Deformable Patch-based NCC Measure for Visual Tracking

Xian-guo YU\(^1\), Qi-feng YU\(^1\), Liang XIE\(^1\) and Jin-bo LIU\(^1\)

\(^1\)College of Aerospace Science and Engineering, National University of Defense Technology, Changsha, Hunan, 410073, China

\(^2\)Hunan Provincial Key Laboratory of Image Measurement and Vision Navigation, National University of Defense Technology, Changsha, Hunan, 410073, China

*Corresponding author

**Keywords:** Visual tracking, Deformable patches, Similarity measurement.

**Abstract.** This paper tries to solve the visual tracking problem in a simple and intuitive way. In particular, we extend the traditional NCC measure with deformable part-based models (DPM) to get a new image similarity measurement and feed it into the particle filtering framework for visual object tracking. The proposed target model consists of a root template and a set of patch templates. The root is the whole target image, and patches are sub-regions of the target image. NCC responses are calculated for the root template with test images, and every patch template with their corresponding sub-regions of test images. The likelihood between the target and a test image is calculated by integrating all template responses. Instead of directly accumulating all responses, we employ the idea of DPM that combines global and local responses using an elastic model to get a robust image likelihood measure. We also explore an exhaustive patch representation mechanism by introducing the dense NCC feature map. The proposed feature map enables the usage of all patches in the target region for target description and similarity measure. Finally, we provide both qualitative and quantitative experiments against state-of-the-arts on the TB50 dataset. Experimental results show that the proposed similarity measure is effective for visual tracking under challenging scenarios.

**Introduction**

Visual object tracking is an important research field in computer vision with various applications in auxiliary driving systems, UAV based ground targets tracking, vision surveillance and so on. Lots of tracking algorithms have been proposed so far to tackle this problem\(^1,2\). However, challenges brought by illumination variations, background clutters, target deformations, occlusions and real-time computation still prevent many algorithms to be applied in real life.

In a typical tracking system, target representation is considered a key point which directly decides the likelihood measure and the inference model used in tracking\(^1\). Target representation involves what kind of features to use and how to describe targets by those features. The most frequently used features are gray values\(^3\), histograms\(^4\), HOG\(^5\), etc. In this paper, we propose to use the NCC features for visual tracking. NCC features are induced from the NCC measurement\(^6\). We propose the concept of NCC feature vectors and the dense form NCC features, as well as how to use them to describe targets. New image likelihood measurement is given based on the proposed feature representation and object model. NCC features have a lot of advantages. Firstly, they are easy to compute. Each NCC feature vector is just a mean-subtracted unit vector. The dense NCC features can also be efficiently calculated by means of integral images. Secondly, they are invariant to affine illumination changes. This advantage is inherited from the general NCC measure\(^6\). Thirdly, they are location sensitive. NCC is widely used in image matching because it is accurate in locating targets in images. The proposed NCC features share these advantages thus can alleviate the model drift problem in tracking.

DPM\(^7\) has achieved great success since its first appearance in the 2007 VOC challenge. It represents objects by a root template and several part templates and uses an elastic model to combine global and local templates rather than stacking them casually. In this work, we borrow the idea of
DPM to build an elastic object model. We explore a simple yet effective part representation strategy with NCC features. Then global and local cues are collected and elastically combined to measure the similarity of a test image with the target.

As tracking algorithms are becoming increasingly complex[9], we show that simple methods can also achieve state-of-the-art performance while being efficient. Specifically, we augment the NCC measure with deformable patch-based models to obtain a robust image likelihood measure. The new object model and likelihood measure is then fed into the particle filtering framework for object tracking. The proposed method is simple and intuitive. Experimental results indicate that it is effective for a variety of tracking scenarios.

Related Work

NCC[6] is a classical image measurement which has been widely used in image matching. It is used to form the baseline tracker of the VOT challenges[10] and a building block of the PROST algorithm[11]. NCC tracker is robust with illumination changes but susceptible to target deformations thus it is often accompanied by optical flow[12].

Local representation based tracking algorithms are effective in handling appearance changes like partial occlusions and local deformations. The fragments-based tracker[4] represents object by multiple patches. Each patch is described by a gray histogram of 8 bins. EMD distance is used to compare patch-to-patch similarity. The tracking is achieved by adding votes from all patches to find the target position and scale. CMT[13] uses a keypoint based object model. Keypoints are initially detected in the target image and then mapped to subsequent frames by matching and optical flow tracking respectively. A special voting scheme is devised to find the target transformation.

Sparse techniques have been applied in the visual tracking field since the success of L1-tracker[14]. While sparse algorithms show power in handling appearance changes, the local sparse representations[15,16] are proved to be more effective than holistic methods[2]. The ASLA tracker[15] uses an alignment-pooling technique to exploit partial and spatial information of the target, which is demonstrated to be robust to alignment errors and background clutters[2]. The incremental PCA algorithm[3] is used to update each sparse template independently. The ODLT algorithm[17] works in a similar way. It extracts local image patches and represent them by sparse coefficients with respect to an over-complete dictionary. Then a classifier is trained online to distinguish the target from background based on sparse decomposition coefficients.

Deformable part models has been applied to visual tracking[18,19]. The part-based tracker[19] model the relationship between parts by online latent SSVM learning, which avoids lengthy initialization of part positions. Deformable models are proven to be very effective in object detection. While in object tracking, it is less popular. One of the reasons is that deformable models are very time demanding both in training and in detection process. Our work tries to make a simple and effective deformable model for tracking. We do not account for large deformations as in[20]. We found that by explicitly taking slight deformations into object model can we make a robust tracker.

Proposed Method

As local models are gaining popularity recently, two basic problems are generally overlooked: how to select patches and how to combine patch responses. This paper tries to deal with these problems in a straight-forward way. Firstly, we see that an intuitive idea to select patches is to use all the patches. Secondly, we borrow the idea of DPM that elastically combines all part responses elastically to account for target deformations. Therefore, a new part-based model is proposed. And we demonstrate that with this idea, the simplest NCC measure can be extended for robust tracking.

In this section, we firstly introduce the deformable part-based model based on NCC measure. We then enable exhaustive patch selection by introducing the dense NCC feature map.
Deformable Patch-based NCC Tracker

We describe the target by global and local templates, which correspond to patches inside target image. The templates are organized in a deformable part-based model to take advantage of target information at different levels. A new image similarity measure is then built on the proposed model with the NCC measure as basic component. We finally get a deformable patch-based NCC tracker (DPNCC).

![Initial frame](image)

**Figure 1.** Target model initialization: equally spaced patches of a certain size are extracted from the target region and are used to initialize the patch templates. The root template is initialized by the whole target image.

Our target model consists of a root template and a set of patch templates. The root template is a holistic representation of the target, and is represented by \( R \). A patch template is a rectangular template describing a sub-region within the target. Patch templates are represented by \( P_1, P_2, ..., P_n \) respectively. The relative positions of the patches to the root are fixed, represented by \( \omega_1, \omega_2, ..., \omega_n \).

Let \( M \) represent the target model. Then the similarity measure of a test image \( I \) with the target is defined as

\[
L(M, I) = NCC(R, I) + \sum_{i=1}^{n} \max_i (NCC(P_i, I(\omega_i + \delta_i)) - \phi(\delta_i))
\]  

(1)

where \( I(\omega_i + \delta) \) denotes a sub-region of \( I \) whose upper-left corner is defined by \( \omega_i + \delta \) and the size is same as \( P_i \), \( \delta = (\delta_x, \delta_y) \) is a small displacement representing deformations. \( S \) is a constant prior confining the deformation range. \( \phi(\delta) \) is the deformation cost corresponding to \( \delta \). In practice, a Gaussian shaped deformation cost function is used. Eq. (1) is actually an integration of the deformation rule and the structural rule used in DPM. It combines global and local templates in an elastic model so that target structure and local deformations are accounted for at the same time.

\( NCC(\cdot, \cdot) \) is the normalized cross-correlation measure. The NCC measure of two images is defined as

\[
NCC(A, B) = \sum_{x,y} (A'(x, y) \cdot B'(x, y)) = \text{dot}(A', B')
\]  

(2)

where \( A' \) and \( B' \) are mean-subtracted and L2-normalized images of \( A \) and \( B \) respectively. The normalization function is defined as

\[
\text{normalize}(A) = \frac{A - \text{mean}(A)}{\text{norm}(A - \text{mean}(A))}
\]  

(3)

We call the normalized vector a NCC feature, which is zero-mean and unit-length.

At the initial time, the target image is extracted and resized to a fixed size, e.g. 32x32, we select equally spaced patches with some overlaps within the target area as shown in Fig.1. The root template
and all the patch templates are initialized by the corresponding regions in the target image. We then normalize each template using Eq.(3).

Once initiated, the target model is ready for tracking. We employ the particle filtering framework for tracking. The major difference between different particle filtering trackers is the similarity measurement. In this paper, we adopt Eq.(1) instead.

The model updating problem is solved by an interpolation strategy

\[
T_{\text{new}} = \text{normalize}(T_{\text{old}} + \beta(I - T_{\text{old}}))
\]

(4)

\(T_{\text{old}}\) is the previous template, and \(I\) is the observation. Then \(T_{\text{new}}\) is interpolated from \(T_{\text{old}}\) and \(I\).

To avoid model degeneration that all elements in the template become zero, e.g. when \(I = 0\) and \(\beta = 1\), the updating threshold \(\eta > 0\) is introduced here.

\[
T_{\text{new}} = \begin{cases} \text{normalize}(T_{\text{old}} + \beta(I - T_{\text{old}})) & \text{if } \text{NCC}(T_{\text{old}}, I) > \eta \\ T_{\text{old}} & \text{otherwise} \end{cases}
\]

(5)

**Dense NCC Feature Map**

![Figure 2. Compute dense NCC features: each entry is calculated from an image patch. The patch size is fixed beforehand.](image)

The basic DPNCC tracker described in the previous section selects several patches at the initialization stage. The idea of using patch templates is to provide more cues where the target lies. One might want to select more patches to improve tracking performance. However, increasing the number of patches will lead to a computation rise. In the ultimate case, all patches are selected by sliding a window from the top-left corner to the bottom-right corner in the target region. This exhaustive patch selection scheme will not only make the previous tracker computationally intractable but also bring the problem of model representation. In this paper we introduce the dense NCC feature map to solve this problem.

In the previous section, we have introduced the concept of NCC feature, which is a zero-mean and unit-length vector. The entries in the NCC feature map all are NCC features and are calculated in the same way. As shown in Figure 2, we slide a fixed size window in an image. At each point \((x,y)\), the corresponding patch is extracted and stretched into a single vector. Then we normalize it to a NCC feature to get a feature entry at that position for the feature map. For an image of size \(m \times n\), if the patch size is \(m_1 \times n_1\), then the resulting NCC feature map will be \((m-m_1+1) \times (n-n_1+1)\), and the feature dimensions will be \(m_1 \cdot n_1\).

One of the advantages of the NCC feature map is the computational efficiency achieved by using integral images. To calculate the feature map of an image \(I\), we firstly calculate the integral sum and the integral square sum. Then the local sum and local square sum of every patch within the image region can be easily got, represented by \(S_1\) and \(S_2\) respectively. In the normalization step, we can easily get the mean of each patch and the L2 norm of the mean-subtracted patch by
\[ \text{mean}(\text{patch}) = \frac{1}{N} S_i, \quad \text{norm}(\text{patch}) = \sqrt{S_2 - N \cdot \text{mean}^2} \]  

(6)

where \( N \) is the number of pixels in the patch.

The dense NCC feature map is a vector field where each feature entry is a direction vector. The feature map is not just a concatenation of many independent features. It arranges all features like an image. Thus the operations that are used to process images can be applied to it. Here we propose to smooth the feature map before using it.

Given the target image, the dense NCC feature map is calculated. Due to texture changes on object surface, neighboring NCC features may differ greatly, which means the test image should have exactly the same appearance with the target to achieve a high similarity score. However, object appearance could change during tracking. To enlarge the capture range of each feature entry, we should make each feature not too distinct. We thus propose to smooth the model feature map by convolving it with a Gaussian kernel \( h_\sigma \)

\[ \hat{F} = \text{normalize}(F \otimes h) \]  

(7)

where the normalization function ensures that each entry of \( \hat{F} \) is a NCC feature.

The NCC feature map is a concrete representation of all patches. With the dense feature map, we can reform the target model simply by a root template and a feature map \( M = \{ R, F_0 \} \), where \( F_0 \) is the feature map of \( R \). We then reform the similarity measure of Eq. (1) as

\[ L(M, I) = \text{NCC}(R, I) + \sum_{x,y} \max_{|\delta| \leq \delta} \left( L_\delta - \varphi(\delta) \right) \]  

(8)

where \( L_\delta \) is a 2D response matrix obtained by

\[ L_\delta(x, y) = \text{dot}(\hat{F}_0(x, y), F(x + \delta_x, y + \delta_y)) \]  

(9)

where \( \hat{F}_0 \) is the smoothed version of the model feature map. \( F \) is the feature map of the test image. \( \hat{F}_0(x, y) \) denotes a feature entry inside \( \hat{F}_0 \). Both \( \hat{F}_0(x, y) \) and \( F(x + \delta_x, y + \delta_y) \) are NCC feature vectors. The \( \max(\cdot, \cdot) \) function in Eq. (8) picks maximum for each point across different response matrices obtained with different \( \delta \).

Figure 3. Relationship of NCC response with image overlap: experiment results on different sequences suggest that the NCC value becomes irrelevant when it is too small.

The feature map encourages the usage of all patches to vote for the target. Till now, all patches contribute equally to the final similarity measure. This, however, could bring problems because bad patches could lead to model drift. We investigated the relationship of NCC response with image overlap as shown in Fig. 3. We extract the target region in each frame of the Benchmark dataset and slide it in the whole image to get NCC response. Statistical results on all the sequences suggest that
NCC does not measure the similarity between two images when the NCC response is too low. We thus disregard those patches whose NCC responses are smaller than a certain threshold. We then replace \( L_d(x, y) \) by \( L'_d(x, y) \).

\[
L'_d(x, y) = \max\left( L_d(x, y), \lambda \right)
\]

where \( \lambda \) is the threshold constraining the lower bound of NCC response. In our experiment, we set \( \lambda \) to 0.05.

To update the target model online, we apply Eq.(5) to \( R \) and each entry of \( F_0 \) respectively.

**Experiment**

The effectiveness of the proposed similarity measure for visual tracking has been demonstrated using the particle filtering framework. We use 600 particles for the experiment. The root template size is set to \( 32 \times 32 \). We have tested different settings of patch template size and found size \( 2 \times 4 \) is a good compromise between effectiveness and efficiency. The update thresh \( \eta \) in Eq.(5) is set to 0.2. Increasing \( \eta \) leads to conservative updating strategy and the model may not adapt well to target appearance change. Otherwise, decreasing it leads to model drift with wrong observations. The learning rate \( \beta \) in Eq.(5) is set to 0.02\( \cdot \)NCC\((T_{old}, I)\), which means learning more aggressively when the observed image is more like the template. The deformation range \( S \) is limited to 1 pixel. Increasing \( S \) will lead to a computation rush. And we found that one pixel ambiguity for each patch is enough. On the other hand, it will lead to an apparent performance loss if no deformation is allowed, or \( S \) is set to 0. The proposed algorithm is implemented in C++ code on a X64 platform with 2.3 GHz i5 CPU and without GPU support. The tracker runs 30 fps on average.

We evaluate our tracker against the state-of-the-arts on the TB50 dataset of the Benchmark [2]. The dataset comes with ground truth annotations as well as manually tagged attributes, including illumination variation, occlusion, deformation and so forth. The tracking results are evaluated by two metrics, namely the center location error (CLE) and the overlap score. CLE computes average distance between the tracked object center and the ground-truth position across all the frames. The overlap score is computed by comparing the intersection area and the union area of the tracked target box and the ground-truth box. Both CLE and the overlap score are then thresholded to draw precision plot and success plot respectively.

Throughout the experiment, all parameters are kept unchanged except for the motion variance for the particle filtering based trackers. This is a bit different from the Benchmark evaluation methodology. Particle filtering tracking algorithms, including ours, generally concentrate on the appearance models and the similarity measures used in tracking and ignore the problem of motion model. When the particle distribution does not cover the target state, the trackers tend to fail. This may explain why few particle filtering tracking algorithm appear at top positions in various tracking competitions. In this paper, we are mainly concerned with efficient target representation and similarity measurement. Thus we set different motion sigmas for different sequences. All trackers share the same set of sigmas so we can focus on the effect of different target models on tracking performance. We re-run the particle filtering trackers ASLA[15], IVT[3], L1APG[21], SCM[8] with selected sigmas for all the sequences. Other tracking results of various algorithms are from the Benchmark toolkit.

**Results**

Figure 4 shows some screenshots of the evaluation results of 6 top-performing algorithms, from which we can see that the proposed DPNCC tracker can handle complex scenarios effectively. The *BlurBody* and the *BlurCar2* sequences have severe motion blur. But it seems to be no problem for most trackers. The ASLA failed tracking these two sequences mainly because of abrupt motion rather than image blurring. The *Box* and the *Liquor* sequences pose challenges of full occlusions. None of those trackers except for TLD can handle full occlusions, but the proposed tracker is able to recapture
the target after short-term occlusions, meaning it has a large capture range than other particle filtering algorithms. Struck does not account for scale changes, thus cannot track CarScale. Struck is able to recover when the object size changes back sometimes. For the difficult DragonBaby sequence, our tracker is able to capture the target sometimes. This is due to that we only update when the NCC response of the test image with the target template is above a threshold. The failure of the proposed method in tracking Human3 and Woman sequences indicate our lackage of the ability to deal with target pose change. All the 6 trackers failed on sequence MotorRolling, where the target takes up a small fraction of the window.

Figure 5 corresponds to the OPE evaluation results on the TB50 dataset. The proposed method achieves comparable result with top-performing algorithms. Noting that other trackers including ASLA, SCM and Struck, uses either local sparse representations or the online structured learning, all are much more complex than our method. Meanwhile, our tracker runs in real-time. Fig. 6 shows detailed success plots under various challenging scenarios. The experiment shows that the proposed method is able to handle different challenging scenarios effectively.

Figure 4. Qualitative results on challenging scenes. The sequences from top to bottom are BlurBody, BlurCar2, Box, CarScale, DragonBaby, Dudek, Human3, Liquor, MotorRolling, Walking2, Woman respectively.
Figure 5. Precision plot and success plot of the top 10 trackers on the TB-50 dataset. The average distance error and the average overlap score for each tracker across all the sequences are included in the legends.

Figure 6. Success plots with respect to 4 challenges including fast motion, background clutter, motion blur illumination variation.

Conclusion

In this paper, we propose a new deformable patch-based target model and corresponding similarity measure. Instead of using complex learning methods, we start from the simplest NCC measure and explore a naive and intuitive way to tackle the visual tracking problem. Surprisingly, the experimental result shows that the proposed method achieves comparable results with complex local sparse based algorithms, noting that our method is simple and runs in real-time. Besides, the major computation burden of the proposed algorithm is to calculate dot-productions between matrices in Eq. (9), where GPU could be more effective.

The efficiency and effectiveness of the proposed similarity measure has been demonstrated for visual tracking under the particle filtering framework. In future work, we will explore to use the target model more smartly, in more pattern recognition applications.

Acknowledgment

This work is supported by the National Key Scientific Instrument and Equipment Development Project under Grant No. 2013YQ140517.
References


