TRACKING-OPTIMAL PRE- AND POST-PROCESSING FOR H.264 COMPRESSION IN TRAFFIC VIDEO SURVEILLANCE APPLICATIONS

E. Soyak [a], S. A. Tsaftaris [a,b] and A. K. Katsaggelos [a]

[a] Department of Electrical Engineering and Computer Science, Northwestern University
2145 Sheridan Rd., Evanston, IL 60208, USA
[b] Department of Radiology, Feinberg School of Medicine, Northwestern University,
737 N. Michigan Avenue Suite 1600, Chicago, IL 60611
email: {e-soyak, s-tsaftaris}@northwestern.edu, aggk@eecs.northwestern.edu

ABSTRACT

The compression of video can reduce the accuracy of automated tracking algorithms. This is problematic for centralized applications such as transportation surveillance systems, where remotely captured and compressed video is transmitted to a central location for tracking. In typical systems, the majority of communications bandwidth is spent on representing events such as capture noise or local changes to lighting. We propose a pre- and post-processing algorithm that identifies and removes such events of low tracking interest, significantly reducing the bitrate required to transmit remotely captured video while maintaining comparable tracking accuracy. Using the H.264/AVC video coding standard and a commonly used state-of-the-art tracker we show that our algorithm allows for up to 90% bitrate savings while maintaining comparable tracking accuracy.

Index Terms— Urban traffic video tracking, transportation, video compression, preprocessing, postprocessing

1. INTRODUCTION

Non-intrusive video imaging sensors are commonly used in traffic monitoring and surveillance. For some applications it is necessary to transmit the video data over communication links. However, due to increased bitrate requirements this assumes either expensive wired communication links or that the video data is being heavily compressed to not exceed the allowed communications bandwidth. Current transportation video solutions utilize older video compression standards and require dedicated wired communication lines. Recently H.264/AVC has started to be used in transportation applications, significantly reducing the link bandwidth requirement. However, most video compression algorithms are not optimized for traffic video data, nor do they take into account the possible data analysis that will follow at the control center. As a result of compression the visual quality of the data will suffer, but more importantly the tracking accuracy and efficiency are severely affected.

The field of video object tracking is quite active, with various solutions offering strength/weakness combinations suitable for different applications. For urban traffic video tracking most applications involve a background subtraction component for target acquisition such as the one developed in [1], and an inter-frame object association component such as those developed in [2, 3].

Most tracking algorithm models account only for the native statistics of video objects, and as a result distortion of these statistics by noise sources such as compression severely degrade their accuracy. In [4] a lightweight encoder-embedded LMMSE filtering algorithm is presented, showing significant success in maximizing post-compression video PSNR. However, the gains are in the domain of fidelity rather than automated trackability. In [5], special consideration is given to post-compression tracking, and a novel method of spatially concentrating bitrate via a Region of Interest derived according to pixel intensity statistics is presented. While shown to be effective in reducing bitrate requirements, this method does not attempt to limit frame-by-frame variation of low tracking interest but concentrates on defining a spatial region where interesting events commonly occur.

Many video compression systems today use Block-based Motion Compensation, where temporal redundancy is eliminated via the use of block motion vectors and frequency-transformed residuals. For typical traffic surveillance systems the camera is stationary, and the majority of bitrate is spent representing temporal changes to the scene such as capture noise or small changes to lighting. Not only are these changes costly to encode, but in most systems their imperfect compression is highly misleading to tracking algorithms.

In this work we present an algorithm that identifies and filters out such events in video, thereby allowing for compression resources to be focused temporally on events of tracking interest. Given the special requirements of centrally controlled traffic surveillance systems, it is necessary to limit resource requirements, such as memory and processing power, for any technique seeking to counter the effects of video distortion on tracking. The algorithm presented herein is low in complexity and is readily deployable as a simple modular add-on to low processing power remote nodes of centralized traffic video systems. It makes no assumptions about the operation of the video encoder (such as its motion estimation or rate control methods) and is thus suitable for use in a variety of systems. The resulting bitstreams are standard-compliant, thereby guaranteeing interoperability with other standard-compliant systems.

In Section 2 we discuss the effects of video compression on the efficiency of tracking algorithms, focusing on the distortion of features commonly used in real-time video object tracking. In Section 3 we propose our method of bitrate concentration on critical temporal events, for which we show experimental results in Section 4. Finally we present concluding remarks in Section 5.

2. COMPRESSION DISTORTION OF TRACKING

While the active field of video object tracking contains a large variety of algorithms, most of these systems share some fundamental concepts. In reviews of object tracking presented in [6] it is shown that most algorithms operate by modeling and segmenting foreground

and background objects. Once the segmentation is complete and the targets located, the targets are tracked across time based on key features such as spatial edges, color histograms and detected motion boundaries. The segmentation models and key features for a particular tracking application are chosen based on the application's goals and parameters. For example, color histograms can be useful when tracking highway vehicle activity during the day, but less useful under low light conditions at night. In [5] we discuss the effects of video compression especially debilitating for tracking algorithms.

In order to optimize for tracking quality a metric to measure tracking accuracy is required. In [7] a state-of-the-art review for video surveillance performance metrics is presented. Due to their pertinence in traffic surveillance for our work we choose the Overlap, Precision and Sensitivity metrics presented therein. *Overlap* (OLAP) is defined in terms of the ratio of the intersection and union of the Ground Truth (GT) and Algorithm Result (AR) objects,

$$OLAP = \frac{GT_i \cap AR_i}{GT_i \cup AR_i}, \tag{1}$$

where GT_i are the segmented objects tracked in uncompressed video, the AR_i those tracked in compressed video, $\cap$ the intersection of the two regions and $\cup$ their union. *Precision* (PREC) is defined in terms of the average number of True Positives (TPs) and False Positives (FPs) per frame as

$$PREC = \frac{\overline{TP}}{\overline{TP} + \overline{FP}}, \tag{2}$$

where TPs are objects present in both the GT and AR, while FPs are objects present in the AR but not in the GT. An FP is flagged if an object detected in the AR does not overlap an equivalent object in the GT (i.e. $OLAP(AR_i, GT_i) = 0$). *Sensitivity* (SENS) is defined in terms of TPs and False Negatives (FNs) as

$$SENS = \frac{\overline{TP}}{\overline{TP} + \overline{FN}}, \tag{3}$$

where FNs are objects present in the GT but not in the AR. An FN is flagged if an object detected in the GT does not overlap an equivalent object in the AR (i.e. $OLAP(GT_i, AR_i) = 0$). We define the aggregate tracking accuracy A as

$$A = (\alpha * OLAP) + (\beta * PREC) + (\gamma * SENS), \tag{4}$$

where α, β and γ are weighting coefficients. Given that OLAP, SENS, PREC are all in the range [0, 1], no normalization of A is necessary as long as $\alpha + \beta + \gamma = 1$.

3. PROPOSED METHOD

The proposed algorithm seeks to minimize the bitrate required for a given level of tracking accuracy. At reasonable compression ratios frame-to-frame variations to pixel intensity, such as due to capture noise or local changes to lighting, can be imperfectly represented in compressed video. For example, such variation may be sampled sparsely over time rather than on every frame, leading to seemingly random block refreshes in the decoded video. Such changes may be interpreted as significant motion and thus be highly misleading to many trackers. Our algorithm seeks to suppress such changes in compressed video, thereby reducing both the required bitrate and post-compression tracking inaccuracies.

Our algorithm operates in two distinct parts: (a) we model, detect, and remove temporal events of low tracking interest as a preprocess to compression, and (b) after decoding of the video at the receiver we use the estimated modeling parameters from part (a) to synthesize and insert noise into the video prior to tracking. Part (a) aims to minimize the bitrate requirement, while part (b) aims to improve post-compression tracking results.

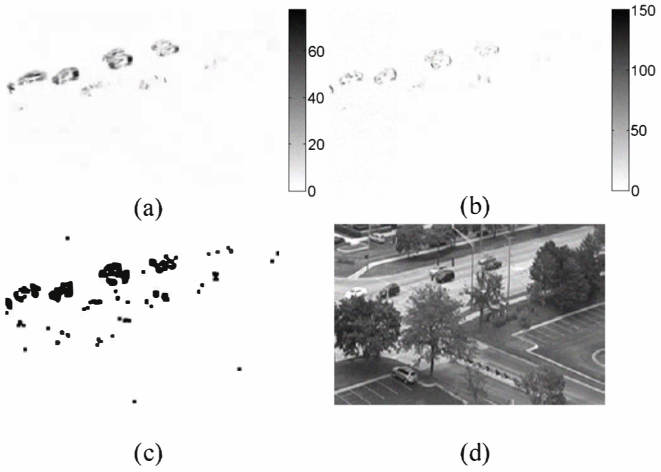

Fig. 1. Sample estimation process for n_t at $t = 18$ of the "Golf" sequence. Here shown are (a) σ_t, (b) Δ_t, (c) M_t and (d) $\hat{v}_t$.

We define temporal variations to pixel intensity of low tracking interest as *noise*. This noise is modeled for each frame t as a Gaussian process with zero mean and standard deviation n_t. We assert that for traffic surveillance video, the majority of temporal variation on each frame will be due to noise. We define σ_t, the standard deviation of each pixel's intensity over the past buffered B frames. Since we expect the majority of values in σ_t to be due to noise, we can estimate the noise standard deviation n_t by taking the mode of values of σ_t, i.e. $n_t = mode(\sigma_t)$. In order to generate a mask M_t, which is a bitmap of pixels in frame t whose variation is of tracking interest, we define Δ_t, the absolute difference in pixel intensities between frames t and $t-1$. Pixels in M_t where Δ_t exceeds n_t multiplied by a confidence coefficient C are marked as 1, while the rest are marked as 0. Therefore for the video sequence v, the estimated high tracking interest video $\hat{v}$ is iteratively computed as:

$$\sigma_t = std([v_{t-B} ... v_t]) \tag{5}$$

$$n_t = mode(\ vec(\sigma_t)\ ) \tag{6}$$

$$\Delta_t = v_t - v_{t-1} \tag{7}$$

$$M_t = |\Delta_t| > C * n_t \tag{8}$$

$$\hat{v}_0 = v_0$$

$$\hat{v}_t = M_t * v_t + \overline{M_t} * \hat{v}_{t-1} \quad \forall\ t > 0\ . \tag{9}$$

In Eq. 5, std is the per-pixel standard deviation, computed over the past B frames. In Eq. 6, $mode$ and vec are respectively the histogram mode and matrix vectorization operations. In Eq. 8, $||$ is the absolute value operation, and $\overline{M_t}$ is the logical inverse of the bitmap M_t. Refer to Fig. 1 for a sample iteration. In the figure, sample values are shown for σ_t (a) and Δ_t (b). The threshold $C * n_t$ is used to derive the mask M_t shown in (c), and finally the $\hat{v}_t$ shown in (d). Note that the encoder receives only the filtered input $\hat{v}_t$.

The derivation of the coefficient C is done based on the desired confidence interval CI. Since we model noise as Gaussian, its distribution is subject to the cumulative Gaussian distribution function Φ. Refer to [8] for further discussion on the derivation and use of Φ in this context. Using Φ, we express the probability of any given pixel variation being due to noise as the probability that it belongs to a zero-mean Gaussian distribution with standard deviation n_t:

$$CI = Prob(|\Delta_t| < C * n_t) = \Phi(C) - \Phi(-C). \tag{10}$$

Basically, the relationship being used here is that for any given Gaussian distribution, a certain percentage of all values are expected to lie within a given multiple of standard deviations from the mean.

The higher C is set, the more likely it will be that pixel variations will be classified as noise. This will reduce the required bitrate R (less noise will be coded), increase the precision $PREC$ (fewer false positives will occur due to miscoding of noise) but decrease the sensitivity $SENS$ (more false negatives will occur as more true motion is misclassified as noise and filtered out). For example, the values of $C = [1, 2, 3]$ are expected to filter out $[68.26\%, 95.45\%, 99.74\%]$ of Gaussian noise. However, the higher values of C will also mean that more actual motion is misclassified as noise and therefore erroneously filtered out.

The value of C can be evaluated offline or via a table lookup to avoid the complexity of evaluating Φ online. Note that the estimated n_t needs to be transmitted to the receiver for the second part of our algorithm – however, given that a single 32-bit number per frame is sufficient for this transmission, we will disregard the additional bitrate required by this operation.

As described above, synthetic noise is reinserted into the decompressed video as part of our algorithm. For each frame t randomly generated Gaussian noise with zero mean and standard deviation n_t is added to the frame, clipping any resulting overflows due to dynamic range. The goal of this post-process is to restore the Gaussian temporal noise characteristics the video possessed prior to compression. Given that many tracking algorithms rely on noise modeling for background subtraction, this step is critical to allow such trackers to be able to tell actual foreground such as vehicles from pixel variations artificially introduced by compression.

For video where multiple color components are available, our algorithm is applied independently to each component. While joint application to multiple components may improve the robustness of the algorithm, special care would have to be taken in this case where low-contrast scenes (such as in low-light or fog) may present a disadvantageous tradeoff between sensitivity and precision.

The modifications we propose for our algorithm are completely modular and can be implemented as a pre- and post-processing add-on to any system with reasonable headroom. We impose no additional latency to the system, and the resulting bitstream is fully standard-compliant. The parameters for the algorithm are the buffer size B, which should be set as high as possible given memory and processing power constraints, and the confidence parameter C, which should be set based on the specific application requirements in terms of $SENS$ and $PREC$.

4. EXPERIMENTAL RESULTS

To verify the gains possible with our algorithm a sample implementation was tested using multiple sequences with differing characteristics such as viewing angle, quality and type of vehicle traffic observed. Details for the sample implementation and experimental procedure in addition to test results are presented below.

The video compression experiments presented herein were performed using the open-source H.264/AVC encoder x264 [9] and the JM 16.0 H.264/AVC reference decoder. The open-source OpenCV [10] "blobtrack" module was used as the object tracker, where the Mean Shift object tracking algorithm presented in [2] was implemented. Prior to tracking, the areas in the scene of interest (such as roads or sidewalks) were manually segmented. This was done to better simulate state-of-the-art traffic surveillance applications, which would concentrate on tracking vehicles and pedestrians as opposed to other objects such as trees or clouds.

	Percentage Reduction in DFD		
	C=1	C=2	C=3
Golf	70.53	83.73	95.23
Camera6	53.33	80.34	88.79
dt_passat	83.14	89.61	91.19

Table 1. The reduction in DFD energy due to noise suppression using $C = \{1, 2, 3\}$. The percentage gain is computed by $1 - \frac{|\hat{v}_t - \hat{v}_{t-1}|}{|v_t - v_{t-1}|} \ \forall\, t > 0$.

In order to establish a baseline for tracking performance an unmodified H.264 encoder was run across a range of bitrates, where the bitrate was increased by coarsening the Quantization Parameter (QP) for each operating point. The bitstreams were then decoded to get reconstructed video, which after application of the manual segmentation mask was used for tracking. Accuracy analysis was performed based on Eq. 4. In order to keep the results "application neutral," e.g. without prioritizing precision over sensitivity, equal weights for the the accuracy components were used ($\alpha = \beta = \gamma = \frac{1}{3}$). A causal buffer of size $B = 7$ frames was used, which required a reasonable amount of memory available on most consumer-grade applications. Since these buffered frames are not modified, they can also be used as the frame buffer of the video system, thereby avoiding additional memory requirements for our algorithm. For our experiments 20 seconds of each sequence was processed at 30Hz.

To test our algorithm against the baseline, we perform our described pre-processing (noise estimation and suppression) prior to encoding, feeding $\hat{v}$ instead of the original v into the encoder. After the bitstream is decoded, we perform post-processing by using the estimated n_t to generate and insert random noise into the video prior to tracking. For the results presented herein $C = 2$ and $C = 3$ were used, so that on average respectively 95.45% and 99.74% of noise was expected to be filtered. With our "application neutral" tracking accuracy definition of $\alpha = \beta = \gamma = \frac{1}{3}$, we found that a value of $C = 2$ usually performs slightly better than $C = 3$. Both sets of results are shown here in order to illustrate sensitivity to C.

The following videos were used for testing. The "Golf" sequence (720x480) was shot on DV tape and is a relatively high fidelity source, showing a local road intersection with steady non-rush traffic. As part of the scene there are trees and parking lots for office buildings and a strip mall. The "Camera6" sequence (640x480) was used under the NGSIM license courtesy of the US FHWA. It shows an intersection with light traffic, with trees swaying in the wind and buildings casting reflections of passing cars as part of the scene. This video was MPEG4 intra-only compressed during acquisition and is thus significantly noisier than the "Golf" sequence. The "dt_passat" sequence (768x576) was made available courtesy of KOGS/IAKS Universität Karlsruhe. It shows a busy intersection with steady traffic interrupted by a traffic signal and a light urban rail crossing. This content is uncompressed luminance-only with significant capture noise as well as global illumination changes.

As discussed herein, most of the frame to frame changes to pixel intensity for stationary-camera video content are going to be due to noise. Our algorithm reduces bitrate by suppressing those changes of low tracking interest. In Table 1 the percentage reduction in the displaced frame difference (DFD), i.e. between $|v_t - v_{t-1}|$ and $|\hat{v}_t - \hat{v}_{t-1}| \ \forall\, t > 0$, is shown. As expected here we see that most of the DFD is suppressed as a result of the filtering, with more of it being suppressed the higher C is set. We do not expect values presented in Tab. 1 corresponding to $C = [1, 2, 3]$ to be exactly equal to the Φ-based values [68.26%, 95.45%, 99.74%]. This is primarily because noise is not the only contributor to the DFD, as interesting motion

such as vehicles and pedestrians are also part of it, and therefore not all of the DFD is subject to filtering. Secondarily, given that we estimate n_t on each frame, the accuracy of how much of the DFD we suppress is limited by how well we estimate n_t.

Refer to Fig. 2 for algorithm results using the "Golf" sequence, where we compare the rate-accuracy curves for default compression vs. our algorithm using $C = 2$ and $C = 3$. Note that despite the original content being high-fidelity and thus having litte noise, consistent bitrate gains from 60% to 90% are possible using our algorithm. Refer to Fig. 3 for results using the "Camera6" sequence. Note that once again consistent bitrate gains from 60% to over 90% are possible using our algorithm. Refer to Fig. 4 for results using the "dt_passat" sequence. Observe that here as well consistent bitrate gains from 70% to 90% are possible using our algorithm.

Given that the second part of our algorithm involves random generation of Gaussian noise, there will be some variability across given realizations of randomly generated noise, and therefore possibly in the subsequent tracking accuracy. The experimental accuracy values presented herein represent the mean of 16 experiments, where each time a different realization of random noise was inserted as a post-process. The standard deviation of the accuracy across all the experiments was in the interval [0.005, 0.015], which is reasonable for real-world systems in the operational predictability sense.

5. CONCLUSION

We have proposed a novel method of reducing compressed video bitrate required for a given level of tracking accuracy through filtering of temporal events with low tracking utility. We used statistics based on pixel-level intensity changes over time to estimate and suppress such events prior to compression, and used our estimates to synthesize noise simulating such events prior to tracking. We have demonstrated using a common tracker that our algorithm allows for up to 90% bitrate savings while maintaining comparable tracking quality.

Acknowledgement: This work was supported in part by the Northwestern Center for the Commercialization of Innovative Transportation Technology (CCITT).

6. REFERENCES

[1] S. Cheung, C. Kamath, "Robust Techniques for Background Subtraction in Urban Traffic Video", *Proc. VCIP*, 2009.

[2] D. Comaniciu, V. Ramesh, P. Meer, "Real-Time Tracking of Non-Rigid Objects Using Mean Shift", *Proc. CVPR*, 2000.

[3] A. Senior, A. Hampapur, Y.-L. Tian, L. Brown, S. Pankanti, R. Bolle, "Appearance Models for Occlusion Handling", *Proc. 2nd IEEE Workshop on PETS*, Kauai, Hawaii, USA, Dec. 2001

[4] L. Guo, O. C. Au, M. Ma, Z. Liang, "An Encoder-Embedded Video Denoising Filter Based On The Temporal LMMSE Estimator", *Proc. ICME*, July 2006.

[5] E. Soyak, S. A. Tsaftaris, A. K. Katsaggelos, "Content-Aware H.264 Encoding for Traffic Video Tracking Applications", *Proc. ICASSP*, March 2010.

[6] A. Yilmaz, O. Javed, M. Shah, "Object Tracking: A Survey", *ACM Computing Surveys*, 2006, Vol. 38, No. 4, pp. 13.1-13.45

[7] A. Baumann, M. Boltz, J. Ebling, M. Koenig, H. S. Loos, M. Merkel, W. Niem, J. K. Warzelhan, J. Yu, "A Review and Comparison of Measures for Automatic Video Surveillance Systems", *EURASIP Jour. on Im. and Vid. Proc.*, Vol. 2008, Article ID 824726

[8] A. H. Haddad, "Probabilistic Systems and Random Signals", *Prentice Hall*, 2006, ISBN 0-13-204345-9, pp 77

[9] http://www.videolan.org/developers/x264.html

[10] http://opencv.willowgarage.com

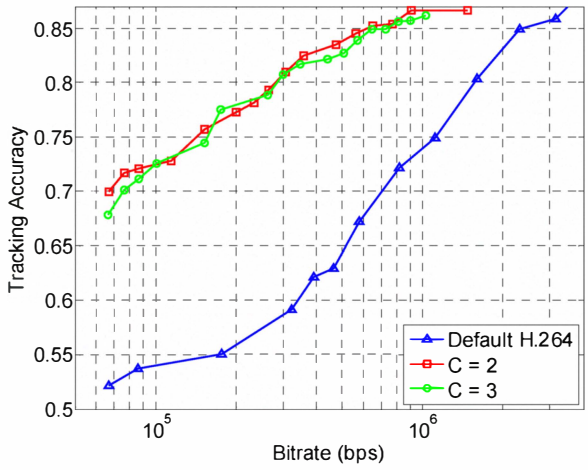

Fig. 2. Algorithm results for the "Golf" sequence (720x480).

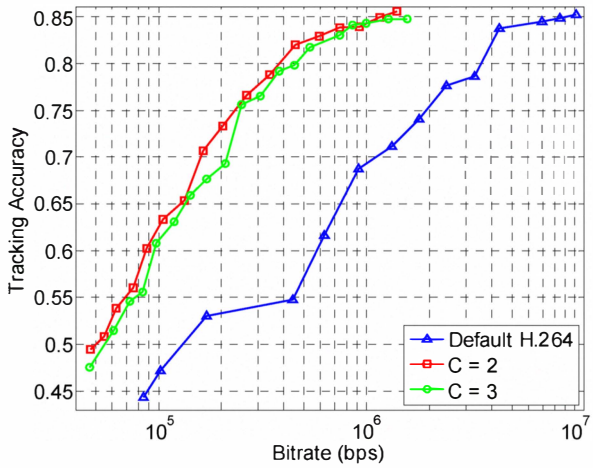

Fig. 3. Algorithm results for the "Camera6" sequence (640x480).

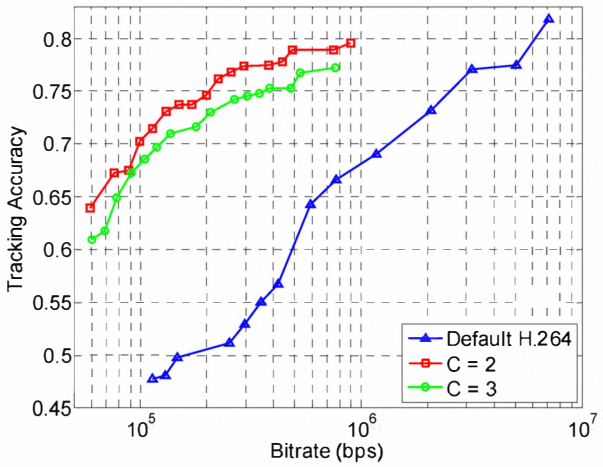

Fig. 4. Algorithm results for the "dt_passat" sequence (768x576).