A Fast Object Detecting-Tracking Method in Compressed Domain

Zenglei Qian, Jiuzhen Liang, Zhiguo Niu, Yongcun Xu, Qin Wu

Internet of things engineering College, Jiangnan University, Binhu District, Wuxi, Jiangsu Province, China

Abstract. The traditional pixel domain tracking algorithms are often applied to rigid objects which move slowly in simple background. But it performs very poor for non-rigid object tracking. In order to solve this problem, this paper proposes a tracking method of rapid detection in compressed domain. Convex hull formed by Self-adaptive boundary searching method and rule-based clustering are adopted for the detector in order to reduce the complexity of the algorithm. At the tracking stage, Kalman filtering is used to forecast the location of the objective. Meanwhile, as the whole process is completed in the compressed domain, it can meet the real-time requirement compared with other algorithms. And it tracks the target more precisely. The experimental results show that the proposed method has the following properties: 1) more advantages in tracking small-sized objects; 2) a better effect when track a fast moving objects; 3) faster tracking speed.

1 Introduction

It's known that long-term target tracking in a video stream is one of the most important subjects in computer vision. It has broad application in many fields such as security surveillance, intelligent transportation. The main job of tracking is to estimate the location of the target in the subsequent frames. Given a bounding box defining the object of interest in a single frame, our goal is to automatically determine the bounding box of the object or indicate that the object is not visible in succeeding frames that follows. The video stream is processed at certain frame rate and the process may run indefinitely long. Although numerous algorithms have been proposed in literature, object tracking remains a challenging problem due to target appearance change caused by pose, illumination, occlusion, and motion, etc. So far, no single tracker has been able to deal with all these factors at the same time.

There are two major groups of approaches to segment and track moving objects in a video sequence, distinguished by the domain in which they operate: pixel domain and compressed domain.

Z. Kalal et al. designed a novel framework (TLD[13]) that decomposed the long-term tracking task into three subtasks: tracking, learning, and detection. But it does not perform well in case of full out-of-plane rotation and tracks a single object. Zhang et al.[34] proposed an effective and efficient tracking algorithm with an appearance model based on features extracted. A very sparse measurement matrix was adopted to efficiently compress features from the foreground targets and background. Wu et al. proposed a multi-scale tracking algorithm[30] which combines the random projection-based appearance model with the bootstrap filter framework. The proposed scale-invariant normalized rectangle feature was adopted to characterize a target object of different scales to be tracked.

Pixel domain approaches mentioned above have the potential for higher accuracy, but also require higher computational complexity. "Compressed-domain" approaches make use of the data from the compressed video bit stream, such as motion vectors (MVs), block coding modes, motion-compensated prediction residuals or their transform coefficients, etc.[14, 15, 7] Though the lack of full pixel information often leads to relative lower accuracy. The main advantage of compressed-domain methods in practical applications is their lower computational cost.[19, 11, 27]Currently, The compression standard these methods commonly used is highly compressed digital video coding standard (H.264/AVC), which was formulated by the Joint Video Team ITU-T Video Coding Experts Group (VCEG) and MPEG coalition (JVT, Joint Video Team).[28, 29] It plays an extremely important role in the field of video retrieval and monitoring for its fast segmentation algorithm. Therefore, compressed-domain methods are thought to be more suitable for real-time applications [38, 3], as a mean shift clustering is used in [3] to segment moving objects from MVs and partition size in H.264 bit-stream.[4, 5] However, some of them are still characterized by high complexity[31, 32, 16]. You et al. [32] presented an algorithm to track multiple moving objects in H.264/AVC compressed video based on probabilistic spatio-temporal MB filtering and partial decoding. Their work assumes stationary background and relatively slow-moving objects.

In this paper, we propose and discuss a Fast Object Detecting-Tracking Method in Compressed Domain, firstly, the tracker estimates the objects motion between consecutive frames under the assumption that the frame-to-frame motion is limited and the object is visible. While the tracker is likely to fail and never recover if the object moves out of the camera view. Once the tracker failed to track the target, the detector will correct tracking results in time and reduce the occurrence of the lost. We will make an integration of the tracking results and detecting results, draw the best prediction result as the final output.

2 Related work

The long-term tracking can be approached either from tracking or from detection perspectives. Tracking algorithms estimate the object motion. Trackers only require initialization. They are fast, and produce smooth trajectories. On the other hand, they accumulate errors during runtime (drift) and typically fail if the object disappears from the camera view. Research in tracking aims at developing increasingly robust trackers that track "longer". The post failure behavior is not directly addressed. Detection-based algorithms estimate the object location in every frame independently. Detectors do not drift and do not fail if the object disappears from the camera view. However, they require an offline training stage and therefore cannot be applied to unknown objects.

2.1 Detection

In recently years, Many methods have been developed for moving object detection in H.264/AVC bitstream domain. Sabirin et al. presented a graph-based algorithm[26] for detecting and tracking moving objects in H.264/AVC bitstream domain. The spatio-temporal graph first constructed represents the blocks with non-zero motion vectors or non-zero residues and their relations between two frames. A method proposed in [17] is that it uses both motion vectors and macroblock information, including macroblock data sizes, types, and partitions, to track and detect the target even when it stops. Because motion vector information is not generated when the moving object stops, use of only motion vector tracking cannot consistently track the target consistently. However, by using the addition of macroblock information,[17] demonstrate how target tracking can be maintained when the target stops. The [17] proposed method thereby improves target tracking. Khatoonabadi et al.[12] presented a method for tracking moving objects in H.264/AVC-compressed video sequences using a spatio-temporal Markov random field (STMRF) model. Sabirin et al.[24] detecting and tracking moving objects by treating the encoded blocks with non-zero motion vectors and/or non-zero residues as potential parts of objects in H.264/AVC bitstreams. Fang Yumin et al.[8] proposed a novel object segmentation approach using background estimation. To reduce the processing time for the subsequent global motion compensation, the inter-frame coding modes are applied to estimate the background region.

2.2 Tracking

Object tracking is the task of estimation of the object motion. We use Kalman filter to complete our task tracking. Kalman filter is the minimum-variance state estimator for linear dynamic systems with Gaussian noise [20]. Even if the noise is non-Gaussian, the Kalman filter is the best linear estimator. In addition, Kalman filter is the minimum-variance linear state estimator for linear dynamic systems with non-Gaussian noise [21]. For nonlinear systems it is not possible, in general, to implement the optimal state estimator in closed form, but various modifications of Kalman filter can be used to estimate the state. These modifications include the extended Kalman filter[21], the unscented Kalman filter[10] , and the particle filter[6]. Meanwhile, [37] presents an extended Kalman filter adopted to estimate the current location based on the estimated parameters and the previous location estimate. In [2], the strong Kalman filter is enhanced with the improved time-variant recession matrix and applied in tracking the given video target with fast mobility. The enhanced strong Kalman filter is found to be more robust than generic Kalman filter in resisting process uncertainties to which the fast mobile target is usually subjected. To deal with the actual process of tracking moving objectsappear in occlusion and interference problems caused bycamera movement, [9] designs a moving object tracking system using the algorithm based on Camshift and Kalman filter.

3 Framework of Detection-Tracking in Compressed Domain

The aim of this section is to provide a framework for detecting-tracking moving objects system in compressed domain based on MVs(see Fig. 1 for the overall framework). As we see, the compressed bit-stream is partially decoded by the H.264/AVC decoder which generate the motion vector. Then the MV of the current frame(t-th frame) is fed into the preprocessor to get the more stable and reliable MV field. Followed that, the corresponding field is entered into the detector in which we obtained multiple motion objects regions by means of boundary searching and rule-based clustering. Under the assumption that the frame-to-frame motion is limited and the object is visible, our tracker estimates the motion of the target between consecutive frames by Kalman filter. Finally, the result which is integrated with the tracker and detector is regarded as the final output.

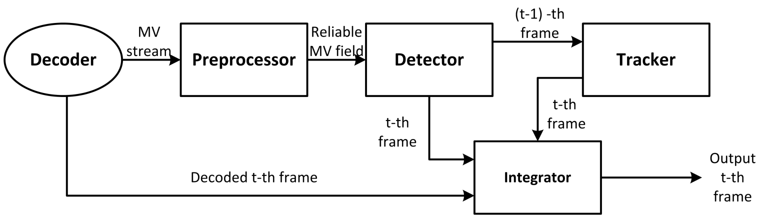

Fig. 1. CDT method framework

3.1 Preprocessor

To filter the noisy motion vectors, a spatio-temporal motion vector consistency filter (STF) is presented in paper[18]. Let $(x,y)^t$ represents the pixel coordinate (x,y) in the t-th frame, and $mv\,(x,y)^{t\to t_{ref}}$ represents the motion vector of pixel (x,y) in the t-th frame with respect to the reference frame t_{ref}.

Backward iterative cumulate This is accomplished by dividing the cumulative motion vector by frame number, according to Motion Vector Normalization Equation (**??**)(1)(2).

$$N\left(MV^{t\to t_{ref}}\right) = \frac{MV^{t\to t_{ref}}}{t - t_{ref}} \tag{1}$$

$$P\,(i,j)^{t\to t_{ref}} = (i,j)^t$$

$$P(i,j)^{t\to(t-1)} = (i,j)^t + MV^{i,j}(t) = \left(\hat{\imath},\hat{\jmath}\right)^{t-1} \tag{2}$$

$$P(i,j)^{t\to(t-k)} = P\left(P(i,j)^{t\to(t-k+1)}\right)^{t\to(t-k)} \quad (k>2)$$

$N\left(MV^{t\to t_{ref}}\right)$ is the normalized motion vector. In this section, the necessity of backward iterative cumulate is demonstrated by the MV denoising theory in the time domain.

Spatial consistency Spatial consistency is calculated between the current MVs and neighborhood MVs in distance(see Equation(3)(4)).

$$MVNCI^{i,j}_{amp} = \left| \frac{MV^{i,j}_{amp} - \frac{1}{N}\sum\limits_{(i',j')\in\theta} MV^{i',j'}_{amp}}{\frac{1}{N}\sum\limits_{(i',j')\in\theta} MV^{i',j'}_{amp}} \right| \tag{3}$$

$$MVNCI^{i,j}_{dir} = \frac{1}{N}\sum_{(i',j')\in\theta} \left| MV^{i,j}_{dir} - MV^{i',j'}_{dir} \right| \tag{4}$$

$MVNCI^{i,j}_{amp}$ represents the degree of $MV^{i,j}_{amp}$ deviated from it neighborhood nonzero MV, $MV^{i',j'}_{amp}$ is the surrounding MVs, θ represents eight-neighborhood structures and N represents the number of nonzero motion vector of the neighborhood.

Temporal consistency Vector matching ratio is used to analyze the MVs compared with MVs of the same position in the previous frame. Temporal consistency between the consecutive frames is calculated by Equation (5) and (6).

$$R\left(\overrightarrow{a},\overrightarrow{b}\right) = \begin{cases} 0 & if\ \overrightarrow{a} = \overrightarrow{b} = (0,0) \\ \frac{\left\|\overrightarrow{a}-\overrightarrow{b}\right\|}{\left\|\overrightarrow{a}\right\|+\left\|\overrightarrow{b}\right\|} & otherwise \end{cases} \tag{5}$$

$$MVTCI\left(MV^{i,j}(t)\right) = \prod_{k=1}^{n} R\left(MV\left(P(i,j)^{t\to t-k+1}\right), MV\left(P(i,j)^{t\to t-k}\right)\right) \tag{6}$$

3.2 Detector

The detection algorithm alone is unable to narrow the search area because the compressed stream has no color information as object feature, we propose a fast moving objects detection algorithm based on boundary searching with convex hull. The algorithm consists of three parts as follows.

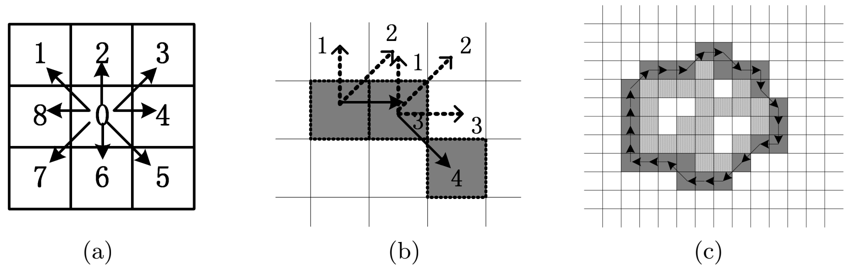

Fig. 2. Adaptive boundary blocks scanning. (a) Eight-direction template. (b) the s-canning order of the eight-direction. The dotted lines and serial numbers represent the order of scanning. (c) The grey blocks represent the boundary blocks after scanning.

Adaptive boundary searching Without inner information of moving objects, we use local search by locating a moving region instead of global search, so that to reduce the time complexity. This algorithm sets a 3×3 offset template which is used to represent the eight-direction neighborhood block of the current search block(see Fig.2(a)).

The zero is the current block, and 1 to 8 are searching blocks in eight-direction which saved the offset value relative to the current search block, as described in Equation (7).

$$(x_{cur}, y_{cur}) = (x_{cur}, y_{cur}) + (x_{offset}, y_{offset}) \tag{7}$$

After the eight-direction template is defined, we define the first initial search block using the raster scan method and ensure that its searching blocks are boundary blocks by scanning blocks in clockwise direction(as shown in Fig.2). In the current direction, the block is not a moving block, we should turn to the block in next direction and so on, until the eight searching directions is completed or found a motion block. The target in tracking is only related with its contour, getting only the convex hull points instead of all points in the target can greatly reduce the computational complexity. Graham scanning method is token in this paper(see Fig.3).

Rule based clustering After the procedure discussed above, some scattered connected regions of moving objects are got. In this section the final moving object is obtained by clustering the scattered motion subregions following some rules. The number of iterations often depends on the number of global motion vector blocks. And the consumption of the iterative time decides the segmentation algorithm performance. In this paper, clustering the motion subregions can greatly reduce the number iterations.

This paper puts forward the clustering rules contained of two elements: one is the direction and amplitude of motion vectors, the other is the distance between motion subregions. Two new distances are defined in Equation (8) and Equation (9).

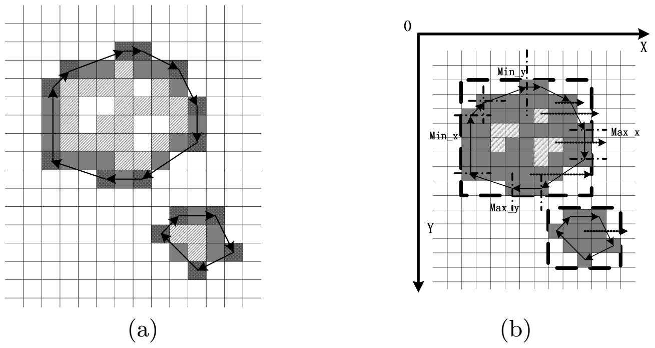

Fig. 3. In (a), the mesh blocks, grey blocks and twill blocks are the Mask after filtering in motion vector field, the grey blocks represent the boundary blocks on the boundary searching, the intermediate-blank blocks represent the missing blocks, and the mesh blocks are convex hull points. In (b), the rough borders are searching area, whose internal is used to fill points by ray method for avoiding repeated searching.

$$p_{dis} = \underset{1<i<n, i \neq j}{Hausdorff} \|\overrightarrow{p_i}, \overrightarrow{p_j}\| \tag{8}$$

$$p_{NCI} = \mu_1 p_{amp}^{(i)(j)} + \mu_2 p_{dir}^{(i)(j)} = \mu_1 \frac{\left|MV_{amp}^{(i)} - MV_{amp}^{(j)}\right|}{MV_{amp}^{(j)}} + \mu_2 \frac{\left|MV_{dir}^{(i)} - MV_{dir}^{(j)}\right|}{MV_{dir}^{(j)}} \tag{9}$$

Where $p_{amp}^{(i)(j)}$ and $p_{dir}^{(i)(j)}$ represent deviation in amplitude and direction respectively, $\overrightarrow{p_i}$ and $\overrightarrow{p_j}$ are the positions of block i and j, respectively. In this way, the motion subregions having common feature are clustered into a whole moving object. The constraint condition is $\mu_1 + \mu_2 = 1$.

3.3 Tracker

Object tracking is the problem of estimating the positions and other relevant information of moving objects in image sequences. Our tracker is based on Kalman filter[22], which is a classical motion prediction model and optimal recursive algorithm. The Kalman filter solves the general problem of estimating the state $x \in R^n$ of a discrete-time controlled process that is governed by the linear stochastic difference equation (10) and (11).

$$X_k = AX_{k-1} + BU_{k-1} + W_{k-1} \tag{10}$$

$$Z_k = HX_k + V_k \tag{11}$$

Where X_k is the system state at time k, U_k is the amount system control, A, B is system parameters, Z_k is measured values of time k, H is a parameter of the measurement system. The random variables Wk and V_k represent the process and measurement noise respectively. According to the system model, we can predict the present state based on the former state in equation , and P is denoted as the covariance in equation 12,13.

$$\hat{X}_k = A\hat{X}_{k-1} + BU_{k-1} \tag{12}$$

$$P_k^- = AP_{k-1}A_k^T + Q \tag{13}$$

P_k^- is the covariance corresponding to $\hat{X_k^-}$, Q is the covariance of the system process. Eq.12 and Eq.13 are the predicted results of system. Combing the predicted and measured values, we can get the following iterative (14)-(16).

$$\hat{X}_k = \hat{X}_k^- + K_k(Z_k - H\hat{X_k^-}) \tag{14}$$

$$K_k = P_k^- H^T \left(HP_k^- H^T + R\right)^{-1} \tag{15}$$

$$P_k = (I - K_k H) P_k^- \tag{16}$$

Where K_k is the Kalman Gain, and of which the basic principle can be represented by (12) to (16).

4 CDT Implementation

This section describes our implementation of the CDT framework.

4.1 Initialization

In order to track the target, we need to artificially specify an object as target from the first frame, and the decoded stream should be preprocessed. At first, the first frame of the video sequence is decoded, and the initial bounding box is artificial demarcated. The real tracking target will be obtained by the detector. Spatial similarity of the bounding boxes in the detector and artificial demarcated is measured by overlap between bounding boxes, which is defined as a ratio between intersection and union.

4.2 Object detector

The detector scans the input image by searching multiple moving objects and decides which one is the tracking target we wanted. The algorithm of boundary searching is shown in Algorithm 1.

Algorithm 1: Adaptive Boundary Searching

```
Algorithm1 (Mask, Bpoint, Tukenum)
  Data: Mvfield // MV of current frame
  MVsize = sizeof(Mvfield) // size of the current frame
  Result: Mask // moving object mask in 1 or 0,
               // 1 means object, 0 means background
```

```
        Bpoint // the boundary points of motion subregions
        Tukenum // the number of the motion subregion
  begin:
    tukenum := 0, Mask := 0 in height*width;
    for i:=1:Mvsize(Mask(i)==1), do
        if Mvfield(i) is interval in MV threshold && Mask(i)=0 then
            save startpoint with address of Mvfield(i);
            while i is not startpoint, do
                for j:=8-direction searching, do
                    if Mvfield and Mask is appropriate, then
                        Mask(j):=1,Bpoint:=Mvfield(j)
                if 8-direction searching is not find block, then
                    return the original path;
        tukenum++, Bpoint:=Coverxhull(Bpoint);//get the convex
        //hull vertices remove the points inner convex hull;
end
```

In this way, the number of iteration is decreased a lot, thus the computational complexity is reduced. The *Mask* and *Bpoint* contain the scattered motion regions are the inputs of the clustering algorithm's inputs. Then the clustering algorithm is shown in Algorithm 2.

Algorithm 2: Rule Based Clustering

```
Algorithm2 (Edge, Objectnum)
  Data: Bpoint // the boundary points of motion subregions
  Tukenum // the number of the motion subregion
  Meanmv //the average of MV in x and y direction of subregion
  Result: Edge // the points of convex hull of final objects
        Objectnum // the number of the final objects
  begin:
    for i := 1:tukenum do
        for j :=i+1:tukenum do
            pdis(i,j):= compute the Hausdorff distance between
            Bpoint(i) and Bpoint(j);
            if pdis(i,j) < Tdis then
                pamp(i,j) := compute the deviation degree
                in amplitude of MeanMv;
                pdir(i,j):= compute the deviation degree
                in direction of MeanMv;
                pNCI := u1pamp(i,j)+u2pdir(i,j);
                if pNCI < TNCI then
                    Edge := Convexhull(Bpoint);
    Objcetnum:= update tukenum;
end
```

The threshold of Rule-based clustering needs to be initialized, where the value of μ_1 and μ_2 are different for different videos. If the target is moving slowly and its moving direction keeps unchanged, its moving direction can be determined by the MVs in the motion subregions. And the value of μ_2 need to be set large, generally 0.8. If the target is moving quickly and its direction changes rapdily, then the motion vector will be more messy. In that case, MV has major influence on determining the degree of deviation in magnitude, the value of μ_1 requires to be large, and generally 0.6. Fig.4 shows the segmentation performance of our detecting algorithm, it works well.

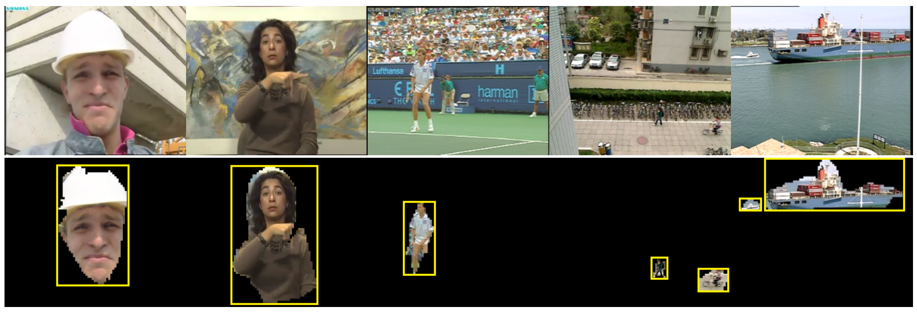

Fig. 4. The performance of segmentation in detector, these video sequence is respectively called 'foreman', 'silent', 'stefan', 'walk' and 'container' in the version of JM12.2 of H.264/AVC.

4.3 Object tracking

The tracking component of CDT is based on Kalman Filter tracker, which represents the object by a bounding box(called tBB) and estimates its motion between consecutive frames. Here we need to consider two issues. Firstly, the input of tracker is the output of detector. When the detector fails to detect, the tracker can automatically recognize the false detection and avoid unnecessary errors. Secondly, the output of the detector depends entirely on the boundary searching and rule based clustering algorithm. Note that, the detector and the tracker propose in this paper an independent of each other. The tracker can do a smooth filtering and prediction to the detector, and the result is re-applied to the detector and correct it.

Failure detection The tracker can do a preliminary identification of the output of the detector. We define predition position of the target at the i-th frame as $tBB\,(i)$, and the output of detector at the i-th frame is defined as $dBB(i)$. We define e is $e = tBB(i) - dBB(i)$. If the deviation e is greater than two times the average of the moving distance, we consider the output of detector is failure. the trace $tBB(i+1)$ is predicted according to $dBB(i-1)$, and so on. See Fig.5.

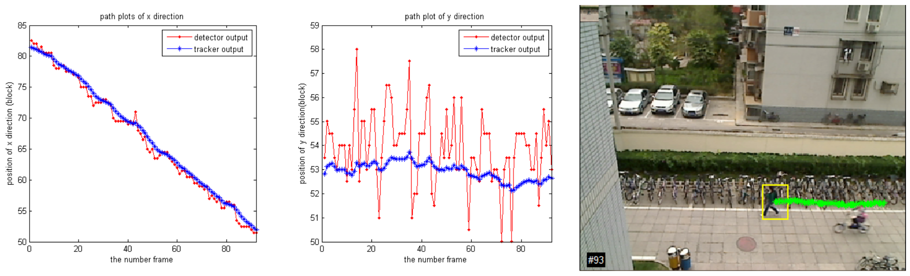

Fig. 5. The output path of detector and tracker. The first two plots are the path of x direction and y direction, the third figure is trajectory diagram of the video 'Walk'.

5 Experiments and Analysis

Our experiment is implemented in MATLAB R2011b. The system configuration is: Intel(R), Core(TM), i5-3470@3.20GH, 4GB memory, the OS is Microsoft Windows 8. In order to test the effectiveness of our tracking algorithm, we compare it with six algorithms, MIL[1], WMIL[33], STC[36], FCT[35], TLD[13]. The experiments compare the tracking error, error rate and frame rate. Five compressed video sequences are used in this experiment. All sequences were encoded using the H.264/AVC JM12.2 encoder as shown in table.1. The initialized bounding box, of which including the four elements: the first two x,y coordinate of the upper left corner of the bounding box, the others coordinate of the length and width of the target object.

Table 1. Information of video sequences used in the experiments

Video Sequence	Numbers of Frames	Initial Bounding Box
Boat	180	(63,106,224,79)
Container	170	(61,40,230,71)
Stefan	90	(195,74,102,176)
Coastguard	82	(105,116,99,48)
Walk	92	(316,194,27,36)

5.1 Tracking accuracy experiment

From Fig.6, one may find that the backgrounds in the test sequences a, b, d are relative simple, so other tracking algorithms learn good models from training samples which produce good tracking results. However, the third test sequence is tracking a object which moves rapidly and the background is complex and changing rapidly. In that case, the limitations of other tracking algorithms are exposed. The rapid movement leads to great changes of learning samples, which makes it impossible for other algorithms to track the object accurately. And

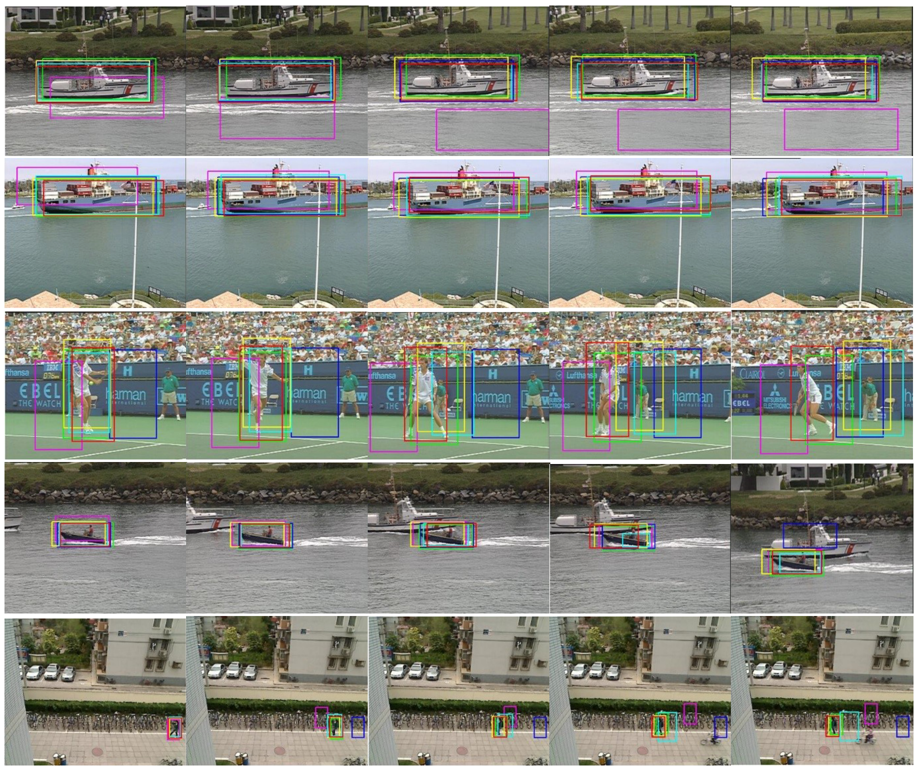

Fig. 6. Screenshots from some of the sampled tracking results(From top to bottom: *a.* Boat *b.* Container *c.* Stefan *d.* Coastguard *e.* Walk). red is CDT, purple is MIL, green is WMIL, blue is STC, yellow is FCT, cyan is TLD

the fifth sequence is a non-rigid object with complex background. The STC, TLD and MIL algorithms can not track the object well. In this paper, the CDT Algorithm detector is based on motion vector. And the motion vector is predicted from the inter frames, which contains part of the real motion information. So the algorithm is less sensitive to the speed of the object and background complexity.

Tracking error rate is an important index in detection tracking algorithm. In this paper, we define that the object is incorrectly tracked when the distance between the center of the tracker to the stand center of the moving object is more e_T. The value of e_T is depended on the size of the moving objects. We set $e_T = \mu D_0$, where D_0 is the size of the movement object. μ is [0,0.2]. It can be more objective to observe the changes of different tracking error rate with the error threshold for these settings. As shown in figure.7,8.

5.2 Frame rate tests

In this experiment, maximum framerate, minimum, frame rate, average framerate are calculated for maximum frame rate, minimum, frame rate, average frame

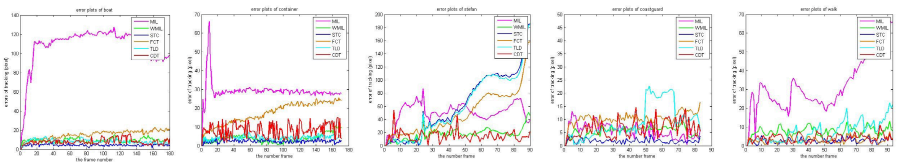

Fig. 7. Error plots for all video sequences labeled with ground truth.

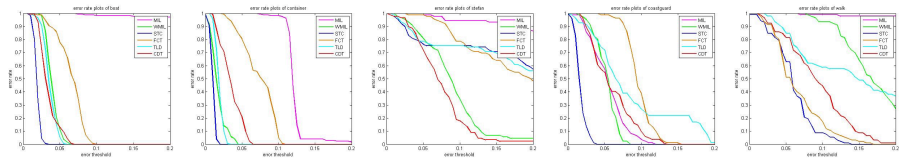

Fig. 8. Error rate plots for all video sequences.

rate by the six algorithms. The results are shown in Table 2. From this table, one may find that both the FCT and CDT algorithms have obvious advantages in the frame rate. Especially for the CDT algorithm, it has outstanding advantages on the average frame rate and high frame rate, but not much effect on the minimum frame rate. The reason is that the core of the CDT algorithm is the detector, which takes the majority time in the whole algorithm. Boundary searching and clustering rules depend on the number of motion vectors and their dispersion, large numbers of motion vectors and dispersion increase the time consumption of the detector algorithm. For example, the container sequence, the tracking object is larger, and in the stefan sequence, there is lots of noise due to the fast movement of the object. However, as for the average frame rate, it still has good advantages.

6 Conclusions

This paper presents a method of rapid detection in compressed domain and tracking target. At the tracking stage, Kalman filtering is used to forecast the location of the object. Convex hull formed by Self-adaptive boundary searching method and rule-based clustering are adopted for the detector in order to reduce the complexity of the algorithm to a large extent. Meanwhile, as the whole process is completed in the compressed domain, it can better meet the real-time requirement comparing with other algorithms. Meanwhile, it tracks the object more precisely. However, due to the fact that less features can be used in a compressed domain. The real-time is gained at the expense of the loss certain accuracy of tracking. In our experiment, more intense shaking will be observed as side effect. The reason is that the smallest unit of image information in compressed domain is a 4×4 block. In our further research, we will come up with more effective way to decrease the shaking.

Table 2. Frame rate comparsion of six algorithms

video sequence	Fps	CDT	MIL	WMIL	STC	FCT	TLD
Boat	Max-fps	**42.06**	10.24	24.56	28.72	**33.09**	14.56
	Mean-fps	**36.08**	5.97	20.92	21.67	**32.20**	13.31
	Min-fps	11.03	3.82	9.52	**14.28**	**18.27**	12.29
Container	Max-fps	**38.08**	8.68	24.56	27.89	33.05	17.54
	Mean-fps	**26.42**	5.76	20.12	21.49	**31.78**	13.37
	Min-fps	6.97	3.88	7.71	9.51	**18.26**	**11.96**
Stefan	Max-fps	**59.69**	9.75	23.18	32.02	**32.32**	16.62
	Mean-fps	**30.90**	7.22	21.35	28.90	**32.06**	12.95
	Min-fps	3.70	5.65	**10.57**	10.19	**18.32**	7.94
Coastguard	Max-fps	**57.33**	9.04	24.79	**36.62**	33.05	19.93
	Mean-fps	**39.43**	6.94	20.94	30.76	**31.81**	16.30
	Min-fps	**16.59**	5.53	8.99	15.60	**18.41**	11.18
Walk	Max-fps	**59.48**	9.03	24.40	**41.35**	33.89	19.88
	Mean-fps	**50.29**	6.80	20.15	**32.11**	27.68	16.08
	Min-fps	**23.28**	5.09	7.67	11.38	**18.42**	12.47

References

1. Babenko, Boris and Yang, Ming-Hsuan and Belongie, Serge.: Visual tracking with online multiple instance learning. Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on. (2009) 983–990
2. Chen, Ken and Zhao, Xuemei and Xu, Tiefeng and Napolitano, MR.: Enhanced strong Kalman filter applied in precise video tracking for fast mobile target. Progress in Informatics and Computing (PIC), 2010 IEEE International Conference on. **2** (2010) 875–878
3. Chen, Yue-Meng and Bajic, Ivan V.: A joint approach to global motion estimation and motion segmentation from a coarsely sampled motion vector field. Circuits and Systems for Video Technology, IEEE Transactions on **9** (2011) 1316–1328
4. Chen, Yue-Meng and Bajic, Ivan V and Saeedi, Parvaneh.: Moving region segmentation from compressed video using global motion estimation and Markov random fields. Multimedia, IEEE Transactions on **3** (2011) 421–431
5. Chen, Yue-Meng and Bajic, Ivan V and Saeedi, Parvaneh.: Motion segmentation in compressed video using Markov random fields. Multimedia and Expo (ICME), 2010 IEEE International Conference on. (2010) 760–765
6. Doucet, Arnaud and De Freitas, Nando and Gordon, Neil.: Sequential Monte Carlo methods in practice. Springer (2001)
7. Fei, W and Zhu, S: Mean shift clustering-based moving object segmentation in the H. 264 compressed domain. IET image processing. **4** (2010) 11–18
8. Fang, Yuming and Lin, Weisi and Chen, Zhenzhong and Tsai, C and Lin, C.: A Video Saliency Detection Model in Compressed Domain. (2013)
9. Huang, Shengluan and Hong, Jingxin.: Moving object tracking system based on camshift and Kalman filter. Consumer Electronics, Communications and Networks (CECNet), 2011 International Conference on. (2011) 1423–1426
10. Julier, Simon J and Uhlmann, Jeffrey K.: Unscented filtering and nonlinear estimation. Proceedings of the IEEE. **3** (2004) 401–422

11. Käs, Christian and Nicolas, Henri.: An approach to trajectory estimation of moving objects in the h. 264 compressed domain. Advances in Image and Video Technology. (2009) 318–329
12. Khatoonabadi, Sayed Hossein and Bajic, Ivan V.: Video object tracking in the compressed domain using spatio-temporal Markov random fields. Image Processing, IEEE Transactions on **1** (2013) 300–313
13. Kalal, Zdenek and Mikolajczyk, Krystian and Matas, Jiri.: Tracking-learning-detection. Pattern Analysis and Machine Intelligence, IEEE Transactions on **34** (2012) 1409–1422
14. Liu, Zhi and Lu, Yu and Zhang, Zhaoyang: Real-time spatiotemporal segmentation of video objects in the H. 264 compressed domain. Journal of visual communication and image representation. **3** (2007) 375–290
15. Liu, Zhi and Lu, Yu and Zhang, Zhaoyang: An efficient compressed domain moving object segmentation algorithm based on motion vector field. Journal of Shanghai University (English Edition). **12** (2008) 221–227
16. Liu, Zhi and Zhang, Zhaoyang and Shen, Liquan.: Moving object segmentation in the H. 264 compressed domain. Optical Engineering. **1** (2007) 017003–017003
17. Maekawa, Erii and Goto, Satoshi.: Examination of a tracking and detection method using compressed domain information. Picture Coding Symposium (PCS), 2013. Waseda University, Shinjuku-ku, Tokyo, Japan.(2013) 141–144
18. Moura, Ronaldo C and Hemerly, Elder Moreira.: A spatiotemporal motion-vector filter for object tracking on compressed video. Advanced Video and Signal Based Surveillance (AVSS), 2010 Seventh IEEE International Conference on. (2010) 427–434
19. Mezaris, Vasileios and Kompatsiaris, Ioannis and Boulgouris, Nikolaos V and Strintzis, Michael G." Real-time compressed-domain spatiotemporal segmentation and ontologies for video indexing and retrieval. Circuits and Systems for Video Technology, IEEE Transactions on. **14** (2004) 606–621
20. Rhodes, Ian B.: A tutorial introduction to estimation and filtering. Automatic Control, IEEE Transactions on **6** (1971) 688–706
21. Simon, Dan.: Optimal state estimation: Kalman, H infinity, and nonlinear approaches. John Wiley & Sons. (2006)
22. Salmond, David.: Target tracking: introduction and Kalman tracking filters. Target Tracking: Algorithms and Applications (Ref. No. 2001/174), IEE. (2011) 1–1
23. Soyak, Eren and Tsaftaris, Sotirios A and Katsaggelos, Aggelos K.: Low-complexity tracking-aware H. 264 video compression for transportation surveillance. Circuits and Systems for Video Technology, IEEE Transactions on. **10** (2011) 1378–1389
24. Sabirin, Houari and Kim, Munchurl.: Moving object detection and tracking using a spatio-temporal graph in H.264/AVC bitstreams for video surveillance. Multimedia, IEEE Transactions on. **3** (2012) 657–668
25. oyak, Eren and Tsaftaris, Sotirios A and Katsaggelos, Aggelos K.: Content-aware H. 264 encoding for traffic video tracking applications. (2010) 730–733
26. Sabirin, Houari and Kim, Munchurl.: Moving object detection and tracking using a spatio-temporal graph in H. 264/AVC bitstreams for video surveillance. **3** (2012) 657–668
27. Treetasanatavorn, Siripong and Rauschenbach, Uwe and Heuer, Jörg and Kaup, André.: Bayesian method for motion segmentation and tracking in compressed videos. Pattern Recognition. (2005) 277–284
28. Treetasanatavorn, Siripong and Rauschenbach, Uwe and Heuer, Jörg and Kaup, André.: Model based segmentation of motion fields in compressed video sequences

using partition projection and relaxation. Visual Communications and Image Processing 2005. (2005) 59600D–59600D
29. Treetasanatavorn, Siripong and Rauschenbach, Uwe and Heuer, Jörg and Kaup, André.: Stochastic motion coherency analysis for motion vector field segmentation on compressed video sequences. Proc. WIAMIS.(Apr. 2005). (2005)
30. Wu, Yunxia and Jia, Ni and Sun, Jiping.: Real-time multi-scale tracking based on compressive sensing. The Visual Computer. Springer (2014) 1–14
31. You, Wonsang and Sabirin, MS Houari and Kim, Munchurl.: Moving object tracking in H. 264/AVC bitstream. Multimedia Content Analysis and Mining. (2007) 483–492
32. You, Wonsang and Sabirin, MS Houari and Kim, Munchurl.: Real-time detection and tracking of multiple objects with partial decoding in H. 264/AVC bitstream domain. IS&T/SPIE Electronic Imaging. (2009) 72440D–72440D
33. Zhang, Kaihua and Song, Huihui.: Real-time visual tracking via online weighted multiple instance learning. Pattern Recognition. **1** (2013) 397–411
34. Zhang, Kaihua and Zhang, Lei and Yang, Ming-Hsuan.: Real-time compressive tracking. Computer Vision–ECCV 2012 Springer (2012) 864–877
35. Zhang, Kaihua and Zhang, Lei and Yang, M.: Fast Compressive Tracking. (2014)
36. Zhang, Kaihua and Zhang, Lei and Yang, Ming-Hsuan and Zhang, David.: Fast Tracking via Spatio-Temporal Context Learning. arXiv preprint arXiv:1311.1939. (2013)
37. Zhang, Li and Chew, Yong Huat and Wong, Wai-Choong.: A novel angle-of-arrival assisted extended Kalman filter tracking algorithm with space-time correlation based motion parameters estimation. Wireless Communications and Mobile Computing Conference (IWCMC), 2013 9th International. (2013) 1283–1289
38. Zeng, Wei and Du, Jun and Gao, Wen and Huang, Qingming.: Robust moving object segmentation on H. 264/AVC compressed video using the block-based MRF model. Real-Time Imaging. **4** (2005) 290–299
39.
40. Alpher, A.: Advances in Frobnication. J. of Foo **12** (2002) 234–778
41. Alpher, A., Fotheringham-Smythe, J.P.N.: Frobnication revisited. J. of Foo **13** (2003) 234–778
42. Herman, S., Fotheringham-Smythe, J.P.N., Gamow, G.: Can a machine frobnicate? J. of Foo **14** (2004) 234–778
43. Smith, F.: *The Frobnicatable Foo Filter.* GreatBooks, Atown (2009)
44. Wills, H.: Frobnication tutorial. Technical report CS-1204, XYZ University, Btown (1999)