# Online Content-aware Video Condensation

Shikun Feng, Zhen Lei, Dong Yi, Stan Z. Li*
Center for Biometrics and Security Research & National Laboratory of Pattern Recognition
Institute of Automation, Chinese Academy of Sciences
shi.k.feng@gmail.com {skfeng,zlei,dyi,szli}@nlpr.ia.ac.cn

## Abstract

*Explosive growth of surveillance video data presents formidable challenges to its browsing, retrieval and storage. Video synopsis, an innovation proposed by Peleg and his colleagues, is aimed for fast browsing by shortening the video into a synopsis while keeping activities in video captured by a camera. However, the current techniques are offline methods requiring that all the video data be ready for the processing, and are expensive in time and space. In this paper, we propose an online and efficient solution, and its supporting algorithms to overcome the problems. The method adopts an online content-aware approach in a stepwise manner, hence applicable to endless video, with less computational cost. Moreover, we propose a novel tracking method, called sticky tracking, to achieve high-quality visualization. The system can achieve a faster-than-real-time speed with a multi-core CPU implementation. The advantages are demonstrated by extensive experiments with a wide variety of videos. The proposed solution and algorithms could be integrated with surveillance cameras, and impact the way that surveillance videos are recorded.*

## 1. Introduction

In the past decade, the world has seen an explosive growth in surveillance video data. Security applications have great demands for efficient technologies for fast video browsing, retrieval and storage.

The easiest approach to efficient browsing includes fast forwarding [2] or video skimming [15]. Unfortunately, skipped frames may lead to missing some video contents. Content-adaptive skipping of frames [11] is thus proposed but it achieves lower efficiency. Another approach, video summarization [8], uses key frames as a synopsis to represent the original video. Like a slide show, this method loses not only the dynamic aspect of video but also video contents. The space-time video montage [6], as an alternative, analyzes both the spatial and temporal information distribution of the input video. Inspired by seam carving [1], the ribbon carving based method [9] considers a ribbon rather than a whole frame as the smallest processing unit, and removes the ribbons without activities. However, it achieves low condensation ratio, and may fail when adjacent objects have different speeds and directions.

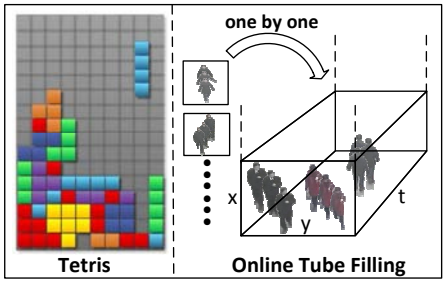


Figure 1. The main idea is inspired by the Tetris game (left). A key part of the proposed algorithm is online tube filling (right).

A significant progress in achieving high condensation ratio is video synopsis [12, 13, 14] by Peleg and his colleagues. Video synopsis enables a viewer to browse a day long video recording in just a few minutes. The algorithm is organized into two phases. The first phase does preprocessing, where moving objects are extracted and inserted to an object database. In the second phase, objects are picked up by users from database (offline), and an energy function is minimized to determine the play time of such objects in synopsis video. As the newer work of Peleg, [19] generates the coherent video synopsis based on clustering of similar activities. Following the video synopsis framework, [18] presents a set theoretical method to determine the play time of objects in synopsis video. Recently, an algorithm called direct shift collision detection [7] aims to make the second phase of video synopsis real time. However, the result may contain occlusions between nonadjacent objects.

However, the above offline methods in the video synopsis framework have the following drawbacks: (1) they require that all the video data be ready for the optimization; (2) and are therefore computationally expensive by process-

*Stan Z. Li is the corresponding author.

978-1-4673-1228-8/12/$31.00 ©2012 IEEE 

ing the data all at once; (3) they need huge memory to store all the tubes and backgrounds; (4) the length of final synopsis video is determined by the user rather than the content of input video, which is impractical in reality since the user does not know the density of input video beforehand.

In this paper, we propose a novel solution, called online content-aware video condensation, to overcome the above problems to achieve efficient video condensation. The solution achieves the same goal as video synopsis; however, in a content-aware approach, it uses recent video content to automatically determine the length of condensed video. The core is an online tube filling algorithm (a tube is a frame-sequence of an object), inspired by the game Tetris, as illustrated in Fig.1; the processing is performed stepwise, online, thus needs a low memory consumption, and is applicable to endless video. In order to achieve high-quality results, a novel tracking method called sticky tracking is proposed to overcome drawbacks of existing tracking methods when applied to video condensation. It not only works well in the case of poor object segmentation, but also keeps the chronological order of nearby objects. The proposed solution can achieve faster-than-real-time speed when implemented on a multi-core CPU. It could be integrated in the surveillance cameras, and impact the way that surveillance videos are recorded and stored.

## 2. Sticky Tracking

Tube is the smallest processing unit in video condensation. It is defined as the frame-sequence of the object in video. Examples are shown on the right picture of Fig. 1 where there are five tubes. The role of tracking is to generate such tubes after object segmentation. Many tracking methods have been proposed to date in the field of video analysis [21], however, those may not be entirely suitable for video condensation. The following example illustrates the problems: as illustrated in Fig. 2a with two objects, where object 2 is occluded by object 1 at time $t$. Fig. 2b shows an ideal result of a common blob tracking method [20]: Two tubes are generated, but both are disabled. Because only one blob is matched to object 1 at time $t$, something not belonging to object 1 will burst into view in the condensed video part. Meanwhile, because no blob is matched to object 2, tube 2 will disappear abruptly and then appear again in the view. As a result, both tubes will cause blinking effect, deteriorating user experience. Fig. 2c is the result of an ideal method that would produce the most accurate result. However, such an optimal tracker would fail too: Part of tube 2 is lost due to the occlusion at time $t$, and this also causes blinking.

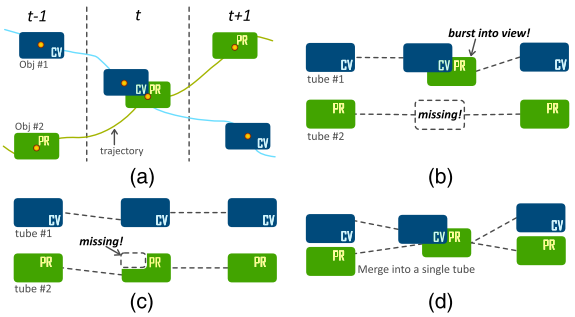


Figure 2. (a) Occlusion between two objects. Results of (b) common blob tracking, (c) optimal tracking and (d) sticky tracking.

Sticky tracking is proposed to reduce such blinking effect. It is based on the following idea: if occlusions happen to two or more tubes, they will be merged into a single tube, as if they are sticking together in Fig. 2d. Note that the goal of sticky tracking is very different from that of traditional tracking methods. The key point of sticky tracking is to launch merging before matching, that is, if two or more tubes in the target list are matched to the same blob by the similarity analysis at time $t$ (in this paper we use the nearest distance between the points on the tubes' boundary as similarity measure, and if occlusion happens, such distance will be set to zero), they will be merged into one tube; then repeat this process until there is no merging; the relationship of matches, which can be "one target-to-many blobs", is determined at last.


Figure 3. Sticky tracking results with a surveillance video from [9]. A unique color presents a single track; if two or more parts have the same color, they are considered as a single track.

In addition to reducing blinking, sticky tracking presents another advantage, being able to amend poor object segmentation (under- and over-segmentation). The case of under-segmentation can be treated as object occlusion, for which sticky tracking is proposed. Fig. 3 shows some sticky tracking results for the other case of over-segmentation. In (a), the head and the body of the same person were considered as two objects due to the over-segmentation at frame 732. When the segmentation becomes correct at frame 733, the head and body are merged into a single tube by using sticky tracking. In (b), a person with a suitcase splits into two objects at frame 4548 due to occlusions by the fence, and sticky tracking still successfully considers them as the same object.

Moreover, sticky tracking has an ability to keep the chronological order of objects when they are close to each other (*e.g.*, taking a conversation): If the distance between

objects is less than a threshold (10 pixels in experiments), sticky tracking will consider them as a whole object, therefore, their chronological order will be preserved.

# 3. The Online Content-aware Framework

## 3.1. Offline VC as Optimization Problem

The offline video condensation (VC) [13] condenses the video by rearranging tubes in such a way that objects that appear sequently in the original video can appear simultaneously in condensed video. Constraints applied for the said purpose, includ keeping chronological order, avoiding collisions between objects, consistency with background and so on. Therefore, it can be viewed as a problem of constrained optimization. The offline VC reassigns a time label to each tube by minimizing the following energy function:

$$E(\boldsymbol{\ell}) = \sum_{i \in \boldsymbol{Q}} E_u(\ell_i) + \sum_{i,j \in \boldsymbol{Q}} E_p(\ell_i, \ell_j), \qquad (1)$$

where $\boldsymbol{Q}$ denotes the whole tube set, and $\ell_i$ denotes the time label of tube $i$ taking a value in the label set: $\mathcal{L}_{\text{offlineVC}} = \{1, \cdots, M\}$. Here $M$ denotes the frame number of the final condensed video, which should vary across videos and is actually set by users. $E_u$ and $E_p$ denote respectively the unary and pairwise energy functions encoding the constraints [13]. Minimizing such energy function is time-consuming due to the large search space, and needs to store all the tubes in memory thus requires huge storage space.

## 3.2. Online VC as Stepwise Optimization

In contrast to offline VC, the online VC determines the time label of each tube one by one, rather than all at once. This can be treated as a stepwise optimization problem. Specially, for the current tube $i$, we have

$$E(\ell_i) = E_u(\ell_i) + \sum_{j \in \boldsymbol{Q'}} E_p(\ell_i \mid \ell_j), \qquad (2)$$

where $\boldsymbol{Q'} \subset \boldsymbol{Q}$ denotes the subset of processed tubes which changes dynamically and is reasonably much smaller as compared to the whole set $\boldsymbol{Q}$, and $\ell_i$ will take a value in a much smaller label set:

$$\mathcal{L}_{\text{onlineVC}} = \{1, \cdots, n\}, \qquad (3)$$

where $n$ denotes the frame number of temporary condensed space, with $n \ll M$. $\ell_j$ is the known time label of processed tube $j$. The objective of the basic online method is to deal with tubes by optimizing Equ. (2) at each step, to give good approximations to optimal solutions to the problem in Equ. (1). Compared to Equ. (1), the search space of optimizing Equ. (2) is much smaller. Besides, there is no need to store all the tubes $\boldsymbol{Q}$ in memory but just those in $\boldsymbol{Q'}$. Therefore, it can achieve real-time tube processing with low memory cost.

## 3.3. Online Processing Structure

A specific processing structure using the optimization criterion in Equ. (2) is illustrated in Fig. 4. It includes six main stages:

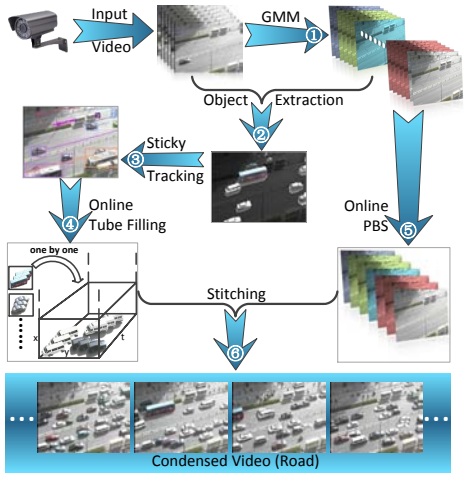


Figure 4. OVC Processing Structure.

**S**1: Background modeling: Generate the background sequence using Gaussian mixture models [16].

**S**2: Moving object extraction: Extract moving objects using a graph-cut segmentation method [17].

**S**3: Tube extraction. Extract tubes using the proposed sticky tracking method, which will be introduced shortly.

**S**4: Stepwise optimization of time labels: Find optimal labels of tubes using online tube filling.

**S**5: Online principal background selection (PBS): Select a size-fixed subset of backgrounds using online selection method. See [5] for more details.

**S**6: Object stitching: Stitch tubes into selected backgrounds (S5) using a modified Poisson image editing [13] according to the corresponding time labels (S4).

## 3.4. Online V.S. Offline Framework

The processing structure in Fig. 4 enables online real-time performance owing to advantages brought about by the key stage: online tube filling. It is such the step makes our method different from offline video synopsis [13]. One can see Figure 13 in [13] for comparison of the processing framework. In video synopsis, all video data (tubes and backgrounds) should be ready before optimization, which means that such offline framework needs huge memory to store them and much more time to find solution in large search space. In other words, users have to wait for a long computational time to borrow condensed video after the video is finished, and such property of offline framework will reduce the user experience to a low degree. Besides, in video synopsis the length of condensed video is determined by users who don't know the density of input video

beforehand. In contrast, the online framework does it by the content of input video automatically.

## 4. Online Tube Filling

The idea of online tube filling comes from Tetris, suggesting to deal with tubes one by one, rather than all at once as in [13]. The objective of Tetris is to manipulate the tetrominoes (the colorful polyominoes in the left picture of Fig. 1) with the aim of creating a horizontal line of blocks without gaps, then such a line can be cleared. If the player is smart enough, the game can go on forever. Similarly, this online method treats tubes as tetrominoes, and regards a 3D condensation space as the playing field of video condensation. Now, our job is to design a smart player for the tube filling game by optimizing Equ. (2).

### 4.1. Two Level Cache Condensed Space

Two level cache condensed spaces are introduced as the playing field of tube filling game as shown in the left picture of Fig. 5. These two spaces are with different size: the first level, $L1$, cache space is $w \times h \times n$ while the second level, $L2$, cache sapce is $w \times h \times \infty$, where $w$ and $h$ denote the width and height of frame respectively, and $n$ denotes the frame number in the cache space, which is the same with the one in Equ. (3). The dynamic tube set $\boldsymbol{Q'}$ in Equ. (2) is made of tubes in these two level cache condensed spaces.

The $L1$ cache space is the real playing field of the tube filling game. That is to say, an incoming tube is limited to select a certain frame of the $L1$ cache space from $n$ frames as the starting position to show. Besides, the function of $L2$ cache space is to receive the tail of the coming tube if $L1$ cache space cannot hold the whole tube.

### 4.2. Embedding Collision Cost

Collision is one of the most important criteria in the online tube filling. In [13], the collision cost is defined as the volume of two tubes' space-time overlap weighted by their activity measures. However, this cost ignores tubes with small size. To overcome this drawback, a new collision cost will be necessary to give high penalty for this case.

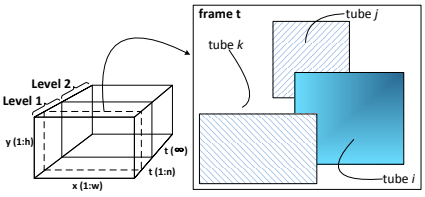

Figure 5. Two situations of tubes collision.

Assuming the current tube $i$ and a processed tube $j \in \boldsymbol{Q'}$ were placed at the location $\ell_i$ and $\ell_j$ respectively, the collision cost $E_c(\ell_i \mid \ell_j)$ between them is defined as:

$$E_c(\ell_i \mid \ell_j) = \sum_{t \in t_i \cap t_j} \Big( \mathcal{I}^t_{i,j} \cdot A^t_{i,j} + (1 - \mathcal{I}^t_{i,j}) \cdot B^t_{i,j} \Big), \quad (4)$$

where $t_i \cap t_j$ is time intersection of tube $i$ and $j$ in the cache condensed space, $A^t_{i,j}$ denotes the penalty caused by the situation that tube $i$ occludes tube $j$ at time $t$, and $B^t_{i,j}$ denotes the penalty caused by the situation that tube $i$ is occluded by tube $j$ at time $t$. See two cases in Fig. 5: tube $i$ occludes tube $j$ while tube $i$ is occluded by tube $k$ at frame $t$. Additionally, $\mathcal{I}^t_{i,j} \in \{0, 1\}$ is an indicator: if tube $i$ occludes tube $j$, then $\mathcal{I}^t_{i,j} = 1$, otherwise, $\mathcal{I}^t_{i,j} = 0$. Next, $A^t_{i,j}$ and $B^t_{i,j}$ in Equ. (4) are formulated as follows:

$$A^t_{i,j} = \begin{cases} O^t_{i,j} & \text{if } \frac{O^t_{i,j}}{\text{Area}^t(j)} < \beta \\ \kappa \cdot \text{Area}^t(i) & \text{otherwise,} \end{cases} \quad (5)$$

$$B^t_{i,j} = \begin{cases} O^t_{i,j} & \text{if } \frac{O^t_{i,j}}{\text{Area}^t(i)} < \beta \\ \kappa \cdot \text{Area}^t(i) & \text{otherwise,} \end{cases} \quad (6)$$

where $\text{Area}^t(\cdot)$ denotes the area function of the input tube at time $t$, $O^t_{i,j} \in [0, \min(\text{Area}^t(i), \text{Area}^t(j))]$ denotes the area intersection of tube $i$ and $j$ at time $t$, and $\beta \in [0, 1]$ denotes the maximal tolerable occlusion rate. In this paper, we consider the collision cost as the unique constraint: $E_u(\cdot) \equiv 0$ and $E_p = E_c$.

### 4.3. Energy Minimization for Tube Filling

#### 4.3.1 Greedy Minimization

As one of the simplest strategies for tube filling, greedy optimization selects the location with least collision cost:

$$\mathscr{L}(i) = \arg\min_{\ell_i} \sum_{j \in \boldsymbol{Q'}} E_p(\ell_i \mid \ell_j). \quad (7)$$

This greedy optimization makes the locally optimal choice at each stage with the hope of making the condensed space full with least collision at last. However, the greedy strategy makes a decision without considering future tubes, so there is still room for further improvements.

#### 4.3.2 Roulette Wheel Selection

Roulette wheel selection, known as a genetic operator used in genetic algorithms for creating the basis of the next generation [10], can be utilized as an optimization method for tube filling. First, we denote the function of the collision rate of tube $i$ when it is placed at location $\ell_i$:

$$\text{CR}_i(\ell_i) = \sum_{j \in \boldsymbol{Q'}} E_c(\ell_i \mid \ell_j) \Big/ \sum_{t \in t_i} \text{Area}^t(i), \quad (8)$$

Then we assign each location $j$ a fitness for the tube $i$: $\mathcal{F}_i(j) = 1/(1 + \text{e}^{-\omega(\beta/2 - \text{CR}_i(j))})$, where $\omega$ and $\beta/2$

control the shape and the central location of the sigmoid function respectively. The next step is to calculate each location's probability of being selected: $\mathbb{P}_i(\ell_i) = \mathcal{F}_i(\ell_i)/\sum_{j=1}^{n}\mathcal{F}_i(j)$. Finally, we can carry out the roulette wheel selection using the above probabilities. It is obvious that candidate locations with a low probability will be less likely to be selected, however, there is still a chance that they may be; this is an advantage, because such strategy gives the future tubes chance to obtain better locations.

### 4.4. Content-aware Tube Filling

One of the important issues in tube filling is how to decide whether $L1$ cache condensed space is full of tubes. Before a decision can be made, we need to decide whether a new tube $i$ can be filled into the cache space:

$$\mathrm{CR}_i\Big(\mathscr{L}_{\mathrm{greedy}}(i)\Big) > \tau, \tag{9}$$

where the function $\mathrm{CR}_i(\cdot)$ is defined in Equ. (8), $\mathscr{L}_{\mathrm{greedy}}(i)$ denotes the result of greedy selection of tube $i$ by Equ. (7), and $\tau$ the maximum tolerable threshold. The current tube $i$ cannot be filled into the cache space if the inequality (9) holds, then such tube will be added to a temporary list, and we simply assume the $L1$ cache condensed space is full once the length of this temporary list reaches a limit.

As the $L1$ space is full, its content is stitched to selected backgrounds for condensed video, then it will be cleared and the first $n$ frames of $L2$ space will be pushed into the $L1$ space. The benefit of this mechanism is that the length of the condensed video is determined by the content of the input video. This is what call content-aware condensation.

## 5. Experimental Results

In experiments, we use graph cut based background subtraction [17] as accurate segmentation of moving objects. However, the graph cut is computationally expensive, thus can't achieve real-time processing. Therefore, we design a *FIFO* thread pool to process the frames with graph cut. Table 1 shows the relationship between the speed of graph cut and the size of the thread pool (thread number). It runs at a 8 cores @ 2.66 GHz computer with frame size $320 \times 240$. It can be seen that such multi-thread cut enables faster-than-real-time processing.

| Thread Num. | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| Speed (fps) | 21 | 40 | 57 | 71 | 85 | 95 | 98 | 110 |

Table 1. The relationship between speed and thread number

The system of online content-aware video condensation is evaluated with extensive experiments. Nine surveillance videos are used. They are taken from indoor and outdoor environments. The final condensed videos are generated


Figure 6. The online condensed result for IndoorGTTest2 [3]. (a) Original Frames. (b) Condensed Frames.

by setting the size of thread pool of object segmentation to 7 with the greedy optimization for online tube filling. Fig. 6 shows the condensed result of surveillance video "IndoorGTTest2" [3] by our online method. It can be seen that the essence of the video condensation is to simultaneously display the moving objects (two men in this case) originally in different frames, which is a means for fast browsing of activities in videos. Fig. 7 shows that as the maximum tolerable threshold $\tau$ and max temporary list size (MTLS) increase, the condensation ratio grows. Additionally, compared to [9], our method has a far higher condensation ratio (the condensation ratio reported from [9] is lower than 3 while the max one of our method is 239).

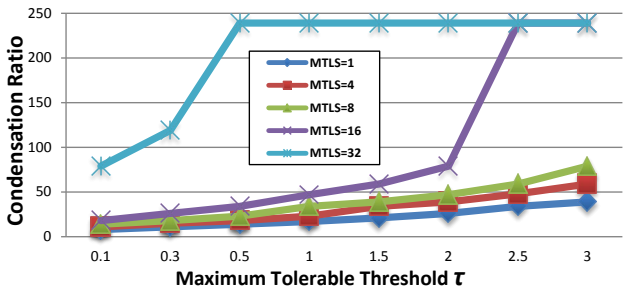


Figure 7. Relationship among condensation ratio, $\tau$ in Equ. (9), and MTLS (max temporary list size), with the "Overpass" [9].

| | Video Description | #Frame | #Tube | Speed (fps) | CR |
|---|---|---|---|---|---|
| 1 | Outdoor, $320 \times 240$ | 138583 | 1336 | 85 | 9.2 |
| 2 | Garden, $320 \times 240$ | 33826 | 374 | 94 | 26 |
| 3 | Park, $352 \times 288$ | 10221 | 56 | 71 | 102 |
| 4 | Exit, $320 \times 240$ | 81538 | 421 | 78 | 47 |
| 5 | IndoorGTTest1 [3] | 2659 | 9 | 88 | 26 |
| 6 | IndoorGTTest2 [3] | 1750 | 23 | 79 | 17 |
| 7 | Overpass [9] | 23950 | 59 | 211 | 119 |
| 8 | Road, $320 \times 240$ | 16897 | 343 | 77 | 16 |
| 9 | Park2, $352 \times 288$ | 11278 | 46 | 83 | 113 |

Table 2. The results of online content-aware video condensation on various types of surveillance videos. CR-Condensation Ratio.

The condensed results are summarized in Table 2 [1]. The results show that in addition to the aforementioned advantages, our method can achieve faster-than-real-time processing speed. Besides, in Table 2 the lowest condensation ratio is 9.2 while the highest one is 119, and this means that our method has adaptive condensation ratio varying from video to video according to the video content. This is more reasonable than by setting a fixed condensation ratio in video synopsis [13]: Just imagine how to condense a very-heavy-density video in a ratio of 100, and what's worse, the user doesn't know it beforehand. The nine surveillance videos of Table 2 are processed automatically without manual intervention, and our method can take endless surveillance video and generate endless condensed video.

| Method | Speed | Memory | CR | CA | Blinking Effect |
|---|---|---|---|---|---|
| VS [13] | 10 fps | High | User | No | High |
| RC [9] | Slow | Huge | Low | Yes | – |
| Ours | 70 fps+ | Low | High | Yes | Low |

Table 3. Comparison between our method and the state-of-the-art video condensation methods. Abbreviation: VS-Video Synopsis, RC-Ribbon Carving, CR-Condensation Ratio, CA-Content Aware.

Table 3 gives a summary of comparison between our online method and the state-of-the-art video condensation methods in terms of speed, memory used, condensation ratio, content aware and blinking effect. It shows that compared to other methods, our method has a faster speed with lower memory, and a content-aware condensation ratio with better visual quality. Someone may ask what happens if the non-online approach [13] is simply performed on a block-by-block basis. Actually, such way is unrealistic because non-online approaches need to set a fixed condensation ratio for each block, however, the density of each block is unknown beforehand.

## 6. Conclusions

Online content-aware video condensation has been proposed as an efficient mean for fast indexing and browsing of activities in video. Extensive experiments have clearly demonstrated the advantages. The online method can process live video in real time, thus it could be integrated inside the surveillance cameras. As a result, the security operator can take a fast browsing of condensed video at any time without waiting.

However, the video condensation technology by its functionality has some limitations: 1) It's difficult to achieve good condensation ratio for videos with dense activity, and it outputs nearly as long as the original video. This limitation could not be tackled apparently in the current frameworks. 2) It is difficult to condense a video containing objects that last for a long time. Keeping such endless tubes not only leads to a low condensation ratio but also consumes huge memory. Dividing tube with least blinking effect may be a solution.

The online method currently uses local minimizers. As it is inspired by the game Tetris, existing AI algorithms for the Tetris game [4] could be incorporated to better approximate the global solution, which is another future direction.

[1]The video results are at http://www.cbsr.ia.ac.cn/users/skfeng/ovc/

## Acknowledgement

This work was supported by the Chinese National Natural Science Foundation Project #61070146, #61105023, #61103156, #61105037, National IoT R&D Project #2150510, European Union FP7 Project #257289 (TABULA RASA http://www.tabularasa-euproject.org), and AuthenMetric R&D Funds.

## References

[1] S. Avidan and A. Shamir. Seam carving for content-aware image resizing. In *ACM SIGGRAPH*, New York, NY, USA, 2007. ACM.

[2] H. Benjamin, H. Markus, W. Daniel, and H. Gunther. Information-based adaptive fast-forward for visual surveillance. *Multimedia Tools and Applications*, 55:127–150, 2011.

[3] L. Brown, A. Senior, Y. Tian, J. Connell, A. Hampapur, C. Shu, H. Merkl, and M. Lu. Performance evaluation of surveillance systems under varying conditions. In *PETS*, pages 79–87, 2005.

[4] T. Christophe and S. Bruno. Improvements on learning tetris with cross entropy. *International Computer Games Association Journal*, 2009.

[5] S. Feng, S. Liao, Z. Yuan, and S. Z. Li. Online principal background selection for video synopsis. In *ICPR*, volume 0, pages 17–20. IEEE Computer Society, 2010.

[6] H.-W. Kang, X.-Q. Chen, Y. Matsushita, and X. Tang. Space-time video montage. *CVPR*, 2:1331–1338, 2006.

[7] S. Kasamwattanarote, N. Cooharojananone, S. Satoh, and R. Lipikorn. Real Time Tunnel Based Video Summarization Using Direct Shift Collision Detection. *Advances in Multimedia Information Processing-PCM 2010*, pages 136–147, 2010.

[8] C. Kim and J.-N. Hwang. An integrated scheme for object-based video abstraction. In *ACM MM*, pages 303–311. ACM, 2000.

[9] Z. Li, P. Ishwar, and J. Konrad. Video condensation by ribbon carving. *TIP*, 18:2572–2583, 2009.

[10] M. Mitchell. *An introduction to genetic algorithms*. The MIT press, 1998.

[11] N. Petrovic, N. Jojic, and T. S. Huang. Adaptive video fast forward. *Multimedia Tools Appl.*, 26:327–344, 2005.

[12] Y. Pritch, A. Rav-Acha, A. Gutman, and S. Peleg. Webcam synopsis: Peeking around the world. In *ICCV*, pages 1–8, 2007.

[13] Y. Pritch, A. Rav-Acha, and S. Peleg. Nonchronological video synopsis and indexing. *TPAMI*, pages 1971–1984, 2008.

[14] A. Rav-Acha, Y. Pritch, and S. Peleg. Making a long video short: Dynamic video synopsis. In *CVPR*, pages 435–441, 2006.

[15] M. A. Smith. Video skimming and characterization through the combination of image and language understanding techniques. *CVPR*, 0:775, 1997.

[16] C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. In *CVPR*, pages 2246–2252, 1999.

[17] J. Sun, W. Zhang, X. Tang, and H.-Y. Shum. Background cut. In *ECCV*, pages 628–641, 2006.

[18] M. Xu, S. Z. Li, B. Li, X. Yuan, and S. Xiang. A set theoretical method for video synopsis. In *Multimedia Information Retrieval*, pages 366–370, 2008.

[19] A. H. Y. Pritch, S. Ratovitch and S. Peleg. Clustered synopsis of surveillance video. In *AVSS*, 2009.

[20] T. Yang, S. Z. Li, Q. Pan, and J. Li. Real-time multiple objects tracking with occlusion handling in dynamic scenes. In *CVPR*, pages 970–975, 2005.

[21] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38, December 2006.