

Camera Networks Dimensioning and Scheduling with Quasi Worst-Case Transmission Time

Viktor Edpalm

Axis Communications, Lund, Sweden

viktor.edpalm@axis.com

Alexandre Martins

Axis Communications and Department of Automatic Control, Lund University, Lund, Sweden

alexandre.martins@axis.com

Karl-Erik Årzén

Department of Automatic Control, Lund University, Sweden

karlerik@control.lth.se

Martina Maggio

Department of Automatic Control, Lund University, Sweden

martina@control.lth.se

 <https://orcid.org/0000-0002-1143-1127>

Abstract

This paper describes a method to compute frame size estimates to be used in quasi Worst-Case Transmission Times (qWCTT) for cameras that transmit frames over IP-based communication networks. The precise determination of qWCTT allows us to model the network access scheduling problem as a multiframe problem and to re-use theoretical results for network scheduling. The paper presents a set of experiments, conducted in an industrial testbed, that validate the qWCTT estimation. We believe that a more precise estimation will lead to savings for network infrastructure and to better network utilization.

2012 ACM Subject Classification Computer systems organization → Embedded systems, Computer systems organization → Real-time systems, Networks → Network components

Keywords and phrases worst-case transmission time, H.264, bandwidth estimation, video compression, network access scheduling, multiframe model, camera network

Digital Object Identifier 10.4230/LIPIcs.ECRTS.2018.17

Funding This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. Alexandre Martins, Karl-Erik Årzén, and Martina Maggio are members of the LCCC Linnaeus Center and the ELLIIT Excellence Center at Lund University.

1 Introduction

In the modern interconnected world, multiple devices share access to networking resources, such as transmission bandwidth. For some of these devices — e.g., video surveillance cameras connected to a monitoring station [21] — access to networking resources is often more critical than access to computing resources [27–29]. Scheduling network access is therefore crucial for the satisfaction of real-time requirements [1, 18, 25, 26], like the timely transmission of surveillance videos from different cameras [22].



© Viktor Edpalm, Alexandre Martins, Karl-Erik Årzén, and Martina Maggio; licensed under Creative Commons License CC-BY

30th Euromicro Conference on Real-Time Systems (ECRTS 2018).

Editor: Sebastian Altmeyer; Article No. 17; pp. 17:1–17:22



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

A typical video surveillance system comprises of multiple cameras disseminated over an area. These cameras continuously record a specific scene, it being an office space, a parking lot, a road, or any other alternative. The recorded scenes are of course different from one another, but their characteristics do not evolve significantly over time. A camera that is installed outdoor in a parking lot will record similar scenes, mostly involving cars and people, in different light conditions. At the same time, a camera that is pointing to a highway lane will (most likely) record either an empty road, or the passage of cars. A common challenge in the video surveillance industry is to tailor the entire infrastructure of the surveillance system to achieve a certain level of quality, while keeping the cost as limited as possible. Today, the video industry is mainly focused on using IP cameras, which stream videos that are compressed using the H.264 standard. In order to tailor the infrastructure, one must be able to anticipate how much data each camera in the system is expected to produce, given its unique set of internal characteristics and settings — e.g., position, placement, surrounding environment, etc. Such an estimate can be conservative, assuming that video frames are not compressed. Currently, conservative techniques are adopted for practical applications [1, 18, 22, 29]. However, conservativeness greatly increases infrastructure cost and limits the network usage. Non-conservative estimates have the potential of reducing the operational cost of video-surveillance systems. The challenge explored in this paper is therefore the estimation of the amount of data produced by each camera in the surveillance system.

We motivate our investigation by drawing a parallel between network scheduling for video surveillance camera systems and CPU scheduling. Using the periodic task model, a set $\mathcal{T} = \{\tau_1, \dots, \tau_p\}$ of p tasks execute on a given hardware platform. Each $\tau_i = \{E_i, P_i, D_i\}$ is activated at precise time instants, determined by the period P_i , and must meet a given deadline D_i . For scheduling policies to be effective at ensuring the satisfaction of deadline constraints in complex systems, schedulers use information about the Worst-Case Execution Time (WCET) E_i of a task τ_i on the given hardware.

Similarly, a set $\mathcal{C} = \{c_1, \dots, c_p\}$ of p surveillance cameras transmits video streams to a monitoring station. Each camera c_i has a given frame rate f_i , denoting the number of frames that the camera captures in a second. The frame rate has a direct implication on the transmission requirements of the camera, its inverse $1/f_i$ being equal to the activation period. For simplicity, we can assume that the deadline to transmit the currently captured frame is equal to the period. Hence, in this setting, reusing well-known CPU scheduling algorithms for network access depends on determining the Worst-Case Transmission Time (WCTT) for video frames. From the theoretical perspective, the task set model is not as simple as a set of periodic tasks, and can be described using a multiframe model [16], as will be shown in the following. Also, video encoders are very complex and the frame size depends heavily on the encoded scene. We therefore cannot compute precise upper bounds — e.g., using static analysis or formal methods — that guarantee that the given size is never exceeded. We therefore limit ourselves to the computation of quasi Worst-Case Transmission Times (qWCTT). We have experimentally verified that our estimate of the upper bound is valid in most cases and we have not encountered any case in which a frame exceeding our estimated upper bound is not a result of software bugs.

This paper contributes to the state of the art of real-time systems (and real-time surveillance video streaming) by:

- Determining a combination of measurable parameters that can accurately predict the expected H.264 frame sizes;
- Computing reasonable estimates of upper bounds for the qWCTT of frames of different

■ **Table 1** Nomenclature: Acronyms.

Acronym	Brief Explanation
GOP	Group of Pictures: Set of one I-frame and multiple P- and B-frames. The number also represents the amount of frames between two consecutive I-frames
HDR	High Dynamic Range: Technique used to enhance video, that typically allows frames to include more details and be sharper
IDR	Instantaneous Decoding Refresh: I-frame that imposes a refresh, i.e., following frames must not need any information from frames prior to the IDR I-frame
QP	Quantization Parameter: Compression parameter defined in the H.264 standard, higher numbers indicate more information loss
SAO	Size of Average Object: Reflects the expected distance to an object in an image, determined by factors like the zoom level, field of view, and lens type, as well as placement of the camera
WCET	Worst-Case Execution Time: Upper bound on the time it takes for a task to execute on a given hardware platform
WCTT	Worst-Case Transmission Time: Indicates the maximum time it takes to transmit a frame of the video using the available network bandwidth
qWCTT	quasi Worst-Case Transmission Time: Indicates a non-exact upper bound for the transmission time of a frame using the available network bandwidth

types over a network, casting the problem of scheduling switched Ethernet network access into a multiframe non-preemptive scheduling problem;

- Conducting a thorough experimental campaign to validate our findings and the given models, providing parameters for different camera models and circumstances and allowing researchers to use the derived models to verify real-time properties on the network transmission time.

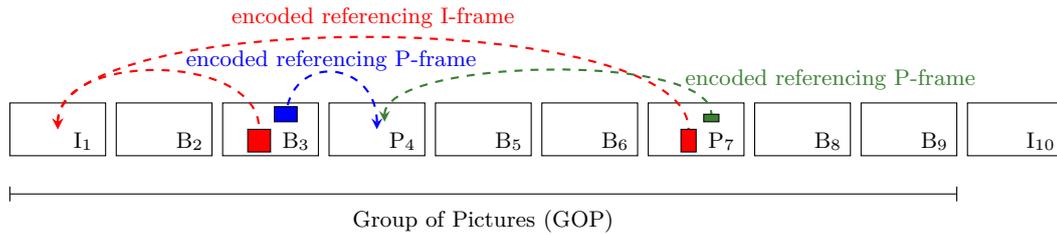
From the industrial perspective, the relevance of this paper is in enabling infrastructure tailoring for a video surveillance system and selecting quantities like the total required network bandwidth to guarantee a given video stream quality.

In the following, we review the H.264 standard and terminology in Section 2. Section 3 then discusses our models; enumerating the parameters, explaining how to measure them when needed, and showing the equations used to determine the frame sizes. Section 4 presents related efforts and Section 5 shows experimental results obtained with 6 different cameras in a laboratory environment and 24 different real-life surveillance scenarios. We finally conclude the paper in Section 6.

2 Background on Video Encoding

This section provides a brief overview of H.264, also called MPEG-4 part 10 AVC, which currently is the *de facto* standard for video encoding and decoding¹. Table 1 presents a recap of the acronyms used in the paper.

¹ The first official version H.264 version was approved in March 2003 [17, 30] and has since evolved over time. The standard now includes more features and modes, the latest version being approved in April 2017 [11]. The MPEG LA organization administers most of the licenses for patents applying to this standard.



■ **Figure 1** H.264 frame sequence: I-frames, P-frames, B-frames, and Group of Pictures.

H.264 is a video compression standard that defines how a video should be decoded. The implementation of the encoding is left to the manufacturer's discretion. The standard describes a *block based hybrid codec*, i.e., a video is decomposed in blocks of data for encoding. To allow for video compression, H.264 uses motion-compensated encoding, i.e., it describes a frame by referencing parts of other frames, thus capturing the motion of objects across different frames [30]. A stream encoded with H.264 contains a sequence of frames, these frames are not necessarily encoded following the display order or time they were captured. Based on the frame encoding, it is possible to distinguish between three different types of frames: Intra frames (I-frames), Predicted frames (P-frames), and Bi-directional predicted frames (B-frames).

- I-frames are (usually²) self-contained. An I-frame contains the full image and does not need additional information in the decoding process. In terms of encoding, these are fast and easier to encode, as all the information should be present in the resulting frame and no extra buffer containing other frames are necessary. In terms of size, on the contrary, these are the most space-consuming type of frames.
- P-frames are encoded using information contained in the current frame and in previous ones (up to the last self-contained I-frame). In the encoding of a P-frame, part of the image can be encoded using references to previous ones with extra information to reproduce the difference, instead of repeating the information. This allows the encoder to compress the frame, reducing its size, at the cost of additional computation and buffering.
- B-frames are encoded using both information from previous frames and information from following frames. In a B-frame, the encoder can introduce references to frames that come next, in display order, with respect to the current one being encoded. B-frames require the most computational capacity for the encoding, but are usually the lightest in terms of space consumption.

Figure 1 shows a sequence of 10 frames. The first nine frames in the example denote a Group of Pictures (GOP). A GOP consists of an I-frame followed by a sequence of B-frames and P-frames. The I-frame can be marked as an Instantaneous Decoding Refresh (IDR), meaning that the following frames do not need information from frames prior to that one in the sequence. If all the I-frames are marked as IDR points, the decoding of each GOP is independent, otherwise it is not. The sequence of frame types is determined and fixed by a high-level controller before the frame encoding starts.

² If an I-frame is marked as an Instantaneous Decoding Refresh (IDR), its encoding is self-contained. In most cases, this is true, but there are certain conditions in which this does not hold. Since we are interested in estimating the upper bounds, we can safely assume that the upper bound of an I-frame is self-contained.

- by referring to a block in a previous frame, containing a positional vector, the frame reference, and the residual information (Motion Compensation, P-block), or
- by referring to block in a previous or future frame, containing a positional vector, the frame reference, and the residual information (Motion Compensation, B-block).

The Motion Estimation function determines the cost for the four choices and selects the most appropriate one for the current macroblock.

The residual information is then Transformed, Scaled, and Quantized according to a Quantization Parameter (QP) to reduce its size. This is the only step where there is information loss and the higher the QP value, the higher the loss of information. The scaling, inverse transform and the deblocking filter allow the encoder to reconstruct the output frame and buffer it for future encoding. The entropy coding function uses lossless statistical compression to produce the final output frame.

3 Frame Size Estimation

The aim of this paper is to estimate an upper bound for the size of encoded video frames, to aid a potential external network manager towards a better scheduling of network capacity. The largest improvement is given when information-rich frames (I-frames) are treated separately from frames that can contain references to previous and future frames (P- and B-frames). The small difference in size of P- and B-frames and the similarity in the methods used for their construction justify the use of the same upper bound estimate for the two frame types. We therefore devise two models: an Intra Frame model for I-frames and an Inter Frame model for P- and B-frames. In Section 3.1 we explain what are the implication for network access scheduling. In Section 3.2 we describe the model we use for the estimation of the upper bound of the size of I-frames. In Section 3.3 we describe how to derive upper bounds estimates for P-frames and B-frames. In the following, we use \propto to indicate proportionality.

3.1 Scheduling implications

Assume it is possible to compute an upper bound estimate for the size of I-frames, denoted with I^* and an upper bound estimate for the size of P- and B-frames, denoted with P^* . Knowing the network speed \mathcal{N} , e.g., 100 Mbps, one can then translate these bounds into knowledge of the WCTT for the two types of frames in the network. The GOP parameter specifies how many “dynamic” (P- and B-) frames there are in between two “static” (I-) frames.

In fact, when a set $\mathcal{C} = \{c_1, \dots, c_p\}$ of p surveillance cameras share the same network, one can say that the i -th camera behaves according to the multiframe task model [16]. The camera has a vector of execution times $[E^0, E^1, \dots, E^{\text{GOP}-1}]$ and a single period and deadline, equal to the inverse of the frame rate $1/f_i$. E^0 is then equal to the upper bound estimate on the transmission time of the I-frame I^*/\mathcal{N} and all the other execution times $[E^1, \dots, E^{\text{GOP}-1}]$ are equal to the upper bound estimates on the transmission time of the P-frame, i.e., P^*/\mathcal{N} . This allows us to reuse theoretical results developed for the specific model [5, 10, 15, 32] or for its generalizations [4, 7, 9, 14, 19, 24, 31]. In particular, once we have determined the WCTTs for the different frame types, we can use the analysis on non-preemptive scheduling of multiframe tasks [3, 6] to determine schedulability properties for a set of video-surveillance cameras communicating over switched Ethernet [2].

As video encoders are very complex software elements, we cannot really compute an upper bound with static analysis or formal methods, that would guarantee that the size will never exceed the one predicted. However, we can compute an approximation (estimate) of

such upper bound, that is proven conservative in most cases. We believe that the very few circumstances in which the size of frames exceeds the computed values are due to problems and bugs of the execution of video-surveillance software. Therefore, we refer to I^*/\mathcal{N} and P^*/\mathcal{N} using the term quasi Worst-Case Transmission Times (qWCTT).

3.2 Intra Frame Model – I-frames

To determine the upper bound estimate I^* for the size of I-frames, we isolate the principal components that influence the amount of information included in the frame. Many acronyms and symbols are defined in the rest of the section. Table 2 contains a summary of the terms and constants that are needed for the estimation. The second column of the table contains a letter explaining how the value is obtained: [C] for computed, [K] for known, [M] for measured. Section 3.4 contains details on how to measure the [M]-parameters given a scene and a camera model.

Three different components influence the size of the frame: (i) the resolution of the video r , (ii) the compression level I_c , (iii) the actual camera and scene parameters I_a . There are many alternatives to write an expression of how each of these factors influences the size of the resulting frame. We decided to express I_c and I_a as scaling factors with respect to the resolution of the frame, therefore writing I^* as the product of the three terms,

$$I^* = \{r \cdot I_c \cdot I_a\}. \quad (1)$$

We now provide details for each of these terms separately.

- **Resolution r .** The frame resolution r is the number of pixels in the frame. Its value is equal to the product of the height h and the width w of the frame, $r = w \cdot h$. The resolution is linked to the number of macroblocks in the frame, therefore it influences its size directly.

- **Compression level I_c .** We denote with I_c the influence of the compression, $I^* \propto I_c$. The compression level QP determines the loss of information in each macroblock. From the H.264 standard, we infer that “an increase of 1 in QP corresponds to an increase of the quantization step size by approximately 12%” [30] (an increase of 6 means an increase of the quantization step size by a factor of 2).

In order to properly capture this relationship, we define a reference QP, denoted with QP^{ref} , and express I_c as a function of the difference between the current value and the reference value, $\Delta\text{QP} = \text{QP} - \text{QP}^{\text{ref}}$. We select $\text{QP}^{\text{ref}} = 28$ as the baseline. This choice is arbitrary, but represents a commonly used value, and does not affect the generality of the approach. ΔQP is used to scale the frame sizes between two compression levels, according to the relationship $I_c = 2^{-\frac{\Delta\text{QP}}{6}}$. The expression in Equation (1) thus becomes

$$I^* = \{r \cdot I_c \cdot I_a\} = \left\{r \cdot 2^{-\frac{\Delta\text{QP}}{6}} \cdot I_a\right\}. \quad (2)$$

- **Actual camera and scene parameters I_a .** The last component that influences the size of an I-frame includes a mix of camera and scene parameters, that we denote with I_a for “actual”. I_a includes two different terms, $I_a = I_d + n_{c,\ell}$. The first one, I_d , is related to how many details the scene has and how well the camera is able to retain that information. The second one, $n_{c,\ell}$ is related to the amount of noise generated in the camera. I^* then becomes

$$I^* = \{r \cdot I_c \cdot I_a\} = \left\{r \cdot 2^{-\frac{\Delta\text{QP}}{6}} \cdot I_a\right\} = \left\{r \cdot 2^{-\frac{\Delta\text{QP}}{6}} \cdot (I_d + n_{c,\ell})\right\}. \quad (3)$$

The detail influence I_d , captures how the scene details and their perception at the camera level affect the size of the frame. These can be separated into two categories: (i) scene-dependent parameters (each camera reacts differently to them, but they are a property of the scene), (ii) camera-dependent parameters. Parameters in the first category should be measured, while parameters in the second category are either measured or known, e.g., available from the camera manufacturer.

In the first category, we include the scene illumination ℓ , the scene detail level d_s , and the nature parameter n . In the second category, we include the camera detail level d_c , the enhancing factor e induced by features like High Dynamic Range (HDR), and the Size of the Average Object (SAO) in the scene, which depends for example on the zoom level enforced by the camera. The resulting I_d is the product of all these factors. In fact, the factors are known or measured as the relative difference that they produce in the I-frame size.

The frame size is greatly influenced by the illumination of the surroundings ℓ , given that more light allows the camera to capture the scene better while the absence of light hides details in the image. The value of ℓ represents the ratio between the current illumination level and a reference one, it is measured in a controlled environment with predetermined light levels. The result of the measurement is a value $\ell \in \mathbb{R}^+ \mid 0.25 \leq \ell \leq 1$. We consider three different light levels: low, medium, and high. A low light scenario is a scene recorded at night time, without any major light sources. A medium illumination scene is a night time scenario, with some light source illuminating the scene. A high illumination scene is a daylight scene, or a well lit indoor environment such as an office or a store. The high illumination scenario used as basis for scaling the remaining ones. This means that each camera at high light level has $\ell = 1$, and values for middle and low level are scaling factor that decrease the size of the frame. Given a camera model, these values can be determined experimentally as described in Section 3.4.

Directly connected with the light factor, is the level of details in the scene d_s . The scene detail level represents how many details there are in a scene, and can be measured in the field based on the different scenes. The resulting value is a number $d_s \in \mathbb{R}^+ \mid 500 \leq d_s \leq 2000$ expressed in millibits per pixel. Section 3.4 describes how to conduct field measurements.

We have experimentally found that the detail influence is also highly correlated to the amount of nature in the scene—lawns, bushes, trees, and similar. These features increase the difficulty of the encoding process, forcing the encoder to include more details in the resulting image, especially in the presence of wind. A high level description of the scene (e.g., a road, a garden, an office) allows one to provide an estimate of the amount of nature present in the frames. The nature factor n is expressed as the portion of the scene that includes natural elements, $n \in \mathbb{R}^+ \mid 0 \leq n \leq 1$. It can be easily measured on the field by taking a frame and computing a rough estimate. Typically, indoor scenes have a nature factor $n = 0$, while forest scenes have a nature factor $n = 1$. Common values for an outdoor parking lot are between 0.5 and 1. The factor included in the computation is $(1 + n)$, as the presence of nature only adds complexity to the scene, compared to the baseline.

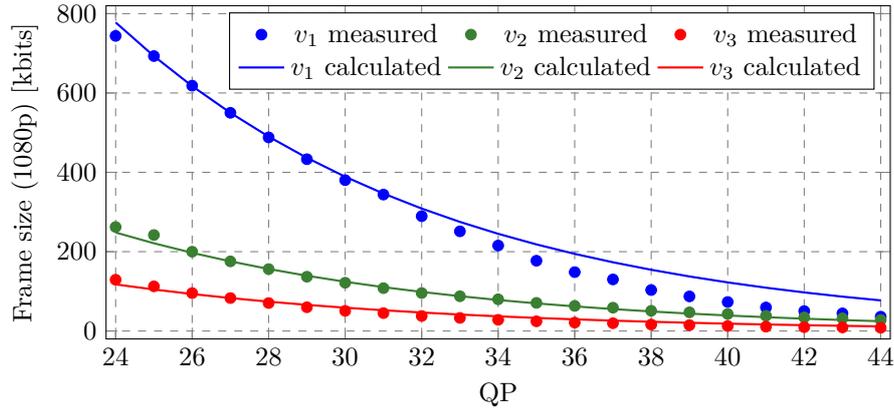
The camera properties should be taken into account when computing the detail influence. The factor d_c is used to scale the frame size taking into account factors like the sensor types, lenses properties, etc. The constant value d_c represents how well the camera captures the details in the scene and how sharp they are. A measure of d_c can be obtained with respect to a standard camera. The camera detail level d_c can be measured for a given camera as detailed in Section 3.4.

The dynamic range of the scene, together with the camera's ability of capturing it through various image enhancement techniques such as HDR is modelled using the enhancement factor, e . If one assumes that the different light ranges have the same bitrate characteristics and that the camera auto-exposure will select the range filling the most pixels then $e \in \mathbb{R}^+ \mid 1 \leq e \leq 2$.

■ **Table 2** Terms and Constants used in the Estimation of the upper bound for the I-frame size.

Acronym or Symbol		Brief Explanation	Range or Typical Values
d_c	[M]	Camera detail level: camera specific constant that reflects the camera capacity to retain scene details	$d_c \in \mathbb{R}^+ \mid 0.1 \leq d_c \leq 10$
d_s	[M]	Scene detail level: indicates the total amount of details in the scene	$d_s \in \mathbb{R}^+ \mid 500 \leq d_s \leq 2000$
e	[M]	Enhancement factor, indicates the effectiveness of High Dynamic Range (HDR) or similar technology	$e \in \mathbb{R}^+ \mid 1 \leq e \leq 1.35$
h	[K]	Height of a frame in pixels	~200–4320
I^*	[C]	Upper bound on the size of I-frames	
I_a	[C]	Influence of camera and scene	
I_c	[C]	Influence of the compression level QP	
I_d	[C]	Influence of the detail level	
ℓ	[M]	Scene illumination: it indicates the luminance (amount of light) in the scene, lower values indicate less light	$\ell \in \mathbb{R}^+ \mid 0.25 \leq \ell \leq 1$
n	[M]	Nature factor: amount of nature (trees, bushes, etc) in the scene	$n \in \mathbb{R}^+ \mid 0 \leq n \leq 1$
$n_{c,\ell}$	[M]	Noise level: camera specific constant indicating the amount of noise in the camera, capturing characteristics like sensor size and type; lower values indicate indoor high light and higher values low-light environments	$n_{c,\ell} \in \mathbb{R}^+ \mid 1 \leq n_{c,\ell} \leq 500$
QP	[K]	Quantization Parameter: reflects the frame compression, higher numbers indicate more information loss	$QP \in \mathbb{N}^+ \mid 1 \leq QP \leq 51$
QP^{ref}	[K]	Reference value used in measurements for the Quantization Parameter QP	28
ΔQP	[K]	$QP - QP^{\text{ref}}$	
r	[K]	Frame resolution (number of pixels in the frame)	~64000–35389440
SAO	[M]	Size of Average Object: reflects the expected distance of an object in an image, determined by factors like the zoom level, field of view, and lens type, and placement of the camera	$SAO \in \mathbb{R}^+ \mid 0.5 \leq SAO \leq 1.5$
w	[K]	Width of a frame in pixels	~320–8192

There are two corner cases, 1 and 2. $e = 1$ describes a scene with no additional dynamic range to capture, such as an indoor scene or a foggy day scene. $e = 2$ describes a scene where half the the frame is low dynamic and the other half is high dynamic, such as an indoor scene with large windows. An average value for all real world scenarios lays in between the two. The cameras that we tested had on average a 35% larger I-frame size when HDR was enabled, inducing $e \in \mathbb{R}^+ \mid 1 \leq e \leq 1.35$.



■ **Figure 3** Measured I-frame sizes and calculated ones for different videos, varying QP.

Another important factor affecting the I-frame size via I_d is the size of typical objects and details in the scene, denoted with the term SAO. This parameter can be approximated based on a combination of the distance to the scene, the zoom level and the field of view. The effect of this is to reduce the I-frame size for scenes where the objects are large, since the amount of details in a typical object usually does not scale with resolution. Section 3.4 provides an explanation of how to estimate this parameter.

The last parameter that we need to include is $n_{c,\ell}$, which captures the influence of noise generated in the camera (which in the end influences the size of the I-frame). We assume that the camera is the only source of noise, but the parameter value varies with the amount of light ℓ . In fact, the amount of noise is in direct relation to the scene noise level. The more light there is, the more sensor saturation, the more photons the sensor receives, and the less noticeable the camera noise becomes. The noise level is heavily camera dependent, and related to both hardware (optics and sensor) and software (exposure strategies, noise filtering technologies, and image settings). Depending on the different light conditions ℓ , the noise level can be measured. Values are $n_{c,\ell} \in \mathbb{R}^+ | 1 \leq n_{c,\ell} \leq 500$. The procedure to measure $n_{c,\ell}$ is described in Section 3.4.

Considering all the contributions to the upper bound estimate I^* , and substituting I_d and $n_{c,\ell}$ in Equation (3), we can finally write

$$I^* = \dots = \left\{ r \cdot 2^{-\frac{\Delta QP}{6}} \cdot (\ell \cdot d_s \cdot (1+n) \cdot d_c \cdot e \cdot \text{SAO} + n_{c,\ell}) \right\}, \quad (4)$$

obtaining our desired expression for the I-frame size upper bound estimate.

Figure 3 illustrates the results that we obtain using Equation (4) with a default camera. The figure represents data obtained with three different 1080p videos: v_1 , v_2 , and v_3 . The videos were encoded using different QP values in a standard setup where we know lighting conditions, detail level of both the scene and the camera, the size of objects, the enhancement features and the noise. We record I-frame sizes during the encoding with varying QP values, shown as dots in the Figure. The three lines represent the estimation obtained with Equation (4), which upper bounds the dot in almost every case.

3.3 Inter Frame Model – P-frames and B-frames

The same reasoning we used to estimate the upper bound of I-frames can be used to estimate the upper bound of the size of frames that can be encoded referencing macroblocks in other frames. The three components that provide contributions to the size of a P- and

■ **Table 3** Additional Terms and Constants used in the upper bound for the P-frame size.

Acronym or Symbol	Brief Explanation	Range or Typical Values
f_{inf}	[K] Inferior frame rate limit	2
f_{sup}	[K] Superior frame rate limit	120
f_s^{ref}	[K] Reference frame rate used for the motion level measurement	30
f_s	[K] Number of frames per second in the video (saturated)	$f_s \in [f_{\text{inf}}, f_{\text{sup}}]$; ~20–40
P^*	[C] Upper bound on the size of P-frames	
P_a	[C] Influence of camera and scene	
P_c	[C] Influence of the compression level QP	
P_d	[C] Influence of the detail level	
P_m	[C] Influence of motion	
μ_s	[C] Motion level: fraction of the image that is expected to be moving	$\mu_s \in \mathbb{R}^+ \mid 0 \leq \mu_s \leq 1$
μ_x	[M] Motion encoder efficiency: reflects the ability of efficiently encode moving object, an encoder with a large motion search window will have a low motion cost	$\mu_x \in \mathbb{R}^+ \mid 0 \leq \mu_x \leq 1$

B-frame are the same. We use P-frames as our basis, as we expect the encoder to be slightly more successful in encoding B-frames, therefore P-frames should represent an upper bound estimates for B-frames too. Table 3 summarizes the additional terms that are defined in this Section.

Using scaling factors with respect to the resolution (as we did for the I-frame), we define

$$P^* = \{r \cdot P_c \cdot P_a\}. \quad (5)$$

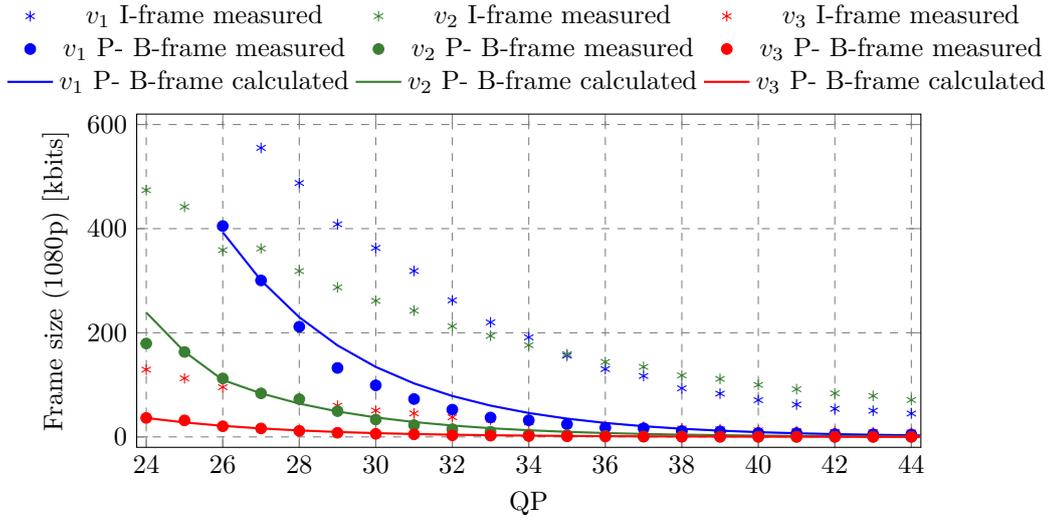
The first element contributing to the size of the frame is the resolution of the image r . The second and third components respectively are related to compression (P_c) and to the actual parameters of the camera and scene (P_a).

P-frames are highly correlated with neighboring frames, due to the compression algorithm. This makes the compression factor for P-frames larger than the one for I-frames and $P_c < I_c$. The relation between the compression parameter (QP) and frame size that we used for I-frames does not apply for P-frames due to this correlation. We introduce this by changing the compression term (the base 5 experimentally achieved through curve fitting):

$$P_c = 5^{-\frac{\Delta\text{QP}}{6}}. \quad (6)$$

P_a can again be split into two parts, one part relative to the influence of the detail level P_d and the noise $n_{c,\ell}$, which is the same term used for the I-frames, $P_a = P_d + n_{c,\ell}$. The difference between I_d and P_d , on the contrary, lies in the motion detected in the image. The encoding algorithm tries to find motion, starting from the same macroblock position in buffered images. We therefore encode $P_d = P_m \cdot I_d$, defining P_m as a multiplicative gain that explains the effect of motion on the resulting frame size, refining Equation (5) into

$$P^* = \{r \cdot P_c \cdot P_a\} = \left\{ r \cdot 5^{-\frac{\Delta\text{QP}}{6}} \cdot (P_m \cdot I_d + n_{c,\ell}) \right\}. \quad (7)$$



■ **Figure 4** Measured P- and B-frame sizes and calculated ones for different videos, varying QP.

The influence of motion on the P-frame size P_m is affected by three factors: (i) the frame rate f_s , (ii) the scene motion level μ_s , and (iii) a camera motion cost, which reflects how well the H.264 encoder captures encoding of moving objects, which we call motion encoder efficiency μ_x .

- **Saturated frame rate f_s :** P_m is directly linked to the frame rate of the video: the lower the frame rate, the more difference there will be between consecutive frames, the larger the motion step will be and the more objects would have moved. This larger gap will translate into higher chances of a motion miss by the encoder, and leads to higher bandwidth consumption. At extremely high frequencies or extremely low frequencies, the frame rate effect saturates. We therefore impose thresholds on the frame rate, forcing it to belong to the interval $[f_{\text{inf}}, f_{\text{sup}}]$. We have experimentally determined good values for f_{inf} and f_{sup} and respectively set them to 2 and 120. Using experimental data, we have determined that P_m is proportional to the inverse square of the video frame rate $P_m \propto \sqrt{f_s^{-1}}$.
- **Scene motion level μ_s :** The motion level of a scene is a measurable quantity at a certain reference frame rate f_s^{ref} , in our case equal to 30. This means that $P_m \propto \mu_s \cdot \sqrt{f_s^{\text{ref}}}$. The motion level determines the portion of the image that has moved from one frame to the next. If accurately known, μ_s can be uniquely used and varied per frame. However, since the primary use case of our upper bound is to estimate the required bandwidth there is a strong added benefit in simplifying the analysis. For simplification, we only use a generic set of possible motion levels: high, medium, and low. For high motion scenes, μ_s is typically around 0.15. For medium motion scenes, its value is around 0.07, and for low motion scenes 0.01.
- **Motion encoder efficiency μ_x :** The motion encoder efficiency is a measurable quantity per camera. The camera encoding capabilities are often dependent on the encoder capabilities and efficiency. The motion encoder efficiency can be measured, as explained in Section 3.4.

Including all the terms specified above, one can write $P_m = \mu_s \cdot \mu_x \cdot \sqrt{f_s^{\text{ref}}/f_s}$, and therefore, substituting P_m in Equation (7), we obtain our upper bound estimate

$$P^* = \dots = \left\{ r \cdot 5^{-\frac{\Delta QP}{6}} \cdot \left(\mu_s \cdot \mu_x \cdot \sqrt{f_s^{\text{ref}}/f_s} \cdot I_d + n_{c,\ell} \right) \right\}. \quad (8)$$

Figure 4 illustrates the results that we obtain using Equation (8) with a default camera with known parameters. The figure represents data obtained with three different 1080p videos, v_1 , v_2 , and v_3 . The lines represent the estimation obtained with Equation (8), which upper bounds the measured values, plotted with dots. We also report the measured size of I-frames for the same videos with asterisks.

3.4 Model Calibration

As indicated above, different constants need to be measured for the various cameras and scenes, in order to be able to extract meaningful numbers for Equations (4) and (8). These characteristics can be grouped in different sets: (i) platform-related, (ii) camera-related, and (iii) scene-related.

- **Platform-related characteristics.** The motion encoder efficiency μ_x is related to the platform (mostly the encoder) that is being used. In principle, the scene is also important in this case, but a scene-independent approximation can be computed. For each encoder generation and brand, the estimation of μ_x is done by isolating the encoder, or an equivalent encoder model, with a series of predetermined video sequences, encoded using varying compression.
- **Camera-related characteristics.** The three characteristics that we need to measure among the camera-related ones are d_c , $n_{c,\ell}$, and e . They are respectively: (d_c) the ability of the camera to retain scene details, ($n_{c,\ell}$) the amount of noise that the camera generates in specific light conditions, and (e) the enhancement factor added by technology like HDR. These are constants that summarize many different physical elements like the sensor size and quality.
- **Scene-related characteristics.** Four scene-related characteristics should be measured: the scene level detail d_s , the amount of nature n , the Size of the Average Object in the scene, SAO, and the amount of light ℓ .

Measurements should be collected in a reproducible environment. In our case, we collected the data in a dedicated laboratory. The main idea is to be able to reproduce certain scene conditions. The environment must contain different levels of details. It should be possible to shoot videos of areas with few or no details, as well as others with many details. It should also be possible to control the amount of light, at least to reproduce three different light conditions — high, medium, and low. Finally, there should be some reproducible source of motion, e.g., a fan or a toy train. The position of the camera with respect to the scene should be fixed in advance and should be reproducible as well. Figure 5 shows the laboratory in which the tests to compute the above mentioned parameters were conducted. Most measurements are conducted using a reference camera, and then for a new camera some additional data is collected to compare the camera to the reference one.

To determine the parameters we follow a specific procedure, both for the reference camera and for the model that we are trying to profile: (i) we record (repeatable) scenes with no motion, motion, no details, details, in three different light levels; using the compression level QP^{ref} ; (ii) we extract the frame sizes for all the I-frames and P-frames in the video; and (iii) we compute statistics for the videos, the average and maximum size of I- and P-frames.



■ **Figure 5** Image laboratory used to determine characteristics related to the camera and the scene.

For the camera detail level d_c , we compute the average frame size (for all the set of recorded videos), including both I-frames and P-frames and compare them with the values obtained with the reference camera. Denoting with S_{avg} the average frame size of the camera under test and with $S_{\text{avg}}^{\text{ref}}$ the one of the reference camera, $d_c = S_{\text{avg}}/S_{\text{avg}}^{\text{ref}}$. We repeat the same considering only low light conditions, and compute $n_{c,\ell}$ exactly using the same formula. The value of e for a given camera is determined by computing the ratio of the average frame sizes with HDR activated and deactivated.

To compute scene-level measurements, there are two alternatives. The first one is to physically record videos from the location where the camera should be installed, and the second one is to film similar scenes multiple times, and re-use the average measured parameter for similar scene types. We denote with $S_{I,\text{avg}}$ the average size of I-frames measured in bits for these measurements. We also want to collect videos done with the zoom level set to 50% for this case. The average size of the I-frames for this zoom level is indicated with $S_{I,50\%,\text{avg}}$.

The reference camera is used to measure the scene detail level d_s . Using the set of videos recorded from similar or the same scene, d_s is computed as the average size of I-frames expressed in millibits per pixel, $d_s = S_{I,\text{avg}} \cdot 1000/r$. The scene illumination ℓ is measured by comparing the laboratory result with the scene results using the actual camera to be used. From the laboratory results, we take videos recorded in high illumination scenes and compute the average size of I-frames for these videos as $S_{I,\ell=1,\text{avg}}$. We then compute ℓ as $\ell = S_{I,\ell=1,\text{avg}}/S_{I,\text{avg}}$. The amount of nature n is computed by looking at how many pixels in a frame are covered by nature.

Finally, we need to measure the size of the average object SAO. SAO is determined as $S_{I,50\%,\text{avg}}/S_{I,\text{avg}}$. The SAO levels can be, for simplicity, divided into three levels: large, medium, and small. As a general rule of thumb, one can determine the SAO level for 1080p video such as: (i) Large SAO: Objects taking up more than 1% of the pixels. An example is a licence plate camera, commonly setup to capture mainly a car with sufficient margin around it. (ii) Medium SAO: Objects are between 1% and 0.01% of the pixels. This is the most common case. (iii) Small SAO: Objects are very small, less than 0.01% of the pixels. This is sufficient only for scene awareness, i.e. knowing what happened in the scene, but does not permit to identify objects.

4 Related Work

The ultimate goal of this paper is to enable scheduling of network bandwidth in a video-surveillance system, utilizing the available bandwidth as much as possible. This goal can be achieved in many different ways.

One alternative to better utilize network resources is to reduce the amount of sent information by exploiting better compression and enhanced encoding. A lot of research has been devoted to adapting video stream quality to fit network channels [1, 12, 13, 20, 21, 23]. For example, adaptive strategies have been developed for MJPEG encoding [1, 23], MPEG-2 [12], and MPEG-4 [13]. Another alternative offer variable network channels [22, 29]. In this work, we investigate estimation of the WCTT for frames over a network, which is related to these works, but takes a different route. The aim of this paper is to devise a reasonably accurate model to aid scheduling decisions, without introducing adaptation.

To the best of our knowledge, there are two known alternative methods to estimate the frame size, and in turn the expected video bandwidth needed for the video transmission. These methods are based on other encoding methods (respectively MJPEG and MPEG-4) and aim to provide an estimate of the expected frame sizes. To the best of our knowledge, we propose the first frame size estimation for MPEG-4 part 10 AVC (H.264).

We denote the MJPEG method with **LIN**. This method only considers the compression parameter (QP for H.264 videos), and scales the frame size linearly according to such a parameter that we name q_l . Given a maximum size, identified with the term s_{max} , the frame size $s(q_l)$ is computed as $s(q_l) = q_l \cdot s_{max}$. The parameter q_l indicates the quality of the encoding, and relates, as indicated previously, to the Quantization Parameter QP. The scale and logic used are different and in MJPEG $q_l \in [0.01, 1.0]$, 1 being the lowest compression and 0.1 the highest, therefore $q_l = 1.01 - (QP/51)$. In the case of a 1080p YCbCr color video with 8 bits per pixel, $s_{max} = 1920 \cdot 1080 \cdot 8 \cdot 3 = 49766400$ [bits per frame]. This model is used for example in [22, 23] to devise a control strategy to determine the quality to be applied given a target bandwidth consumption.

We call the MPEG-4 model **RQM**. This model is used in [1] and described in [8]. It uses curve fitting to determine the parameters of a rate-distortion curve, modeled with a Gaussian random variable. Denoting with α a constant accounting for overhead bits, with β a constant that varies with the resolution and amount of motion in the video, with q_r the compression level for MPEG-4 ($q_r \in [1, 31]$), and with γ a constant that varies depending on the frame type (paper [8] providing recommended bounds of $\gamma \in [0.5, 1]$ for I-frames and $\gamma \in [0.5, 1.5]$ for P-frames), the size of the frame can be written as $s(q_r) = \alpha + \beta \cdot 1/q_r^\gamma$.

Notice that neither **LIN**, nor **RQM** compute proper upper bounds. They rather compute estimates of the frame size. We therefore do not expect them to be suitable for upper bounding the size of frames and obtaining WCTTs.

5 Experimental Results

In this section we present our experimental evaluation. We conducted many tests with different cameras and in different scenarios to validate the upper bounds estimates computed with our technique. We present two different categories of tests. Section 5.1 shows the results obtained for a controlled environment and a repeatable video, comparing our estimation strategy with state-of-the-art techniques. Section 5.2 presents a stress-test where we report the aggregate results of a large experimental campaign.

To conduct a comprehensive evaluation, we used 6 different camera models, and deployed them in 24 real-life (surveillance) scenarios. We refer to the different camera models using

■ **Table 4** Measured camera-related parameters.

Model	d_c	$n_{c,l}$ (high ℓ)	$n_{c,l}$ (medium ℓ)	$n_{c,l}$ (low ℓ)	μ_x
A	1.00	2.50	2.75	22.2	0.450
B	0.98	0.25	2.75	230.0	0.450
C	1.23	0.35	1.10	102.0	0.450
D	0.54	0.75	4.05	5.6	0.400
E	0.81	1.25	12.00	35.0	0.400
F	1.03	2.25	2.7	119.0	0.425

letters from A to F. Camera A was used as reference camera for the parameter estimation discussed in Section 3.4. To show the versatility of the model we use different parameters, resolutions, etc. Also, Camera C is a thermal camera. Table 4 contains the camera-related parameters that do not change with the scenario. Parameters that change with the scenario will be discussed in the corresponding sections.

5.1 Frame-by-Frame Evaluation

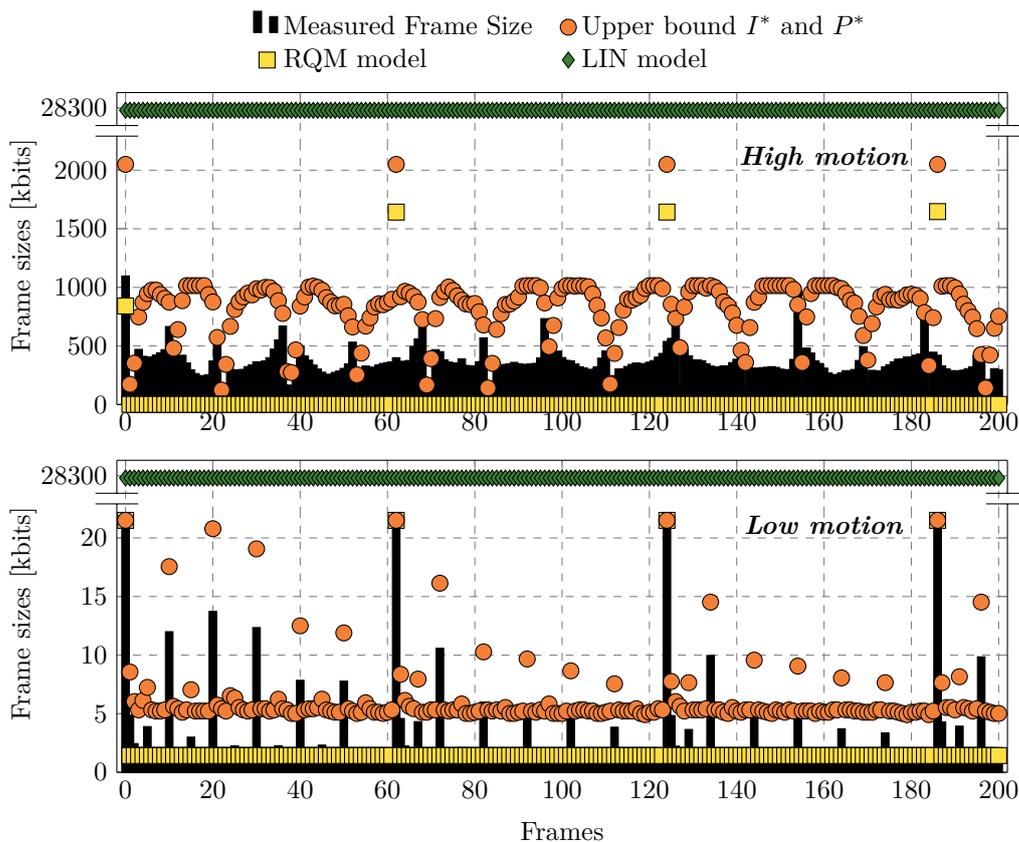
We present here a first validation experiment done with our reference Camera A. We recorded two videos of the same scene in our laboratory. The scene has a lot of details. The laboratory allows us to move the camera with predictable motion and control the amount of movement introduced in the image. Our aim is to show a frame-by-frame comparison between our frame size estimation and the state-of-the-art techniques discussed in Section 4.

The two videos differ in the amount of motion that is introduced³. A toy, present in the scene, allows us to introduce very limited but non-zero motion in both cases. In the first video, we also sharply changed the position of the camera. This simulates a fast movement for a video-surveillance camera. In the second video we kept the camera still, thus the only movement comes from the toy. The first video is characterized by a large amount of motion μ_s , while the second video has a very low μ_s .

The Camera A parameters for the two videos are: camera level detail $d_c = 1$, enhancement factor $e = 1.35$ (HDR), width $w = 1920$ [pixels], height $h = 1080$ [pixels], frame rate $f_s = 25$ [frames per second], QP = 29, noise level $n_{c,\ell} = 2.5$, motion encoder efficiency $\mu_x = 0.45$, GOP = 64. The scene parameters are: no nature, $n = 0$, very good illumination, $\ell = 1$, scene detail $d_s = 780$ [millibit per pixel], and size of the average object SAO = 1.

Figure 6 shows the results we obtained for the two videos. Each plot represents 200 frames of one video, the top one being the high-motion one and the bottom one being the low-motion case. The black bars represent the real frame sizes measured after the encoding. The circles represent the estimated upper bound on the frame sizes provided by the algorithm presented in this paper. The squares show the estimate produced by the LIN model, which does not take into account the difference between I, P, and B frames. Finally, the squares represent the estimate produced by the RQM model.

³ The two videos are available online: <https://www.youtube.com/watch?v=614BbbhD56M> (high-motion), and <https://www.youtube.com/watch?v=q4j3L1Vr01s> (low-motion). We have manipulated them to also visually show the motion vectors detected for both the original videos: <https://www.youtube.com/watch?v=5Yrx1GhadsY> (high-motion), and <https://www.youtube.com/watch?v=cfr08CZQa-E> (low-motion)



■ **Figure 6** Results of the comparison experiment with the high- and low-motion video.

For the RQM model, we used to the low-motion video to tune the parameters α , β , and γ , as recommended in [1]. The tuning resulted in $\alpha = 0.55$ and $\beta = 1.7$. As γ changes depending on the frame type, we fit $\gamma_I = 0.5$ and $\gamma_P = 4$ separately. The RQM tuning resulted in average errors on I-frames and P-frames respectively of 1.80% and 1.38%, which indicate very good performance for the low motion video. The square points in the lower plot of Figure 6 are therefore *a posteriori* estimations, and are clearly a very good fit for the video, despite the presence of a few outliers. The RQM model neglects motion — i.e., the β parameter is not sufficient to take motion into account. In fact, when the parameters determined with the low-motion video are used for *a priori* estimating the size of the frames in the high-motion video, the estimate frame size greatly underestimates the real value. The RQM approximation is therefore not a good fit to upper bound the size of the frames.

On the contrary, the LIN model gives very conservative results for both the high- and low-motion video, as its only parameter is a translation of the encoding quality QP. These are too conservative to be used in any practical setting, since the estimates are roughly 30 times as large as the real values. The LIN approximation is therefore also not a good upper bound for the size of the frames.

In the case of our upper bound estimates I^* and P^* , the circles represent for both plots *a priori* estimates based on the parameters that we have selected and on a standard computation of the motion level μ_s based on the percentage of pixels that differ from one image to the next (which could be determined before the encoding step). Roughly, the computed upper bound estimates are twice as large as the real values. While this could be

■ **Table 5** Parameters and results of the experiments conducted with 6 cameras in 24 real-life surveillance scenarios.

	f_s	QP	GOP	μ_s	ℓ	d_s	SAO	b_r	\hat{b}_r
1a (A)	25	28	62	$\approx 1\%$	1	780	1.00	1040	1275
1b (A)	25	28	62	$\approx 3\%$	1	780	1.00	1600	1806
1c (A)	25	28	62	$\approx 9\%$	1	780	1.00	3200	3398
1d (A)	12	32	32	$\approx 1\%$	1	780	1.00	544	600
1e (A)	12	32	32	$\approx 3\%$	1	780	1.00	720	723
1f (A)	12	32	32	$\approx 11\%$	1	780	1.00	1200	1219
2a (B)	15	28	62	$\approx \{2, 3\}\%$	$\{1, 0.8\}$	810	1.00	794	991
2b (B)	15	28	62	$\approx 0\%$	1	710	0.45	78	208
2c (C)	15	28	62	$\approx 1\%$	0.8	820	0.45	243	287
2d (A)	15	28	62	$\approx \{3, 5\}\%$	$\{1, 0.8\}$	990	0.45	669	765
2e (C)	15	28	62	$\approx 1\%$	1	810	1.00	513	761
2f (C)	15	28	62	$\approx 1\%$	$\{1, 0.8\}$	1400	1.00	333	490
2g (C)	15	28	62	$\approx 5\%$	1	920	0.45	409	456
2h (F)	15	28	62	$\approx 0\%$	0.8	710	0.45	45	96
2i (A)	15	28	62	$\approx 0\%$	$\{1, 0.5\}$	780	1.10	722	793
2j (F)	15	28	62	$\approx 4\%$	0.8	780	1.00	139	144
2k (A)	15	28	62	$\approx \{4, 3\}\%$	$\{1, 0.5\}$	780	1.00	194	220
3a (A)	25	28	32	$\approx 21\%$	1	1200	1.00	10000	10051
3b (A)	25	28	32	$\approx 4\%$	1	1200	1.00	2800	3116
4a (C)	30	18	32	$\approx 6\%$	1	660	1.00	4215	4551
4b (C)	30	18	32	$\approx 2\%$	0.5	780	1.00	4966	5321
5 (D)	25	24	4	$\approx 2\%$	1	990	1.00	42500	46529
6 (E)	25	32	32	$\approx 4\%$	0.5	660	1.00	2837	2878
7 (A)	15	36	30	$\approx 20\%$	1	1050	1.00	620	681

reduced with a more conservative setup of parameters, we believe that there could be a risk of cases in which the real frame size exceeds the upper bound estimate. In the full length of the two videos (low-motion 751 frames, high-motion 376 frames) this never happens for the low-motion case, and happens five times for the high-motion case. Inspecting these five occurrences prompted us to suspect some capturing error or some encoding miss, possibly due to the sharp movement.

5.2 Stress test

The purpose of the stress test is to verify that we obtain a reasonably good estimate of the bandwidth consumed by cameras to transmit their frame streams to a base station. We deployed our cameras in real-life surveillance scenarios and collected video streams for a time up to five days. We then measured the expected bandwidth consumption using estimates of the parameters (e.g., instead of computing precisely the motion level μ_s , we guessed it based on the type of recorded scene). We compared the measure expected bandwidth with the real bandwidth requirements — the videos' bitrates. The characteristics of the tested scenarios and the obtained results are summarized in Table 5, where b_r represents the bitrate, and \hat{b}_r its estimate.

The scene in scenarios 1a–1f is a highly illuminated parking lot, recorded with camera A ($e = 1.35, w = 1920, h = 1080$). Scenarios 2a–2k are videos from the surveillance system of a hotel complex. Camera B ($w = 1920, h = 1080$) in scenario 2a points at the reception entrance. In scenario 2b, Camera B ($w = 1920, h = 1080$) captures the emergency exit. Camera C ($w = 1920, h = 1080$) in scenario 2c films the control room. Camera A ($w = 1920, h = 1080$) in 2d is directed to the parking entrance. Camera C ($w = 1920, h = 1080$) in 2e films the reception. Camera C ($w = 1280, h = 720$) in 2f captures the corridor with shops. In 2g, Camera C ($w = 1280, h = 720$) is directed towards the elevator. Camera F in 2h films the staircase. Camera A ($w = 1280, h = 720$) in 2i streams a parking lot with nature $n = 0.5$. Camera F ($w = 704, h = 480$) in 2j and Camera A ($w = 704, h = 480$) in 2k film parking lots without nature. When the table contains two numbers for the motion level μ_s and for the light ℓ , this means that in the estimation the numbers are adjusted for day and night capture. The set includes first the day and then the night value. The value of e is set to 1 for 2b, 2c, 2e, 2f, 2h, 2j, which means HDR is turned off. In the other scenarios, HDR is turned on with a contribution of $e = 1.35$. The two instances of Camera A ($e = 1.35, w = 1920, h = 1080$) used in scenario 3a and 3b are placed in bridges on the highway and monitor car traffic. The two instances of Camera C ($e = 1$) of scenario 4a and 4b monitor a perimeter of a parking lot and the parking lot itself. In 4a the resolution is set to $w = 640, h = 480$, while in 4b the resolution is set to $w = 384, h = 288$. In scenario 5, Camera D ($e = 1, w = 3840, h = 2160$) streams a 4k video of the corner of a city street. Camera E ($e = 1.35, w = 3072, h = 1728$) in scenario 6 is filming a shipyard loading dock. Finally, in scenario 7 Camera A ($e = 1.35, w = 1280, h = 720$) is facing a city intersection.

Despite the high variety of scenes, the varying light conditions, the different cameras, and the different motion levels, the estimated bitrate \hat{b}_r (upper bound estimate) is always higher than the measured bitrate b_r . In most cases, the two values are very similar to one another (see for example scenario 1e or 3a). In a few cases, like 2b and 2h, it is possible to see that the upper bound overestimates the video bitrate (respectively 2.65 and 2.13 times as large). However, we believe these numbers provide a reasonable upper bound estimate and permit to correctly dimension the network bandwidth, aiding scheduling decisions.

6 Conclusion and Future Work

In this paper we presented a practical contribution on how to derive upper bounds estimates for the size of video frames in a streaming system. We have discussed which characteristics influence the bandwidth requirements of different cameras, derived models for the upper bound estimates of the size of I-, P-, and B-frames. We have also systematized the knowledge on the involved quantities and parameters. We divided such quantities into parameters that are known, characteristics that are measurable, and values that are computable. We have then taken the measurable characteristics and discussed how to conduct field tests to obtain reasonable values for them, and — when possible — how to guess based on the environmental conditions. Some parameters can be more or less easily estimated online (motion, light level, noise level, scene type...). Estimating these parameters on the source could lead to a more accurate and less pessimistic short term prediction. More frame by frame tests as well as highly challenging scenarios will also be ran in order to enhance the model.

The derivation of reasonable upper bounds estimates for the WCTT allows us to precisely formulate the problem of allocating network bandwidth to a set of cameras in a switched Ethernet network environment and to reuse well-known scheduling results. We have shown with a thorough experimental campaign that our estimated upper bounds are more reliable, and closer to the real frame sizes than state-of-the-art estimation techniques.

A proper estimation of the frame sizes is the key to properly dimension network infrastructures for real-time video-surveillance systems. Our results demonstrated that we can dimension the network infrastructure, being able to accurately predict the bitrate consumption of video streams. Our findings have a significant industrial relevance, as they permit to reduce the infrastructure cost and allows us to reuse known scheduling results.

References

- 1 Luís Almeida, Paulo Pedreiras, Joaquim Ferreira, Mário Calha, José Alberto Fonseca, Ricardo Marau, Valter Silva, and Ernesto Martins. Online QoS adaptation with the flexible time-triggered (FTT) communication paradigm. In *Handbook of Real-Time and Embedded Systems*, 2007.
- 2 Björn Andersson. Schedulability analysis of generalized multiframe traffic on multihop-networks comprising software-implemented ethernet-switches. In *IEEE International Symposium on Parallel and Distributed Processing*, pages 1–8, April 2008. doi:10.1109/IPDPS.2008.4536565.
- 3 Björn Andersson, Sagar Chaki, Dionisio de Niz, Brian Dougherty, Russel Kegley, and Jules White. Non-preemptive scheduling with history-dependent execution time. In *24th Euromicro Conference on Real-Time Systems*, pages 363–372, July 2012. doi:10.1109/ECRTS.2012.38.
- 4 Sanjoy Baruah, Deji Chen, Sergey Gorinsky, and Aloysius Mok. Generalized multiframe tasks. *Real-Time Systems*, 17(1):5–22, 1999. doi:10.1023/A:1008030427220.
- 5 Sanjoy Baruah, Deji Chen, and Aloysius Mok. Static-priority scheduling of multiframe tasks. In *Real-Time Systems, 1999. Proceedings of the 11th Euromicro Conference on*, pages 38–45, 1999. doi:10.1109/EMRTS.1999.777448.
- 6 Sanjoy K. Baruah and Samarjit Chakraborty. Schedulability analysis of non-preemptive recurring real-time tasks. In *Proceedings 20th IEEE International Parallel Distributed Processing Symposium*, 2006. doi:10.1109/IPDPS.2006.1639406.
- 7 Samarjit Chakraborty and Lothar Thiele. A new task model for streaming applications and its schedulability analysis. In *Proceedings of the Conference on Design, Automation and Test in Europe - Volume 1, DATE '05*, pages 486–491, Washington, DC, USA, 2005. IEEE Computer Society. doi:10.1109/DATE.2005.26.
- 8 Wei Ding and Bede Liu. Rate control of mpeg video coding and recording by rate-quantization modeling. *IEEE transactions on circuits and systems for video technology*, 6(1):12–20, 1996.
- 9 Pontus Ekberg, Nan Guan, Martin Stigge, and Wang Yi. An optimal resource sharing protocol for generalized multiframe tasks. *Journal of Logical and Algebraic Methods in Programming*, 84(1):92–105, 2015. doi:10.1016/j.jlamp.2014.10.001.
- 10 Ching-Chih Jason Han. A better polynomial-time schedulability test for real-time multiframe tasks. In *Proceedings 19th IEEE Real-Time Systems Symposium*, pages 104–113, Dec 1998. doi:10.1109/REAL.1998.739735.
- 11 ITU-T. Advanced video coding for generic audiovisual services. <https://www.itu.int/rec/T-REC-H.264-201704-I/en>, 2017.
- 12 Anand Kotra and Gerhard Fohler. Resource aware real-time stream adaptation for mpeg-2 transport streams in constrained bandwidth networks. In *IEEE International Conference on Multimedia and Expo*, pages 729–730, July 2010. doi:10.1109/ICME.2010.5583196.
- 13 Anand Kotra and Gerhard Fohler. Resource aware real-time stream adaptation of mpeg-4 video in constrained bandwidth networks. In *Visual Communications and Image Processing*, pages 1–4, Nov 2011. doi:10.1109/VCIP.2011.6116008.

- 14 Shuai Li, Stephane Rubini, Frank Singhoff, and Michel Bourdelles. A task model for tdma communications. In *Proceedings of the 9th IEEE International Symposium on Industrial Embedded Systems (SIES 2014)*, pages 1–4, June 2014. doi:10.1109/SIES.2014.7087455.
- 15 Wan-Chen Lu, Kwei-Jay Lin, Hsin-Wen Wei, and Wei-Kuan Shih. New schedulability conditions for real-time multiframe tasks. In *19th Euromicro Conference on Real-Time Systems (ECRTS'07)*, pages 39–50, July 2007. doi:10.1109/ECRTS.2007.20.
- 16 Aloysius K. Mok and Deji Chen. A multiframe model for real-time tasks. *IEEE Transactions on Software Engineering*, 23(10):635–645, Oct 1997. doi:10.1109/32.637146.
- 17 Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG. "Draft ITU-T recommendation and final draft international standard of joint video specification (ITU-T Rec. H.264/ISO/IEC 14496-10 AVC)", 2003.
- 18 Paulo Pedreiras and Luís Almeida. The flexible time-triggered (FTT) paradigm: an approach to QoS management in distributed real-time systems. In *International Parallel and Distributed Processing Symposium*, 2003.
- 19 Bo Peng and Nathan Fisher. Parameter adaption for generalized multiframe tasks and applications to self-suspending tasks. In *2016 IEEE 22nd International Conference on Embedded and Real-Time Computing Systems and Applications*, pages 49–58, Aug 2016. doi:10.1109/RTCSA.2016.15.
- 20 Naomi Ramos, Debashis Panigrahi, and Sujit Dey. Dynamic adaptation policies to improve quality of service of real-time multimedia applications in IEEE 802.11e WLAN networks. *Wireless Networks*, 13(4):511–535, 2007. doi:10.1007/s11276-006-9203-5.
- 21 Bernhard Rinner and Wayne Wolf. An introduction to distributed smart cameras. *Proceedings of the IEEE*, 96(10), 2008.
- 22 Gautham Nayak Seetanadi, Javier Cámara, Luís Almeida, Karl-Erik Årzén, and Martina Maggio. Event-driven bandwidth allocation with formal guarantees for camera networks. In *IEEE Real-Time Systems Symposium*, 2017.
- 23 Gautham Nayak Seetanadi, Luis Oliveira, Luis Almeida, Karl-Erik Arzen, and Martina Maggio. Game-theoretic network bandwidth distribution for self-adaptive cameras. In *15th International Workshop on Real-Time Networks*, 2017.
- 24 Martin Stigge, Pontus Ekberg, Nan Guan, and Wang Yi. The digraph real-time task model. In *Proceedings of the 2011 17th IEEE Real-Time and Embedded Technology and Applications Symposium*, RTAS '11, pages 71–80, Washington, DC, USA, 2011. IEEE Computer Society. doi:10.1109/RTAS.2011.15.
- 25 Linpeng Tang, Qi Huang, Amit Puntambekar, Ymir Vigfusson, Wyatt Lloyd, and Kai Li. Popularity prediction of facebook videos for higher quality streaming. In *USENIX Annual Technical Conference, USENIX ATC*, 2017.
- 26 Laszlo Toka, András Lajtha, Éva Hosszu, Bence Formanek, Dániel Géhberger, and János Tapolcai. A Resource-Aware and Time-Critical IoT framework. In *IEEE International Conference on Computer Communications*, 2017.
- 27 Bobby Vandalore, Wu-Chi Feng, Raj Jain, and Sonia Fahmy. A survey of application layer techniques for adaptive streaming of multimedia. *Real-Time Imaging*, 7(3), 2001.
- 28 Vilas Veeraraghavan and Steven Weber. Fundamental tradeoffs in distributed algorithms for rate adaptive multimedia streams. *Computer Networks*, 52(6), 2008.
- 29 Xiaorui Wang, Ming Chen, Huang-Ming Huang, Venkita Subramonian, Chenyang Lu, and Christopher D. Gill. Control-based adaptive middleware for real-time image transmission over bandwidth-constrained networks. *IEEE Transactions on Parallel and Distributed Systems*, 19(6), 2008.
- 30 Thomas Wiegand, Gary J. Sullivan, Gisle Bjontegaard, and Ajay Luthra. Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):560–576, jul 2003. doi:10.1109/TCSVT.2003.815165.

- 31 Haibo Zeng and Marco Di Natale. Outstanding paper award: Using max-plus algebra to improve the analysis of non-cyclic task models. In *2013 25th Euromicro Conference on Real-Time Systems*, pages 205–214, July 2013. doi:10.1109/ECRTS.2013.30.
- 32 Areej Zuhily and Alan Burns. Exact scheduling analysis of non-accumulatively monotonic multiframe tasks. *Real-Time Systems*, 43(2):119–146, 2009. doi:10.1007/s11241-009-9085-6.