MDPI

Article

Error Resilient Coding Techniques for Video Delivery over Vehicular Networks

Pablo Piñol [1,*], Miguel Martinez-Rach [1], Pablo Garrido [1], Otoniel Lopez-Granado [1] and Manuel P. Malumbres [1]

[1] Physics and Computer Architecture Department, Miguel Hernández University, 03202 Elche, Spain; mmrach@umh.es (M.M.-R.); pgarrido@umh.es (P.G.); otoniel@umh.es (O.L.-G.); mels@umh.es (M.P.M.)

* Correspondence: pablop@umh.com; Tel.: +34-966-658-366

Academic Editor: name

Version August 21, 2018 submitted to Sensors

Abstract: Nowadays, more and more vehicles are shipped with communication capabilities, not only providing connectivity with onboard devices, but also with off-board communication infrastructures. From road safety (i.e., multimedia e-call) to infotainment (i.e., VoD services), there are a lot of applications and services that may be deployed in vehicular networks, where video streaming is the key factor. As it is well known, these kinds of networks, suffer from high interference levels and low available network resources, being a great challenge to deploy video delivery applications which provide video contents with a minimum quality. In this work, we focus on supplying error resilience capabilities to video streams in order to fight against the high packet loss rates found in vehicular networks. So, we propose the combination of source coding and channel coding techniques. The former ones are applied in the video encoding process by means of intra refresh coding modes and tile-based frame partitioning techniques. And the latter one is based on forward error correction mechanisms in order to recover as many lost packets as possible. We have carried out an extensive evaluation process to measure the error resilience capabilities of both approaches alone in both (a) a simple network packet error probabilistic model, and (b) by means of a realistic vehicular network simulation framework where packet video delivery has been simulated with a high level of details. Results show that forward error correction mechanisms are mandatory to guarantee a minimum received video quality, being highly recommendable the use of the proposed source-coding mechanisms to increase even more the final video quality.

Keywords: video streaming, vehicular networks, error resilience, error concealment, HEVC, RaptorQ

1. Introduction

As defined in Directive 2010/40/EU of the European Parliament, "Intelligent Transport Systems (ITS) are advanced applications which, without embodying intelligence as such, aim to provide innovative services relating to different modes of transport and traffic management and enable various users to be better informed and make safer, more coordinated and smarter use of transport networks". There are lots of applications envisioned for ITS. In [1], we can find a classification of ITS services in four categories: (a) active safety; (b) public service; (c) improved driving; and (d) business and entertainment. Active safety is obviously the core part of the majority of the proposals but, recently, other aspects like driving efficiency are gaining growing attention. In the business world, a lot of promising applications will arise.

One of the technologies that may have multiple beneficial uses in ITS applications is video streaming. Video streaming can be linked to applications that range from digital entertainment (such as video on demand) to road safety (such as emergency video calls), or others related with business

applications (such as contextual advertising or tourist information). Although video streaming over vehicular networks may be very useful, it comes with two big challenges. On the one hand, because of its wireless nature and because of the mobility of its nodes, vehicular networks are packet loss prone environments. On the other hand, digital video has huge requirements regarding resources, both because of the high volume of data with which it deals, and because of the necessity of computing power in order to compress video sequences. This need for compressing video data makes encoded video bit streams vulnerable to packet loss because the video encoding process makes use of redundancy to a certain level, and the loss of some pieces of data may have an adverse effect in large regions of the video sequence.

In order to provide video streaming with protection against data loss, we propose a combination of two different approaches. The first improves bit stream error resilience by acting at source coding level, i.e., adapting and adjusting video encoding in such a way that the compressed bit stream can obtain a better quality level in the presence of data losses. As this approach does not obtain optimal results, in order to complement and enhance the source coding protection, we propose the use of Forward Error Correction (FEC) techniques to minimize the effect of packet losses. The former approach will be based on the High Efficiency Video Coding (HEVC) standard [2], and, for the latter approach, we have selected RaptorQ codes technology [3]. HEVC is the most recent video coding standard developed by the Joint Collaborative Team on Video Coding (JCT-VC). To add robustness to video streams on a source coding level, we have combined three strategies: (a) we have implemented an error concealment mechanism at the decoder that alleviates the effects of missing frames or parts of a frame; (b) we have proposed and studied the division of each frame into a different number of tiles (which is a brand new feature of the HEVC video encoder) in order to stop the multiplication factor of packet loss when every frame is encoded as a whole; and (c) we have proposed several encoding modes that increase the intrinsic error resilience of encoded video bit streams. Regarding the FEC approach, fine tuning of RaptorQ technology has been performed to adjust it to better fulfill the specific video streaming requirements in vehicular networks. Parameters like the protection period, the repair symbol size, and the amount of protection have been evaluated, and the best combination of these parameters has been selected to achieve the maximum recovery features while keeping the protected bit stream values within network constraints. To check the different proposals, a number of tests have been performed with the well-known vehicular traffic simulator SUMO and a combination of network simulators (OMNeT++/MiXim/Veins) using realistic urban scenarios under different network conditions. Results show that, by selecting the appropriate configuration, video streaming successes to keep the reconstructed video quality within suitable bounds.

The rest of the paper is organized as follows. In Section 2, related work is presented. In Section 3, different source coding techniques to provide error concealment and error resilience to the video stream are proposed and evaluated. In Section 4, a channel coding technique based on FEC codes is presented and tested. In Section 5, a complete simulation framework is depicted to evaluate the global performance of the proposed techniques running in realistic network scenarios. At last, in Section 6, the general conclusions and future research directions are given.

2. Related Work

Several works can be found in the literature that address HEVC video evaluation under data loss. In [4], the authors evaluate the efficiency of HEVC and compare it with H.264/AVC [5] for streaming over best-effort networks (the Internet). They use three different video sequences encoded at five different bit rates and use the 1%, 3%, and 5% loss patterns specified in [6]. The frame copy method is used to conceal errors. They configure HEVC and H.264/AVC to obtain videos with the same quality (with no data loss) and then evaluate the perfomance of the *codecs* under loss conditions. They conclude that HEVC is more efficient than H.264/AVC (around 30% - 45%), but H.264/AVC is more resilient to data loss. In [7], the author compares HEVC with H.264/AVC in wireless environments. He uses one video sequence to evaluate packet losses that range from 0% to 40% (in steps of 10%) using

channel conditions borrowed from [8]. He uses DFRM (Decoded Frame Rate Metric), SSIM (Structural SIMilarity), and PSNR (Peak Signal-to-Noise Ratio) metrics to evaluate the transmitted video quality under wireless conditions. His main conclusion is that HEVC is more error resilient than H.264/AVC under tested loss conditions. In [9], the authors have developed a complete framework for testing HEVC under different packet loss rates, bandwidth restrictions, and network delays. This framework first generates a trace file with the bit stream information and then uses this trace file for the streaming simulations. On the receiver side, a log file is created that gathers the transmission results (including the missing packets). In the final stage, the framework decodes the complete bit stream (without any loss) and then overrides the areas corresponding to the missing packets (which are tagged in the log file). The authors use this framework to evaluate concurrent multipath transmission in multihomed mobile networks.

Video streaming in vehicular networks has also been studied by several authors. In [10], the authors enumerate some of the open problems (and proposed solutions) in video streaming in VANETs (Vehicular Ad-hoc NETworks), including link connectivity, error resilience, clustering, and multihop routing. In [11], the authors address the problem of video streaming over urban VANETs. They use the H.264/AVC *codec* and some of its error resilience features, such as Flexible Macroblock Ordering and Redundant Frames, to protect the encoded bit stream. They use GloMoSim (Global Mobile System Simulator) to conduct IEEE 802.11p standard-based vehicular network simulations. In [12], the authors use the HEVC *codec* to encode video sequences in order to evaluate several flooding schemes for soft real-time video transmission in VANETs. The target application can be the timely delivery of video recorded by vehicles involved in an accident. This type of information may be useful for other vehicles located near the involved vehicles and also for public services, which may be located far from the accident site. They evaluate both the packet arrival ratio and PSNR value of the reconstructed video sequences. In [13], the authors do some real experiments of video transmission, by using several digital tablets and their WiFi connection. In one of the scenarios, the tablet inside one vehicle sends a video to the tablet in another vehicle, and in the other scenario a person standing in the sidewalk acting as a road side unit, sends a video to a passing by vehicle. They use three tablets: one of them acts as an access point and the other two act as the video sender and receiver. They use erasure codes, based in XOR operations, in order to protect the video traffic from losses. In [14], the authors propose MERVS, a streaming protocol in which the most relevant video frames are transmitted using the TCP protocol, and the least relevant video frames are transmitted using UDP protocol. TCP is in charge of retransmitting lost frames, so the protection is guaranteed by this protocol. The main restriction is that TCP mechanisms insert a great delay in the transmission and to overcome this drawback the authors use a mechanism called Quick-Start which alleviates the delay to some extent. In [15] and [16], the authors propose two mechanisms called SHIELD and AntArmour, to protect video streaming in vehicular networks, and they make comparisons between them. SHIELD uses the network statistics to make a self-adaptive mechanism in order to provide different levels of protection to network packets. AntArmour uses a scheme which is based in ant colonies to dynamically allocate the precise amount of redundancy in order to protect the video stream.

Forward Error Correction is a well-known technique used to protect data delivery in all types of networks. It is, therefore, also applied in vehicular networks. In [17], the authors use FEC techniques for the protection of real-time multimedia streaming over vehicular networks. They propose a technique that is based on the optimization of the FEC and interleaving parameters under real-time constraints. Interleaving can help reduce the length of error bursts in order to improve correction abilities. In [18], the authors conduct thorough research on the protection of content delivery in vehicular environments searching for the best packet size in order to maximize throughput. They use different FEC techniques to protect data and provide their results in terms of packet arrival ratio and file transfer time.

One of the differences between our work compared to other works is that we have used the real HEVC reference software in the decoding of damaged/incomplete bit streams. The original reference software crashes (it results in an illegal instruction exception and stops working) if it tries to decode a

bit stream with missing parts. This occurs because the decoder expects data with a specific format and when it encounters a gap, it cannot control the execution, and the program stops (without decoding the remainder of the bit stream). In order to use the real reference software decoder with "holey" bit streams, we modified the reference software to be aware of bit stream losses, allowing the decoder to continue its work until the end of the video sequence. Other works "emulate" the behavior of the decoder in the presence of missing parts by decoding the complete bit stream, and after that they remove the parts that correspond to the missing packets. This is not the real behavior of the HEVC decoder because, by proceeding in this way, the decoder can use all the information present in the bit stream for the decoding process and, as we realized in the patching process, some pieces of data depend on others (which may be missing) in the decoding process.

Some works use synthetic video data for their experiments, whereas we use well-known video sequences (belonging to the set in "Common Conditions for the evaluation of HEVC" [19]). Also, in some works, the evaluation of the quality of the decoded video sequence is represented in terms of Packet Loss Ratio (PLR) or other network performance metrics. PLR has influence on the final video quality but it is not a straightforward measure of video quality. The "semantic" meaning of the missing data is more important than the amount of data loss. Thus, we use a video quality measure (PSNR) to measure the final reconstructed video quality.

We have introduced RaptorQ codes, a FEC protection strategy, by using the real software RaptorQ encoder and decoder. This means that when we protect bit stream network packets, a file is first generated with the "source" packets and the "repair" packets, and then a simulation of their transmission through the network is performed. After that, the RaptorQ decoder is used to try to recover missing packets in a real way. Again, no simulation or formulas have been used for the use of this type of FEC.

Regarding vehicular networks, we have used real maps for the experiments, obtained from geographic databases. The motion of the vehicles is accurately simulated by vehicular simulators that take into consideration lanes, traffic lights, crossroads, etc., and the interaction of vehicles with them and with other vehicles (decelerating and stopping, accelerating, etc.). Instead of using standard WiFi communications, the protocols specifically designed for vehicular networks, like the IEEE 802.11p protocol and the IEEE 1609 family of standards (WAVE) have been used for the simulations. This protocols have several differences with other WiFi protocols (802.11a/b/g/n), which do not take into consideration characteristics like the division of the transmission time between the control channel and the service channel, the time guard interval, and others.

To obtain the results of video streaming over vehicular networks, we have conducted individual simulations for every tested combination, i.e., we have not used loss patterns. A video sequence encoded with different encoding settings and different protection parameters may lead to different results in the loss patterns (due to its bandwidth usage, packet rate, network load, etc.).

At last, we have used a mechanism that, to our knowledge, no other work has introduced: the combination of slices together with tiles to take advantage of the combined benefits of each one of them (error resilience features combined with coding efficiency). Also, the introduction of new encoding modes which show different encoding performance and error resilience attributes has not been addressed in other works. Using only one protection method cannot offer enough protection in these networks, so, we use the combination of source coding methods with channel coding methods to provide robust video streaming over vehicular networks.

3. Source Coding Protection

The main objective of this work is to design proposals for robust video streaming over vehicular networks and evaluate them in a realistic environment. These proposals are based on two different (and complementary) approaches. The first approach, based on source coding, focuses on the video codec itself and tries to produce a robust bit stream by means of Error Resilience (ER) and Error Concealment (EC) techniques.

For the encoding of video sequences, the HEVC standard has been selected. In the Call For Proposals for the development of HEVC, one of the targets was doubling its predecessor's (H.264/AVC) efficiency, i.e., to be able to generate a bit stream 50% smaller than H.264/AVC on the same quality level. Some works confirm that the real efficiency gains range from 30% to 45%. This "profit margin" is generally used to stream video sequences using fewer resources (by producing lower bit rates), or to stream sequences with higher resolutions and frame rates using the same resources. However, it is known that coding efficiency uses to be inversely proportional to bit stream robustness. So, we use the HEVC bit rate savings to strengthen the encoded video bit stream (producing a much more robust video stream with the same bit rate as H.264/AVC).

In the encoding process, each frame is divided into small square regions (usually, a 64x64 pixels area). Let's call them blocks. These blocks can be encoded using one of three modes: (a) without any prediction, (b) using spatial prediction, or (c) using temporal prediction. *Spatial prediction* exploits redundancy within a frame. In order to encode a block, it uses previously encoded regions of the same frame to search for pixel information that is similar to that block in order to create a prediction. Then, this prediction is subtracted from the current block and thus we obtain the prediction residual. This method is also called *intra*-frame prediction. *Temporal prediction* uses previously encoded frames to estimate a block prediction by means of the search of a similar block in other frames (called reference frames), which have been previously encoded, decoded, and stored in a buffer. It exploits temporal redundancy, taking advantage of the fact that nearby frames usually contain blocks that are very similar to the current block and so the residuum of the motion compensation is close to zero. This method is also called *inter*-frame prediction. For these three encoding modes, a domain transform is applied to the resulting values, so they are converted into the frequency domain. Then, these coefficients are quantized what introduces a loss of quality (to a certain extent) in the video stream. In the decoding process, the quantized coefficients will be scaled and these scaled coefficients will not be exactly the same as the original ones. This is the reason for the loss of information (lossy compression). Depending on the quantization step, the set of quantized coefficients will be more detailed (lower compression, higher quality), or less detailed (higher compression, lower quality). Finally, the quantized coefficients, together with the side information of the process (selected coding mode, motion vectors, etc.), are compressed by an entropy encoder to conform the final bit stream.

The source coding protection approach consists in tuning the intrinsic features of the video encoding process in order to obtain a robust encoded video stream which may preserve the user's perceived quality of the video, in a packet loss prone environment. So, we propose the use of three HEVC encoding tools: (a) frame partitioning (slices/tiles), (b) intra refresh coding modes, and (c) simple error concealment methods.

3.1. Tileslices, a new frame partition proposal

Slices are regions of an encoded frame that can be independently decoded, regarding other slices of the same frame. Slices are not a new concept in HEVC and they were used in previous standards. The main purpose of slices is providing a certain level of error resilience to a frame. If one slice of a frame gets lost, then the rest of the slices of the frame can be decoded and only the region covered by the missing slice is lost. Slices are independent syntax units inside the encoded bit stream which consist of correlative encoded blocks (Coding Tree Units - CTUs) in raster scan order (see Figure 1(a)).The main disadvantage of dividing a frame into slices is that the coding efficiency is reduced because of (a) the loss of prediction accuracy, since the information in neighbor slices is not available, and (b) the overhead introduced by slice headers.

One of the new features presented in HEVC is the ability to divide a frame into rectangular regions, called *tiles* (see Figure 1(b)), with the aim of incorporating parallel capabilities to video encoders and decoders. Tiles are independently decodable regions of a frame, which are encoded with some shared header information, but they are not independent syntax units of the bit stream. As they do not include heavy headers, and due to their rectangular shape, tiles are more flexible and efficient than slices, but

they are not useful for error resilience purposes because they are not completely independent (as they do not form independent syntax units in the bit stream).

In the search for robust video streaming techniques, while attempting to keep the overhead introduced by slices low, we have devised a new element that we call *tileslice*. First, one frame is divided into several tiles, and then each tile is placed into one slice. Every *tileslice* of a frame is a different syntax unit (because it is a slice), which has a rectangular shape (and this improves coding efficiency over typical slices). This is a possibility that HEVC includes in the definition of the standard, so it is not really a new invention, but, although it is a very simple idea, there are no previous works, to the authors' knowledge, that use this particular combination of slices and tiles. Maybe this is due to the fact that, although slices are well known elements used in previous video coding standards, tiles are relatively new objects in the video coding research timeline.

(a) *Slices* (b) *Tiles*

Figure 1. Representation of a frame divided into five slices (a), and a frame divided into four tiles (b). Slices and tiles consist of Coding Tree Units.

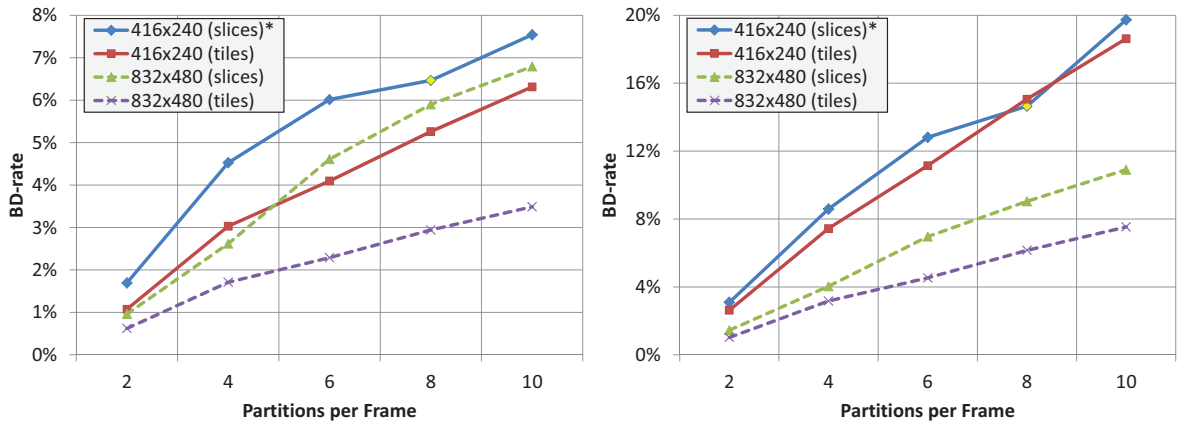
As *tileslices* contain slice headers, overhead introduced by them is not avoidable. So, is it really worth using *tileslices* instead of typical slices? To answer this question, several overhead measurements have been made. From this point forward we will use the term *tile* to refer to what we have defined as *tileslice*, i.e., one rectangular slice that contains one tile. In our tests we have compared the performance of tiles over slices for six different layouts: 1, 2, 4, 6, 8, and 10 slices (or tiles) per frame.

Table 1. Video sequences used in the tests.

Video sequence	Acronym	Resolution	FPS
BasketballDrill	bbd_25	832x480	25
BQMall	bqm_30	832x480	30
Flowervase	fl8_30	832x480	30
Keiba	ke8_30	832x480	30
Mobisode2	mo8_30	832x480	30
PartyScene	psc_25	832x480	25
RaceHorses	rh8_30	832x480	30
BasketballPass	bbp_25	416x240	25
BlowingBubbles	blo_25	416x240	25
BQSquare	bqs_30	416x240	30
Flowervase	fl4_30	416x240	30
Keiba	ke4_30	416x240	30
Mobisode2	mo4_30	416x240	30
RaceHorses	rh4_30	416x240	30

Fourteen video sequences, which belong to the HEVC "common test conditions" [19], have been selected. They are enumerated in Table 1. Seven of them have a resolution of 832x480 pixels and, the

other seven, have a resolution of 416x240 pixels. Each sequence has a frame rate of 25 or 30 frames per second (FPS). The two encoding modes used for the evaluation of HEVC are *All Intra* (AI) mode and *Low-delay P* (LP) mode, which are included in the HEVC reference software [20]. In AI mode, every frame of a video sequence is encoded as an I (*intra*) frame, so temporal prediction is not used at all. This mode has inherent error resilience properties because errors do not propagate to other frames. The main drawback of this mode is that it produces a bit stream with a high bit rate. In LP mode, the first frame of the sequence is encoded as an I frame, and then P (*predictive*) frames are generated for the rest of the sequence. P frames are mainly formed by inter-coded CTUs, i.e., CTUs that are encoded by using motion estimation and compensation using other frames as reference. A P frame can also contain intra-coded CTUs (e.g., if the encoder estimates that those CTUs are more efficiently encoded in an intra way). LP mode is much more efficient than AI mode and it generates a much smaller bit stream than AI mode for the same quality level, but a simple error in a frame is increased and propagated until the end of the sequence, even if the rest of the frames are correctly received. The compression rate of both modes can be selected by means of the Quantization Parameter (QP). This parameter is used by the quantization process, which entails the loss of precision in the transformed coefficients (lossy compression). If a high value is selected for the QP, the generated bit stream has a low bit rate and, correspondingly, low visual quality. If a low value is selected for the QP, the generated bit stream has a high bit rate and high visual quality. For the analysis of HEVC, 4 values for the QP have been used (22, 27, 32, and 37) as it is specified in the common test conditions.



(a) *AI mode* (b) *LP mode*

Figure 2. Average BD-rate for different frame partition layouts (slices/tiles).

In Figure 2, Bjørntegaard Delta Rate (BD-Rate) [21] average values of both sets of source video sequences are shown when using 2, 4, 6, 8, and 10 tiles or slices per frame. BD-Rate is a measurement that shows the relative bit rate increase (or decrease) of one encoding proposal with respect to another one chosen as reference. For the low resolution source videos (416x240), dividing a frame into 8 slices is not possible, so, we have decided to use 7 slices per frame instead, what is not fair when comparing to the 8 tiles per frame option. This issue can be observed in Figure 2 plots, specially in Figure 2(b), where no differences between tiles an slices are found at 8 tiles/slices per frame in low resolution video sequences. We have placed and asterisk (*) in the figure legend to remind this fact. Except in this specific case, all the values in the curves that represent the overhead of tile partitions are lower than those representing the overhead of slice partitions. This is the reason why the tile (*tileslice*) partition will be the favorite frame partition method to protect the video bit streams. The maximum overhead reduction, around 3% less overhead than slices, is obtained with 10 tiles per frame in the 832x480 video sequences for both LP And AI coding modes. By using *tileslices* instead of slices we obtain the same level of error resilience and a better compression performance.

3.2. Intra Refresh Coding Modes

In the previous tests, two encoding modes have been used: All Intra mode and Low-delay P mode. AI mode offers better error resilience properties than LP mode because an error in one tile does not propagate to other frames. On the other hand, LP mode is very sensitive to lost tiles, because P frames are "infected" by erroneous reference frames, and these infected P frames, when used as reference frames, "spread disease" until the end of the sequence. There is also an unpredictable reaction in LP mode in the presence of packet losses. For example, when an LP encoded sequence loses the last frame, then the error affects only that frame. But if the first frame (I) gets lost, then no frame can be correctly reconstructed at all. So, the position of the missing parts is also important when dealing with errors in LP mode. These two encoding modes have been determined as our upper and lower bounds regarding error resiliency. In this work, 7 new encoding modes have been proposed and evaluated, which introduce intra refresh to some extent. Intra refresh is an error resilience technique that forces to periodically intra encode (refresh) certain frame areas into the inter-coded frames, in order to stop the error propagation mentioned before. The 9 encoding methods evaluated are explained below.

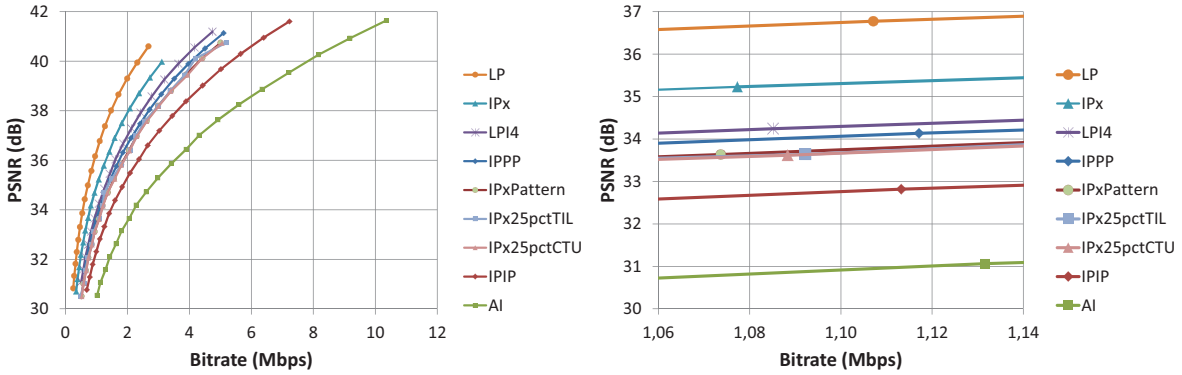
- **AI** - This mode encodes every frame as an I frame. It is considered our upper bound mode.
- **LP** - This mode encodes the first frame as an I frame and the rest of the frames as P frames. Four previous reference frames are available for temporal estimation and compensation for each P frame. It is considered our lower bound mode.
- **IPx** - Similar to LP mode but every P frame uses only the previous frame as reference frame, instead of having 4 reference frames.
- **IPx25pctCTU** - Similar to IPx mode, where 25% of the CTUs are forced to be intra refreshed.
- **IPx25pctTIL** - Similar to IPx mode, where 25% of the tiles are forced to be intra refreshed.
- **IPxpattern** - Similar to IPx mode, where 25% of the tiles are forced to be refreshed following a specific pattern, which covers all the tiles of a whole frame every 4 frames.
- **IPPP** - Similar to IPx mode, with an I frame inserted every 4 frames.
- **LPI4** - Similar to LP mode, with an I frame inserted every 4 frames.
- **IPIP** - I and P frames are inserted alternatively. Every P frame always uses the previous I frame as reference.

The 9 encoding modes can be classified into four groups, depending on the whole-frame refreshing rate: (a) LP and IPx modes only have an I frame at the beginning of the sequence, so no refreshing is performed (0%); (b) IPPP, IPx25pctCTU, IPx25pctTIL, IPxpattern, and LPI4 modes have an average frame-refresh rate of one complete intra refreshed frame every four frames (25%); (c) IPIP mode has an intra frame-refresh rate of one frame out of two (50%); (d) AI mode uses a full rate of intra-refreshed frames (100%). For the following tests, we have selected the BasketballDrill sequence (832x480 pixels, 25 fps, `bbd_25`), which exhibits an average coding performance amongst the 832x480 sequence set. In Figure 3(a), the resulting rate/distortion curves for the 9 encoding modes (for a very wide range of QP values) are plotted. LP mode results the most efficient mode of all, and AI the least efficient.

To evaluate the 9 encoding modes in fair conditions, we have selected an individual QP value for every mode so that they generate a bit stream with a similar bit rate. The QP values used for each encoding mode with the resulting bit rate and PSNR values are listed in Table 2. Figure 3(b) is a zoomed version of Figure 3(a), which shows the curves for the selected QP values. The curves corresponding to encoding modes belonging to group (b) exhibit very similar coding efficiency. On the contrary, in group (a), LP and IPx curves are not close to each other, and LP clearly outperforms IPx regarding coding efficiency, which remarks the importance of the available reference frames. The PSNR values range from 31.07dB (AI) to 36.77 dB (LP) in a no-loss scenario. The list of the 9 encoding modes, ordered by their coding efficiency, is the same order found at plot labeling.

3.3. Error Concealment

Finally, we propose a simple error concealment technique that will hide the missing information to the user in such a way that the received video quality will not be seriously damaged. A well-known



(a) *R/D representation for* `bbd_25` (b) *Zoom in the range of 1.06 to 1.14 Mbps*

Figure 3. (a) Rate/Distortion representation for `bbd_25` sequence encoded with all the encoding modes. (b) Zoom in the range of 1.06 to 1.14 Mbps of Rate/Distortion representation for `bbd_25` sequence.

Table 2. Selected values for the QP parameter and bit rate and the PSNR values obtained for each encoding mode.

Encoding mode	QP	Bit Rate (Mbps)	PSNR (dB)
AI	40	1.13	31.07
IPIP	36	1.11	32.82
IPPP	33	1.11	34.13
LP	27	1.10	36.77
LPI4	32	1.08	34.24
IPx25pctCTU	32	1.08	33.62
IPx25pctTIL	32	1.09	33.64
IPxpattern	32	1.07	33.63
IPx	29	1.07	35.23

EC technique (defined as "zero-MV concealment" in [22]) exploits temporal redundancy in video sequences by applying a motion compensated concealment with the zero motion vector. In this technique, the missing area of a frame is replaced by the co-located area of the previous frame. This technique has a particular case in which a whole frame is missing or corrupted. In this case, the last correctly decoded frame is duplicated in order to conceal the loss of the missing one. This technique is known as "frame copy concealment" in [23]. In order to add error concealment features to video streaming we have modified the HEVC reference software decoder to add the options which allow us to apply these EC techniques, by copying the previous co-located region when part of a frame is missing, instead of rendering a gray region on it, and also by creating a duplicate of the last decoded frame when we encounter a missing frame.

3.4. Evaluation of Source Coding Protection

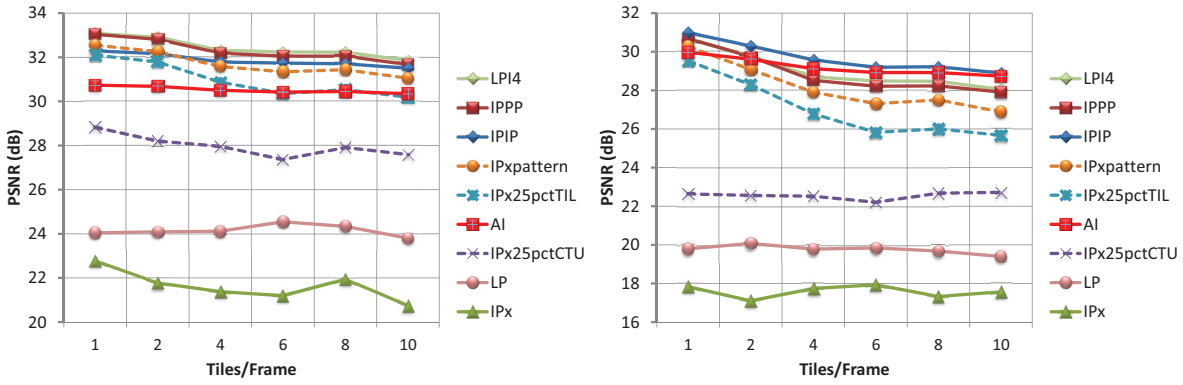
The performance of the proposed encoding modes and the overhead introduced by tiles and slices have already been evaluated in a no-loss scenario. Now, the performance of the 9 intra refresh coding modes with the different tile partition layouts (1, 2, 4, 6, 8, and 10 tiles/frm), under a simple probabilistic network packet loss model, will be evaluated. Six different Tile Loss Ratios (TLR) have been selected for the tests: 1%, 3%, 5%, 7%, 10%, and 20%. For every one of these loss rates, 5 different seeds for the random number generator have been used, and the results have been averaged.

First of all, we have evaluated the decoder EC performance. The frame copy method has been always used in order to obtain a reconstructed video sequence with the same number of frames than the original video sequence. The zero-MV technique has been tested and the differences in the PSNR

value when using this method or not, have been calculated. For every encoding mode and tile loss percentage, we have computed the PSNR values for all the tile partition layouts (1, 2, 4, 6, 8, and 10 tiles/frm). The results show that the differences in the PSNR value are always positive. This means that the EC decoder version provides better PSNR values than the version without EC and the PSNR gain ranges from 0.14 dB up to 3.02 dB. In the two modes without intra refresh, LP and IPx, the resulting PSNR values with both the EC decoder and the non-EC decoder versions are below the bounds which are considered the minimum acceptable, so the values of the difference in PSNR are not relevant. As a product of these observations, from this point forward we will always use the EC decoder version for the reconstruction of video sequences.

Regarding the error resilience attributes related to the encoding modes and the frame tile layouts, in Figure 4 the PSNR values obtained for each encoding mode with a 3% and a 10% TLR are shown. Three of the encoding modes show a non-robust response to tile loss (PSNR values under 29 dB are considered low quality values). For LP and IPx this is the expected result, as they have no intra refreshing strategy. But the bad performance for IPx25pctCTU was not expected, because in this mode 25% of the CTUs (one out of four) are forced to be intra-coded. Previously, this encoding mode showed the same coding efficiency as IPx25pctTIL and IPxpattern (which share the same percentage of intra refreshed areas), but it is clear that a random CTU intra refresh provides worse protection against tile loss than a random tile intra refresh or a pattern tile intra refresh. Why intra refresh on the CTU level is not 100% effective? The reason is that intra-frame encoding uses pixel information (surrounding the CTU) to compute a prediction, which is used both in the encoding process and also in the decoding process. If the pixels used to compute that prediction belong to inter-coded CTUs whose reference frames are corrupted, then the pixels used for the intra-frame prediction are not correct, and, even if the intra CTU is correctly received, it will be incorrectly decoded. If intra refresh is carried out on tile level, as all the CTUs that belong to a tile do not depend on CTUs from outside that tile, then every correctly received intra CTU will be correctly decoded. Intra refreshing on the CTU level does not provide the bit stream with enough robustness.

One of the consistent results throughout all the tile loss percentages tested is that, for a fixed TLR, when the number of tiles per frame increases, the quality decreases. The reason is that the loss of small parts in many frames causes a worse effect than the loss of one big part in only one frame, even if the loss percentage is the same. For 1% and 3% TLR, the most robust encoding modes are LPI4 and IPPP. At 5% TLR, these two modes and IPIP show very similar performance. And at 7% TLR and above, IPIP is the most resilient mode. At 20% TLR, AI mode is over IPIP for 4 tiles per frame layout and above, but the low PSNR values obtained at these points discard it as a good solution.



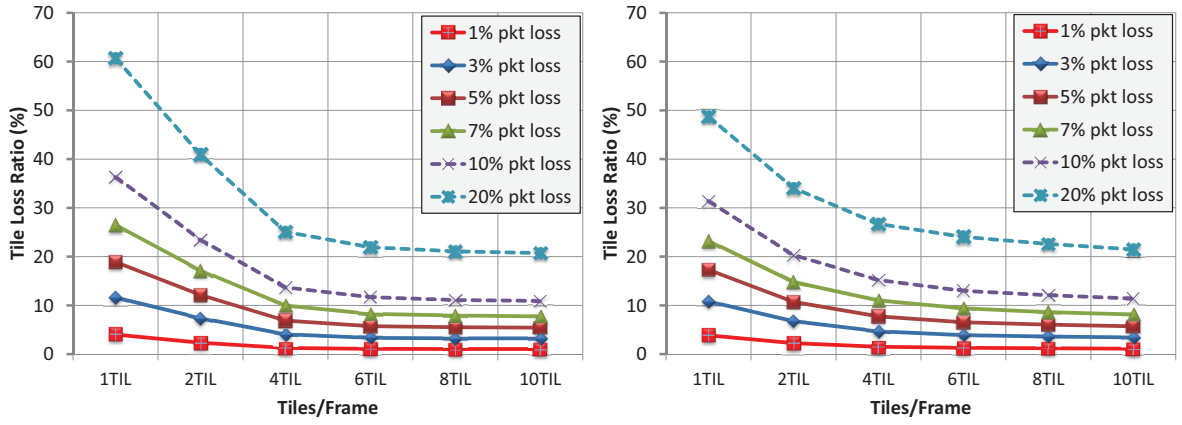
(a) *3% TLR* (b) *10% TLR*

Figure 4. PSNR for all the til/frm layouts and all the encoding modes with 3% and 10% of TLR. Non-FEC bit streams.

Table 3. Average number of Packets Per Tile for every encoding mode.

Packets Per Tile	1 til	2 til	4 til	6 til	8 til	10 til
AI	4.38	2.52	1.37	1.17	1.12	1.09
IPIP	4.37	2.35	1.57	1.27	1.14	1.10
IPPP	4.40	2.46	1.49	1.27	1.17	1.10
IPx	4.18	2.38	1.41	1.14	1.05	1.02
IPx25pctCTU	4.27	2.41	1.41	1.13	1.04	1.02
IPx25pctTIL	4.26	2.42	1.43	1.20	1.08	1.07
IPxpattern	4.19	2.39	1.42	1.19	1.09	1.06
LP	4.30	2.43	1.45	1.20	1.10	1.06
LPI4	4.34	2.35	1.59	1.35	1.23	1.14

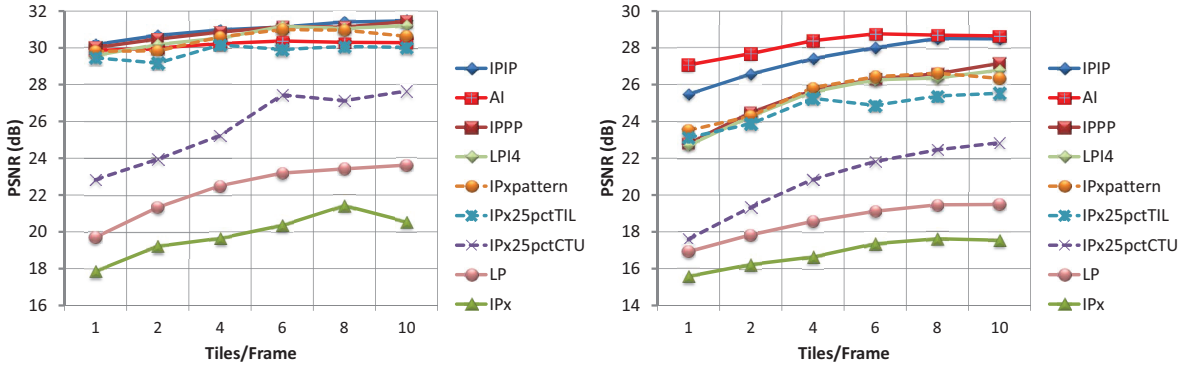
From these experiments, where we introduce fixed TLRs, it seems that the 1 tile per frame layout is more robust than the other layouts. But, in the "real world", a certain network Packet Loss Ratio (PLR) may produce very different TLRs, depending on the proportion between each tile size and the network MTU (Maximum Transfer Unit). The tile size depends on several factors as the frame type (I, P), the number of tiles per frame, and the QP parameter selected (the compression ratio). Now, we study the behavior of the different tiles per frame layouts with fixed PLRs. First of all, note that, if a tile is larger than the network MTU, that tile has to be divided into several network packets, and, if one of these packets gets lost, then the rest of the packets belonging to the same tile are completely useless because that tile cannot be decoded at all. The average number of network packets per tile in the encoded bit streams is shown in Table 3. For the 1 tile per frame layout, the proportion is over 4 packets per tile. This means that the loss of one network packet will probably entail the effective loss of four packets. For the 10 tiles per frame layout, the proportion is near 1 packet per tile. This means that the loss of one network packet will probably entail the effective loss of only that packet.



(a) *AI mode* (b) *LPI4 mode*

Figure 5. TLR obtained for every PLR for the AI and LPI4 encoding modes. Non-FEC bit streams.

Figure 5 shows the TLRs obtained for different PLRs, for both AI and LPI4 modes. It is clearly visible that the 1 tile per frame layout suffers from vulnerability in both modes, so this layout which apparently showed to be the most robust in the previous tests, now provides higher TLRs than the rest of the layouts, for a fixed PLR, which is a drawback. The obtained TLRs are correlated to the values in Table 3, so when the number of packets per frame tends to one, the TLR obtains its minimum value. Dividing a frame into 6, 8, or 10 tiles per frame produces similar TLRs, because the proportions of packets per tile for these three modes are very close.



(a) *3% PLR* (b) *10% PLR*

Figure 6. PSNR for all the til/frm layouts and all the encoding modes with 3% and 10% of PLR. Non-FEC bit streams.

Now, we will show the video quality performance of the different tiles per frame layouts, at a fixed PLR. In Figure 6, the PSNR values for each tile per frame layout at 3% and 10% PLR, is shown. Again, LP, IPx, and IPx25pctCTU obtain very low PSNR values, showing little resilience against errors. The other six modes have increasing PSNR values when the number of tiles per frame increases. This result is consistent throughout all the PLRs tested, therefore, in all the experiments, we have observed that when a high number of tiles per frame is selected, the recovered video sequence obtains a better PSNR value. For moderate packet loss (1%), IPPP and LPI4 are still the best encoding methods; for a medium packet loss percentage (3%), IPPP and IPIP are the best methods; and, for the rest of values (5%, 7%, 10%, and 20%), the best methods are AI and IPIP, although for values of 7% and higher, the PSNR values are very low.

From the evaluations of this section, some conclusions can be drawn. Of the 9 encoding modes evaluated, there are three that do not have good error resilience properties. Two of them (LP, IPx) have provided the expected performance results as they do not use any intra refreshing at all. But the third one (IPx25pctCTU), which refreshes one of every four CTUs, does not have the expected intra-refresh effect in the bit stream. Of the rest of the modes, two of them (LPI4 and IPPP) stand out with respect to the others when the percentage of loss remains low. And another method (IPIP) seems to have good performance when the percentage of loss increases. With respect to the effect of frame partitioning, we have confirmed in all experiments that the higher the number of tiles per frame, the higher the final video quality. However, it is recommended to use no more than 6 tiles per frame, since the benefits in video quality are negligible. Although some of the methods exhibit good error resilience properties, this is not enough to guarantee robust video streaming. In the next section, we will evaluate RaptorQ codes, and will search for the best setup to provide the desirable protection to the video packet stream.

4. Channel Coding Protection

The second proposed ER approach tries to provide redundancy to protect the video packet stream against loss by using RaptorQ codes, an Application Layer Forward Error Correction (AL-FEC) method, which adds redundant data to the bit stream in order to recover lost packets.

4.1. FEC Protection

RaptorQ codes provide AL-FEC protection to video streaming in order to improve video quality by recovering as many lost packets as possible. The cost of using FEC is an additional network bandwidth to carry FEC repair data, and also an increased latency to support packet recovery from a block of packets. Increasing the FEC bandwidth (given a fixed FEC latency) leads to an increase in

data protection. Similarly, increasing the FEC latency (given fixed FEC bandwidth) also leads to an increase in data protection.

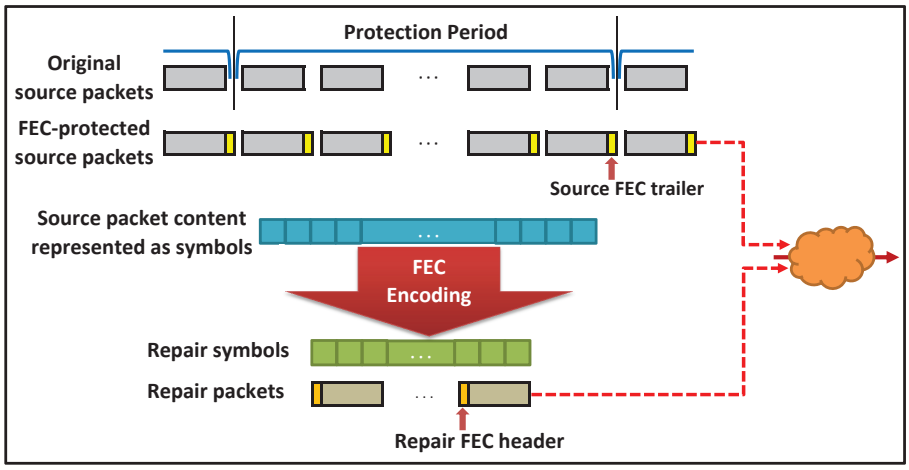


Figure 7. RaptorQ encoding process.

Figure 7 shows the way in which the RaptorQ encoder protects data. First of all, the RaptorQ encoder receives a data stream (*original source packets*) during a specified *protection period*. A 4-byte *FEC-trailer* is added to each received packet thus forming an *FEC-protected source packet stream*. This trailer identifies the packet and the protection period to which it belongs. When an FEC-protected packet is generated, it is then immediately sent through the network. The packet payloads for all source packets within the protection period are gathered and represented as a set of continuous *RaptorQ source symbols*. When the protection period finishes, these source symbols are used to compute *repair symbols* for that protection period by the *FEC encoding* process. The repair symbols are the same size as the source symbols and are carried in *repair packet* payloads. Each repair packet includes a 6-byte *FEC header* that prepends one or more FEC repair symbols. The repair FEC header identifies the associated protection period, the first symbol in the repair packet, and the number of source symbols in the protection period. Both the original source packets (with the FEC-trailer) and the associated repair packets are transmitted within the same protection period, and are used by a receiver application in the recovery process. Within each protection period, a receiver must receive "enough" source and repair symbols (from the FEC-protected source and repair packets) to be able to recover any dropped packets.

Table 4. Average overhead percentage for using a symbol size of 192 bytes, 450 bytes, and 1350 bytes.

Overhead	10% prot.		20% prot.		30% prot.	
percent.	1 til	10 til	1 til	10 til	1 til	10 til
192 B	11.68%	13.05%	22.50%	24.84%	33.36%	36.67%
450 B	14.24%	16.36%	26.96%	30.93%	39.87%	45.63%
1350 B	22.65%	30.17%	44.14%	56.75%	62.20%	83.79%

For the protection of the encoded bit streams, we have used the Qualcomm (R) RaptorQ (TM) Evaluation Kit [24]. For each one of the encoded bit streams (`bbd_25` sequence, using 9 encoding modes, with 6 til/frm layouts), we have generated several protected versions by combining different values for the parameters cited before: *protection level*, *protection period*, and *symbol size*. The properties of the protected versions will determine the most suitable configurations for the protection of the video bit streams. Three different levels of protection have been tested: 10%, 20%, and 30%; seven protection

periods have been used: 133, 166, 200, 250, 333, 500, and 1000 milliseconds; and three symbol sizes have been evaluated: 192, 450, and 1350 bytes.

In Table 4 the average overhead for the three levels of protection, for the 1 til/frm and 10 til/frm layouts and all the protection periods, is shown. The smaller symbol size (192 bytes) obtains the lowest overhead values. This is due to the nature of the source data. As the source packets payloads have variable sizes and they range from small to big packets, a small symbol size optimizes the size of the repair packets. Instead, if we were protecting some other type of data with a constant size payload, then a symbol size matching the payload size would be the most efficient. Another observed feature is that the 1 til/frm layout generates less overhead than the 10 til/frm layout. The overhead values for the rest of the layouts have not been included in these tables for the sake of concision and clarity, but they show monotonically increasing behavior: the higher the number of tiles per frame, the higher the overhead introduced by FEC protection. For a fixed symbol size and a fixed til/frm layout, the overhead values for the different protection periods show monotonically decreasing behavior: the wider the temporal window, the lower the overhead percentage. As the size of the protection period has a direct effect on latency, the selection of the proper temporal window seems clear: the highest acceptable latency will determine the most efficient protection period. In this work, as a "design decision", we have selected a protection period of 333 ms. Depending on the particular ITS application which uses video streaming, the selection of the protection window can vary if the delay requirements are tight or more relaxed.

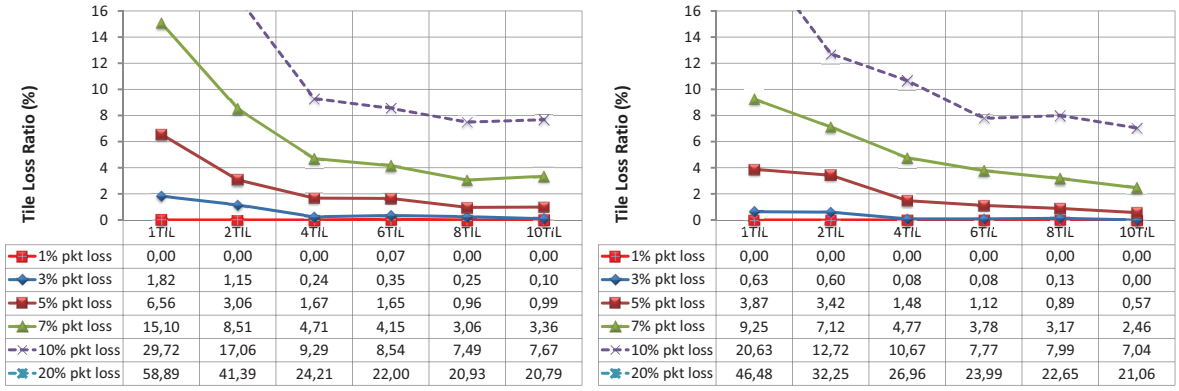
In summary, the parameters selected for the protection of the video bit streams are a symbol size of 192 bytes because of efficiency (lowest overhead) and a protection period of 333 ms (maximum latency allowed, as a design decision). We will maintain the three values for the protection level parameter in order to test their performance against packet losses. After tuning the parameters of the FEC module, we have analyzed the resulting bit rates to determine the overhead introduced by both FEC coding and the use of tiles. On the one hand, the overhead introduced by FEC is close to the protection level (i.e., a protection level of 10% will increase the bit stream size around 10%). The same behavior has been observed in all coding modes defined in this work. On the other hand, increasing the number of tiles per frame also produces an increase on the final bit stream size (see Figure 2). So, when fixing a protection configuration, we have to take into account the size of the resulting bit stream that will depend on (a) the coding mode, (b) the number of tiles per frame, and (c) the FEC protection level.

4.2. Packet Recovery

Although in the previous section we set a protection period of 333 ms because it obtains the most efficient performance, in terms of bit rate overhead, for the highest latency allowed, we want to check if this selection is also efficient regarding data protection. After evaluating the recovery attributes of all the protection periods equal or lower than 333 ms, we have observed that the 333 ms temporal window clearly outperforms the rest of the protection periods. Consequently, the selection of the 333 ms protection period produces both the lowest overhead and the best percentage of recovery.

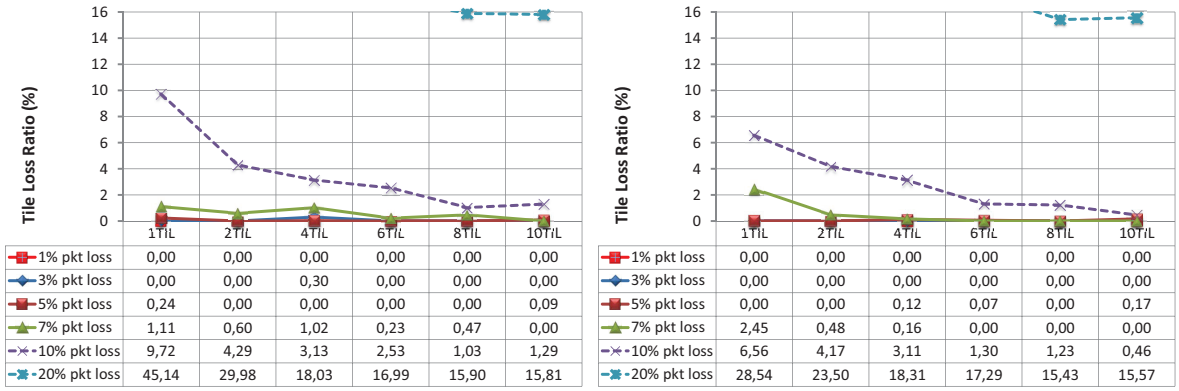
Figure 8 shows the TLR (after FEC recovery) for different til/frm layouts with a protection level of 10% for both the AI and LPI4 modes at different packet loss rates (1%, 3%, 5%, 7%, 10%, and 20%). Figure 9 shows the same data for 20% FEC protection level. In both figures, and for both the AI and LPI4 graphics, the y-axis has been fixed ranging from 0% to 16% to make comparisons between them clearly. Note that the 20% packet loss curve is outside the bounds for this scale (or nearly out), so it is not completely sketched inside the graphic areas, but its values can be numerically inspected in the data tables.

In the case of a protection level of 10%, layouts of 4 til/frm and higher noticeably improve the recovery percentage over 1 til/frm and 2 til/frm layouts (in which the number of packets per tile penalizes the recovery process). A protection level of 10% can almost completely recover a protected bit stream for a packet loss percentage of 1%, and keeps the tile loss percentage really low for a packet



(a) *AI mode* (b) *LPI4 mode*

Figure 8. TLR for every PLR, for the AI and LPI4 encoding modes. FEC-protected bit streams with a protection level of 10% using a protection period of 333 ms and a repair symbol size of 192 bytes.



(a) *AI mode* (b) *LPI4 mode*

Figure 9. TLR for every PLR, for the AI and LPI4 encoding modes. FEC-protected bit streams with a protection level of 20% using a protection period of 333 ms and a repair symbol size of 192 bytes.

loss percentage of 3%. Even at a packet loss of 5%, a protection level of 10% does a good job, especially for 8 and 10 til/frm layouts (where the final tile loss percentage is lower than 1% in both encoding modes). In Table 3 the AI encoding mode obtains an average value of 4.38 packets per tile (1 til/frm layout), and the LPI4 encoding mode obtains a very similar average value of 4.34 packets per tile. But the values of TLR for both modes (in Figure 8) are not as similar. Although the average packets per tile are similar, the I and P frame distribution in each encoding mode is very different. In the AI mode all the frames are I frames. This implies that all the encoded frames have a similar size and the packets-per-tile ratio is nearly constant for every frame. On the contrary, the LPI4 encoding mode inserts an I frame followed by three P frames. P frames are more efficient (and therefore they are smaller) than I frames, so the packets-per-tile ratio of a P frame is lower than that of an I frame. When the packets-per-tile ratio tends to 1 (high number of til/frm) these differences are narrow, but when the packets-per-tile ratio increases (low number of til/frm), the differences increase. This unbalance produces different results in the final tile loss percentage of both modes, and, in this situation, the LPI4 mode yields better recovery results than the AI mode. In the case of a protection level of 20% (Figure 9), values of 5% of packet loss and lower are practically wiped out. For the LPI4 encoding mode, this is also true for 7% of packet loss and 4, 6, 8, and 10 til/frm layouts. The results in both figures show that the protection level parameter does not have to be confused with the "recovery percentage." So, a trade-off between bit stream overhead and packet recovery capabilities should be found depending

on application requirements and network load conditions. The results show that between 4 and 6 tiles per frame with a 10% protection level configurations provide a reasonable good packet recovery performance with a moderated bit stream overhead.

5. Real Scenarios Evaluation

Note that in the previous tests we have compared the results obtained by the different configurations using a simple probabilistic network packet loss model. In a realistic vehicular scenario, different layouts produce different PLRs to deal with, so every configuration does not "work" under the same conditions. Each different configuration of the bit stream (encoding mode, protection level, til/frm, ...) may behave differently through the network and will have to confront a different situation. In this section, we will evaluate all the protection proposals in a realistic vehicular network environment to check their performance in this type of scenarios.

5.1. Test Framework

For the evaluation of error resilience proposals, we have used three main blocks (see Figure 10). The first of these blocks is formed by an open-source vehicular traffic simulator: SUMO (Simulation of Urban MObility) [25]. This simulator models the behavior of vehicles on routes, interacting with other vehicles, junctions, multi-lane roads, traffic lights, etc. The second of the blocks is in charge of the simulation of vehicular network communications, particularly those based on the IEEE 802.11p protocol and the IEEE 1609 family of standards (WAVE). For this task, we have selected OMNeT++ (Objective Modular Network Testbed in C++) [26] together with Veins (VEhicles In Network Simulation) [27], which is based on MiXiM (MIXed sIMulator) project [28]. The vehicular simulator and the network simulator blocks are connected by TraCI (TRAffic Control Interface) [29]. TraCI creates a TCP connection to allow the communication between the two simulators. The third of the blocks deals with video processing. It prepares the encoded bit streams for the tests and also evaluates the quality of the final recovered/reconstructed versions of the streamed video sequences. The encoded bit streams are used to generate *video trace files* which are injected into the simulations. For this purpose, we developed a new module in OMNeT++. This module provides a video sender which reads packets from a trace file to deliver them through a vehicular network scenario. Vehicles tagged as video receivers get video packets and write both a file with the correctly received packets and a file with several network statistics such as PLR. These two files are subsequently used for evaluations.

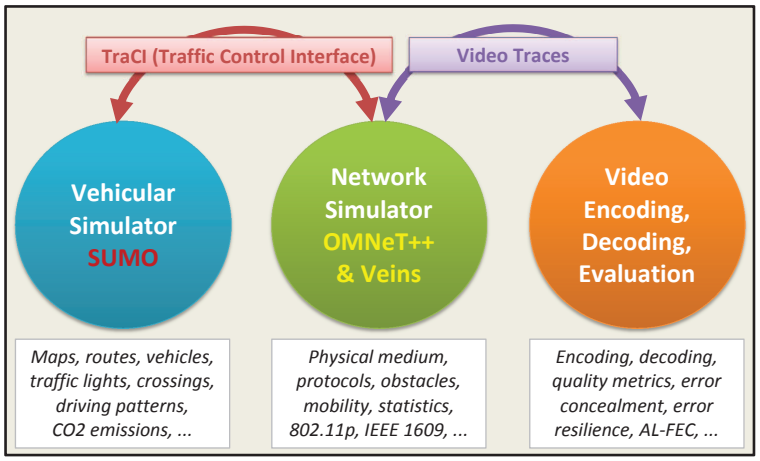
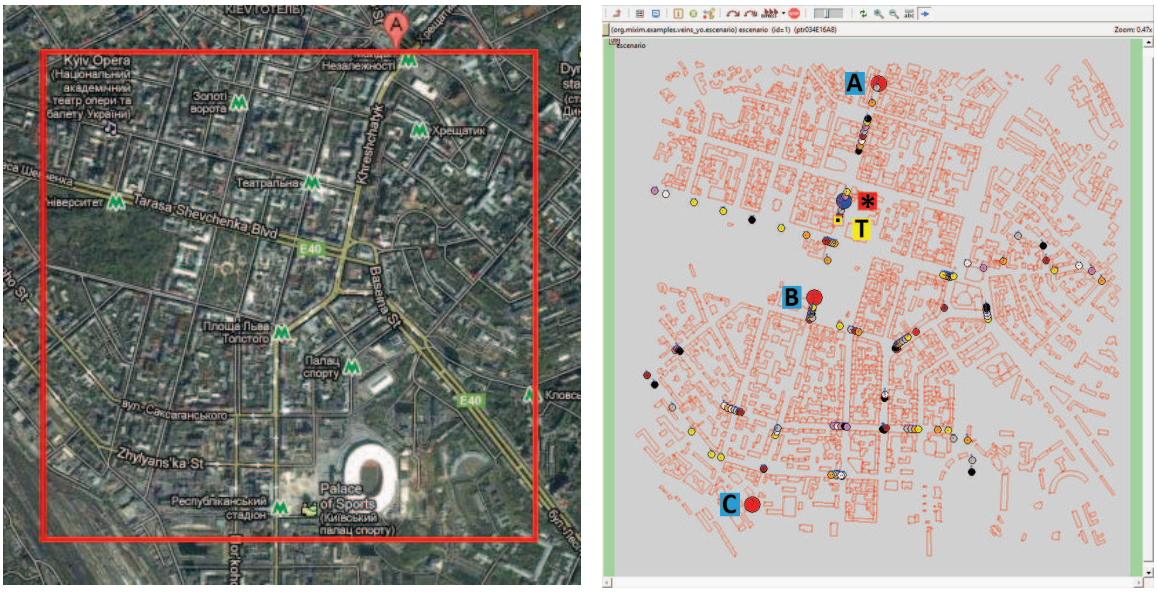


Figure 10. Test framework.

The scenario for the simulations is an area of 2000 m x 2000 m in the city of Kiev (Ukraine). Figure 11(a) shows an orthophoto of the selected region. At the bottom of the picture, the Olympic National Sports Complex can be clearly seen. To obtain the data for the vehicular scenario (lanes,

building, etc.), the OpenStreetMap (OSM) [30] project has been used, as it provides free maps from all around the world. The data obtained from OSM has been converted into SUMO compliant format. The map zone selected for the tests, loaded in OMNeT++/MiXiM/Veins, is displayed in Figure 11(b). Red drawings represent the buildings imported from OSM. Veins uses a file with polygons to sketch the obstacles and to calculate the visibility between two wireless network interface cards. There is a long avenue that crosses this area from north to south. Along the avenue, three Road Side Units (RSUs) have been positioned, tagged A, B, and C in the figure. They are around 1 km apart from each other. The coverage radius of all the wireless devices is 500 m, with a maximum data rate of 6 Mbps. There is a small area between RSUs A and B that is not covered by either of them, so a shaded area appears. There is also a small area halfway between RSUs B and C that is covered by both of them (where their signals overlap). Therefore, we have three different types of areas regarding transmission: (a) areas where a vehicle receives data from only one RSU (**Zone I**); (b) one area where the signal is momentarily lost (between the A and B coverage areas)(**Zone II**); and (c) one area where the vehicle receives the signal from two RSUs (B and C)(**Zone III**). The three RSUs transmit the same video sequence simultaneously in a synchronized and cyclic way. A total of 450 vehicles are inserted into the scenario, driving in different routes, which come and go from the simulation area. At every moment, there are simultaneously around 80 vehicles in the cited area. The maximum allowed speed is 50.4 km/h. Vehicles send ten *beacons* per second through the control channel (following the IEEE 1609.4 multi-channel operations). RSUs send periodically, through the control channel, advertisements of the video service that they offer, indicating the service channel used for the video stream. The video client vehicle (labeled in the figure as *) receives that invitation and commutes to the specified service channel in order to receive the video stream. This vehicle travels along the avenue, receiving the video stream from the RSUs through the specified service channel. Another vehicle drives nearby the video client which can act as a background traffic source (labeled as T in Figure 11(b)) by sending packets through the wireless network at the specified Packets Per Second (PPS) rate. It is used to reproduce different network load conditions. The video client experiences isolated packet losses (mainly due to background traffic) and bursty packet losses (around the limits of RSUs coverage).



(a) *Kiev orthophoto* (b) *Simulation environment*

Figure 11. (a) Partial view of the city of Kiev. The square indicates the area selected for the simulations. (b) Vehicular network scenario in OMNeT++/MiXiM/Veins. *(red circles [A,B,C] = RSUs; blue circle [*] = video client; yellow square [T] = background traffic source; small circles = other vehicles; red polygons = buildings).*

In the experiments, the combination of the bit streams indicated in the previous sections has been used. For the tests, we have selected the BasketballDrill sequence (832x480 pixels, 25 fps, `bbd_25`), encoded with 9 different encoding modes (AI, IPIP, IPPP, IPx, IPx25pctCTU, IPx25pctTIL, IPxpattern, LP, and LPI4) and with 6 different tile layouts (1, 2, 4, 6, 8, and 10 til/frm). For each one of these bit streams, 4 different versions have been generated. One of these versions has been encoded without any FEC protection, but with the appropriate division of tiles into network packets. The other three versions of every bit stream have been FEC protected with RaptorQ codes for 3 different protection levels (10%, 20%, and 30%). Each one of these bit streams (with or without protection) has been delivered by the three RSUs and received by the video client in the described vehicular scenario under 3 different "network conditions." The first group of tests has been conducted without injecting any background traffic. This group of tests is referred to as "ideal conditions" and permits the characterization of each one of the three zones mentioned before. After that, experiments have been performed under 2 further network conditions: one with moderate background traffic (62 packets per second of 512 bytes), and another with more dense background traffic (125 packets per second of 512 bytes), both for protected bit streams and non-protected bit streams. The tests with the non-protected bit streams serve to evaluate the performance of the encoding modes and the frame layouts, regarding error resilience. The tests with the FEC protected bit streams serve to evaluate the specific level of protection added by RaptorQ codes. In the following section, the results obtained will be presented and analyzed.

5.2. Analysis of Results

Before explaining the results obtained, we want to remember the difference between two mentioned metrics: PLR and TLR. PLR is the percentage of network packets that get lost, including the repair packets, if any. TLR is the percentage of tiles that cannot be used in the decoding process because one or more of the network packets which carry parts of that tile get lost and cannot be recovered by FEC means. These two concepts are not equivalent, nor directly proportional, as shown before. We can have a low PLR that leads to a high TLR and vice versa.

5.2.1. Ideal Conditions

The first set of experiments in the vehicular scenario tries to evaluate the situation under ideal conditions, i.e., when no background traffic is injected into the service channel that is used by the RSUs to stream the video.

- Without FEC protection

In **Zone I**, under these conditions, every packet sent by the RSU is correctly received by the client vehicle for all the combinations of coding modes and tiles per frame. So, we have a PLR value of 0%. Consequently, no tile is missing in any of the sequences and the quality of the reconstructed video sequences is the quality directly provided by the encoding/decoding process (shown in Table 2). The coding efficiency of each one of the encoding modes prevails, so, in the absence of packet loss obtained in Zone I, the LP and IPx modes obtain the best PSNR values.

In **Zone II**, for the non-protected bit streams, we encounter a burst of packet loss corresponding to the area where neither of the RSUs signals arrive. The average PLR in this zone has a narrow range (from 1.98% to 2.01%). The TLR has also a narrow range (from 1.95% to 2.39%). As opposed to the results shown in Figure 5 for probabilistic random packet losses (where the TLR was higher than the PLR, especially in 1 and 2 til/frm layouts), here, the PLR is always very similar to the TLR, even when a low number of tiles per frame is used. The reason is that, when isolated losses occur (like those generated by random packet loss), the real loss of one isolated packet (a low PLR) entails the loss of the whole tile (which may mean the effective loss of several packets and a high TLR), but, when bursty losses occur, like in this case, the loss of consecutive packets that belong to the same tile does not entail an increase in the TLR. For this reason, in Zone II, the PLR and TLR have analogous values. In this zone, the best two modes regarding the PSNR value (averaging the whole range of

til/frm layouts) are LPI4 and IPPP, with 33.74 dB and 33.71 dB values, respectively. The IPx25pctTIL and IPxpattern modes also show good performance. As the TLR is not pronounced, all the encoding modes have PSNR values over 30 dB, even the LP, IPx, and IPx25pctCTU modes, which showed poor performance in the presence of packet loss. Under these conditions, the AI mode obtains the lowest PSNR value. Comparing the different tiles per frame layouts, no layout performs much better than the others, regarding the PSNR value.

In **Zone III**, where the signals of two of the RSUs overlap, the PLR is around 22%-23%. As the two RSUs are 1 km away from each other, they cannot "see" each other (because the coverage radius of the wireless devices is 500 m). These collisions produce a high loss rate in the overlapped area, which is wider than the shaded area in Zone II. Some bursts of data loss appear in this zone. As in Zone II, the TLR values are only slightly higher than the PLR values. But, in this zone, the TLR values are very high and this fact produces very low values for PSNR. The same behavior as that for Zone II is observed regarding the tiles per frame layouts: all of them show very similar performance. Therefore, we can conclude that when bursty losses occur, the til/frm layout does not have any influence on the PLR value, nor on the PSNR value.

- *With FEC protection*

Now we will analyze the performance of the FEC protection in **Zone II** and **Zone III** under these ideal conditions (without any background traffic load). Before that, it is useful to remember that the FEC protected bit streams are composed of two types of packets: (a) *protected source packets*, which include video data plus a 4-byte FEC trailer, and (b) *repair packets*, which may be used in the recovery process to recover the missing packets. In order to evaluate FEC recovery efficiency, the ARPLR (After-Recovery Packet Loss Ratio) metric measures the percentage of the source video packets missing over the total source video packets of the sequence, once the FEC recovery process has been carried out. In Table 5, the PLR, ARPLR, and TLR values for three of the encoding modes (AI, IPIP, and IPPP), protected with three levels (10%, 20%, and 30%), averaged over all the frame layouts, for the two mentioned zones, are shown. The rest of the encoding modes have shown a similar behavior so, for the sake of concision, only three of them have been included in the table.

Table 5. PLR, ARPLR, and TLR for FEC-protected bit streams under "ideal conditions" (0 pps) for zones II and III for AI, IPIP, and IPPP encoding modes for three levels of the protection level parameter (10%, 20%, and 30%) averaged over all the til/frm layouts.

	Zone II			Zone III		
	PLR	ARPLR	TLR	PLR	ARPLR	TLR
AI / 10%	2.00%	1.94%	2.03%	23.49%	29.05%	30.25%
AI / 20%	1.99%	1.78%	1.81%	23.94%	30.95%	32.37%
AI / 30%	2.01%	1.63%	1.67%	24.32%	28.62%	29.55%
IPIP / 10%	1.98%	1.89%	1.95%	23.08%	27.71%	28.22%
IPIP / 20%	2.00%	1.57%	1.62%	23.50%	28.05%	28.59%
IPIP / 30%	2.00%	1.42%	1.49%	24.05%	30.89%	31.61%
IPPP / 10%	2.00%	1.93%	1.98%	23.09%	26.49%	26.95%
IPPP / 20%	2.00%	1.78%	1.86%	23.55%	26.90%	27.45%
IPPP / 30%	2.00%	1.78%	1.89%	23.99%	28.43%	29.19%

In **Zone II**, the PLR values for the protected bit streams are very similar to those of the non-protected bit streams. The ARPLR values for this zone do not show good efficiency in the recovery of missing packets. In this zone, RaptorQ codes are not able to recover lost packets because of the bursty nature of the losses, so the ARPLR values are not much better than the PLR values, nor are they better than the PLR values of non-protected bit streams. RaptorQ codes have good recovery properties when dealing with isolated losses, but a minimum number of packets need to be received

for them to work properly, which is not the case when bursty losses are encountered. The recovery of missing packets from a burst of losses would need a very long protection window for any of the FEC techniques available (which would introduce a long delay) or else the incorporation of other complementary techniques like *interleaving*, which converts bursty losses into isolated losses, but which also introduces a non-negligible delay in the transmission process. So shaded areas (where no packet can be received) are not suitable for live video delivery and a readjustment of the RSUs location in order to eliminate these "black zones" may be the most appropriate action. In this zone, as happened with the non-protected bit streams, the ARPLR values do not either generate high TLR values.

In **Zone III**, the PLR is around 23%-24%. The bursty nature of losses in this zone makes FEC recovery unfeasible, as in Zone II. Furthermore, there is a striking fact in Zone III measurements: the ARPLR values are higher than the PLR values. How can this happen? The PLR metric takes into consideration the total number of packets in the FEC-encoded packet stream, which includes protected packets and repair packets. If losses are bursty and FEC recovery is not effective, only protected packets (which include video data) will be translated into source video packets (by just taking away the trailer). As repair packets may recover few (or none) source video packets they are not useful and the proportion of missing source video packets over the original number of source video packets may produce a higher value for the ARPLR than for the PLR. As a consequence of the high TLR values in this zone, the reconstructed video sequences can be considered useless. A possible solution to the problem found at overlapping areas resides in using different service channels for each one of the overlapping RSUs and incorporating a horizontal handover mechanism to guarantee a nearly uniform coverage area.

5.2.2. Background Traffic Conditions

Now, we present the results of the tests where a moderate background traffic load (62 pps/512 bytes) and a high background traffic load (125 pps/512 bytes) have been used.

- Without FEC protection

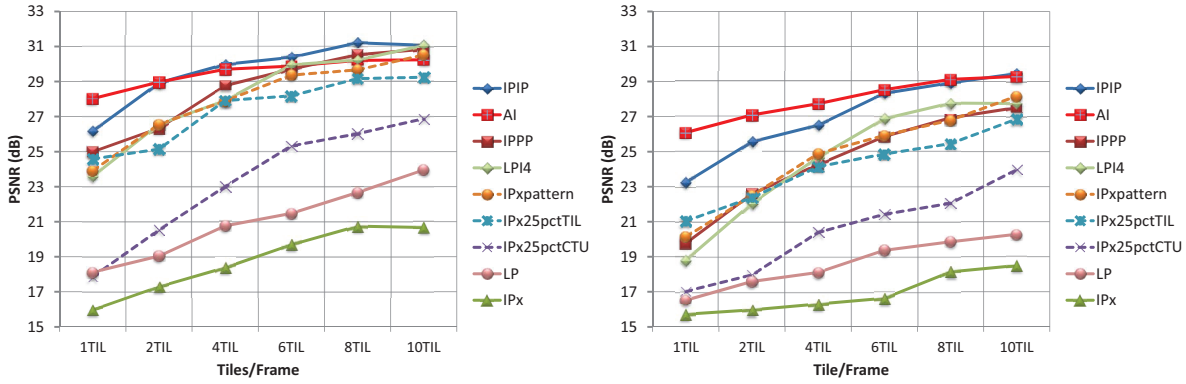
In Table 6, the PLR and TLR values for the bit streams without FEC protection, averaged for all the encoding modes in the three zones under consideration using both traffic loads are shown. It can be generally observed that, as the number of tiles per frame increases, the PLR value diminishes, and again, the poor performance of layouts with a low number of tiles per frame appears.

In **Zone I**, the PLR obtained is exclusively due to background traffic (for this zone in the absence of background traffic, a PLR value of 0% is obtained). Only for the three higher numbers of tiles per frame (6, 8, and 10) and the 62 pps background network load conditions the TLR values keep under reasonable limits. The quality of video sequences for this zone and 62 pps is depicted in Figure 12(a). The LP, IPx, and IPx25pctCTU encoding modes show poor performance, in line with the previous tests. As it was verified, 25% of intra refreshed CTUs does not contribute enough to make video robust. The IPIP encoding mode turns out to be the most effective (for a high number of tiles per frame). For dense traffic conditions, 125 pps (Figure 12(b)), nearly all of the PSNR values for the different modes and til/frm layouts are under 29 dB and are considered of very low quality. So, for dense traffic conditions, FEC protection becomes mandatory.

In **Zone II** and **Zone III**, the PLR value results from the combination of isolated packet losses and bursty packet losses. The high TLR values for these two zones clearly show that all the bit streams received in these two zones are completely useless, since the received packets produce very low quality video sequences.

Table 6. PLR and TLR for non-protected bit streams under 62 pps and 125 pps background traffic conditions for zones I, II, and III for every til/frm layout averaged over all the encoding modes.

62 pps/ 512 B	Zone I		Zone II		Zone III	
	PLR	TLR	PLR	TLR	PLR	TLR
1 til/frm	7.82%	28.10%	16.87%	40.21%	27.32%	43.55%
2 til/frm	6.80%	14.56%	15.42%	26.06%	26.60%	34.14%
4 til/frm	5.49%	7.44%	13.77%	17.16%	25.65%	28.04%
6 til/frm	4.43%	5.13%	12.15%	13.49%	24.72%	25.33%
8 til/frm	3.73%	4.00%	10.99%	11.68%	24.01%	24.22%
10 til/frm	3.26%	3.42%	9.94%	10.34%	23.56%	23.61%
125 pps/ 512 B	**Zone I**		**Zone II**		**Zone III**	
	PLR	TLR	PLR	TLR	PLR	TLR
1 til/frm	14.32%	44.63%	24.89%	54.02%	31.26%	54.17%
2 til/frm	14.01%	28.43%	24.71%	39.85%	30.78%	42.24%
4 til/frm	12.68%	16.88%	24.03%	29.37%	29.69%	33.40%
6 til/frm	10.69%	12.35%	21.75%	23.92%	27.65%	28.84%
8 til/frm	8.82%	9.52%	19.52%	20.53%	26.68%	27.16%
10 til/frm	7.43%	7.79%	17.80%	18.35%	25.82%	25.97%



(a) *62 pps* (b) *125 pps*

Figure 12. PSNR for every til/frm layout and every encoding mode. (Zone I; without FEC protection; 62 pps and 125 pps background traffic conditions).

- *With FEC protection*

Regarding the FEC protected bit streams, in Table 7, the performance of RaptorQ codes in **Zone I** for a background traffic of 62 pps and 125 pps for different protection levels (10%, 20%, and 30%) and different tiles per frame layouts averaged over all the encoding modes is shown.

In **Zone I**, under moderate traffic conditions (62 pps), a protection level of 30% gets almost 100% of packet recovery for all the layouts of tiles per frame (only at 1 tile per frame, a very low ARPLR value of 0.02%, which produces a residual TLR of 0.05% remains). Encoding a video sequence at 10 tiles per frame produces a bit stream around 9%-12% larger than encoding it at 1 tile per frame. If we protect the bit stream with a protection level of 30%, the tile-layout overhead can be avoided in Zone I under these precise conditions. Also, a protection level of 20% nearly completely recovers all the missing source video packets, so the layouts with medium and low numbers of tiles per frame can be used, avoiding extra overhead and still obtaining the best video quality. If a protection level of 20% or 30% is used, then the most efficient encoding mode (LP) provides the best quality, and, therefore, it is the best candidate to be used under these network conditions. In this zone (and also in Zone II), as the number of tiles per frame increases, the PLR value decreases. Note that, for a certain til/frm layout,

Table 7. PLR, ARPLR, and TLR for FEC-protected bit streams under 62 pps and 125 pps background traffic conditions for Zone I for every til/frm layout for three levels of the protection level parameter (10%, 20%, and 30%) averaged over all the encoding modes.

Zone I	62 pps			125 pps		
	PLR	ARPLR	TLR	PLR	ARPLR	TLR
1 til / 10%	6.96%	3.56%	11.80%	14.19%	12.68%	38.56%
1 til / 20%	6.89%	0.15%	0.40%	14.55%	6.06%	16.34%
1 til / 30%	6.31%	0.02%	0.05%	14.22%	0.56%	1.62%
2 til / 10%	6.44%	2.36%	4.97%	13.99%	12.78%	25.38%
2 til / 20%	6.34%	0.16%	0.26%	14.36%	6.31%	12.13%
2 til / 30%	6.05%	0.00%	0.00%	13.78%	0.75%	1.33%
4 til / 10%	5.47%	1.37%	1.85%	12.81%	10.89%	14.80%
4 til / 20%	5.17%	0.02%	0.03%	12.07%	2.68%	3.54%
4 til / 30%	5.10%	0.00%	0.00%	11.90%	0.25%	0.30%
6 til / 10%	4.57%	0.64%	0.72%	10.58%	7.26%	8.40%
6 til / 20%	4.49%	0.00%	0.00%	10.21%	0.80%	0.89%
6 til / 30%	4.30%	0.00%	0.00%	10.50%	0.00%	0.00%
8 til / 10%	3.77%	0.15%	0.16%	9.03%	5.12%	5.56%
8 til / 20%	3.76%	0.04%	0.04%	8.50%	0.20%	0.23%
8 til / 30%	3.70%	0.00%	0.00%	8.79%	0.00%	0.00%
10 til / 10%	3.20%	0.08%	0.08%	7.68%	3.27%	3.47%
10 til / 20%	3.17%	0.06%	0.06%	7.71%	0.18%	0.19%
10 til / 30%	3.11%	0.00%	0.00%	7.58%	0.14%	0.14%

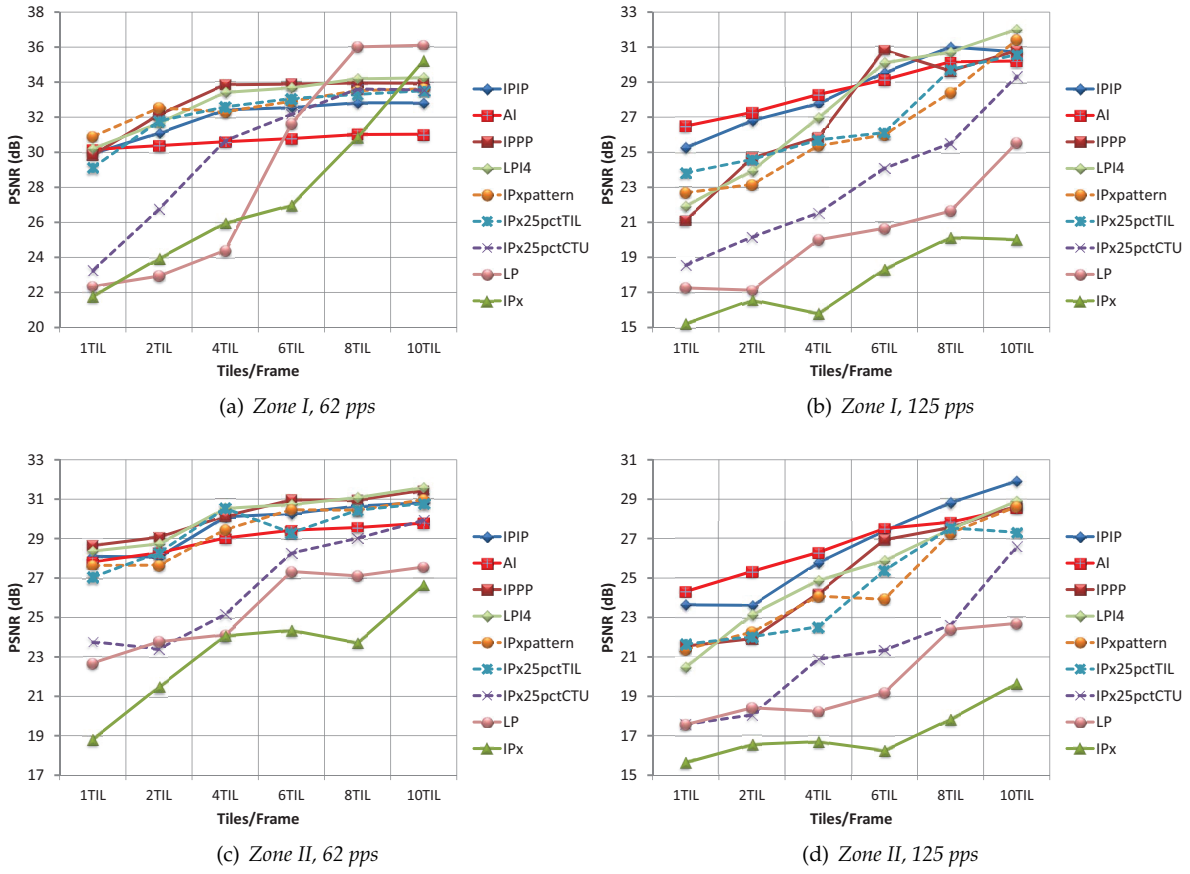
the three different levels of FEC protection produce very similar PLR values (even more, when the protection level increases, the PLR value slightly decreases).

In the experiments where a more dense background traffic is used (125 packets per second of 512 bytes) the PLR values obviously increase. Under these conditions, a protection level of 30% can restore almost all the missing packets for every layout, and a protection level of 20% also obtains good ARPLR values, mainly for layouts with a high number of tiles per frame. If Table 7 is compared with Table 6 (125 pps; bit streams without FEC protection), it is observed that the PLR values are very similar. This means that the added FEC protection does not have an adverse effect in the PLR values obtained.

Regarding the encoding modes performance, in Figure 13, the PSNR values for every encoding mode and every til/frm layout, with a protection level of 10% are shown. In Figure 13(a), with moderate traffic conditions in Zone I, the plotted curves recommend the use of the IPPP and LPI4 encoding modes for the 4 and 6 til/frame layouts, and the use of the LP encoding mode for the 8 and 10 til/frm layouts. For the 1 and 2 til/frm layouts, the IPxpattern encoding mode shows the best PSNR values. By comparing Figure 13(a) and Figure 12(a), the benefit of using RaptorQ codes against isolated losses, even for a moderate protection level of 10%, is proven.

In Figure 13(b), the PSNR values for Zone I and a background traffic of 125 pps are shown. The best quality is obtained for a layout of 10 til/frm and the LPI4 and IPxpattern encoding modes. For the 62 pps traffic conditions and a protection level of 20% (Table 7), the recommendation was to use a low number of tiles per frame layout (to reduce overhead) and the LP encoding mode (which provided the best PSNR value). If the LP encoding mode and a layout with low number of tiles per frame (1, 2, or 4 til/frm) are selected and the network conditions change into 125 pps, the TLR values move away from 0% (3.54% - 16.34%), and PSNR values are low. Therefore, as network conditions in a vehicular environment are continuously changing, an encoding mode, such as LP, which is so sensitive to packet loss (even for low TLR), is always a risky choice (unless 0% TLR could be guaranteed by high protection levels).

For **Zone II**, the PLR, ARPLR, and TLR values for background traffic conditions of 62 pps and 125 pps are higher than those in Zone I. By correlating TLR values and PSNR of many tests, we have



(a) *Zone I, 62 pps*

(b) *Zone I, 125 pps*

(c) *Zone II, 62 pps*

(d) *Zone II, 125 pps*

Figure 13. PSNR for every til/frm layout and every encoding mode. (FEC protection level of 10%).

observed that for the four most robust modes (IPPP, LPI4, IPIP, and AI), TLR values under 15% may produce PSNR values over 29 dB. Taking this observation into consideration, we can state that for moderate network load conditions (62 pps), the 10% protection level is considered enough protection for layouts equal or greater than 4 tiles per frame, as can be seen in Figure 13(c). We can also conclude that, for tight conditions (125 pps), the combination of a high number of tiles per frame (6, 8, 10) with high a level of protection (30%), may also provide some robustness, even for this hard-to-manage zone, as can be seen in Figure 13(d).

6. Conclusions and Future Work

From all the results obtained in the tests and simulations, several findings can be highlighted.

Vehicular scenario. Under "ideal conditions" (no background traffic) in areas with good coverage of only one RSU, all the network packets arrive to the video clients, so the video quality is directly determined by the efficiency of the encoding mode used. In shaded zones where the signal does not reach, and in zones with overlapping signals, bursty losses appear. Under background traffic conditions, even for moderate traffic, isolated packet losses appear for areas with good signal coverage, mainly due to collisions and the WAVE multichannel operations functioning.

Frame layout. For isolated losses, the selected layout has a direct influence on the PLR and TLR values. As the number of tiles per frame increases (a) the PLR value decreases and (b) for the same PLR value, a lower TLR value is obtained. When losses are bursty in nature the layout has little influence because (a) the PLR is very similar for all the til/frm layouts and (b) the TLR value is in line with the PLR value because missing packets in a burst probably belong to the same tile(s). For moderate network load conditions the 4 and 6 til/frm layouts have in general a good performance, and for tight

network conditions a high number of tiles per frame (8, 10) are needed to provide robustness to the video bit stream.

Encoding modes. The encoding modes that are inherently sensitive to errors (LP, IPx) provide the best performance in the total absence of losses, but rapidly worsen in the presence of losses, even for low TLR values. The encoding modes that have an implicit error resilience (AI, IPIP) are at a disadvantage because they start off with low quality values, therefore they only obtain better PSNR values for pronounced losses, and, in those situations, the final quality is very near (or even under) the acceptable minimum quality. The "combined" encoding modes take advantage of their characteristics to provide the best quality in the majority of cases when packet losses appear. From all the "mixed" encoding modes proposed, the IPPP and LPI4 encoding modes show the best performance in most of the situations. The IPxpattern encoding mode (and, in some ocassions, the IPx25pctTIL encoding mode) offers decent results. The IPx25pctCTU encoding mode, in spite of using intra refresh in the same proportion as the IPxpattern and IPx25pctTIL encoding modes, does not provide the expected performance; therefore, it is rejected as a protection mechanism.

RaptorQ codes. RaptorQ codes have good performance when dealing with isolated losses, but they do not solve the problem of bursty losses. In those situations the use of other mechanisms, such as *interleaving*, may be used to convert bursty losses into isolated losses. Even though, FEC channel coding is necessary in video streaming over vehicular networks because the protection provided by source coding is not enough to guarantee video delivery. The combination of the two protection approaches (RaptorQ codes and source coding) offers good performance: when RaptorQ codes are not able to recover all the missing packets, the source coding protection mitigates the propagation and multiplication effect of errors.

Real-time video streaming. In order to select a suitable setup for the real-time delivering of video in urban vehicular networks, we should recommend the use of an encoding mode which has both good compression efficiency while providing a certain level of robustness. To this extent, the LPI4 mode is the best positioned mode. When it comes to select a til/frm layout, the combination of low overhead, good error resilient features, and a low avalanche effect (when translating PLR into the resulting TLR), mid values, like 4 and 6 til/frm are the best tradeoff. At last, if we are protecting video streams delivered under full coverage areas, a protection level of 10% (in combination with a good encoding mode and til/frm layout), may provide the required robustness against isolated losses, to guarantee an appropriate user experience.

Future work. The two main research efforts in which we are targeting our future work are: (a) the application of Quality of Service methods to the prioritization of data packets through the vehicular network, by means of 802.11p priority queues, in order to provide more robustness to those video packets which are more relevant in the final quality of the reconstructed video, and, (b) the introduction of feedback mechanisms in the network to measure the real time packet loss ratio and network delay, in order to create an adaptive version of the protection mechanisms presented in this work.

Author Contributions: Conceptualization, P.P., O.L.-G. and M.P.M.; Methodology, P.P., M.M.-R. and P.G.; Software, P.P.; Investigation, P.P., M.M.-R. and P.G.; Writing—Original Draft Preparation, P.P., M.M.-R., P.G., O.L.-G. and M.P.M.; Writing—Review & Editing, P.P., M.M.-R., P.G., O.L.-G. and M.P.M.; Project Administration, M.P.M. and O.L.-G.; Funding Acquisition, M.P.M. and O.L.-G.

Funding: This research was supported by the Spanish Ministry of Economy and Competitiveness under Grant TIN2015-66972-C5-4-R, co-financed by FEDER funds (MINECO/FEDER/UE).

Conflicts of Interest: The authors declare no conflict of interest.

1. Schoch, E.; Kargl, F.; Weber, M.; Leinmuller, T. Communication Patterns in VANETs. *Communications Magazine, IEEE* **2008**, *46*, 119–125. doi:10.1109/MCOM.2008.4689254.

2. Bross, B.; Han, W.J.; Ohm, J.R.; Sullivan, G.J.; Wang, Y.K.; Wiegand, T. High Efficiency Video Coding (HEVC) text specification draft 10. Technical Report JCTVC-L1003, Joint Collaborative Team on Video Coding (JCT-VC), Geneva (Switzerland), 2013.
3. Luby, M.; Shokrollahi, A.; Watson, M.; Stockhammer, T.; Minder, L. RaptorQ Forward Error Correction Scheme for Object Delivery. IETF RMT Working Group, Work in Progress. RFC 6330, 2011.
4. Oztas, B.; Pourazad, M.; Nasiopoulos, P.; Leung, V. A Study on the HEVC Performance over Lossy Networks. Electronics, Circuits and Systems (ICECS), 2012 19th IEEE International Conference on, 2012, pp. 785–788. doi:10.1109/ICECS.2012.6463542.
5. ITU-T.; ISO/IEC JTC 1. Advanced Video Coding for Generic Audiovisual Services. *ITU-T Rec. H.264 and ISO/IEC 14496-10 (AVC) version 16* **2012**.
6. Wenger, S. NAL Unit Loss Software. Technical Report JCTVC-H0072, Joint Collaborative Team on Video Coding (JCT-VC), San Jose, 2012.
7. Psannis, K.E. HEVC in wireless environments. *Journal of Real-Time Image Processing* **2015**, pp. 1–8. doi:10.1007/s11554-015-0514-6.
8. Liu, H.; Zhang, W.; Yang, X. Error-resilience Packet Scheduling for Low Bit-rate Video Streaming over Wireless Channels. Proceedings of the 2006 IEEE International Symposium on Circuits and Systems. ISCAS 2006., 2006, pp. 1 – 4. doi:10.1109/ISCAS.2006.1692622.
9. Nightingale, J.; Wang, Q.; Grecos, C. HEVStream: A Framework for Streaming and Evaluation of High Efficiency Video Coding (HEVC) Content in Loss-prone Networks. *IEEE Transactions on Consumer Electronics* **2012**, *58*, 404–412. doi:10.1109/TCE.2012.6227440.
10. Vineeth, N.; Guruprasad, H. A Survey on the Techniques Enhancing Video Streaming in VANETs. *International Journal of Computer Networking, Wireless and Mobile Communications* **2013**, *3*, 37–46.
11. Qadri, N.; Altaf, M.; Fleury, M.; Ghanbari, M. Robust Video Communication over an Urban VANET. *Mobile Information Systems* **2010**, *6*, 259–280.
12. Torres, A.; Calafate, C.T.; Cano, J.C.; Manzoni, P.; Ji, Y. Evaluation of flooding schemes for real-time video transmission in VANETs. *Ad Hoc Networks* **2015**, *24*, 3–20. doi:10.1016/j.adhoc.2014.07.030.
13. Mammeri, A.; Boukerche, A.; Fang, Z. Video Streaming Over Vehicular Ad Hoc Networks Using Erasure Coding. *IEEE Systems Journal* **2016**, *10*, 785–796. doi:10.1109/JSYST.2015.2455813.
14. Xie, H.; Boukerche, A.; Loureiro, A.A.F. MERVS: A Novel Multichannel Error Recovery Video Streaming Protocol for Vehicle Ad Hoc Networks. *IEEE Transactions on Vehicular Technology* **2016**, *65*, 923–935. doi:10.1109/TVT.2015.2397862.
15. Immich, R.; Cerqueira, E.; Curado, M. Shielding video streaming against packet losses over VANETs. *Wireless Networks* **2016**, *22*, 2563–2577. doi:10.1007/s11276-015-1112-z.
16. Immich, R.; Cerqueira, E.; Curado, M. Efficient high-resolution video delivery over VANETs. *Wireless Networks* **2018**. doi:10.1007/s11276-018-1687-2.
17. Pasin, M.; Petracca, M.; Bucciol, P.; Servetti, A.; De Martin, J. Error Resilient Real-Time Multimedia Streaming over Vehicular Networks. DSP Workshop for In-Vehicle Systems and Safety; , 2009.
18. Calafate, C.T.; Fortino, G.; Fritsch, S.; Monteiro, J.; Cano, J.C.; Manzoni, P. An Efficient and Robust Content Delivery Solution for IEEE 802.11p Vehicular Environments. *Journal of Network and Computer Applications* **2012**, *35*, 753 – 762. doi:10.1016/j.jnca.2011.11.008.
19. Bossen, F. Common Test Conditions and Software Reference Configurations. Technical Report JCTVC-L1100, Joint Collaborative Team on Video Coding (JCT-VC), Geneva (Switzerland), 2013.
20. HM Reference Software vers. 9.0, https://hevc.hhi.fraunhofer.de/svn/ svn_HEVCSoftware/tags/HM-9.0.
21. ntegaard, G.B. Improvements of the BD-PSNR model. Technical Report VCEG-M33, Video Coding Experts Group (VCEG), Berlin (Germany), 2008.
22. Girod, B.; Farber, N. Error-Resilient Standard-Compliant Video Coding. In *Signal Recovery Techniques for Image and Video Compression and Transmission*; Springer US, 1998; pp. 175–197.
23. Bandyopadhyay, S.; Wu, Z.; Pandit, P.; Boyce, J. An Error Concealment Scheme for Entire Frame Losses for H.264/AVC. Sarnoff Symposium, 2006 IEEE, 2006, pp. 1–4. doi:10.1109/SARNOF.2006.4534755.
24. Qualcomm (R) RaptorQ (TM) Evaluation Kit, http://www.qualcomm.com.
25. Krajzewicz, D.; Erdmann, J.; Behrisch, M.; Bieker, L. Recent Development and Applications of SUMO - Simulation of Urban MObility. *International Journal On Advances in Systems and Measurements* **2012**, *5*, 128–138.

26. OMNeT++ Discrete Event Simulator, http://www.omnetpp.org.
27. Sommer, C.; German, R.; Dressler, F. Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis. *IEEE Transactions on Mobile Computing* **2011**, *10*, 3–15. doi:10.1109/TMC.2010.133.
28. MiXiM - Mixed Simulator, http://mixim.sourceforge.net.
29. TraCI - Traffic Control Interface, http://sourceforge.net/apps/mediawiki/ sumo/index.php?title=TraCI.
30. OpenStreetMap, http://wwww.openstreetmap.org.