IEICE TRANS. INF. & SYST., VOL.E98–D, NO.11 NOVEMBER 2015

LETTER

Mixture Hyperplanes Approximation for Global Tracking

Song GU[†a)], ***Member***, **Zheng MA**[†], ***and*** **Mei XIE**[††], ***Nonmembers***

SUMMARY Template tracking has been extensively studied in Computer Vision with a wide range of applications. A general framework is to construct a parametric model to predict movement and to track the target. The difference in intensity between the pixels belonging to the current region and the pixels of the selected target allows a straightforward prediction of the region position in the current image. Traditional methods track the object based on the assumption that the relationship between the intensity difference and the region position is linear or non-linear. They will result in bad tracking performance when just one model is adopted. This paper proposes a method, called as Mixture Hyperplanes Approximation, which is based on finite mixture of generalized linear regression models to perform robust tracking. Moreover, a fast learning strategy is discussed, which improves the robustness against noise. Experiments demonstrate the performance and stability of Mixture Hyperplanes Approximation.

key words: *template tracking, Mixture Hyperplanes Approximation, fast learning, regression*

1. Introduction

Object tracking has many applications in computer vision such as surveillance, vision-based control and visual reconstruction. Many tracking approaches focus on selecting one of the moving object's features as in [1]–[3]. These methods construct a classifier according to the features of the object and the background to distinguish them, and are called feature-based tracking. On the other hand, global-based tracking approaches depend their ability to treat complex templates or patterns that cannot be modeled by local features. They are robust and have been extensively used. Some very successful learning based template trackers are proposed by [4], [5]. They are based on learning linear predictors to efficiently compute template warp parameter updates. Thanks to extensive training, these methods are very fast and tend to avoid local minima. They treat the tracked object (template) as a whole, and find the relationship between the intensity difference of the target and the moving parameters in two consecutive frames. Traditional global-based tracking methods such as [4], [5] abstract directly the template's color as feature, and construct a motion model by warping function. The object is tracked based on the assumption that the relationship between the intensity difference and the warping parameters in two successive frames is linear. This will result in large error in real-time tracking. [6] proposed a second-order approximation of the image difference and achieved a higher convergence rate. However, it still suffers from high complexity which makes it unsuitable for real-time tracking. In addition, an appropriate high-order model is hard to select. In this paper, we propose a new framework that addresses this problem.

This paper proposes a novel global-based tracking approach based on a mixture of generalized linear models by allowing the approximation of the parameters in a data-driven way without specifying the distribution approximation in advance. Mixture model estimation and generalized linear model theory are utilized with Expectation Maximization (EM) algorithm to regress the object motion model. In current global-based tracking approaches, such as [4], the computation of the linear predictors requires the costly inversion of a large, template specific matrix. Instead of computing the inversion of the whole matrix, a high dimension matrix is mapped to a low rank space by simple matrix transformation in this paper. The proposed fast learning strategy decreases the time consuming of the inverse operation. Moreover, all noise outside of the low rank space has no effect when the fast learning strategy is implemented. This will increase the robustness against noise in our framework. We will prove this in our experiments.

2. Mixture Hyperplanes Approximation

Our proposed approach is based on finite mixture models algorithm which can express a global information of the ground truth model more accurately than single model. In this section, Mixture Hyperplanes Approximation will be analyzed in detail. We adopt almost identical notations as those proposed by [4], [7] in order to make the reading easier.

2.1 Template and Motion Model

We select a region in the first frame of a video sequence, which defines the region of interest as *target region* that we want to track. The location of the *target region* in an image is defined by R which stores the position of all the corner points. Without loss of generality, the *target region* is defined by a rectangle in this paper. Suppose $I(R, t)$ is a vector of the brightness values of n_p sample points within *target*

Manuscript received February 12, 2015.
Manuscript revised July 24, 2015.
Manuscript publicized August 13, 2015.

[†]The authors are with the School of Communication and Information Engineering, University of Electronic Science and Technology of China, P.R.China.
[††]The author is with the School of Electronic Engineering, University of Electronic Science and Technology of China, P.R.China.
a) E-mail: gusong1215@sohu.com

DOI: 10.1587/transinf.2015EDL8040

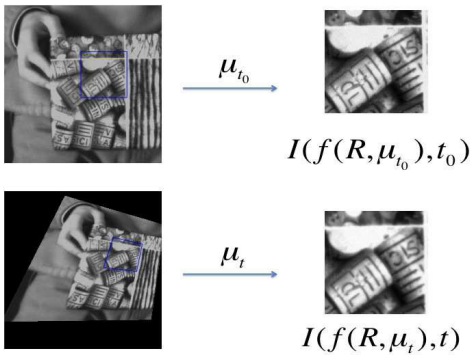


Fig. 1 Example of template and motion model

region instead of all the pixels in it at time t. Note that R is different at each time. Then $I(R,t_0)$ is the brightness values of *target region* in the first frame, which is defined as the *template*, t_0 is the initial time. The relative motion between the object and the camera induces the change in the position of the *template* in the image. The motion can be modeled by a parametric *motion model* $f(R,\mu_t)$, where μ_t denotes a set of parameters at time t. In our implementation, homography is used. Homography motions can be modeled by using an eight parameters model, given by the following matrix

$$H = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & 1 \end{pmatrix} \tag{1}$$

So μ_t is an 8×1 vector and is defined as $\mu_t = [a_{11}, a_{12}, a_{13}, a_{21}, a_{22}, a_{23}, a_{31}, a_{32}]^T$. In the *motion model*, function f can be written as a product of matrices

$$f(x,\mu_t) = H(\mu_t)x \tag{2}$$

where $x \in R$ is written with homogeneous coordinates x=(sx,sy,s) and H is a 3×3 matrix. We define $I(f(R,\mu_t),t)$ as the brightness values of transformed *target region* because of the relative motion, and $I(f(R,\mu_{t_0}),t_0)$ as the brightness values of initial transformed *target region* to *template*. Figure 1 shows an example of template and motion model.

2.2 Mixture Hyperplanes Approximation

Given a *target region* in the first frame, the corresponding transformations and brightness values are stored in μ_{t_0} and $I(f(R,\mu_{t_0}),t_0)$, respectively. From [4], we can get the warping parameter updates function formulated

$$\delta\mu = A\delta i \tag{3}$$

where $\delta\mu = \mu_t - \mu_{t_0}$ and $\delta i = I(f(R,\mu_{t_0}),t_0) - I(f(R,\mu_t),t)$.

The key of tracking object correctly is to find a suitable hyperplane approximation matrix A. However, from simple numerical application in [4], we know that the relationship between both parameters is not linear completely. If we solve this problem simply by a single linear model, it will result in larger error. Certainly, we can construct a nonlinear model to solve this question as [6]. Although, faster convergence rates for larger convergence areas can be additionally obtained by using a high-order instead of a first-order approximation of the error function, it is difficult in finding an appropriate nonlinear model to regress the predictor in different magnitudes of displacements and different template size. [8] has showed an experiment where linear predictors are superior to [6]. Moreover, an inappropriate high-order model is more sensitive to noise than linear model. It is illustrated in our experiments. Based on our opinion, it is often found that improved performance can be obtained by combining multiple linear models together. We consider M linear model components, each governed by its own predictor A_m. Then the warping parameter updates Eq. (3) can be modified as

$$\delta\mu = \sum_{m=1}^{M} \pi_m A_m \delta i \tag{4}$$

where π_m is a weight for each component. It also can be viewed as the confidence of each component, and it satisfies $\sum_{m=1}^{M} \pi_m = 1$ *and* $\pi_m > 0, \forall m$.

2.3 Hyperplane Approximation Learning

The key of our approach is to find appropriate parameters A_m and π_m for each component during the learning phase, where A_m is an $8 \times n_p$ matrix and π_m is a scalar. The learning process uses n_t random transformations on the template, where $n_t \gg n_p$. These transformations are small disturbances $\delta\mu_i = \mu_{t_0} - \mu^i_{t_0}, i = 1,2,\ldots,n_t$ to the initial transformed parameters. As a consequence, this introduces a change in the image brightness values to *template*, δi_i, where $\delta i_i = I(f(R,\mu_{t_0})) - I(f(R,\mu^i_{t_0})), i = 1,2,\ldots,n_t$. We construct matrix $Y = [\delta\mu_1, \delta\mu_2, \ldots \delta\mu_{n_t}] = [y_1, y_2, \ldots, y_{n_t}]$ and $H = [\delta i_1, \delta i_2, \ldots, \delta i_{n_t}] = [x_1, x_2, \ldots, x_{n_t}]$, where H is an $n_p \times n_t$ matrix, Y is an $8 \times n_t$ matrix. Inspired by [9], the assumption that the dependent variable follows a Gaussian distribution is relaxed in the generalized linear model framework. This signifies that $y|x \sim N(y|\mu,\beta^{-1})$, where $N(\bullet)$ is the multivariate Gaussian distribution, μ is the mean vector which is equal to $A_m x$ with respect to each component, and β^{-1} is covariance matrix. Then the mixture distribution of target can be written as $p(y|\theta) = \sum_{m=1}^{M} \pi_m N(y|\mu,\beta^{-1})$, where θ is the vector of all parameters which consists of the component weights and the component specific parameters. Given a data set of observations $\{x_n, y_n\}$ $n = 1,2,\ldots,n_t$, the log likelihood function for this model, then takes the form $lnp(Y|\theta) = \sum_{n=1}^{n_t} ln(\sum_{m=1}^{M} \pi_m N(y_n|A_m x_n,\beta^{-1}))$.

In order to maximize this likelihood function, EM algorithm [9] is adopted to obtain the following parameters

$$\pi_m = \frac{1}{n_t} \sum_{n=1}^{n_t} \gamma_{nm} \tag{5}$$

$$A_m = Y R_m H^T (H R_m H^T)^{-1} \tag{6}$$

$$\frac{1}{\beta} = \frac{1}{n_t} \sum_{n=1}^{n_t} \sum_{m=1}^{M} \gamma_{nm} (y_n - A_m x_n)^2 \tag{7}$$

where

$$\gamma_{nm} = E[z_{nm}] = p(y|x,\theta^j) = \frac{\pi_m N(y_n|A_m x_n, \beta^{-1})}{\sum_{l=1}^{M} \pi_l N(y_n|A_l x_n, \beta^{-1})} \quad (8)$$

To improve invariance to illumination changes, normalization is used on the extracted image data by imposing zero mean and unit standard deviation.

2.4 A Fast Learning Strategy

Considering Eq. (6), the result of HR_mH^T is a $n_p \times n_p$ matrix, and usually n_p is a large number. So it is evident that the computation of A_m is time-consuming due to the inverse of HR_mH^T. To increase the learning speed, we propose to use the inverse of small matrix instead of the one of large matrix. Setting $C = YR_mH^T$ and $B = HR_mH^T$, it leads to $C = A_mB$. Transforming Eq. (6), we obtain $I = A_mBC^T(CC^T)^{-1}$. Setting $D = BC^T(CC^T)^{-1}$, we obtain $I = A_mD$, where D is an $n_p \times 8$ matrix. To learn A_m, we compute

$$A_m = (D^T D)^{-1} D^T \quad (9)$$

Note that R_m is a diagonal matrix, it leads to $R_mH^T = (HR_m)^T$.

It is noteworthy to mention that the computation of the matrix A_m involves two matrix inverse such as $(CC^T)^{-1}$ and $(D^T D)^{-1}$. Both are 8×8 matrices. The advantage of fast learning strategy is as follows:

- Computing the inverse of two 8×8 matrices is much faster than computing the one of an $n_p \times n_p$ matrix, especially when n_p is a large number.
- The n_p is a variable with respect to the size of *target region*. However, the rows of matrix Y is constant when homography is used. That is to say that original learning speed is related with the size of *target region*, but the fast learning speed is unrelated with it.
- Considering Eq. (6), note that R_m is a diagonal matrix again, YR_m corresponds to weight for each column of matrix Y, and YR_mH^T corresponds to orthogonally project weighted Y on H. Similarly, since $BC^T = HR_mH^THR_mY^T$, where $R_mH^THR_m$ is a diagonal matrix, Eq. (9) corresponds to orthogonally project weighted H on Y. Given that we project weighted H on Y, all noise outside of the low rank space represented by Y has no effect. This makes Eq. (9) less sensitive to noise than Eq. (6). We will illustrate the tracking performance against noise of our fast learning approach in experiments section.

Inspired by [10], we divide all data set into M groups randomly, and then initialize each component predictor A_m based on these groups. The computing of initialization is based on [4]. β is initialized to the reciprocal of the true variance of the set of target values y_i. Note that the error of each data set takes the form $y_i - A_m x_i$ with respect to each component. π_m is initialized as $\frac{1}{M}$. In learning stage, the maximum iteration times are 5 times. Alg.1 formalizes the applied learning approach.

Algorithm 1 Mixture Hyperplanes Learning

input: H, Y, M
output: A_m , π_m
function LEARNING(H, Y, M)
Divide the matrix Y and H into M subsets by randomly selecting the columns of them correspondingly
Compute A_m by [4]
Initialize other parameters such as:$\pi_m \leftarrow \frac{1}{M}$ and β
for iteration = 1… **do**
Compute γ_{nm} by Eq. (8)
Update π_m by Eq. (5)
Estimate A_m by Eq. (9)
Update β by Eq. (7)
end for
end function

2.5 Mixture Hyperplanes Tracking

Tracking stage proposed in [4] is adopted as our tracking algorithm. We slightly modify [4] and instead Eq. (3) as warping parameter update function with Eq. (4). By experiments, we find that updating motion parameter μ_t iteratively can improve the tracking performance as well. The iteration number is 3 in our approach.

3. Experiments

We conduct three experiments. The first experiment provides quantitative comparisons between our two approaches (original learning and fast learning) and two alternative approaches, HA [4] and ESM [6], in different types of motions and levels of noise. The second experiment evaluates the performance of our approaches and two alternative approaches in challenge sequence. The performance of each experiment is measured by calculating

$$\frac{1}{4}\sum_{i=1}^{4} \| \vec{x_{ti}} - \vec{x_{gi}} \|_2 \quad (10)$$

where $\vec{x_{ti}}$ is the one of the four-tracked corner points coordinate, and $\vec{x_{gi}}$ is the one of the benchmark. We illustrate a real time tracking example of our approach in the last experiment.

For the implementation of HA, we programmed binaries by ourselves based on [11]. Homography warping algorithm is from [11] which is publicly available. To improve invariance to illumination changes, normalization is used. ESM algorithm is also programmed by ourselves based on the publicly available binaries. All of the algorithm and our proposed algorithm are implemented in MATLAB 7.10.0.

3.1 Basic Motions Comparison

In this section, we analyze the performance of our approaches on the robustness of tracking with respect to different basic transformations such as rotate, translate, scale (zoom in and zoom out) and view point change. Especially

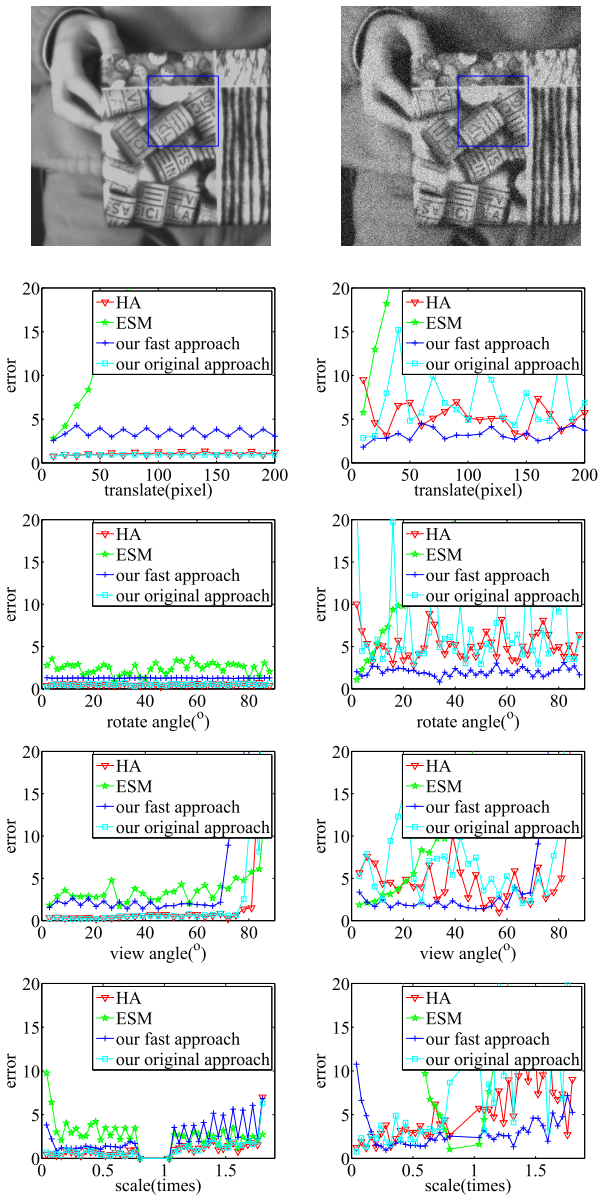
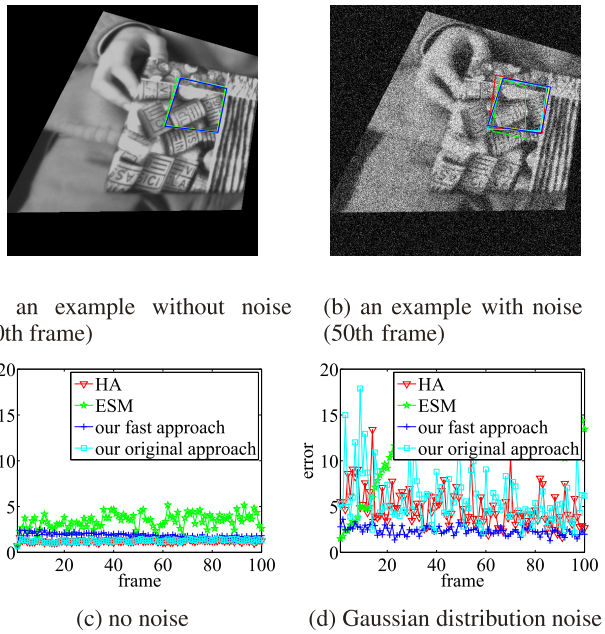


Fig. 2 Basic transformation examples

view point change is the rotation about the vertical axis. Figure 2 shows the image and the *target region* we want to track. The size of *target region* is 150×150. For decreasing the number of sample data, subsample is adopted on the *target region*. The step of subsample is 10, and the length of the input vector δi is 225. The left column of Fig. 2 illustrates the performance of four approaches when no noise is induced, and the right column shows the performance when images is corrupted by Gaussian distribution noise. When no noise is induced, our original approach and HA perform better than other approaches, and our fast learning approach has a slightly worse tracking robustness as shown in the left column of Fig. 2. However, our fast learning approach performs almost similarly regardless of noise and outperforms the other approaches in terms of sensitivity to noise as illustrated in the right column of Fig. 2.



(a) an example without noise (50th frame)

(b) an example with noise (50th frame)

(c) no noise

(d) Gaussian distribution noise

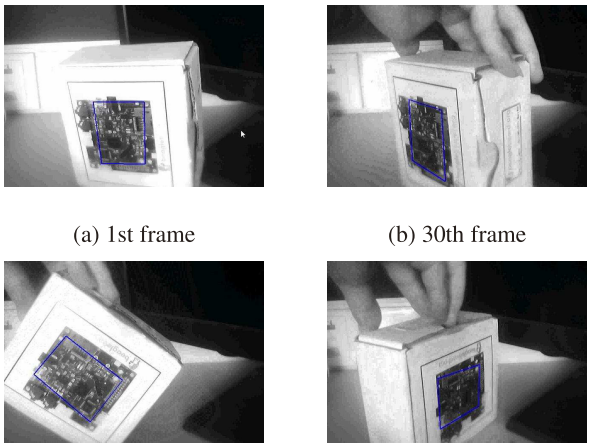
Fig. 3 Successive movements examples. 3(a) and 3(b) show two examples of tracking result by four different methods: the region marked by yellow is the ground truth; the region marked by red is HA; green is ESM; blue is our fast approach; cyan is our original approach. 3(c) and 3(d) show the quantitative comparison of the tracking performance by four different methods.

3.2 Successive Movements Comparison

In this section, a sequence of 100 images with small inter-frame displacements is constructed. † Figure 3(c) illustrates the performance of the four approaches without any added noise, and Fig. 3(d) shows the performance with Gaussian distribution noise. Although the four methods perform almost the same result when no noise is induced, ESM performs the worst in this sequence when noise is induced as Fig. 3(d). It suggests that high-order model is more sensitive to noise than linear model. Moreover, when Gaussian distribution noise is added in the sequence, the disturbance is more stable in our fast learning approach than HA and our original approach. From Fig. 3(d), our fast learning approach improves the tracking performance when noise is induced.

Recently, some other tracking methods have proposed. In [12], ASIFT image matching algorithm extends the SIFT method to a fully affine invariant device. Although it permits to reliably identify features that have undergone very large affine distortions, the number of key points in *target region* will affect the performance of tracking system. A lack of key points will decrease the system's tracking performance especially when the target object region is small. In [13], a consistent low-rank sparse tracker is proposed that builds upon the particle filter framework for tracking. To allevi-

†www.robots.ox.ac.uk/ cmei/SingleViewpointTracking.html



(a) 1st frame (b) 30th frame

(c) 125th frame (d) 220th frame

Fig. 4 Examples of real time tracking

ate the problem of misalignment between dictionary templates and particles, a dense sampling of the object and the background solution is adopted. When the target object region is large and the number of samples is not large enough, the system's tracking performance will be decreased also. Our approach fits for the object of arbitrary size because it only uses the simple global information of the object. It allows the approximation of the parameters in a data-driven way without specifying the distribution approximation in advance. Moreover, our fast learning strategy makes it possible to learn large templates and large affine distortions.

3.3 Real Time Tracking

In this section, a real time tracking is performed by our fast learning approach. The *target region* is defined as Fig. 4(a). From Fig. 4, our approach can track the object accurately.

4. Conclusion

To the best of our knowledge, our approach is the first discussion on homography-based tracking by finite mixture hyperplane models. In this paper, we have shown an original improvement of the tracking algorithm proposed by [4]. The key idea is to regress the ground truth model using mixture hyperplanes instead of a single one. The advantages of mixture models are that they do not require the distribution to be specified in advance, and they can express a global information of the ground truth model more accurately than single model. Moreover, we propose a fast learning strategy not only to overcome the dimension disaster, but also to decrease the effect of noise. Experiments demonstrated that our approach outperforms single model solutions.

In this paper, the mixing parameters are independent of the input variables. For example, each π_m is responsible for all samples in our approach. Allowing the mixing parameters to depend on the input data will yield better result. We suspect more is to come.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No.61271288, No.61301270), and the Research Fund for the Doctoral Program of Higher Education of China (No.20130185130001, No.20130185120014).

References

[1] R.T. Collins, Y. Liu, and M. Leordeanu, "Online selection of discriminative tracking features," Pattern Anal. Mach. Intell., IEEE Trans., vol.27, no.10, pp.1631–1643, 2005.
[2] Z. Kalal, J. Matas, and K. Mikolajczyk, "Pn learning: Bootstrapping binary classifiers by structural constraints," Comput. Vis. Pattern Recognit. (CVPR), 2010 IEEE Conference on, pp.49–56, IEEE, 2010.
[3] H. Grabner and H. Bischof, "On-line boosting and vision," Comput. Vis. Pattern Recognit., 2006 IEEE Computer Society Conference on, pp.260–267, IEEE, 2006.
[4] F. Jurie and M. Dhome, "Hyperplane approximation for template matching," Pattern Anal. Mach. Intell., IEEE Trans., vol.24, no.7, pp.996–1000, 2002.
[5] S. Holzer, M. Pollefeys, S. Ilic, D.J. Tan, and N. Navab, "Online learning of linear predictors for real-time tracking," in Computer Vision–ECCV 2012, pp.470–483, Springer, 2012.
[6] S. Benhimane and E. Malis, "Homography-based 2d visual servoing," Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on, pp.2397–2402, IEEE, 2006.
[7] G.D. Hager and P.N. Belhumeur, "Efficient region tracking with parametric models of geometry and illumination," Pattern Anal. Mach. Intell., IEEE Trans., vol.20, no.10, pp.1025–1039, 1998.
[8] S. Holzer, S. Ilic, and N. Navab, "Adaptive linear predictors for real-time tracking," Comput. Vis. Pattern Recognit. (CVPR), 2010 IEEE Conference on, pp.1807–1814, IEEE, 2010.
[9] C.M. Bishop et al., Pattern recognition and machine learning, springer New York, 2006.
[10] S.J. McKenna, Y. Raja, and S. Gong, "Tracking colour objects using adaptive mixture models," Image and vision computing, vol.17, no.3, pp.225–231, 1999.
[11] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework," International journal of computer vision, vol.56, no.3, pp.221–255, 2004.
[12] J.M. Morel and G. Yu, "Asift: A new framework for fully affine invariant image comparison," SIAM Journal on Imaging Sciences, vol.2, no.2, pp.438–469, 2009.
[13] T. Zhang, S. Liu, N. Ahuja, M.H. Yang, and B. Ghanem, "Robust visual tracking via consistent low-rank sparse learning," Int. J. Comput. Vis., vol.111, no.2, pp.171–190, 2014.