PCA based appearance model for tracking

Vincent Zhang

May 2, 2016

1 Abstract

In this project, we studied PCA based appearance model in visual tracking. We implemented PCA AM module in MTF framework and attempted different ways of formulating PCA. In the experiment sections, we demonstrate the tracking performance of offline PCA when combined with different search methods. PCA + Forward Compositional has shown great tracking accuracy and robustness. It indicates a good potential for online PCA method. In the end, we suggests the future work in how we can use it with nearest neighbor to improve the accuracy.

2 PCA-based Appearance Model (AM)

Search method usually relies on a similarity metric which determines how close a patch is from the template. This similarity is computed mostly directly from image intensity, for example Sum of Squared Difference (SSD). It is very sensitive to view point and illumination change. The idea of PCA-based AM is to reduce this sensitivity by incorporating different appearance into the eigen basis.

Let I denote the matrix of training images, the dimension is $n \times p$, where n is the number of pixels in an image, p is the number of images.

We assume the following pre-processing step for the rest of the paper: subtract the mean image I_{mean} from all images, denote the centered image as I_c.

There are two ways of constructing the PCA basis.

Approach I:

Compute covariance matrix $cov(I_c^T I_c)$, apply eigen decomposition to it to get the eigenvectors Y. Then we project I_c onto the eigenvector to get the image basis: $B = I_c * Y$. Here I_c is $n \times p$, Y is $p \times k$, B is $n \times k$, where k is the number of eigen basis the we want to keep which is smaller than the number of training images.

For a given new image, I_{new}, we subtract I_{mean} from it, project it onto eigen basis to get the coefficient $y_{new} = B^T I_{new}$, where $B\, y_{new} = I_{new}$. This coefficient y_{new} is used as the *feature vector*. Its length is only $k < p$. This raises a concern in its practicality in tracking as a feature vector since it may not contain enough information. It seems to be well addressed by the approach II below.

Approach II:

Similar to approach I, except that instead of computing covariance matrix $cov(I_c^T I_c)$, we now do $cov(I_c I_c^T)$. The truncated eigenvector B can be directly used as the image basis. This is equivalent to doing $SVD(I_c) = U\Sigma V^T$. The eigen basis has a dimension of up to the number of pixels. This means we have a richer set of basis which may give better representation.

Similarity metric:

Once we have computed the basis, we may use it in two different ways as explained in this section:

2.0.1 Eigen coefficients as the feature:

Suppose we have two new images, I_1, I_2 and a template image I_T. We first do the mean subtraction $\hat{I}_i = I_i - I_{mean}$, $i = 1, 2$. then compute the coefficients y_i for each image by $y_i = B^T \hat{I}_i$. We also compute the feature for the template $y_T = B^T \hat{I_T}$.

SSD, NCC can both be applied to this feature vector. It turns out that using coefficients is almost equivalent of using intensities.

We are showing here that the SSD score between similar patches will still be similar with the eigen coefficients as the feature. Same thing can be shown for NCC. This suggests that eigen coefficients are not a better metric than pure SSD.

For example, for similar I_1 and I_2,

$$\begin{aligned}
SSD(I_1, I_T) - SSD(I_2, I_T) &= \sum (I_1 - I_T)^2 - \sum (I_2 - I_T)^2 \\
SSD(y_1, y_T) - SSD(y_2, y_T) &= \sum (B^T I_1 - B^T I_T)^2 - \sum (B^T I_2 - B^T I_T)^2 \\
&= \sum (I_1 - I_T)^T BB^T * (I_1 - I_T) - \\
&\quad \sum (I_2 - I_T)^T BB^T * (I_2 - I_T) \\
&= BB^T \sum (I_1 - I_T)^2 - \sum (I_2 - I_T)^2 \\
&= BB^T [SSD(I_1, I_T) - SSD(I_2, I_T)]
\end{aligned}$$

where, $BB^T \approx Identity$.

2.0.2 Reconstruction error as similarity function

We use the basis to reconstruct the current patch and use the reconstruction error (SSD) as the similarity

$$f = \hat{I}_t - BB^T \hat{I}_t$$

where $\hat{I}_t$ denotes the mean subtracted image $I_t - I_{mean}$ at frame t.

This is often how PCA-based tracking is formulated in the literature [1, 2].

Note that there's a similar formulation which is basically doing SSD between the current patch and the reconstructed template:

$$f = \hat{I}_t - BB^T \hat{T}$$

This template can either be T_0 or T_{t-1} (with mean image subtracted). We found that T_{t-1} yields better accuracy than T_0, which makes sense since T_0 only reflects the old appearance. It is in fact almost the same as the reconstruction error where we are doing $\hat{I}_t - BB^T\hat{I}_t$, since I_t is similar to I_{t-1} in most cases.

3 Experiments

The section is divided into three part, where PCA-based appearance model is combined with other search methods.

The main goal is to figure out how PCA works in a perfect setting, where the basis can be computed from the ground truth to accurately incorporates the appearance information. This "cheating" way allows us to evaluate the best performance that can be achieved by PCA. Further work can be done to replace the offline PCA computation with online PCA update.

For all three parts, the PCA uses a local basis where SVD is applied only on a number of recent frames prior to the current one, instead of using all the possible appearance. We found that global basis misleads the search in some occasions since it is inherently biased towards the appearance that shows up more in the sequence. If the current frame is of a rare appearance, the reconstruction error will be fairly high and there may not present a good minimum.

I'm not including the tracking image sequence since it's already presented in the demo. The experiment is described below with some analysis on the similarity plot and Jacobian/Hessian followed.

The two harder image sequences used in the demo are "Acronic" and "Bear" sequence in the PAMI dataset.

3.1 PCA + Particle Filter

It works well for 3 DOF but not for higher DOF since the it's highly sensitive to the σ in the Gaussian noise model. Fine-tuning is often needed to adjust the parameters to individual sequence.

3.2 PCA + IC

With gradient-based search methods like IC and FC, we can do higher-DOF tracking.

PCA + IC works better than SSD in 8 DOF tracking in that it tracks the object for a longer time. But it still fails later in the tracking sequence. It is possible that by incorporating the PCA basis into the template (we use BB^TT instead of T), we are introducing some error to the Jacobian $\frac{\partial f}{\partial T}\frac{\partial T}{\partial p}$. This error accumulates which causes the tracker to fail.

3.3 PCA + FC

It works extremely well. In most test sequences, it was able to track accurately all the time. The only exception is that it failed due to a sudden large illumination change, but I could not reproduce it. This kind of change is hard for other types of tracker and we found that only trackers specifically designed for this particular change works consistently, such as lscv [3]. Further testing on more

sequence and dataset is needed to find a case where PCA + FC consistently fails.

Again, we're attributing the performance to the Jacobian $\frac{\partial f}{\partial I}\frac{\partial I}{\partial p}$, which now does not depend on the basis (strictly speaking, it does, since $\frac{\partial f}{\partial I}$ includes BB^T, but not as much as it affect IC). In the meantime, the basis is making the template more robust which leads to a much better performance than SSD.

3.4 Analysis

We first look into how the similarity function changes as the tracked patch moves on x and y axis, shown in Fig. 2. This plot is generated from the first frame of the book sequence 'nl_bookI_s3': 1.

We see that the similarity metric is very similar between SSD and PCA. PCA has a slightly steeper gradient which may leads to faster convergence speed. Note that for a through analysis, it needs to be shown for the frame where ssd fails to better understand the difference. I haven't got the time but plan to implement later.

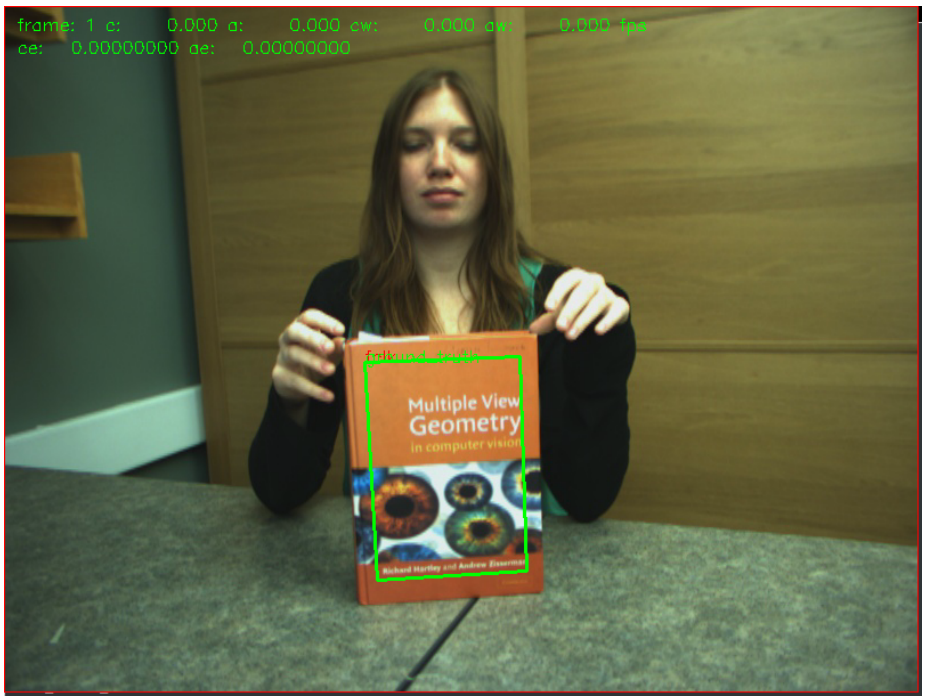
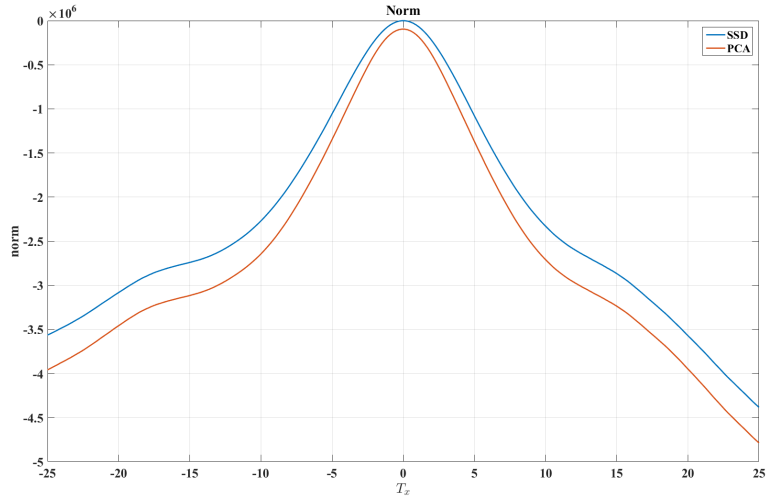


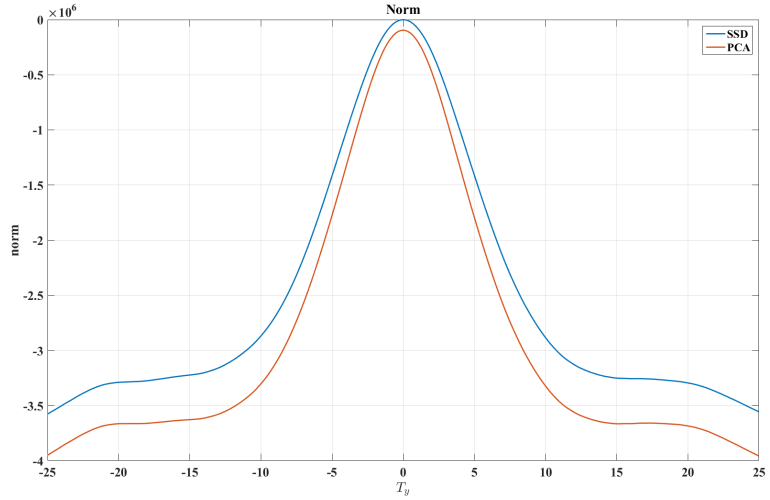
Figure 1: Frame 0

We also verified that the Jacobian was correctly implemented by comparing with numerical Jacobian, shown in Figure 3. They almost completely overlap which shows the correctness of the implementation.

As an attempt to figure out why IC did not work as well as FC. we looked into the Hessian, which is shown in Figure 4. We mainly look at two curves here: the solid red line 'Std2' and the solid cyan line 'InitSelf2'. They refer to the true Hessian, as described in Eq.(4) in [4] and initial self hessians described

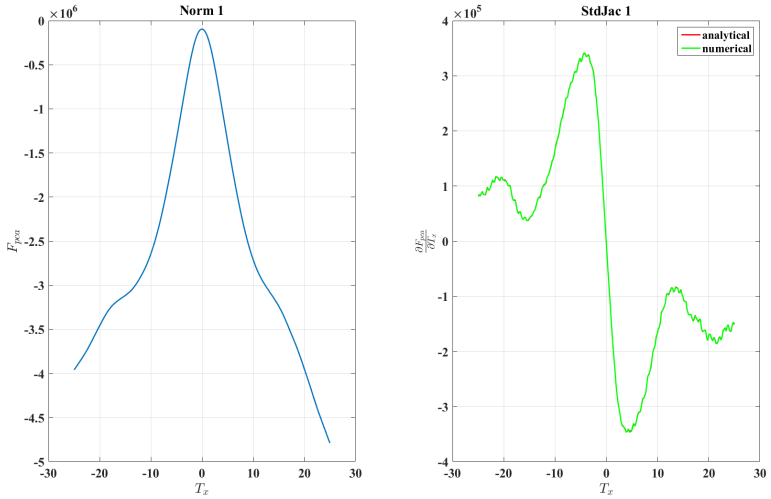


(a) Similarity Plot with x translation

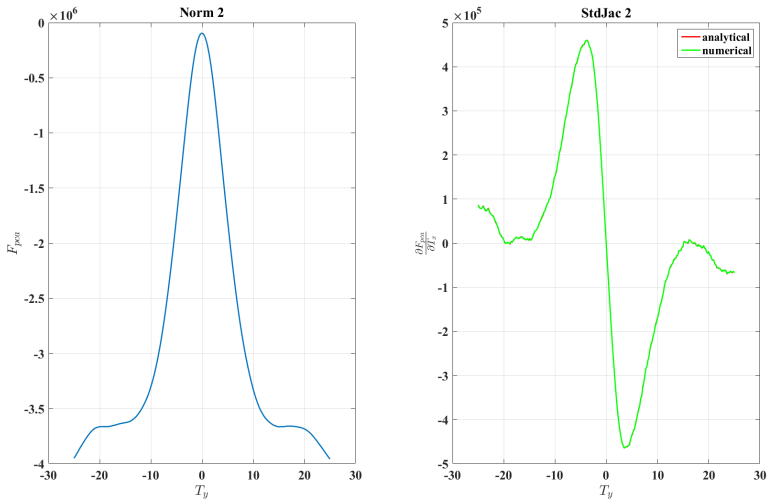


(b) Similarity Plot with y translation

Figure 2: Similarity Plot with translation



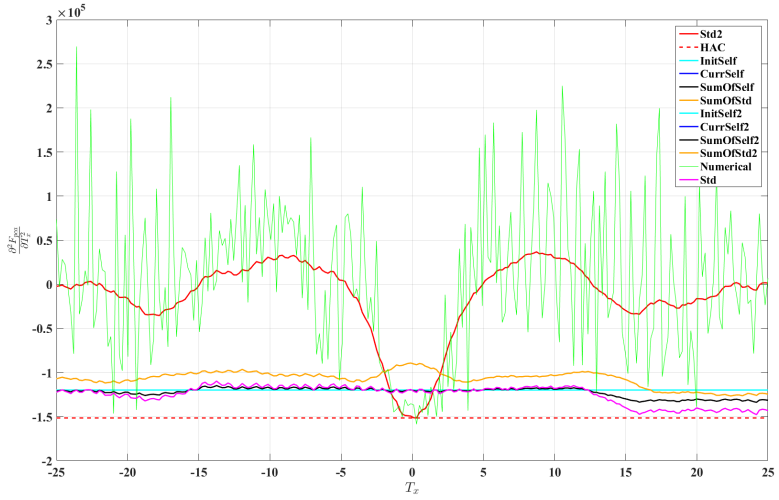
(a) Jacobian under x translation



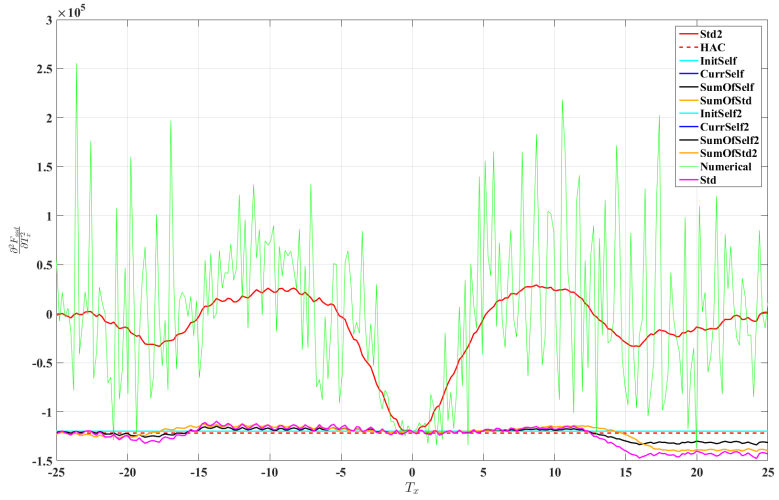
(b) Jacobian under y translation

Figure 3: Jacobian under translation

in Eq. (14), respectively. We can see two things here: 1. From the equation, the initial self hessian depends heavily on the template I_0, which is changed into $BB^T\hat{I}_0$ in PCA. This difference may introduces error. 2. For SSD + FC/IC, the true hessian can be approximated with the initial hessian which usually yields good performance. At translation $Tx = 0$, true hessian is in fact equal to the approximated hessian, which can be found also in the plot. However, for IC, it's not equal anymore, since we are using $BB^T\hat{I}_0$ to approximate it, where true Hessian at $Tx = 0$ is using $\hat{I}_0$. FC+PCA on the other hand, does not suffer from this inaccuracy.



(a) Hessian of PCA



(b) Hessian of SSD

Figure 4: Hessian

4 What's up next?

1. Online PCA:
   Implement online PCA update to replace the offline computation

2. Benchmark IC vs FC:
   Conduct quantitative analysis on IC vs FC to understand the difference when combined with PCA method

3. NNIC+PCA:
   PCA + particle filter has been implemented in IVT. But no one has applied it with NN as search method.

   A reflection on NNIC approach is that the NN is done simply by SSD on intensity. To make this step more accurate, we can introduce PCA reconstruction error. Concretely, we keep an online eigen basis which is computed by the new image and previous few images in the history. Then for NN, we simply choose the one that gives the lowest reconstruction error instead of the one that gives lowest SSD. For testing, we can start with offline basis. For each frame, I store a basis based on this frame and the frames before it. We tested that doing an SVD on a 2500 by 50 matrix in eigen library takes about 0.1 seconds. So the run time performance will be okay. The hypothesis is that this NN+PCA+IC approach will yield better tracking performance by adding more accuracy to NN and more robustness to IC. We can then compare PCA+NNIC with PCA+FC.

4. Wait, are we ignoring eigenvalues ?
   Existing approaches in the literature only take advantage of PCA by keeping a fixed amount of eigen vectors. We are ignoring a potentially important factor: the eigen values. It should show the distinction between high-frequency and low frequency component. Also it can tell us about the confidence of each eigen vector in its power to reconstruct the original image. An open question that I would specifically like to explore is, when using PCA, instead of just combining the basis to reconstruct the image, can we use them separately, in a probabilistic setting? What if we implement a Bayesian framework in use with nearest neighbor where we search for neighbors for each basis image and decide which one, or which subset of the basis vectors to use to achieve better performance as the appearance changes? This idea is inspired by discussion with Martin today about choosing proper eigen basis on-the-fly.

References

[1] David A Ross, Jongwoo Lim, Ruei-Sung Lin, and Ming-Hsuan Yang. Incremental learning for robust visual tracking. *International Journal of Computer Vision*, 77(1-3):125–141, 2008.

[2] Michael J Black and Allan D Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. *International Journal of Computer Vision*, 26(1):63–84, 1998.

[3] R. Richa, M. Souza, G. Scandaroli, E. Comunello, and A. von Wangenheim. Direct visual tracking under extreme illumination variations using the sum of conditional variance. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 373–377, Oct 2014.

[4] Abhineet Singh and Martin Jägersand. Modular tracking framework: A unified approach to registration based tracking. *CoRR*, abs/1602.09130, 2016.