EasyChair Preprint

№ 6764

# Compressed Domain Consistent Motion Based Frame Scoring for IoT Edge Surveillance Videos

Lakshya, Venkata Suneel Kota, Mallikarjuna Rao Voleti and Shivraj Singh



October 3, 2021

# Compressed Domain Consistent Motion based Frame Scoring for IoT Edge Surveillance Videos

Lakshya[1], Venkata Suneel Kota[1], Mallikarjuna Rao Voleti[1], and Shivraj Singh[2]

[1] Samsung R&D Institute India - Bangalore, Bangalore, India
{lakshya.01,suneel.kota,vm.arjun}@samsung.com
[2] CCS University, Meerut, India
shivrajpundir@gmail.com

**Abstract.** IoT Edge is a major active technical front. Among others, video surveillance is one of the most common use cases for IoT Edge. However, there is a need to analyze the surveillance stream, due to the sheer size of the generated data. The stream can be analyzed in either the Edge environment itself or send to a cloud for analysis. There are two main constraints to be considered. First is the associated bandwidth cost to send the data to a cloud server for processing. It underlines the need to reduce the data sent to a cloud. Second is the computational and memory constraints of the devices in the IoT Edge Environment. It implies that only computationally cheap and fast algorithms can be allowed to run in the Edge Environment. However, generally highly effective algorithms require more computational resources and memory. Pruning of uninterested frames is a viable methodology that can potentially reduce the bandwidth cost and the resources utilized. We have developed a fast, computationally cheap, and effective frame scoring algorithm that scores frames based on the consistent motion. The algorithm works in the compressed domain using H.264 encoded motion vectors, by which it saves on the resources spent to decode the video stream. The algorithm can be used to prune uninteresting frames, while the interesting frames can be send either to the cloud or processed further in the edge itself.

## 1 Introduction

Video surveillance is one of the most frequent use cases for home monitoring services. Naturally, due to the sheer size of data being generated as a video feed, there is a need to analyze the data and extract useful information. It is preferable that the generated data be processed inside the Edge environment only. However, there is generally a lack of resources available to run such services. Not only are the devices available in Edge already resource-constrained, but there is a multitude of other essential services like WebRTC running on them.

It is also generally true that there is a correlation between the effectiveness of an analyzing service and the resources utilized. Consequently, either the algorithm needs to be run on the cloud or compromise has to be made on the effectiveness of the results. However, there is a middle ground available through pruning. In a typical video surveillance situation, most of the frames that do not contain consistent motion are ”uninteresting” and need not be considered. Hence, a fast, computationally cheap, and effective frame scoring algorithm can be utilized to weed out ”uninteresting” frames.

We focus our attention on the compressed-domain to realize the frame scoring algorithm, the reason being the processing and storage cost incurred to decode and store the stream in the pixel domain. We only analyze motion vector(MV) information available in the compressed stream to generate frame scoring. There is extra information [17] that can be further utilized to refine the results, such as DCT(Discrete Cosine Transform) coefficients and the Macroblock (MB) type. However, they would incur additional processing and memory costs.

Although, since single MV is assigned to entire MB whose minimum size is 4x4, we work at the level of 4x4 MBs. It implies, we use a MV field size is 4x4 times less than the size of original frame, which also helps in reducing execution time.

We provide the following contributions through this paper.

- We make novel observations regarding the limitations of existing approaches to capture orientation consistency information in videos.
- Based on these observations, we propose our $RLV$ function.
- We use this function to develop a fast, computationally cheap and efficient frame scoring algorithm that can be used to prune uninteresting frames.

## 2 Related Works

There are several tasks for which compressed domain techniques have been proposed. [3] used mean, standard deviation and saliency percentage values of x and y components of motion vectors and DCT coefficients to define frame feature vector. The task of high-speed action recognition is undertaken by [18]. They used motion vector approximated optical flow based similarity between query video and input video. The authors of [6,7] propose to consider Group of Pictures(GoP - P and B frames between two I frames) as a single unit and analyze DCT image of I-Frames to provide GoP scoring. [4,13] provide task specific video summarization techniques.

With respect to motion vector analysis following motion detection and video summarization tasks have interesting techniques. In [9,19] intensity, spatial orientation and temporal orientation maps are extracted from MVs. The spatial and temporal orientation maps are created by applying entropy functions to the values in a fixed spatial and temporal neighborhood of the point, respectively. [16] builds on the pixel domain video synopsis technique of [12]. They first apply median filter to the motion vector field and then extract LBP(Local Binary Pattern) features for each sub-macroblock They then use kernel density estimation on extracted LBP features to extract objects in the frame. While performing the task of moving object extraction in [2], the authors first perform global motion estimation, followed by a median filter. Then, weights are assigned to each MB based on the maximum difference with the 8 spatial neighbors. They go on to apply clustering and MRF to perform segmentation. In [20] first motion vector clipping is done. Then, the absolute difference of angle of a given point is calculated with the angle of spatial neighbors to gain confidence for the point. The authors of [1] present an interesting bi-directional motion vector accumulation technique. In which motion vectors of previous and forward frames are projected back to the current frame based on their motion vectors. Authors of [14] threshold the difference between intensity and

angle of the MVs of a macroblock and its reference MB. Although authors of [10] instead of using MV, use information like the type of sub-macroblocks and quantization parameter information in their work, they present the interesting idea of either one-step spatiotemporal processing or two-step spatial and temporal processing. In [8, 15] permutation and combination of techniques mentioned above is used.

## 3 Background

It can be seem from the previous section(2), that for us baseline contains not only the group of approaches for motion detection, but, also the pre-processing steps involved in video summarization activities, which also try to capture the consistent motion information for the frame. For example, In [19], intensity, spatial orientation, and temporal orientation maps are extracted from MVs, before combining them with visual maps extracted from RGB image pixels. However, there is no unique identifier for these pre-processing techniques, due to which in this section we give a brief description for the components used in this technique.

But before that, we describe a general template that we have observed in prior-arts to perform MV Analysis. We have observed that related works have closely followed this template to perform MV analysis.

### 3.1 MV Analysis General Template

- Calculate multiple maps using the MVs. A map can be created by defining a neighborhood set around a point, for each point in the MV field and a function to operate on the neighborhood set. A special Intensity Map - (shows intensity of MV at each pixel) is also generally used.
- Transform all maps, except Intensity map, using the following function.

$$f(a) = \begin{cases} 0, & a > NBH_{th} \\ NBH_{th} - a & a \leq NBH_{th} \end{cases}$$

  , where $NBH_{th}$ is a threshold whose value is set based on desired expected upper bound of function operating on the neighborhood.
- Multiply the maps together to get a Final Map.
- Score of a frame is simply the mean of all the values in the Final map.
- If the score is below defined threshold ($frame_{th}$) the frame can be pruned.

Figure-1, shows an example of MV analysis algorithm with this template. Map-1 uses Neighborhood Set-1 and $f_1$. Similarly Map-2 uses Neighborhood Set-2 and $f_2$. Map-3 corresponds to special Intensity Map.

### 3.2 Prior-arts

We have observed following definitions of neighborhood set around a point and functions operating on them.

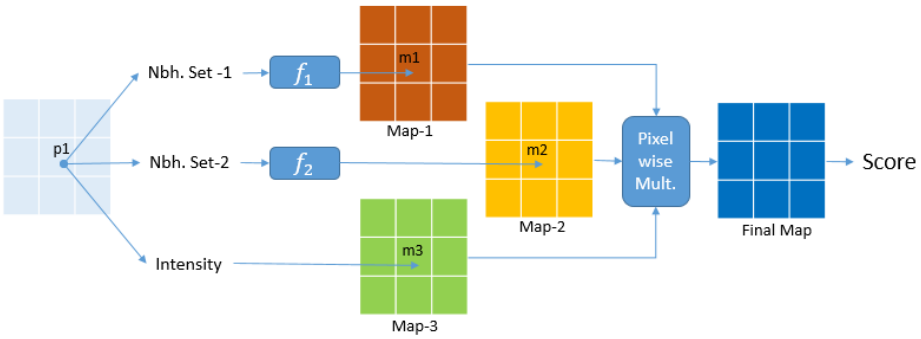


Fig. 1: Template example with two custom Maps(Map-1, Map-2) and one Intensity Map(Map-3).

**Neighborhood Set definitions for a point**

First, let

$$H(x) = \frac{x-1}{2} \tag{1}$$

*Spatial Neighborhood.* The spatial neighborhood($NBH$) of a point/pixel $p_{i_0,j_0,t_0}$ ( $i_0^{th}$ row and $j_0^{th}$ column on the $t_0^{th}$ frame in a video) with respect to a size $N$x$N$ spatial window is defined as following set

$$NBH_{i_0,j_0,t_0} = \{p_{i,j,t} : i_0 - H(N) \leq i \leq i_0 + H(N), j_0 - H(N) \leq j \leq j_0 + H(N),\ t = t_0\} \tag{2}$$

In above equation, we assumed that $N$ is an odd integer, as usually is the case, but similar equation can be written for even case. Also, note that current pixel is also considered as part of the Spatial Neighborhood.

*Temporal Neighborhood.* The temporal neighborhood of a point $p_{i_0,j_0,t_0}$ with respect to a size $N$ temporal window is defined as following set

$$NBH_{i_0,j_0,t_0} = \{p_{i,j,t} : i = i_0, j = j_0, t_0 - H(N) \leq t \leq t_0 + H(N)\} \tag{3}$$

*SpatioTemporal Neighborhood.* The spatiotemporal neighborhood of a point $p_{i_0,j_0,t_0}$ with respect to a size $N$x$N$x$M$ temporal window is defined as following set

$$NBH_{i_0,j_0,t_0} = \{p_{i,j,t} : i_0 - H(N) \leq i \leq i_0 + H(N), j_0 - H(N) \leq j \leq j_0 + H(N), \\ t_0 - H(M) \leq t \leq t_0 + H(M)\} \tag{4}$$

*Forward and Backward Accumulated Temporal Neighborhood.* We refer the readers to [1] for the explanation of this neighborhood set.

**Functions used to operate on the neighborhood set**

*Entropy over angle values - $Entropy$.* A histogram is created using the angles of the MVs in the neighborhood of $p_{i,j,t}$. Respective bin probability is calculated by dividing the bin value with summation of bin values. Then entropy is calculated over the resultant probabilities of the histogram as

$$Entropy_{i,j,t} = -\sum_{b=1}^{b=N} P(b)log(P(b)) \tag{5}$$

, where $N$ is the number of bins and $P(b)$ represents the probability of $b^{th}$ bin.

*Average of absolute difference* Absolute difference between the certain aspect of the pixel and its neighborhood is used. For a point $p_{i,j,t}$,

$$ABSD_{i,j,t} = \frac{1}{N}\sum_{k=1}^{k=N} |f(MV_{i,j,t}) - f(MV_{NBH_{i,j,t}(k)})| \tag{6}$$

, where $N$ is the number of points in the neighborhood, $NBH_{i,j,t}(k)$ represents the kth point in set $NBH$ and $|.|$ represents the absolute function. $f$ is a function that can be intensity of the vector or angle of the input vector. We refer to former as $ABSDoI$ and later as $ABSDoA$. The angle in $ABSDoA$ for a vector $V$ is calculated as $A(V) = arctan(\frac{Vy}{Vx})$

## 4 Observations

After checking multiple maps generated from prior art methods, we have made **novel observations** about the following caveats in the existing functions(Section-3.2) used to operate on any neighborhood set.

- Ideally, the value of the function at a point $p_{i,j,t}$ should be able to differentiate when the neighborhood values are similar or different than the value of $p_{i,j,t}$. However, $Entropy$ function is not able to do so. For example in figure 2, entropy values[3] of figure 2a(= 0.1514) and 2c(= 0.0) are more closer rather than that of figure 2b(= 0.2764) and 2c(= 0.0) or figure 2a(= 0.1514) and 2b(= 0.2764).
- If the function directly uses angle magnitude, there will be an erroneous blind spot. Consider for example that values of angle lie in the range [0,360], then even though magnitude wise 0 and 360 are separate, they actually point to the same direction.
- If the function directly uses angle magnitude, then even zero vectors are treated as vectors in some direction. Exact value depends on how angle is calculated from the $MVY_{i,j,t}$ and $MVX_{i,j,t}$, but even zero vectors are assigned angle values in the range [0,360]. Extra processing needs to be utilized to make sure that this phenomenon doesn't cause erroneous results, it might even hinder writing optimal code.

[3] Red and white pixels go into separate bins while calculating entropy on 3x3 spatial Neighborhood

Both $Entropy$ and $ABSDoA$ suffers from the above mentioned drawbacks. While, $ABSDoI$ doesn't even utilize the orientation information available within the MVs. However, orientation/angle consistency is an important information. For example, the difference between the motion of tree leaves and human movement is that of orientation consistency. Tree leaves will tend to show noisy motion with inconsistent orientation consistency, while, human movement tend to be more smoother.

(a) a (b) b (c) c

Fig. 2: Various possible values in a spatial neighborhood of size 3x3. Red and white corresponds to different angle values for the pixel MV.

## 5 Proposed $RLV$ and Pruning Algorithm

### 5.1 Relative variance over Normalized MVs - $RLV$

Given the incapability of existing operating functions to efficiently extract orientation consistency. We propose a new function, $RLV$, to operate on the neighborhood points.

Generally, variance of a given set of values is defined w.r.t. the mean of those values. However, we propose

- use variance w.r.t the point for which neighborhood is defined i.e. relative variance.
- relative variance to be calculated over normalized MV($normMV$ = unit vector in the direction of MV) rather than MVs themselves.

Given that, we define $RLV_{i,j,t}$ to be the sum of relative variance of $normMVX$ and $normMVY$ values in the $NBH_{i,j,t}$ w.r.t $normMVX_{i,j,t}$ and $normMVY_{i,j,t}$, respectively.

Mathematically, for a point $p_{i,j,t}$, $RLV_{i,j,t}$ is defined as

$$RLV_{i,j,t} = \frac{1}{N}\sum_{k=1}^{k=N}(normMVX_{i,j,t} - normMVX_{NBH_{i,j,t}(k)})^2 + \frac{1}{N}\sum_{k=1}^{k=N}(normMVY_{i,j,t} - normMVY_{NBH_{i,j,t}(k)})^2 \quad (7)$$

Assuming the general definition of vector difference, we can re-write it as

$$RLV_{i,j,t} = \frac{1}{N}\sum_{k=1}^{k=N} I(norm(MV_{i,j,t}) - norm(MV_{NBH_{i,j,t}(k)})))^2 \quad (8)$$

,where $I(.)$ refers to the Intensity. Equation - 8 has a very nice interpretation, because given two unit vectors V1 and V2.

$$\text{Intensity of } (V1 - V2) \propto \text{Angle btw } (V1, V2) \tag{9}$$

From equation 8 and 9, it can be deduced that $RLV$ map is a measure of angle consistency between the neighborhood of the pixel and the pixel. Our $RLV$ function is free of all the limitations mentioned in Section-4 and still incorporates angle consistency information into the algorithm.

Moreover, our $RLV$ only uses frame level operations as opposed to pixel level operations. Our $RLV$ function uses sum of relative variance w.r.t. $MVX$ and $MVY$. Consider, just the relative variance for $MVX$, it can be written as

$$RLV \text{ w.r.t. } MVX = \frac{1}{N}\sum_{k=1}^{k=N}(normMVX_{i,j,t} - normMVX_{NBH_{i,j,t}(k)})^2 \tag{10}$$

which can be rearranged as,

$$RLV \text{ w.r.t. } MVX = \frac{1}{N}(\sum_{k=1}^{k=N} normMVX^2_{NBH_{i,j,t}(k)}) + normMVX^2_{i,j,t} - (2 * normMVX_{i,j,t} * \frac{1}{N}\sum_{k=1}^{k=N} normMVX_{NBH_{i,j,t}(k)}) \tag{11}$$

Equation-11 utilizes neighborhood set mean and squared-mean ,and the $MVX_{i,j,t}$ value itself at a pixel $p(i,j,t)$ to calculate $RLV$ w.r.t. $MVX$. All of these can be easily computed at frame level for each pixel $p_{i,j,t}$ using fixed convolution kernels.This allows code optimization through techniques such as vectorization, which can help reduce execution time significantly. We were able to use these techniques to implement $RLV$ in Halide-C++(more details in Section-6.5).

### 5.2 Our Pruning Algorithm

Our Pruning Algorithm is based on the template mentioned in Section-3.1.

- We first apply a Median filter to the input Motion vector frame/field.
- We use only two maps in our algorithm
  - Map-1: Use $RLV$ function to operate on a temporal neighborhood($N$ = 7).
  - Intensity Map
- To calculate $NBH_{th}$, we first define an angle $\theta$ and then use following formulae, which represent the intensity square of the difference of two unit vectors, $\theta$ angle apart.

$$NBH_{th} = \sin^2(\theta) + (1 - \cos(\theta))^2 \tag{12}$$

In Section-6, we provide the results and ablation studies to justify our choices for the algorithm. We also show how our $RLV$ method outperforms other prior-art functions.

| Method | $NBH_{th}$ | $Frame_{th}$ | f1Score | precision | recall | accuracy |
|---|---|---|---|---|---|---|
| $ABSDoA$ | 30 | 2 | 0.734 | 0.627 | 0.885 | 0.641 |
| $ABSDOI$ | 30 | 15 | 0.755 | 0.634 | 0.932 | 0.661 |
| $Entropy$ | 3.83 | 0.01 | 0.749 | 0.628 | 0.946 | 0.65 |
| $Entropy - Norm$ | 3.83 | 0.01 | 0.721 | 0.561 | **0.99** | 0.56 |
| $RLV$ | **0.268** | **1e-6** | **0.8536** | **0.843** | 0.864 | **0.834** |

Table 1: Results by utilizing different operating functions

# 6 Results

## 6.1 Dataset Details

We use 20 videos that are similar to typical surveillance videos for generating results. In each video, frames containing interesting motion have been identified. By interesting, we subjectively mean movements of humans, cars, pets, etc., while the motion of leaves, water, etc. has been label as uninteresting.

- 16 of the videos are part of dataset2014 from ChangeDetection.net, [5]. However, in dataset2014 the ground truth is only provided for Region of interest(RoI). The provided ground truth also marks static objects. Hence, we re-annotated the dataset at frame-level. Following are the video names, busStation, copyMachine, cubicle, office, peopleInShade, PETS2006, sofa, winterDriveway, abandonedBox, backdoor, blizzard, boats, canoe, fall, streetLight and snowFall.
- 4 videos are taken from video summarization dataset by [11]. However, since they provided dynamic object mask as ground truth, we didn't have to annotate them ourselves. Following are the video names, Building, Cesta4, Transitway1 and Transitway2.

## 6.2 RLV vs others

Through these experiments, we want to prove that our $RLV$ function is better than other available approaches. For these set of experiments, we generate two maps for the scoring algorithms. One map uses spatial neighborhood($N$=5), while the other uses temporal neighborhood($N$ = 5). But, both maps are operated by the same function. Table-1 summarizes the F1-scores achieved.

For $Entropy$, both [9, 19], advocate to normalize the $Entropy$ maps to a range of [0,1], we believe that it is counter productive, because normalization kills the relative value of maps across frames.

- In Table-1, $Entropy$ method doesn't utilize normalization.
- In Table-1, $Entropy - Norm$ utilizes normalization and suffers f1Score drop.

It can be seen from the Table-1, that our proposed $RLV$ function far exceeds the results generated by other functions used throughout the literature.

| Method | $frame_{th}$ | f1Score | precision | recall | accuracy |
|---|---|---|---|---|---|
| Spatial | 1e-4 | 0.766 | 0.670 | 0.895 | 0.694 |
| Temporal | 1e-4 | 0.862 | 0.828 | 0.900 | 0.84 |
| Temporal-pooling | 1e-05 | 0.851 | 0.784 | 0.930 | 0.818 |
| Spatial & Temporal | 1e-06 | 0.853 | 0.843 | 0.864 | 0.834 |
| SpatioTemporal | 1e-7 | 0.859 | 0.894 | 0.825 | 0.848 |
| Fwd & Bwd Accu. | 1e-8 | 0.745 | 0.76 | 0.749 | 0.727 |

Table 2: Results by utilizing different Neighborhood Definition

### 6.3 Ablation Study: Results comparing neighborhood choices

Through these experiments, we want to justify our choice of using Temporal neighborhood set in the first map of our pruning Algorithm. In this section we provide results for following combination of neighborhood choices. Number of neighborhoods type involved in a choice also determine the number of maps considered in that frame scoring algorithm. We use $RLV$ function to operate on the neighborhood values, with $NBH_{th}$ = 0.267(corresponds to $\theta = 30$degrees)

- Spatial: A map of spatial neighborhood($N$ = 5) is used.
- Temporal: A map of temporal neighborhood($N$ = 5) is used.
- Temporal-pooling: Before any processing, raw MV field is processed by a 4x4 average pooling kernel(further explained in Section-6.5). A map of temporal neighborhood($N$ = 5) is used.
- Spatial and Temporal: A map of spatial neighborhood($N$ = 5) and map of temporal neighborhood($N$ = 5) is used.
- SpatioTemporal: A map of spatiotemporal neighborhood($N, M$ = 5) is used.
- Forward and Backward Accumulated Temporal Neighborhood: A map of forward and backward accumulated temporal neighborhood($N$ = 5)

Table-2 shows the results for various settings. It can be seen that temporal neighborhood map alone performs better than rest of the approaches. Hence, we have used temporal neighborhood map in our pruning algorithm(Section-5.2). A possible reason for spatial neighborhood map's negative effect is that spatial neighborhood map intrinsically applies a morphological erosion operations to all the edges of an object boundary.

A peculiar result is the low score for Forward and Backward Accumulated Temporal Neighborhood. We believe that the main cause is that if distance between the frames is more than 1, then accumulation happens with the sum of MVs between the frames rather than being recursively accumulated.

### 6.4 Ablation Study: Results by applying auxiliary techniques

In this section we apply several auxiliary techniques that can be used in conjunction with the template described in Section-3.1 and present the compiled results. For experiment in this section, we compare results with a baseline method in which we use $RLV$ function to operate on a spatial neighborhood($N$ = 5) and a temporal neighborhood($N$

= 5) with $NBH_{th}$ = 0.267. Results for baseline method are available in table-2, in the 'Spatial & Temporal' row. Results for following experiments are shown in Table-3.

**MV clipping** MV clipping is a technique in which all the MV with intensity lower than $clipping_{thresh}$ are set to zero. As shown in Table-3, there is a reduction in the f1Score by applying MV clipping w.r.t. baseline model. We have observed that the main reason for this is that MV clipping removes MVs corresponding to human movements, which tend to have lower MV as compared to those of cars,etc. As shown in the Table-3, there is a significant drop in the recall values corresponding to MV clipping.

**Median filter** Median filter has been used throughout the literature and rightly so. As, shown in Table-3, there is slight increase in the f1Score, owing to the increase in the precision score.

| Technique | variable | value | $Frame_{th}$ | f1Score | precision | recall | accuracy |
|---|---|---|---|---|---|---|---|
| MV Clipping | $clipping_{thresh}$ | 1 | 1e-7 | 0.781 | 0.884 | 0.699 | 0.78 |
| Median filter | $filter\_size$ | 5x5 | 1e-5 | 0.857 | 0.878 | 0.836 | 0.843 |

Table 3: Results by applying auxiliary Techniques

### 6.5 Execution Time Details

We have realized two possible ways to further reduce the time taken by our algorithm.

- 79.1% of the MBs in a frame are actually 16x16 coded. Thus, we can further reduce the size of the original MV field by 4x4. Hence, by applying a average pooling we can reduce the size of MV field to 16x16 of original frame. Performance metrics are shown in the Table-2.
- As mentioned in the Section-5, our $RLV$ can be implemented by frame level codes, which in turn allows us to use batch processing in order to reduce the time of execution at the expense of memory. Optimal batch size can be calculated based on the hardware and memory configurations of the device.

Table-4 shows the execution time for the baseline algorithm and the two alternatives applied independently to the baseline on full HD videos(1920 x 1080). The time shown in the table is the time taken by our algorithm to process the frame, excluding the time taken for extracting MV from the frame. We are using a python based implementation on an Ubuntu18.04 OS on top of an Intel Core i7-7700 CPU @ 3.60 GHz. We also implemented our algorithm in Halide C++ with optimized vectorized operations, results for same have been presented as well. It can be seen that our Halide implementation is extremely fast and optimal to use in IoT-Edge scenario.

| Method | Language | Time per frame(ms) |
|---|---|---|
| Baseline | Python | 20 |
| Pooling | Python | 16 |
| Batch[Batch size = 50] | Python | 11 |
| Baseline | Halide-C++ | 1.8 |
| Pooling | Halide-C++ | 0.7 |

Table 4: Execution Time

## 7 Conclusion

In this paper, we made novel observations regarding the limitations of existing approaches to capture orientation consistency information in videos. Based on these observations, we proposed our $RLV$ function to operate on a neighborhood set. We used this function to develop a fast, computationally cheap and efficient frame scoring algorithm. The frame scoring algorithm can be utilized in the IoT-Edge domain to prune uninteresting motion for the task of video summarization. This pruning methodology not only reduces the further processing cost but also the bandwidth cost if the further processing happens on a cloud.

## References


1. Babu, R.V., Ramakrishnan, K.R., Srinivasan, S.H.: Video object segmentation: a compressed domain approach. IEEE Transactions on Circuits and Systems for Video Technology **14**(4), 462–474 (2004). https://doi.org/10.1109/TCSVT.2004.825536 2, 4
2. Chen, Y., Bajić, I.V., Saeedi, P.: Moving region segmentation from compressed video using global motion estimation and markov random fields. IEEE Transactions on Multimedia **13**(3), 421–431 (2011). https://doi.org/10.1109/TMM.2011.2127464 2
3. Dong, P., Xia, Y., Feng, D.D.: Real-time storyboard generation for h.264/avc compressed videos. In: 2012 IEEE International Conference on Multimedia and Expo. pp. 544–549 (2012). https://doi.org/10.1109/ICME.2012.49 2
4. Erol, B., Lee, D.., Hull, J.: Multimodal summarization of meeting recordings. In: 2003 International Conference on Multimedia and Expo. ICME '03. Proceedings (Cat. No.03TH8698). vol. 3, pp. III–25 (2003). https://doi.org/10.1109/ICME.2003.1221239 2
5. Goyette, N., Jodoin, P., Porikli, F., Konrad, J., Ishwar, P.: Changedetection.net: A new change detection benchmark dataset. In: 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops. pp. 1–8 (2012). https://doi.org/10.1109/CVPRW.2012.6238919 8
6. Herranz, L., Martinez, J.M.: An efficient summarization algorithm based on clustering and bitstream extraction. In: 2009 IEEE International Conference on Multimedia and Expo. pp. 654–657 (2009). https://doi.org/10.1109/ICME.2009.5202581 2
7. Herranz, L., Martínez, J.M.: A framework for scalable summarization of video. IEEE Transactions on Circuits and Systems for Video Technology **20**(9), 1265–1270 (2010). https://doi.org/10.1109/TCSVT.2010.2057020 2
8. How-Lung Eng, Kai-Kuang Ma: Spatiotemporal segmentation of moving video objects over mpeg compressed domain. In: 2000 IEEE International Conference on

Multimedia and Expo. ICME2000. Proceedings. Latest Advances in the Fast Changing World of Multimedia (Cat. No.00TH8532). vol. 3, pp. 1531–1534 vol.3 (2000). https://doi.org/10.1109/ICME.2000.871059 3

9. Lai, J.L., Yi, Y.: Key frame extraction based on visual attention model. J. Vis. Comun. Image Represent. **23**(1), 114–125 (Jan 2012). https://doi.org/10.1016/j.jvcir.2011.08.005, https://doi.org/10.1016/j.jvcir.2011.08.005 2, 8
10. Laumer, M., Amon, P., Hutter, A., Kaup, A.: Compressed domain moving object detection by spatio-temporal analysis of h.264/avc syntax elements. In: 2015 Picture Coding Symposium (PCS). pp. 282–286 (2015). https://doi.org/10.1109/PCS.2015.7170091 3
11. Po Kong Lai, Décombas, M., Moutet, K., Laganière, R.: Video summarization of surveillance cameras. In: 2016 13th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS). pp. 286–294 (2016). https://doi.org/10.1109/AVSS.2016.7738018 8
12. Pritch, Y., Rav-Acha, A., Peleg, S.: Nonchronological video synopsis and indexing. IEEE Transactions on Pattern Analysis and Machine Intelligence **30**(11), 1971–1984 (2008). https://doi.org/10.1109/TPAMI.2008.29 2
13. Sugano, M., Nakajima, Y., Yanagihara, H.: Automated mpeg audio-video summarization and description. In: Proceedings. International Conference on Image Processing. vol. 1, pp. I–I (2002). https://doi.org/10.1109/ICIP.2002.1038186 2
14. Szczerba, K., Forchhammer, S., Stottrup-Andersen, J., Eybye, P.T.: Fast compressed domain motion detection in h.264 video streams for video surveillance applications. In: 2009 Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance. pp. 478–483 (2009). https://doi.org/10.1109/AVSS.2009.78 2
15. Wang, R., Hong-Jiang Zhang, Ya-Qin Zhang: A confidence measure based moving object extraction system built for compressed domain. In: 2000 IEEE International Symposium on Circuits and Systems (ISCAS). vol. 5, pp. 21–24 vol.5 (2000). https://doi.org/10.1109/ISCAS.2000.857353 3
16. zheng Wang, S., yuan Wang, Z., min Hu, R.: Surveillance video synopsis in the compressed domain for fast video browsing. Journal of Visual Communication and Image Representation **24**(8), 1431–1442 (2013). https://doi.org/https://doi.org/10.1016/j.jvcir.2013.10.001, https://www.sciencedirect.com/science/article/pii/S1047320313001818 2
17. Wiegand, T., Sullivan, G.J., Bjontegaard, G., Luthra, A.: Overview of the h.264/avc video coding standard. IEEE Transactions on Circuits and Systems for Video Technology **13**(7), 560–576 (2003). https://doi.org/10.1109/TCSVT.2003.815165 2
18. Yeo, C., Ahammad, P., Ramchandran, K., Sastry, S.S.: High-speed action recognition and localization in compressed domain videos. IEEE Transactions on Circuits and Systems for Video Technology **18**(8), 1006–1015 (2008). https://doi.org/10.1109/TCSVT.2008.927112 2
19. Yu-Fei Ma, Hong-Jiang Zhang: A model of motion attention for video skimming. In: Proceedings. International Conference on Image Processing. vol. 1, pp. I–I (2002). https://doi.org/10.1109/ICIP.2002.1037976 2, 3, 8
20. Zen, H., Hasegawa, T., Ozawa, S.: Moving object detection from mpeg coded picture. In: Proceedings 1999 International Conference on Image Processing (Cat. 99CH36348). vol. 4, pp. 25–29 vol.4 (1999). https://doi.org/10.1109/ICIP.1999.819460 2