Low Delay Video Transmission Strategies

Christoph Bachhuber, Eckehard Steinbach,
Artur Mrowca
Technische Universität München
Chair for Media Technology
christoph.bachhuber@tum.de,
eckehard.steinbach@tum.de

Martin Reisslein
Arizona State University
Department of Electrical Engineering
reisslein@asu.edu

Abstract

Video cameras are increasingly utilized to provide real-time feedback in automatically controlled systems such as autonomous driving and robotics. For such highly dynamic applications, the end-to-end (E2E) latency is critical. We present an analysis of latency in a point-to-point video transmission system and approaches to reduce latency in such setups. To realize the delay reduction, we propose a novel frame skipping and preemption approach. Simulation results demonstrate significant improvements of the approach in terms of E2E latency and channel usage over traditional video transmission.

1. Introduction

Cameras have the potential to replace many sensors for process control [8, 17]. There are several advantages of using cameras in combination with machine vision algorithms over dedicated sensors. Video cameras are comparably low cost and universally usable. They can replace for example radar sensors [1] or mechanical sensors by using visual tracking. Cameras can also replace multiple sensors at once and are compared to low-level hardware sensors more future-proof through improvements in computer vision software. Already today, there are tasks that can be best performed by cameras, for example to inspect granule or powder that passes by on a conveyor.

There are still many challenges to overcome before cameras can be widely used as sensors not only for slow inspection, but as part of fast feedback loops for control applications. One key problem is the performance of machine vision algorithms. These algorithms need to precisely extract information such as position and orientation of an object from a video feed in real time, what is nowadays often done with between 10 and 60 frames per second [9]. Another challenge is the E2E delay. If a control loop includes a video transmission chain with low E2E delay, the dead time of the chain is low, enabling better control compared to a transmission chain with larger delay. State-of-the-art video transmission systems achieve a delay of 70ms to 100ms. In contrast to this, the E2E delay of applications of the envisioned 'tactile internet' should be at most 1ms [7]. To that end, we investigate options to achieve low latency for real-time video transmissions.

1.1. Related Work

There are a couple of fields that relate to this paper: delay analyses of video transmission systems, publications on keyframe selection and buffer analyses of video transmission applications.

Baldi et al. [3] analyze the E2E delay of video transmission, but they do not include the delay induced by frame refresh and they unify the delay from the encoder and the frame grabber without quantizing it. A frame grabber is used to connect a camera with a non-standard interface to a computer vision system. Furthermore, they do not analyze any delays after the decoding unit, but unify all delays as processing delay. Vinel et al. [23] present a coarse delay analysis as part of their overtaking assistance system. In their analysis, they consider 5 delay contributors: encoder, transmitter buffer, channel, receiver buffer and decoder. They do not analyze the camera refresh delay, camera processing delay nor any delay related to the display. Song et al. [21] offer a more detailed analysis for statistically uniform intra-block refresh. They conclude that the worst case (maximum) E2E delay is caused by the maximum size frame and consequently suggest an intra refresh coding scheme. Their analysis does not include any delay from the recording camera or processing delay of the display. Furthermore, they don't offer a statistical analysis of the delays. Schreier et al. [19, 20] analyze the E2E latency of different H.264 coding modes. In their analysis, the largest delay contributors are coders and the transmitter

and receiver buffers. They also do not include the camera and display delays in their analysis.

Keyframe selection or frame skipping describe the discipline of not transmitting, storing or showing every video frame produced by a camera. It is used to reduce the bitrate of the video. Liu et al. [12] drop insignificant frames of often static lecture videos, whilst not having low delay constraints. Doulamis et al. [6] utilize a neural network to predict key frames in real time to cope with low and variable bandwidth connections. There are other less relevant approaches [22, 11] that find positions for independently coded frames (Inter Frames) by optimizing rate and distortion over an entire video, which is not applicable to our low delay use case.

There are many papers giving detailed analyses about transmission buffers. Navakitkanok et al. [15] analyze the frame dropping effect of a delay constraint on the transmission buffer. Consequently, they suggest a rate control method to allocate the number of bits per encoded frame. This improves the buffer behavior and reduces the number of dropped frames. Other papers such as Ribas-Corbera et al. [18] and Zang et al. [25] do an extensive analysis of the transmission buffer and suggest rate control again via bit allocation.

Our contributions with respect to the presented papers are a more detailed analysis of all the delay components, in particular the camera and display delay. In the course of this analysis, we offer a delay distribution, not only maximum or mean values. Furthermore we propose an intelligent frame skipping method to transmit high-priority frames and a preemption mechanism to guarantee the fastest possible transmission of these frames. Finally, we are to our best knowledge the first to analyze differing frame rates for the camera, encoder, network and display in the video transmission chain.

2. Modeling and Analyzing Delay in Video Transmission

This section analyzes the blocks of a video transmission chain in terms of their contributions to E2E delay to point out where potential for reducing delay in video transmission lies.

2.1. Cut-Through Operation

Some of the blocks can perform cut-through (CT) processing. CT operation means that the sending of a packet, in our case a frame, starts before that frame is completely received and/or processed. For our investigations, the atomic unit are the bits representing the raw or encoded frame. CT-like operation is common practice in camera and display electronics, see figure 1. The received first bytes of a frame are directly processed and forwarded to the next element in the chain. Such a tight CT operation is possible because of the reliable connection speeds.

The alternative to CT is store-and-forward, the frame is first completely received and stored in memory. After the processing of the frame data is finished, sending to the next block in the chain is initiated. Examples for this operation mode can be found in software en- and decoders.

2.2. Block Model

We present an analysis of the fundamental point-to-point video transmission chain for the human observer, which is depicted in Figure 1. It comprises the entire E2E delay and is comparable to the model for a machine vision setup, as explained in subsubsection 2.2.13. We define E2E delay T as the time period between when a visible event in the field of view of the camera first takes place and the first time this event is shown on the display at the end of the chain. This E2E delay for the human observer is also called Glass-To-Glass (G2G) latency. The blocks' behaviours, their parameters, metrics and, where available, cut-through abilities are explained in the following. A summary of definitions, parameters and metrics is given in Table 1.

2.2.1 Camera: Frame Refresh

In general, the camera refresh rate is not synchronized with the time of a real-world event taking place. Therefore, the event can happen at any time while the camera sensor is exposed to light. For example, an event takes place just after the exposure has started. In this case, an entire frame period passes before the sensor is read out and the data containing the event is sent to the next block. A frame period in the camera is the inverse of the camera refresh rate. In our model, we lift frame rate constraints and different blocks can have different and varying frame rates, as shown in Table 1. Cameras have limited frame rates because of the readout time of the image sensor and the limited bandwidth of the interface to the next processing block. For example, a camera with 60Hz refresh rate has a frame period of approximately 16.7ms. The other extreme is if an event occurs just before the end of a frame period. In this case, almost no additional delay is caused by the camera refresh. Overall, the delay t_{CFR} from the camera refresh can be modeled as distributed uniformly between zero and the camera frame period because the event can take place at any time during exposure, unrelated to the end/beginning of exposure. This means that the higher the frame rate which a camera uses, the smaller the worst case delay. A 500Hz camera for example has a worst case delay of 2ms.

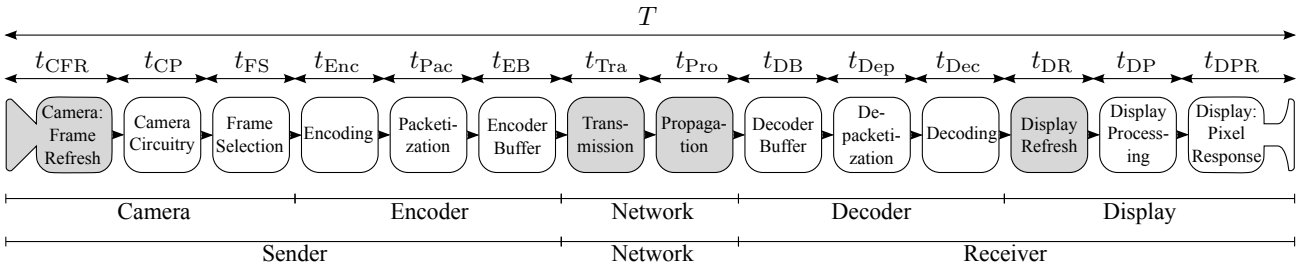

Figure 1. Video processing and transmission chain. Blocks with gray background don't have functionality, but represent a delay. White blocks embody functionality as well as delay.

Type	Variable	Relations	Comment
Definitions	Θ	$\Theta = \{$ CFR, CP, FS, Enc, Pac, EB, Tra, Pro, DB, Dep, Dec, DP, DR, DPR $\}$	Set of video delay components
	$t_i, i \in \Theta$ [s]		Delay of component i
	$p_i(t), i \in \Theta$	$t_i \sim p_i(t)$	Probability density function of the delay of component i
Parameters	$t_{\min}$ [s]	$t_{\min} = \max_{i \in \Theta}(t_i)$	Frame selection: Minimum time between two frames
	$t_{\max}$ [s]	$f_{\mathrm{Base}} = \frac{1}{t_{\max}}$	Frame selection: Maximum time $t_{\max}$ between two frames to sustain a base frame rate f_{Base}.
	f_{Cam} [Hz]	$t_{\mathrm{CFR}} \sim \mathcal{U}\left(0, \frac{1}{f_{\mathrm{Cam}}}\right)$	Camera frame rate
	f_{Enc} [Hz]	$t_{\mathrm{Enc}} = \frac{1}{f_{\mathrm{Enc}}}$, $f_{\mathrm{Base}} \leq f_{\mathrm{Enc}} \leq f_{\mathrm{Cam}}$	Encoder frame rate
	f_{Dis} [Hz]	$t_{\mathrm{DR}} \sim \mathcal{U}\left(0, \frac{1}{f_{\mathrm{Dis}}}\right)$	Display frame rate, also represents the machine vision frame rate
	c [Mbps]		Channel bandwidth
	t_{E} [s]	$t_{\mathrm{E}} \sim \mathcal{E}(\lambda_e)$	Exponentially distributed event inter-occurrence time with mean $1/\lambda_e$
	r [Mbps]	$r \sim \mathcal{P}(\lambda_r)$	Poisson distributed frame sizes with mean λ_r
Metrics	T [s]	$T = \sum_{i \in \Theta} t_i \sim \mathcal{G}(\mu, \sigma)$	E2E delay, where $\mathcal{G}(\mu, \sigma)$ is a placeholder for a general distribution of the E2E delay T with mean μ and variance σ^2.
	β		Ratio of dropped frames
	Δt		Mean display frame period

Table 1. Parameters and metrics of the video transmission delay model

2.2.2 Camera Circuitry

The camera electronics comprise two main parts. First the camera sensor, from which the pixel data is read out. The readout process and/or the defined exposure time typically limit the frame rate f_{Cam}. Second, the camera processing block applies elementary image processing such as gain, offset, white balance, analog digital conversion to the raw image. As these functions are implemented in hardware, the resulting processing delay t_{CP} is typically in the submillisecond domain. For the Allied Vision Guppy Pro cameras, these processing steps take $t_{\mathrm{CP}} = 710\mu\mathrm{s} \pm 62.5\mu\mathrm{s}$[1], for example.

The camera processing is considered to be a CT operation. Using the Allied Vision Guppy Pro again as example, we saw that these cameras adjust the transfer rate such that the transfer time t_{TraC} of a frame equals t_{CFR}.

2.2.3 Frame Selection

This block implements a preprocessing step that applies a frame selection algorithm in order to reduce E2E latency, as described in subsection 3.1. In first CPU-based prototypes, the latency t_{FS} of this block has shown to vary around 100μs up to a maximum delay of 150μs for store-and-forward operation mode on images with VGA (640x480 pixels) resolution. This number is expected to decrease significantly when applying optimized algorithms and implementing it in hardware.

When implemented in hardware, a cut-through mode is feasible. In this mode, only a part of the image is analyzed before the decision whether it shall be skipped or not is made. Investigations about how big this ratio shall be are currently done.

2.2.4 Encoding

The encoding of a video can take widely differing times t_{Enc}, depending on the implementation, hardware or software encoder, and the used standard, e.g. AVC/H.264 or HEVC/H.265. In a low delay video application, it is prohibitive to use a group of pictures structure in which inter-coded frames depend on intra-coded frames from the future. Such non-causal coding makes reordering of frames necessary and introduces several frame periods of delay. Instead, in typical low delay video applications, intra-only encoding is used, in which the frames are encoded independent of each other. Intra-only encoding has a significantly smaller compression factor than inter-coding, which exploits temporal dependencies between successive frames. Therefore, the encoder is the block in which we mostly trade off latency against compression efficiency. Intra-only hardware encoders have an encoding propagation delay of as little as 250μs[2]. This delay describes the propagation latency of a pixel through the encoder, not the time t_{Enc} it takes to encode an entire frame. The frame encoding time is equal to the inverse of the frame rate f_{Enc} at which an encoder can process frames. Only the encoder propagation delay is relevant to the E2E delay as long as the encoder is not fed a too high frame rate from the previous block. If necessary, color space conversion is applied before encoding. From the delay perspective, this is seen as part of encoding.

From the cut-through perspective, we distinguish software and hardware encoders. State-of-the-art software encoders are unable to perform cut-through processing. They write the completely encoded frame to the memory when they have finished encoding. For these, only the frame encoding time t_{Enc} is relevant to E2E delay. In the remainder of this paper, we solely discuss software encoders.

2.2.5 Packetization, Depacketization

The encoded bits have to be packetized and depacketized. For the Maximum transmission unit with a size of 1500 bytes, this takes a time t_{Pac} and t_{Dep} of 12.5μs for UDP or TCP connections on Gigabit Ethernet.

2.2.6 Encoder buffer, decoder buffer

Buffer behavior has been extensively studied [15, 18, 25], especially in video rate control papers. A highly filled buffer introduces a large delay t_{EB} or t_{DB}. We do not give an explicit description of the buffer delay here, as it will be investigated in Sections 4 and 5. Using rate control, the bitrate of the video is fitted as close as possible to the available bitrate of the channel. Assuming a well-working channel rate adaption, we keep the buffer load as small as possible to minimize the resulting delay. Buffer size is limited to give an upper boundary for the delay induced by buffering.

2.2.7 Transmission

Depending on the available data rate of a channel or interface connection, a packet with defined size needs time to be transmitted. For example a 1500 byte packet on a 1Mbit/s channel needs a transmission time $t_{\mathrm{Tra}} = 12$ms. Therefore, the transmission delay of a frame over the network channel depends on the channel rate and the size of an encoded frame. The transmission delay t_{TraC} of a frame for example from the camera to the frame selector is the size of the raw frame divided by the camera interface speed. The last transmission time we consider here is the transmission time from the decoder to the display t_{TraD}.

[1] From page 136 of the Allied Vision Technologies Guppy Pro Technical Manual V4.1.0

[2] SOC Technologies MPEG-4 H.264/AVC Video Encoder

2.2.8 Propagation

Signals on a cable require a nonzero amount of time t_{Pro} to propagate. We give the propagation speed of electrical signals relative to the speed of light c$\approx 3 \cdot 10^8 \frac{\mathrm{m}}{\mathrm{s}}$ in vacuum. Electrical signals in copper travel between 0.66c (RG-58/U coaxial cable, ca. $1.98 \cdot 10^8 \frac{\mathrm{m}}{\mathrm{s}}$) and 0.95c (Open wire, ca. $2.85 \cdot 10^8 \frac{\mathrm{m}}{\mathrm{s}}$)[10]. Assuming a 100m long copper cable for signal propagation, resulting delays range from 0.35μs to 0.51μs and are insignificant for such an application. But for longer connections, this quickly becomes a relevant factor. For example from 300km of connection distance, the propagation delay, lower bounded by the speed of light, is at least 1ms. We apply the speed of light delay as lower propagation delay bound.When we assume a more complex network over which the signal propagates, the propagation delay t_{Prop} can become highly variable due to blocking and re-routing.

2.2.9 Decoding

Decoding is analog to encoding and because of the similar structure in low-complexity coders with a delay t_{Dec} close to the encoder delay. If necessary, a color space conversion is applied on the raw frame. After that, the image resides in the graphics buffer, waiting for the next display refresh.

2.2.10 Display Refresh

Analog to the camera refresh, the display panel is refreshed at a fixed rate f_{Dis} in classical panels. This means that the image data is read out of the graphics buffer and transferred to the display electronics in constant time intervals f_{Dis}^{-1}. The limited rate is caused by the limited bandwidth of the data transmission interface of the display and the time the panel requires to draw one frame. In traditional displays, refresh time is not related to when the decoder finishes decoding a frame. In the best case, the decoder finishes a frame just before the next display refresh, causing almost no additional delay t_{DR} . In the worst case, the decoder puts out a frame immediately after a display refresh, in which case the most recent frame is not drawn on the display until the next refresh, which takes places after almost one display frame period. The frame period is the inverse of the display refresh rate f_{Dis}. New panel types with dynamic refresh using synchronization techniques such as NVidia G-Sync™or AMD FreeSync™have reached the market recently. These panels can synchronize their display refresh to the refresh of the graphics buffer. This reduces t_{DR} drastically at low video frame rates f_{Base} instead compared to using a fixed refresh display with $f_{\mathrm{Dis}} = f_{\mathrm{Base}}$. But these displays also have a minimal period of time they need to draw one frame, causing a maximum refresh rate. If the video frame rate f_{Base} equals this rate, there is no more gain using the presented display refresh synchronization techniques.

2.2.11 Display Processing

The time t_{DP} describes the delay between when a new pixel value is sent to the display and when the display electronics are ready to change the corresponding pixel with the next panel refresh. This varies widely for different displays: we measured values from less than 1ms on a Samsung 2233BW monitor up to 23ms in a DELL U2412M [reference own delay measurement paper or no reference].

2.2.12 Display: Pixel Response

Pixels in LCD displays do not respond instantaneously to a changed voltage. In modern displays such as the Acer XB270H, the grey to grey response time t_{DPR} goes down to 1ms.

2.2.13 Remarks

All these delays are not perfectly constant due to real-world imperfections such as temperature differences influencing the processing speed of a circuit. Nevertheless, significant delay variance is expected only from the frame refresh in the camera, the encoder and decoder buffer, the transmission, the propagation and the display refresh. In a video transmission setup with low latency components, the display and camera contribute large delays, as found in [2] [update reference if accepted to ICIP!]. The delay from these two components accounts for over half of the average E2E delay. Therefore, the delays from camera and display offer great potential to reduce the E2E latency.

A video transmission system with a machine vision algorithm as sink mostly uses the same blocks. The only difference is after the decoder, where the display is replaced with the machine vision unit, so the remaining delay is the machine vision refresh time. Machine vision algorithms also work at a maximum frame rate which can be much higher than the frame rate of a display, in particular for tracking applications. For low-complexity applications, the 1000Hz barrier has been reached many years ago for example by Nakabo et al. [14] and Watanabe et al. [24]. Such high frame rates of the vision chips result in small worst case refresh delays. Looking beyond fixed rate machine vision, there are first advances towards event-based dynamic vision sensors [5] and algorithms which do computations only if requested[16], eliminating the refresh delay for applications in which events do not take place in quicker succession than the algorithms need to process one frame.

Regarding the potential gains of CT operation: In the sender (Figure 1), at most one camera frame transmission period t_{TraC} can be saved by using CT operation. This is

because in store-and-forward, the frame has to be received only once, and then the blocks can subsequently apply their operations on the frame in the shared memory. In the receiver, one channel frame transmission period t_{Tra} can at most be saved. Again, the gain is because in the store-and-forward mode, the receiver has to wait until the frame has fully arrived and can then quickly operate on it. The CT operation in the receiver causes very little tolerance [quantify?] towards delay jitter from the blocks delivering data to sender and receiver. While this is not an issue for the sender, which is fed frames from the camera, this has to be carefully investigated for the receiver. [Of course, with CT, blocks can process images in parallel, while with store-and-forward, we have to do it sequentially. So if the absolute processing time of the blocks get significant (which it is not currently), we have to take that into account, too.]

3. Low Delay Video Transmission Mechanisms

In the following, we present two novel ways to reduce the E2E delay in a video transmission chain. The first is a frame skipping method that discards video frames without significant information, and the second is a buffer emptying to shorten the waiting time of frames containing significant information, so called event frames. Frames without significant information are called base frames.

3.1. Frame Skipping

As previously stated, the delay from the camera offers a lot of potential to reduce the E2E delay. Therefore, we reduce the delay t_{CFR} by using a high frame rate camera. The central contribution is that we enable the use of a high frame rate camera without needing to change other parameters of the system, such as the processing speed of the encoder or the channel rate of the network. If the frame skipping method was not used, the increased output information rate from the camera would overload for example the network channel, yielding an impracticable system. To control the output information rate such that no block is overburdened, we propose to perform a selection of frames immediately after the capturing process at the camera. The decision whether a frame should be skipped or used for further processing is based on the following three criteria:

Content: the higher the amount of new information in a frame compared to the last non-skipped frame, the more likely a frame is to be chosen for processing. New information can in this case be quantized as the **m**ean **a**bsolute **d**ifference (MAD) or the **s**tructural **sim**ilarity index (SSIM). Other frame selection metrics for change detection can also be used. Alternatively, if the frame skipping unit is placed after the encoder, the encoded frame could be analyzed. But for now we only assume the frame skipping that selects raw

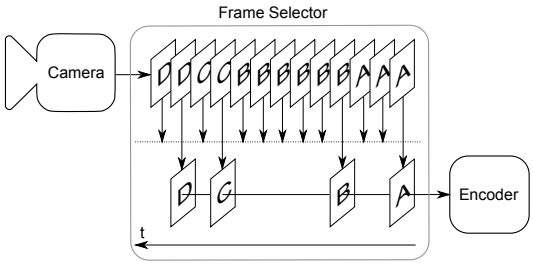

Figure 2. Schematic of the frame selection process.

frames. This way we ensure that a new event which changes the image content significantly has minimal delay. This process is shown in Figure 2. Frames with similar content have the same letter A, B, C or D. From the three similar frames with the letter A, only the first one needs to be transmitted, the subsequent two are not given to the encoder. The next frame transmitted to the encoder is the first frame marked with B, which must have a significant difference to the previous frames, for example in terms of MAD. The choice of skipping or transmitting a frame can for example be done by thresholding the MAD value. It can clearly be seen that the rate of frames coming out of the frame selector is lower than the incoming rate from the camera. The magnitude of the difference depends on the image content and the threshold selection. Please note that in Figure 2, there are multiple frames in the frame selector for illustration purposes only. A real implementation does not store the frames, but decides for every frame as fast as possible whether it should be omitted or not.

Subjective criteria: A strongly varying frame rate, especially in low frame rate domain, decreases the subjective quality of a video. Therefore, we select the frames such that the subjective rating of the resulting video stream is as high as possible while satisfying other constraints, such as the delay and the channel rate. This applies both to human observers and machine vision algorithms as tracking requires regular updates to perform content detection.

Bottleneck component: Every block of the chain has a constant or varying throughput rate in frames per second, caused by the frame transmission time of the respective block. The rate of selected frames may not exceed the rate of the slowest block. Otherwise, frames will have to be dropped or queued by these blocks.

Joining all three criteria into frame selector instructions yields the decision rules presented in Algorithm 1. ΔI is the content difference between the last transmitted frame and the current frame for which the selector shall meet a decision. It can be quantified as MAD, SSIM or other metrics. Δt_{prev} is the time since the last frame was transmitted. If ΔI exceeds a threshold thr and Δt_{prev} is greater than $t_{\min}$,

the frame is skipped in order not to create a frame rate that is too high for the bottleneck block. Otherwise, the frame is transmitted. If ΔI is smaller than thr, the frame is skipped, except for the case that Δt_{prev} is greater than $t_{\max}$. In this case, the frame is transmitted to guarantee a minimal frame rate.

This is a preliminary version of the algorithm which will be applied to the first prototypes. Experience from future experiments might enable a more sophisticated selection process, possibly with an adaptive $t_{\min}$, depending on the state of the network. Additionally, $t_{\max}$ could be adjusted depending on the requirements of the receiving machine vision algorithm. Finally, we are able to adjust thr based on e.g. video content, lighting conditions and camera model.

Another alternative approach to skipping frames would be a drastic drop in quality of the previously encoded frames. If the encoder is still busy encoding that frame, we can order it to allocate no more bits for the following image regions. This is not further investigated in this paper.

Algorithm 1 Frame Selection

```
1:  if ΔI > thr then                                  ▷ Content
2:      if Δt_prev < t_min then                       ▷ Bottleneck
3:          skip frame
4:      else
5:          transmit event frame
6:      end if
7:  else
8:      if Δt_prev > t_max then                       ▷ Subjective criteria
9:          transmit base frame
10:     else
11:         skip frame
12:     end if
13: end if
```

3.2. Preemption

Preemption reduces the E2E delay when an event frame is ready to be transmitted after being encoded, but an old base frame is still being transmitted and therefore occupying the channel. In that case, the new event frame would have to reside in the buffer until the old base frame is completely transmitted. Instead, we flush the buffer from base frames and directly start transmitting the event frame. This is feasible because the base frame does not contain significant information, in contrast to the arriving event frame. Such a scenario is depicted in the message sequence chart in Figure 3a, where frames are being sent from the camera to the unit consisting of frame selector and encoder, from where they are forwarded to the buffer. Frames are represented as blue lines. The light blue area between two frames is the image data that is transferred. Here only the

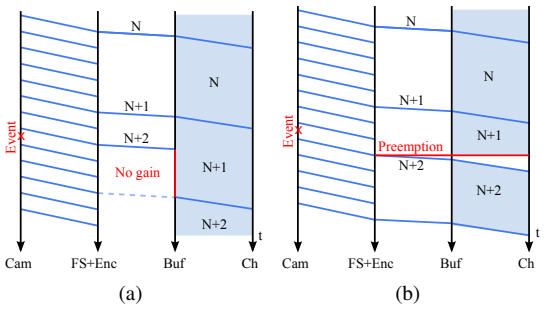

Figure 3. Frame transmission without preemption (a) and with preemption (b)

frame transfer time on the network channel is relevant. In this case, the channel is fully used by base frames, therefore the transmission time t_{Tra} of a frame on the channel takes exactly as long as one maximum frame period $t_{\max}$.

Assume that $t_{\max}$ is chosen such that every 5th base frame is selected, as illustrated in 3a. The frame selector chooses the 2nd frame from the camera after frame N+1 as event frame. The chosen event frame, frame N+2, will not be transmitted earlier than the 5th frame after base frame N+1 would have been transmitted (dashed line). This is because the channel is fully used. Therefore, using a camera and frame selector on a fully loaded channel would not bring any improvement over a classical, five times slower camera without frame selector. To get such an improvement, we need to take further measures when receiving a frame with an event: the previous, old base frame that is still blocking encoding, buffer, channel and maybe even decoding, has to be preempted as shown in Figure 3b. This holds equivalently for the general case, in which the channel is not fully used, but a new event frame arrives in the encoder buffer, while an older base frame is still being transmitted. The frame selector may still not choose events too often. If this is done, an event quickly following another will preempt the earlier event. Therefore, in a burst of events, only the last event would not be preempted, leading to a delayed transmission of the burst of events. This undesirable behavior can be avoided by properly setting the time $t_{\min}$ such that after an event occurred, a short timeout is executed. During this timeout, the frame containing the event can be safely transmitted. The preemption units have a protection to not preempt event frames in case of an incorrectly chosen $t_{\min}$, but to drop the newly arriving ones. This is plausible because it is more important to transmit the initial event of a burst of events rather than a later one.

4. Analysis

We give a short probabilistic analysis to infer an expectation for the E2E delay distribution. We start by modelling the delays of the blocks from subsection 2.2. Both the camera refresh delay t_{CFR} and the display refresh latency t_{DR} are assumed to be uniformly distributed as defined in Table 1.

The remaining relevant delays are the buffering delays t_{EB} and t_{DB} and the transmission delay t_{Tra}. These delays depend on the frame size m, which is determined by the rate control algorithm and the channel rate c. The transmission delay can be computed as $t_{\mathrm{Tra}} = \frac{m}{c}$. Buffer delays of memoryless processes can be calculated with results from queueing theory. In our case, the rate control adjusts the size of frames by changing the encoder quantization parameter depending on the load of the encoder buffer. The arrival process is therefore heavily dependent on the buffer state. The waiting times in such queues can be modeled [13], but that is very complex and not the main topic of this work. We model the buffering delay in a simplified manner in this section and provide a detailed analysis based on a simulation in section 5. We observed an exponential distribution

$$t_{\mathrm{EB}} \sim \mathcal{E}(\lambda_{\mathrm{EB}}) \text{ and } t_{\mathrm{DB}} \sim \mathcal{E}(\lambda_{\mathrm{DB}}) \tag{1}$$

of the buffering delays with $\lambda_{\mathrm{EB}} = 0.5$ and with $\lambda_{\mathrm{DB}} = 10$ with the default parameters in our simulation experiments. The default parameters are the following: the camera frame rate is $f_{\mathrm{Cam}} = 480$Hz, the maximum time $t_{\max} = 0.02$s, the channel rate $c = 20$Mbps and the display refresh rate is $f_{\mathrm{Dis}} = 120$Hz. The remaining components contribute a delay of $t_{\mathrm{rem}} = 2.4$ms, comprising the camera processing, the en- and decoding, packetization, propagation $t_{\mathrm{Pro}} = 0.25\mu$s and display processing ($t_{\mathrm{Dis}} = 1$ms). For parameters without description, we took the values as presented in subsection 2.2. These are also the default parameters used in the evaluation section 5.

The transmission delay t_{Tra} is modelled via a limited mirrored poisson distribution

$$t_{\mathrm{Tra}} \sim c e^{-\lambda_{\mathrm{Tra}}} \sum_{k=0}^{t_{\mathrm{Tra,max}}} \frac{\lambda_{\mathrm{Tra}}^k}{k!} \delta(t + k - t_{\mathrm{Tra,max}}), \tag{2}$$

where c is a normalization factor. The choice of the distribution and the parameters $\lambda_{\mathrm{Tra}} = 1$ and $t_{\mathrm{Tra,max}} = 6$ms are again based on the simulation experiments. The sum iterator k traverses integer milliseconds. All remaining block delays are assumed constant and are summarized in t_{rem} with the probability density function (PDF) $p_{\mathrm{rem}}(t)$.

The sum of these block delays makes the E2E delay T. To retrieve the distribution, we convolve the PDF of the delays. For this to be feasible, the delays have to be mutually independent. This is not given for t_{EB}, t_{DB} and t_{Tra}, but we presume that the effect of this violation will be insignificant. So the PDF of the E2E delay is

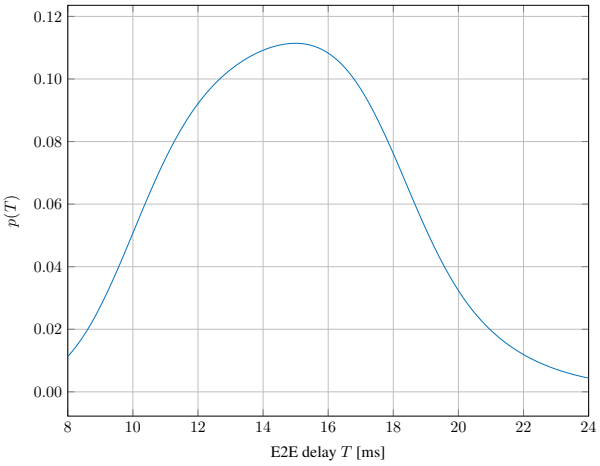

Figure 4. Delay distribution according to the delay model

$$T \sim p(t) = (p_{\mathrm{CFR}} * p_{\mathrm{EB}} * p_{\mathrm{Tra}} * p_{\mathrm{DB}} * p_{\mathrm{DR}} * p_{\mathrm{Rem}})(t), \tag{3}$$

depicted in Figure 4 for the given parameters. In subsubsection 5.2.3, we compare this model to the simulation results.

5. Evaluation

In order to reveal the potential of the proposed improvements, we conduct extensive simulations of a video transmission chain as shown in Figure 1. On this system, the mechanisms described in section 3 are evaluated. The evaluation includes the investigation of the system behavior with respect to the parameters listed in Table 1, as well as a comparison to a system without frame selection.

The simulation is implemented using Omnet++ 5.0b2 and is constructed of blocks including a source transmitting video frames, a background traffic source, an encoder, a sending buffer, a channel, a receiving buffer, a decoder and a display. Those blocks comprise customizable parameters, enabling us to analyze various scenarios. All experiments are performed with a video of length $t_{\mathrm{Rec}} = 10.4$ s with $N_{\mathrm{Event}} = 53$ events, a resolution of 640x480 and sampled with different rates ranging from $f_{\mathrm{Cam}} = 60$ fps ($\mu_{\mathrm{r}} \approx 8.59$ Mbps bitrate of the encoded video) to $f_{\mathrm{Cam}} = 480$ fps ($\mu_{\mathrm{r}} \approx 71.9$ Mbps). We use default parameters as described in section 4. If not otherwise stated, these parameter assignments are in place. A frame is sent from the source (camera) to the target (display). Furthermore, no background traffic is

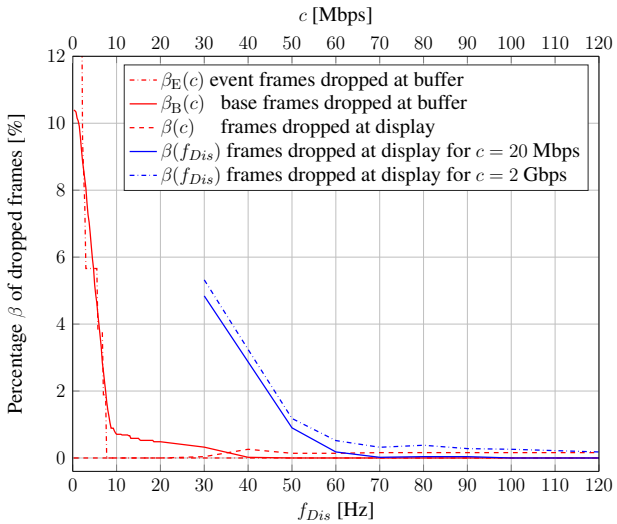

Figure 5. Percentages β of dropped frames for varying c or f_{Dis}.

applied, as the application shows analogous effects as lower absolute channel rates as on comparably crowded channels. Frames are encoded using FFMpeg's H.264 Encoder [4] and segmented according to IEEE 802.3 Ethernet.

5.1 System Performance

In this section the proposed algorithms from section 3 for frame skipping and preemption are evaluated when applied in a transmission system as shown in Figure 1. This evaluation is performed in terms of frames lost during transmission, the average E2E delay and the display frame periods at the display.

5.1.1 Dropped frames

Event and base frames can be dropped at the buffer using preemption as presented in subsection 3.2 or at the display if multiple frames arrive within one refresh cycle.

Therefore simulations are performed to analyze the drop behavior under realistic parametrization, which is a sufficiently small $t_{\min}$ and the default $t_{\max} = 0.02$s adjusted to human perception. The outcome is shown in Figure 5.
The percentage of frames lost in preemption depends on the generation rate of the source and the channel rate c. However, the influence of the generation rate is not existent if the channel provides a realistic high bandwidth and thus, is omitted here. For low c preemption leads to many drops. With $c = 2.25$ Mbps, $\beta_{\mathrm{B}} = 9$ % of all base frames are lost and thus, generated unnecessarily by the source. Also $\beta_{\mathrm{E}} = 9$ % of all event frames are preempted. For lower channel rates, a higher percentage β of all frames is preempted. This is because at low rates frames are kept longer in the buffer and thus, are more likely to collide with event frames. Nevertheless, this effect mitigates with increasing bandwidth being close to zero with channels faster than 40 Mbps.

At the same time, an increase of c leads to frames arriving more frequently at the display, resulting in more display drops. Therefore, we depict the display drops as a function of the display refresh rate f_{Dis} in Figure 5. It can be seen that with ascending display refresh rates, the drop ratio decreases. With a standard $f_{\mathrm{Dis}} = 60\mathrm{Hz} > t_{\max}^{-1} = 50\mathrm{Hz}$ display, the display drops for both channels are around 1%, so we recommend a display with at least this refresh rate for this parameterization.

5.1.2 E2E delay

We evaluate the influence of the display rate f_{Dis}, the channel rate c and the camera rate f_{Cam} on the average E2E delay μ of frames. We obtain the results shown in Figure 6.

The delay t_{DR} contributed by the display refresh on average accounts for $\frac{1}{2} \cdot f_{\mathrm{Dis}}^{-1}$ delay. Thus, small display rates account for significant delay, while for higher rates the delay contribution converges to zero. Additionally, a slow channel results in additional delays at the buffer. This leads to a decreasing E2E delay for increasing channel rates. Moreover, for slow channels event frames have lower latencies than base frames as they are higher prioritized. However, for fast channels, preemption nearly never applies leading to similar E2E delays for base and event frames. Increasing f_{Cam} results in a decrease of the average delay of $\frac{1}{2} \cdot f_{\mathrm{Cam}}^{-1}$ contributed by t_{CFR}, which converges to zero with growing camera rates.

5.1.3 Display frame periods

The ideal interarrival time of frames at the display is the inverse display refresh rate f_{Dis}^{-1}. The interarrival time depends mainly on the source generation rate controlled by $t_{\max}$, the display and camera rate, as well as on the channel rate. The corresponding simulation outcome is depicted in Figure 7.

For small values of $t_{\max}$ (i.e. $t_{\max} < f_{\mathrm{Dis}}^{-1} \approx 0.0083$ s) and a sufficient channel rate, base frames arrive faster than the display refreshes and thus, lead to display frame periods Δt of f_{Dis}^{-1}. However, increasing this parameter reduces the sending rate of base frames which would lead in an eventless case to a linearly growing slope. But as in this case events are present, event frames are sent at the time they occur. Consequently, with growing $t_{\max}$ the curve converges to the inverse average inter-occurrence time of events $\lambda_e = \frac{t_{\mathrm{Rec}}}{N_{\mathrm{event}}} \approx 196.2\ ms$. The curve for increasing display rates converges to $t_{\max}$ for high display refresh rates f_{Dis}. Effects of the camera rate are also masked by $t_{\max}$,

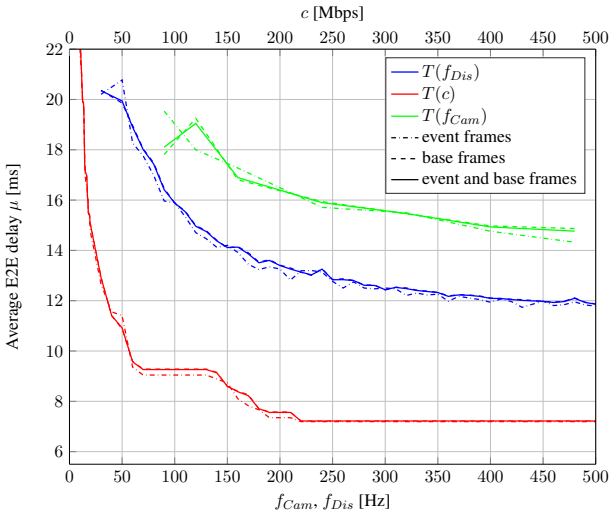

Figure 6. Average E2E delay μ for varying f_{Dis}, c or f_{Cam}.

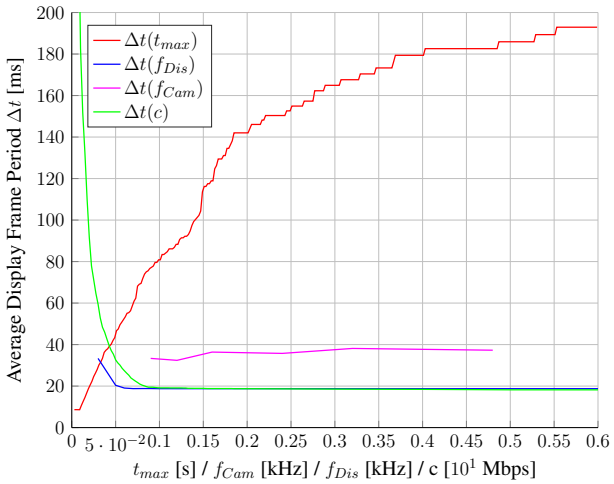

Figure 7. Mean frame periods Δt at display for varying t_{max}, f_{Dis}, f_{Cam} or c.

which leads to an unchanged frame frequency produced by the source. For slow channel rates buffer delays dominate, getting insignificant for high channel rates. Because of this, the curve again converges to t_{max} as $t_{\mathrm{max}} > f_{\mathrm{Dis}}^{-1}$ and $t_{\mathrm{max}} \gg f_{\mathrm{Cam}}^{-1}$. To summarize, the channel has to support the display rate provided by the source, and t_{max} should be bigger than f_{Dis}^{-1}. The camera frame period f_{Cam}^{-1} should be smaller than t_{max}. Both result in the recommendation $f_{\mathrm{Dis}}^{-1} < t_{\mathrm{max}} < f_{\mathrm{Cam}}$.

5.1.4 Channel rate and t_{min}

In order to use the maximum channel bandwidth c while not overloading it, t_{min} can be adjusted to slow down the sending process and to lower the channel load. Thus, in this experiment the maximum $t_{\mathrm{min}} = t_{\mathrm{min}}^{\mathrm{max}}$ was measured that just results in fixed average channel loads as shown in Figure 8 for different channel rates c. To demonstrate the effect of t_{min} it is assumed that every camera frame contains an event. With increasing t_{min} the generation rate and channel load decreases. This can be seen in Figure 8, where less channel load results from higher t_{min} throughout all channel rates. Here this holds for channels up to 6 Mbps. For faster channels t_{min} reduces the channel load, too. Then the channel is sending faster than the maximum frame generation rate which is mainly constrained by t_{max}. The generation rate is constant while the channel rate increases, leading to a mitigating effect of t_{min} on the percental channel load.

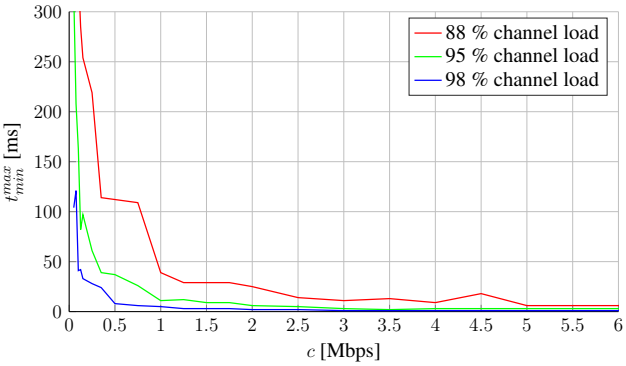

Figure 8. Maximum $t_{\mathrm{min}} = t_{\mathrm{min}}^{\mathrm{max}}$ that satisfies the indicated channel load depending on the channel rate c.

5.2 Comparison to Standard Source

Next, the simulation is executed without the selection and preemption algorithms and the results are compared

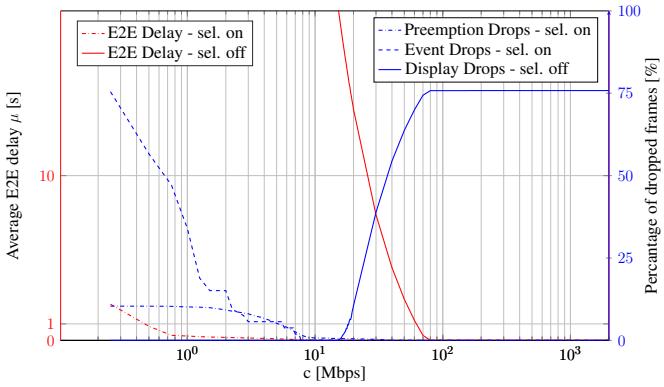

Figure 9. Mean E2E delay μ and percentage of dropped frames for various c. Display drops for applied selection are omitted as values are close to zero.

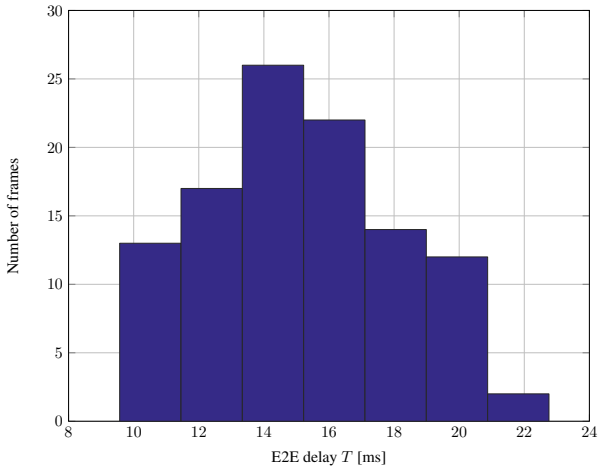

Figure 10. E2E delay distribution of 555 frames

to the outcome with selection applied. For this purpose dropped frames and the E2E delay are investigated. Those parameters are influenced by the channel rate, the camera rate and the display rate. However, as described in subsection 5.1, for realistic scenarios the channel rate is the crucial factor that influences both the system with and without selection applied.

5.2.1 Conservative approach

In this approach all frames that are sampled by the camera are sent. Thus, in Figure 9 it can be seen that for slow channel rates this leads to decreasing E2E delays. This is because of long waiting times at the buffer resulting from a high source generation rate of $f_{\mathrm{Cam}} = 480$ Hz. Also, slow channels give no drops at the display as sending takes longer than the display refresh cycle. In the conservative approach without rate control, the video encoder produces a bitrate of $\mu_{\mathrm{r}} = 71.9$ Mbps. We did not apply rate control to enable a direct comparability of the approaches and because rate control would reduce video quality. Image quality shall remain unchanged, again to be able to compare the approaches. As illustrated with the red graph in Figure 9, the setup is not usable for channels with rates smaller then the encoded video bitrate, it yields average delays of $\mu = 1$ s and above.

5.2.2 Comparison to state-of-the art

As described in subsection 5.1, if selection is applied, the mean E2E delay μ is drastically reduced for slow channel rates. This occurs because the encoder produces a bitrate of just $\mu_{\mathrm{r}} = 8$ Mbps, which is approximately 11% of the previous, full video rate of 71.9 Mbps.

In contrast to the case without selection the sending rate of the source can be adjusted by parametrization of the algorithm parameters. This allows to minimize the number of sent frames without significant information. Also, the E2E delay is already minimal for slow channels of about 1 Mbps if selection is applied, while this convergence is reached at about $c = 80$ Mbps in the conservative case. Lastly, for higher channel rates the number of frames that are sent, but not used at the display, is higher for the conservative approach with about 75.8 % of frames lost. In contrast to that this effect does not occur with the properly parametrized proposed algorithm.

5.2.3 Comparison to Probability model

The E2E delay distribution of 555 frames arriving at the display with the default parameters is depicted in in the simulation histogram in Figure 10. We compare it to the probability model distribution in section 4. The block parameters for creating both the distribution and the histogram are the same defaults, presented in section 4. The shape of the histogram resembles the shape of the distribution. Furthermore, the mean values of the simulation $\mu_{\mathrm{Sim}} = 15.3$ms and the model $\mu_{\mathrm{Prob}} = 14.9$ms are close. The upper and lower limits of the two distributions also meet well. The E2E delay limits of the simulation histogram are $T_{\mathrm{min}} = 9.6$ms and $T_{\mathrm{max}} = 22.6$ms. The integrals $\int_{-\infty}^{T_{\mathrm{min}}} p(T)dT = 0.047$ and $\int_{T_{\mathrm{max}}}^{\infty} p(T)dT = 0.018$ over the area of the distribution which is not covered by samples of the histogram are small. Therefore, we conclude that the imprecisions in the probability model described in section 4 have minor influence on the correctness of the model and we can suggest this model to describe E2E delay in point-to-point video transmission.

6. Conclusions

In this paper, we presented a detailed analysis of the delay sources in a point-to-point video transmission system. We propose the use of two methods, frame skipping and preemption to reduce E2E video delay to make video a low delay information source in coming tactile systems. We analyzed the system and proved the effectiveness of the concept using an event-based simulation. For an unchanged channel, it achieves a significant latency reduction. Alternatively, for constant E2E delay, clear bitrate savings can be obtained.

We are currently implementing prototypes of the principle to evaluate how the frame skipping decision is done in an optimal way. In the course of this, we will investigate the subjective and objective effects of the frame skipping on the video. We are also going to investigate a prototype of the preemption method.

References

[1] N. Alt, C. Claus, and W. Stechele. Hardware/software architecture of an algorithm for vision-based real-time vehicle detection in dark environments. In *Proceedings of the Conference on Design, Automation and Test in Europe*, pages 176–181. ACM, 2008.

[2] C. Bachhuber and E. Steinbach. A system for precise end-to-end delay measurements in video communication. *arXiv preprint arXiv:1510.01134*, 2015.

[3] M. Baldi and Y. Ofek. End-to-end delay analysis of videoconferencing over packet-switched networks. *IEEE/ACM Transactions on Networking*, 8(4):479–492, 2000.

[4] F. Bellard, M. Niedermayer, et al. Ffmpeg. *Availabel from: http://ffmpeg. org*, 2012.

[5] T. Delbrück, B. Linares-Barranco, E. Culurciello, and C. Posch. Activity-driven, event-based vision sensors. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 2426–2429, 2010.

[6] A. Doulamis and G. Tziritas. Content-based video adaptation in low/variable bandwidth communication networks using adaptable neural network structures. In *International Joint Conference on Neural Networks*, pages 4037–4044. IEEE, 2006.

[7] G. P. Fettweis. The tactile internet: Applications and challenges. *IEEE Vehicular Technology Magazine*, 9(1):64–70, 2014.

[8] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361. IEEE, 2012.

[9] A. Handa, R. A. Newcombe, A. Angeli, and A. J. Davison. Real-time camera tracking: When is high frame-rate best? In *Computer Vision–ECCV 2012*, pages 222–235. Springer, 2012.

[10] K. L. Kaiser. *Transmission lines, matching, and crosstalk*. CRC Press, 2005.

[11] H. Lee and S. Kim. Rate-constrained key frame selection using iteration. In *International Conference on Image Processing*, volume 1, pages I–928. IEEE, 2002.

[12] T. Liu and C. Choudary. Real-time content analysis and adaptive transmission of lecture videos for mobile applications. In *Proceedings of the 12th annual ACM international conference on Multimedia*, pages 400–403. ACM, 2004.

[13] A. Mandelbaum and G. Pats. State-dependent queues: approximations and applications. In *Stochastic Networks*, volume 71, pages 239–282, 1995.

[14] Y. Nakabo, M. Ishikawa, H. Toyoda, and S. Mizuno. 1 ms column parallel vision system and its application of high speed target tracking. In *IEEE International Conference on Robotics and Automation (ICRA)*, volume 1, pages 650–655. IEEE, 2000.

[15] P. Navakitkanok and S. Aramvith. Improved rate control for advanced video coding (avc) standard under low delay constraint. In *International Conference on Information Technology: Coding and Computing*, volume 2, pages 664–668. IEEE, 2004.

[16] Z. Ni, C. Pacoret, R. Benosman, S. Ieng, and S. Régnier. Asynchronous event-based high speed vision for microparticle tracking. *Journal of Microscopy*, 245(3):236–244, 2012.

[17] K. Okumura, H. Oku, and M. Ishikawa. High-speed gaze controller for millisecond-order pan/tilt camera. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6186–6191. IEEE, 2011.

[18] J. Ribas-Corbera and S. Lei. Rate control in DCT video coding for low-delay communications. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(1):172–185, 1999.

[19] R. M. Schreier, G. Krishnamurthy, A. Rothermel, et al. Architecture analysis for low-delay video coding. In *IEEE International Conference on Multimedia and Expo*, pages 2053–2056. IEEE, 2006.

[20] R. M. Schreier and A. Rothermel. A latency analysis on h. 264 video transmission systems. In *International Conference on Consumer Electronics*, pages 1–2. IEEE, 2008.

[21] R. Song, Y.-l. Wang, Y. Han, and Y.-s. Li. Statistically uniform intra-block refresh algorithm for very low delay video communication. *Journal of Zhejiang University SCIENCE C*, 14(5):374–382, 2013.

[22] P. Usach-Molina, J. Sastre, V. Naranjo, L. Vergara, and J. M. L. Muñoz. Content-based dynamic threshold method for real-time keyframe selecting. *IEEE Transactions on Circuits and Systems for Video Technology*, 20(7):982–993, 2010.

[23] A. Vinel, E. Belyaev, K. Egiazarian, and Y. Koucheryavy. An overtaking assistance system based on joint beaconing and real-time video transmission. *IEEE Transactions on Vehicular Technology*, 61(5):2319–2329, 2012.

[24] Y. Watanabe, T. Komuro, and M. Ishikawa. 955-fps real-time shape measurement of a moving/deforming object using high-speed vision for numerous-point analysis. In *IEEE International Conference on Robotics and Automation*, pages 3192–3197. IEEE, 2007.

[25] F. Zhang and E. Steinbach. Improved ρ-domain rate control with accurate header size estimation. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 813–816. IEEE, 2011.