bar{\mathbf{w}}_T$ and $\min F$ by (12) and (3), respectively.</p> <p>[Step 4: Backtrack] For $i = T - 1$ downto 2, obtain $\bar{\mathbf{w}}_i$ from $\bar{\mathbf{w}}_{i+1}$ by (9).</p>
--

Fig. 3. Analytical DP for visual tracking.

representation. In Section 5, one feasible method of the quadratic representation will be given.

5 Implementation Issues

5.1 Quadratic representation by approximation

If the cost is not originally quadratic, tracking performance of the proposed method depends on the accuracy of quadratic approximation of $d_i(\mathbf{w}_i)$. For example, if the original cost is given by

$$\delta_i(\mathbf{w}_i) = \sum_{\epsilon} [I_i(\mathbf{w}_i + \epsilon) - I_1(\mathbf{w}_1 + \epsilon)]^2, \quad (13)$$

we must determine the elements $\mathbf{P}_i, \mathbf{q}_i, r_i$ so that $\delta_i(\mathbf{w}_i) \sim d_i(\mathbf{w}_i)$.

In this paper, we derive a quadratic cost $d_i(\mathbf{w}_i)$ by Taylor series expansion, referring KLT tracker [5, 6]. Specifically, $\delta_i(\mathbf{w}_i)$ is expanded to be a second-order polynomial function around a given approximation center $\tilde{\mathbf{w}}_i$.

Now, the remaining problem is how to determine $\tilde{\mathbf{w}}_i$ before the approximation. As noted above, the original cost $\delta_i(\mathbf{w}_i)$ is approximated around $\tilde{\mathbf{w}}_i$ by Taylor series expansion. Thus, if \mathbf{w}_i becomes more distant from $\tilde{\mathbf{w}}_i$, the approximation error between $\delta_i(\mathbf{w}_i)$ and $d_i(\mathbf{w}_i)$ becomes more significant. Consequently, in order to determine $\bar{\mathbf{w}}_i$ at the “true” target position by the proposed method, we must set $\tilde{\mathbf{w}}_i$ close to the true target position for better quadratic approximation; this is obviously a chicken-and-egg problem. The following Section 5.2 will be devoted to relax this problem.

5.2 Iterative updating

The difficulty on deriving a good quadratic cost $d_i(\mathbf{w}_i)$ can be relaxed by iterative updating of the tracking result. The process of the iterative tracking is described as follows:

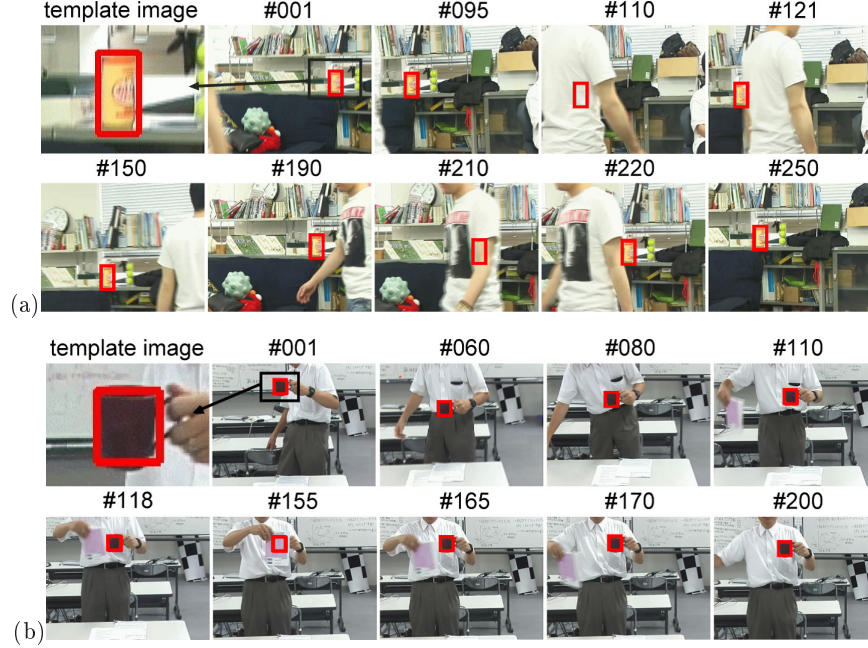


Fig. 4. Results by analytical DP tracker.

1. Find a very rough tracking result $\bar{\mathbf{w}}_1^0, \dots, \bar{\mathbf{w}}_i^0, \dots, \bar{\mathbf{w}}_T^0$ by using, for example, the conventional DP on down-sampled frames (i.e., images whose size $(M \times N)$ is small enough).
2. Set $k = 1$.
3. $\tilde{\mathbf{w}}_i^k \leftarrow \bar{\mathbf{w}}_i^{k-1}$, for all i .
4. Obtain quadratic cost function $d_i(\mathbf{w}_i)$ by Taylor series expansion around $\tilde{\mathbf{w}}_i^k$, for all i .
5. Find updated tracking result $\bar{\mathbf{w}}_1^k, \dots, \bar{\mathbf{w}}_i^k, \dots, \bar{\mathbf{w}}_T^k$ by using the analytical DP tracker.
6. Terminate if k reaches a pre-specified maximum iteration number. Otherwise, $k \leftarrow k + 1$ and goto 3.

Even though the initial tracking result $\bar{\mathbf{w}}_1^0, \dots, \bar{\mathbf{w}}_i^0, \dots, \bar{\mathbf{w}}_T^0$ is not accurate, it will be updated iteratively and converged around the true target position. Note that the initial rough tracking result can also be given by linear interpolation of the points specified manually at several frames.

6 Experimental Results

6.1 Tracking results

Figure 4(a) shows a video sequence captured by a hand-held camera. The target object (an orange color plastic object) was occluded completely by a man during

100~120-th frames and 200~220-th frames. Figure 4(b) shows another video sequence. The target object (a cup) was occluded completely by a book during 120~160-th frames. In both video sequences, the target object underwent neither scaling nor rotation.

In advance to the optimization, the target object template I_1 was manually specified at the first frame (#001). Using this I_1 , $\delta_i(\mathbf{w}_i)$ was evaluated at each frame and then $d_i(\mathbf{w}_i)$ was derived according to the procedure of Section 5. The iterative updating process of Section 5.2 was also employed. Unless otherwise mentioned, the pre-specified maximum iteration number was fixed at 20.

In Fig. 4, the red rectangular shows the target position optimized by the analytical DP tracker. Both results reveal that the proposed method could track the target objects accurately. Even if the target object was occluded completely, the tracking result was still stable. (See the 210-th frame in (a) and the 155th frame in (b).) This shows that the offline tracking with DP is robust against occlusion as expected. Note that the mean-shift tracker failed to track the object due to the occlusion. (See the supplementary video.)

Figure 5 shows the change of $d_i(\bar{\mathbf{w}}_i)$ during the iterative updating process for the video sequence of Fig. 4(a). The initial tracking result (“iteration 0”), which was obtained by the conventional DP on the down-sampled frames, shows that $d_i(\bar{\mathbf{w}}_i)$ was fluctuating and rather large. In contrast, the tracking results after the iterative updating shows that the cost $d_i(\bar{\mathbf{w}}_i)$ decreases monotonically according to the iteration and finally converged into a very small value (except for the frames with occlusion). This result shows the accuracy of the tracking result quantitatively.

6.2 Computation time

Figure 6 shows the computation time per a frame of the analytical DP tracking and the conventional DP tracking. They were measured on a PC (Pentium D).

Since the conventional DP tracking required more than 10 hours without any special treatment for reducing computations, a two-step optimization procedure and a window were introduced for reducing its computations. The first step was the same as that of the iterative updating of the proposed method; that is, a very rough tracking result was obtained on down-sampled frame images as the initial tracking result. At the second step, an $L \times L$ window was set around the initial tracking result. The horizontal axis of Fig. 6 was L for the conventional method.

From several comparative experiments, it was known that L should be 15 pixels at least to have the tracking result as accurate as that by the analytical DP tracking. This fact confirms the superiority of the analytical DP tracking over the conventional method; at $L = 15$, the computation time of the proposed method was about half of the conventional method. Note again that if rotation and scaling are allowed (that is, the dimension of control variables becomes higher), this superiority will become far more significant.

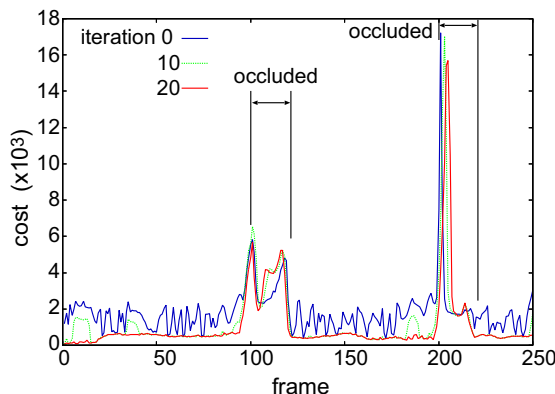


Fig. 5. Change of cost $d_i(\bar{w}_i)$ during iterative updating.

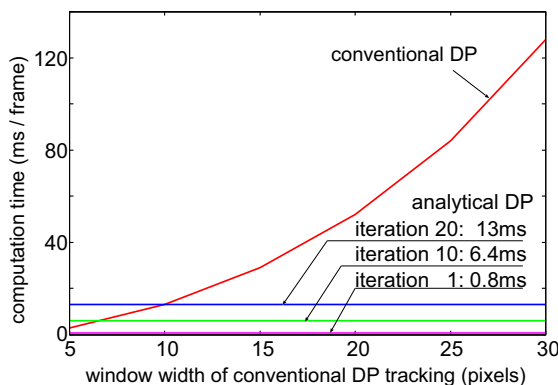


Fig. 6. Computation time.

7 Discussion

7.1 Relation to KLT tracker

The quadratic representation of tracking cost can be found in KLT tracker [5]. KLT tracker is a well-known online tracking method and the target position \mathbf{w}_i is optimized using a quadratic cost function $d_i(\mathbf{w}_i)$ derived by Taylor series expansion of $\delta_i(\mathbf{w}_i)$ around $\hat{\mathbf{w}}_i = \mathbf{w}_{i-1}$. (This assumes that the displacement between the $(i-1)$ th and i th frames is very small.) In order to improve the tracking accuracy, $\bar{\mathbf{w}}_i$ giving the minimum of $d_i(\mathbf{w}_i)$ is then considered as the updated approximation center $\hat{\mathbf{w}}_i$ in KLT. This procedure is repeated until convergence and $\bar{\mathbf{w}}_i$ is determined as the converged \mathbf{w}_i . Then the optimization of \mathbf{w}_{i+1} starts.

This iterative updating process of KLT tracker is very similar to the iterative updating process of the analytical DP tracker introduced in Section 5.2. Especially, they have the same purpose of compensating poor approximation ability of Taylor series expansion. They, however, are different in their optimization strategies. Figure 7 show their iterative updating processes. As noted above, the

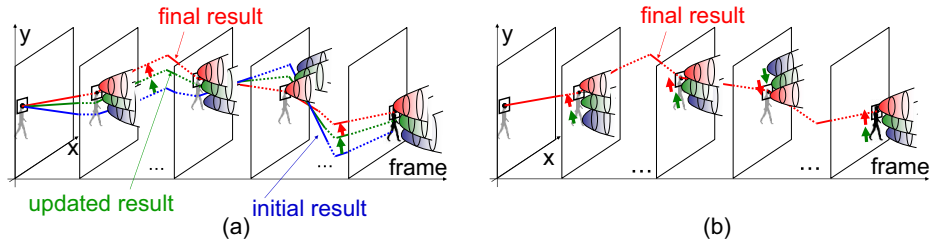


Fig. 7. Iterative optimization of (a) analytical DP tracker and (b) KLT tracker.

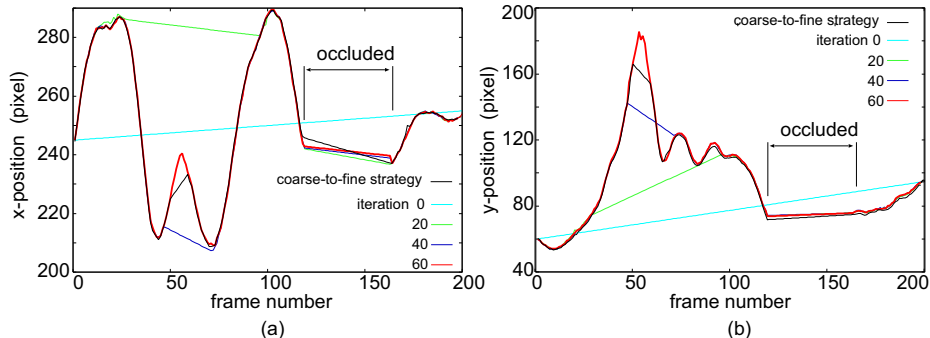


Fig. 8. Tracking result by bi-directional strategy. (a) The trajectory of x_1, \dots, x_T . (b) The trajectory of y_1, \dots, y_T .

iteration of KLT tracker is performed within a frame. In other words, a “local” iterative optimization is done. In contrast, the iteration of the analytical DP tracker is performed for all the tracking positions $\mathbf{w}_1, \dots, \mathbf{w}_i, \dots, \mathbf{w}_T$. Thus a “global” iterative optimization is done. Consequently, we can say that the analytical DP tracker with the iterative updating process is a global optimization version of KLT tracker ¹.

7.2 Bi-directional strategy

Bi-directional tracking in [15] is a tracking method where not only the initial position but also the final position are specified. Even for offline tracking, the specification of the final position is very beneficial to obtain better tracking accuracy.

This idea can be utilized in the iterative updating for improving computational efficiency. First, a linear trajectory connecting the two specified positions is considered as the initial tracking result. Then, it was used as the approximation center to derive quadratic cost function $d_i(\mathbf{w}_i)$ at all i . A difference from the iterative updating process in Section 5.2 is that the approximation center $\tilde{\mathbf{w}}_i^k$ is not set at $\bar{\mathbf{w}}_i^{k-1}$ but selected from $\bar{\mathbf{w}}_{i-1}^{k-1}$ or $\bar{\mathbf{w}}_{i+1}^{k-1}$ according to their scores. This

¹ Precisely speaking, there is another difference between analytical DP tracker and KLT tracker. That is, the former evaluates the smoothness of its tracking result whereas the latter does not.

selection mechanism is introduced for propagating the reliable tracking results around the both ends ($i = 1$ and T) gradually into the middle part ($i \sim T/2$) according to the iteration.

Figure 8 shows the change of the tracking result given by the bi-directional strategy. The propagation of accurate tracking results around both ends can be observed. After 60 iterations, we could have the tracking result which almost equals the result by the iterative updating of Section 5.2 (“Coarse-to-fine strategy” in Fig. 8). The bi-directional strategy does not require the initial tracking result by the conventional method on down-sampled frames and therefore it is more computationally efficient.

7.3 Multiple quadratic functions

In this paper, we assume the cost $d_i(\mathbf{w}_i)$ is represented by a quadratic function, i.e., a unimodal function. Unless the cost is originally quadratic, this representation is not perfect. In fact, the cost by (13) will be a multimodal function.

One possible remedy to improve the representation accuracy is multiple quadratic functions. This is analogous to Gaussian mixture. Specifically, K quadratic functions, $d_{i,k}(\mathbf{w}_i)$, $k = 1, \dots, K$, are prepared at the i -th frame for approximating the multimodal function. When optimizing \mathbf{w}_i by the analytical DP tracker, one quadratic function is selected from the K functions at each frame. It is interesting to note that the selection itself can be done optimally by the conventional DP.

8 Conclusion

In this paper, a novel DP-based tracking method, called analytical DP tracker, has been proposed. Quadratic representation of the tracking cost at each frame allows an analytical solution by DP. This solution is very effective; in fact, given the quadratic cost at each frame, we have globally optimal tracking result with only $O(T)$ computations, where T is the number of frames. In addition to this computational feasibility, it has been revealed experimentally that the analytical DP tracker could provide accurate tracking results against complete occlusions by fully utilizing its offline optimization property. As the practical problem, we have also discussed several techniques to derive the quadratic tracking cost by approximating some original tracking cost. Finally, it was shown that the analytical DP tracker can be considered as an offline version of the well-known KLT tracker.

Future work will focus on the extension for dealing with rotation and scaling. Contour models such as Schoenemann and Cremers [19] can also be employed as an extended target model. As noted before, those extensions will enhance the superiority of the proposed method in computational efficiency over the conventional DP-based tracking methods. In addition, we must investigate more accurate and sophisticated cost representation, such as multiple quadratic functions.

References

1. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. *ACM Comput. Surveys* **38** (2006) 1-45
2. Bellman, R., Dreyfus, S.: *Applied Dynamic Programming*, Princeton University Press (1962)
3. Angel, E.: Dynamic programming for noncausal problems. *IEEE Trans. Automatic Control* **AC-26** (1981) 1041-1047
4. Serra, B., Berthod, M.: Subpixel contour matching using continuous dynamic programming. *CVPR* (1994) 202-207
5. Tomasi, C., Kanade, T.: Detection and tracking of point features. Tech. Report, CMU-CS-91-132, Carnegie Mellon Univ. (1991)
6. Shi, J., Tomasi, C.: Good features to track. *CVPR* (1994) 593-600
7. Sakoe, H., Chiba, S.: A dynamic programming algorithm optimization for spoken word recognition. *IEEE Trans. ASSP* **ASSP-26** (1978) 43-49
8. Montanari, U.: On the optimal detection of curves in noisy pictures. *Comm. ACM* **14** (1971) 335-345
9. Amini, A. A., Weymouth, T. E., Jain, R. C.: Using dynamic programming for solving variational problems in vision. *TPAMI* **12** (1990) 855-867
10. Geiger, D., Gupta, A., Vlontzos, J.: Dynamic programming for detecting, tracking and matching deformable contours. *TPAMI* **17** (1995) 294-302
11. Akgul, Y. S., Kambhamettu, C.: A coarse-to-fine deformable contour optimization framework. *TPAMI* **25** (2003) 174-186
12. Barniv, Y.: Dynamic programming solution for detecting dim moving target. *IEEE Trans. Aerospace and Electronic Systems* **21** (1985) 144-156
13. Arnold, J., Shaw, S., Pasternack, H.: Efficient target tracking using dynamic programming. *IEEE Trans. Aerospace and Electronic Systems* **29** (1993) 44-56
14. Han, M., Xu, W., Tao, H., Gong, Y.: An algorithm for multiple object trajectory tracking. *CVPR* (2004) 864-871
15. Sun, J., Zhang W., Tang, X., Shum, H.: Bi-directional tracking using trajectory segment analysis. *ICCV* (2005) 717-724
16. Dreuw, P., Deselaers, T., Rybach, D., Keysers, D., Ney, H.: Tracking using dynamic programming for appearance-based sign language recognition. *FGR* (2006) 293-298.
17. Buchanan, A., Fitzgibbon, A.: Interactive feature tracking using k-d trees and dynamic programming. *CVPR* (2006) 626-633
18. Wei, Y., Sun, J., Tang, X., Shum, H. -Y.: Interactive offline tracking for color objects. *ICCV* (2007) 1-8
19. Schoenemann, T., Cremers, D.: A combinatorial solution for model-based image segmentation and real-time tracking. *TPAMI* **32** (2010) 1153-1164