Event-based Vision: Understanding Network Traffic Characteristics

Giulia Attanasio*†, Claudio Fiandrino* and Joerg Widmer*
*IMDEA Networks Institute, Madrid, Spain
†Universidad Carlos III de Madrid, Spain
Email: {giulia.attanasio, claudio.fiandrino, joerg.widmer}@imdea.org

***Abstract*—Event-based vision fosters a new way of sensing reality. Event-based cameras work radically differently compared to legacy frame-based cameras because they continuously measure brightness changes at a per-pixel granularity (i.e., events) rather than snapshots of intensity measurements (i.e., frames). Event-based cameras are applied in robotics and augmented and virtual reality applications due to their properties of low-latency, high temporal resolution and dynamic range. For example, they greatly improve unmanned aerial vehicle (UAV) navigation and collision avoidance. While event-based vision is currently restricted to local devices, in the near future applications involving distributed systems will gain momentum, such as the coordination of swarms of UAVs or robots. However, the network traffic characteristics of event-based vision systems are largely unexplored. In this paper, we aim to fill this gap by providing the first study of network traffic generated by event-based cameras. To this end, we employ publicly available data sets and experimentally study properties like the impact of packet/event losses on typical computer vision operations like tracking, and the implications of medium access under contention. We find that complex scenes that incur a high event generation rate are more robust against packet loss due to transmission errors or wireless contention. Conversely, packet loss or delay are more harmful to tracking and visualization operations when the event generation rate is small.**

I. Introduction

Video is a very popular information source for applications in the field of pervasive computing. Such applications stem from human activity recognition [2] (e.g., eye fatigue detection [3], or eldery fall detection [4], [5]) to Virtual/Augmented reality (AR/VR) [6]. Processing video images is common practice with deep neural networks, that are nowadays the foundation for computer-vision operations like object detection, recognition and tracking thanks to their ability of extracting high-level features from a scene [7].

In AR/VR applications, tasks like object detection and recognition over one video frame can last more than 1 s if performed on mobile devices with TensorFlow Lite and convolutional neural networks [8]. Frame skipping and preemption, i.e., interrupting the transmission of one frame in favor of another with more significant content, are other techniques used to reduce the amount of information to be transmitted and processed in order to achieve average glass-to-glass latencies of 21.2 ms. However, even such figures do not meet the very strict latency requirements of future tactile internet applications. For tactile internet, fifth generation (5G) mobile networks include specific ultra reliable and low latency communications (URLLC) [9]. The Third Generation Partnership Project (3GPP) has defined a target user plane latency of 1 ms for both uplink and downlink transmissions, and a reliability requirement of 99.999% to transmit a 32 Byte packet. Current networks are still unable to meet the 1 ms round-trip time (RTT) and besides network specific components, also processing contributes significantly to the latency budget. In a sensor-to-actuator control loop, 0.65 ms are spent by the sensor, wireless and wired communication, leaving only 0.35 ms for processing of data in a local multi-access edge computing (MEC) environment [10]. Video processing imposes a non-negligible time cost because of the intrinsic nature of the video creation. Conventional video tools capture a number of snapshots over fixed periods of times (i.e., the frames) and this generates redundant information, motion blur and the frames are subject to the lighting conditions of the recording. In turn, to extract meaningful information and prior to performing any feature extraction for detection and recognition, the video should undergo additional processing operations to, e.g., improve poor image quality when recording during moonlight [11] or reconstruct the blur-free latent images [12]. All this additional processing negatively affects latency.

In the last years, a new vision paradigm has emerged and gained momentum. Event-based cameras measure brightness[1] changes at a per-pixel granularity (i.e., events) rather than intensity measurements (i.e., frames) [14]. The high temporal resolution and low latency (events are transmitted once detected, at µs resolution, much lower than the legacy frame-based camera exposure time), and the high dynamic range (> 120 dB, whereas high quality frame-based cameras achieve 60 dB) have made them the candidate instrument for robotic applications where real-time interaction is crucial, e.g., drone racing [15]. Other applications involve object tracking, recognition, gesture control, monitoring and surveillance [13].

Event-based cameras have attracted interest of both industry and academia. Recently, Prophesee has launched the first event-based sensor for IoT devices for Industry 4.0[2]. This is a

This work is based on the M.Sc. final work of Giulia Attanasio during her internship at IMDEA Networks Institute while being enrolled at Politecnico di Torino [1].

[1]Similarly to [13], we denote as *brightness* the log level intensity of light.
[2]Available at: https://www.prophesee.ai/event-based-sensor-packaged/

promising news as mass production will enable commercial-scale adoption of the event-based vision paradigm. Prof. Davide Scaramuzza, who pioneered the development of event-based vision since the very beginning, has been recently awarded with a ERC Consolidator grant for his project "Agile Flight: Low-latency Perception and Action for Agile Vision-based Flight" that fosters autonomous UAV flight through event-based vision.[3] To this date, event-based cameras have always been employed on board or in close proximity to the device performing actuation, and streaming such events remotely is largely unexplored.

In this paper, we conduct an empirical study to characterize network traffic generated by event-based vision systems with the objective to identify properties that can help overall system performance. Specifically, this study sheds light on the nature of the traffic itself in relation with the complexity of the scene and the implications of losses and medium access on the performance of computer vision operations like object detection and tracking. To quantify such implications, besides network-related indicators, we further evaluate well known image quality metrics. To the best of our knowledge, we are the first to perform such a measurement study and we believe it provides the research community with a missing piece of the puzzle to foster the adoption of event-based cameras in distributed systems.

In summary, this paper makes the following contributions:

- We study the characteristics of event-based camera datasets made available by the computer vision research community and we identify the relation between the complexity of the scene and, in turn, the event generation and packetization processes.
- We study the properties of network traffic when a single pair of transmitter-receiver 802.11ac stations is transmitting event-based camera traffic. This uncovers the impact of network losses on the performance of computer vision operations such as object tracking.
- We perform measurements with an extensive set of background traffic distributions (originated by the network traffic generator D-IGT [16] and by the corporate network of IMDEA Networks) to analyze the behavior of event-based camera traffic in the presence of medium access contention. This allows to study the combined effect of packet loss and contention on the performance of remote computer vision operations.

The rest of the paper is structured as follows. Section II provides background on event-based vision systems. Section III presents the evaluation framework by illustrating the characteristics of the datasets, the software, and the testbed employed for evaluation. Section IV is the core of our work and illustrates the insights obtained from the measurement analysis. Finally, Section V concludes the work.

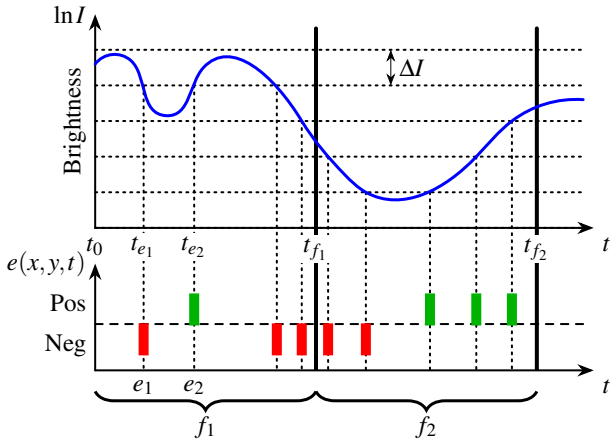


Fig. 1. Event and frame generation. When the brightness crosses a threshold ΔI, a "positive" (Pos) or "negative" (neg) event is triggered events (in green and red respectively), while frames f_1 and f_2 are absolute brightness measurements generated through integration over a fixed time period

II. Background on Event-Based Vision

Legacy vision systems rely on frame-based cameras whose sensors provide *an absolute and direct* brightness measurement per pixel at a low and constant rate (e.g., 30 fps). Conversely, event-based cameras measure the *change of brightness*, or temporal contrast, per pixel. This paradigm is based on Dynamic Vision Sensors (DVSs) that employ a bio-inspired approach to emulate the behavior of the retina [14]. Specifically, the "silicon-retina" of DVSs registers events or spikes, i.e., positive/negative log intensity changes measured at a pixel whenever a threshold is crossed. These events are recorded and transmitted from the pixel array of the sensor in the form of an address-event representation (AER) [17]. Fig. 1 exemplifies the different operational principles behind event- and frame-based cameras. By denoting as $\ln I$ the brightness[4] and ΔI the threshold, the upper plot shows the function mapping the changes of brightness at the pixel p with coordinates (x, y). The lower part of the plot shows the output, i.e., events and frames. Consider now two measurements at t_{e_1} and t_{e_2}, and $E = \ln I|_{t_{e_1}} / \ln I|_{t_{e_2}}$. When

$$\begin{cases} E > \Delta I & \text{the event is "positive";} \\ E < \Delta I & \text{the event is "negative".} \end{cases} \quad (1)$$

Such an approach allows to significantly reduce information redundancy and the latency between photodiode illumination change detection and its off-chip transmission down to ≈ 15 µs. By contrast, the frame-based approach imposes a (slower) fixed integration time to produce one frame, e.g., 33.3 ms for a video recorded at 30 fps resolution. This is shown in Fig. 1 as the periods $t_0 - t_{f_1}$ and $t_{f_1} - t_{f_2}$ which result in frames f_1 and f_2 respectively. As a side effect of triggering events rather than

[3]Available at: https://www.eenewseurope.com/news/nccr-robotics-lab-gets-eu-2-million-european-grant-autonomous-drones

[4]According to [13], "Brightness is a perceived quantity" and the log intensity is typically employed allow to distinguish well uniformly-lighted scenes.

TABLE I
COMPARISON FEATURES EVENT- AND FRAME-BASED CAMERAS

FEATURE	FRAME-BASED	EVENT-BASED
Resource Intensive	Yes	No
Motion Blur	Yes	No
Dynamic Range	Low	High
Latency	High	Low
Information Redundancy	Yes	No

of frames, DVSs achieve much higher dynamic range (in the order 120 dB) that significantly exceeds the one of high-quality frame-based cameras (in the order of 60 dB) [13]. This allows to perform low light video recording, e.g., during moonlight.

Robotic systems have strict timing requirements in order to meet control-loop deadlines and operate in (or close to) real-time. This requires a legacy frame-based camera to operate at a high frame rate which, in turn, produces a large amount of data leaving only few instructions per pixel for processing [18] on current CPUs. In scenarios that involve drones or autonomous driving, it is important to continuously track the environment to estimate the position of a moving object (e.g., counting cars passing by a point of interest [19] or trajectory estimation for autonomous driving of UAVs [20]). In this case, legacy vision frame-based cameras at high frame rates lead to significant motion blur, which makes it harder for the algorithms to infer trajectories precisely.

Table I recaps the main differences between event-based and frame-based camera sensors.

III. THE EVALUATION FRAMEWORK

A. The Datasets

The computer vision research community has made available a number of datasets with recordings from event-based cameras. The datasets typically use the following formats:

- *text format*: provides one event per line in common `txt` files.
- *binary format*: records events in arrays that can be interpreted by the Robot Operating System (ROS).

In both cases, the information recorded as *event* consists of a timestamp (with µs accuracy), pixel position (row and columns) and polarity ("positive" and "negative"). The former format is more suitable for analysis with Matlab or Python, while the latter is used for example for the visualization framework jAER (see § III-B). We resort to the second option for our study.

The considered datasets were generated using different sensors, namely the DAVIS240C (resolution of 240×180 pixels, 120 dB dynamic range, up to 12 MEvents/second, dimensions in mm 56 (h) × 62 (w) × 28 (d)) and DAVIS346 (240×180 pixels, 120 dB dynamic range, 12 MEvents/second, dimensions 40 (h) × 60 (w) × 25 (d)) sensors from iniLabs[5] with 8 Byte event resolution. Scene complexity is an important factor in our choice of the datasets. As event-based cameras capture motion changes, the complexity of a scene is defined in terms of the number of moving objects, their speed and the (moving) background. We therefore select datasets with increasing level of scene complexity and size, which is function of the length of the video recording and the scene complexity itself. Fig. 2 shows all the datasets and depicts in green and red "positive" and "negative" events respectively. Specifically, we consider:

- The *butterfly* dataset (Fig. 2(a)) is a 6 s video recording with a butterfly moving in the vicinity of a flower. Overall it generates 225 876 events, which makes it the smallest dataset in the pool. It is also the one with lowest scene complexity as the velocity of the butterfly and flower motion is slow.
- The *shapes* dataset (Fig. 2(b)) is a 60 s long video with a total number of events 23 126 288. The event-based camera records with increasing levels of rotational motion a set of shapes fixed on a wall with white background.
- The *poster* dataset (Fig. 2(c)) is also a 60 s long video, but features a total number of events much higher than the above dataset (169 350 136). Like *shapes*, this dataset is is also recorded with increasing levels of rotational speed, but the heavy textured background triggers many more events than the white background wall.
- The *drone* dataset (Fig. 2(d)) contains a 73.2 s long video recorded by an event-based camera mounted on a drone flying at a maximum velocity of 12.78 m/s in a room. The scene accounts for a total number of events of 51 023 819. Unlike the previous datasets that were all generated with the DAVIS240C sensor, this dataset was produced with the DAVIS346B mounted on a quadrotor.
- The Multi Vehicle Stereo Event Camera (*drive*) dataset (Fig. 2(e)) comprises a number of measurements taken at day/night with different vehicles (car/motorbike). Specifically, we consider the Outdoor Driving Night 1 dataset which contains a 262 s video recorded by a car during night and accounts for a total of 101 178 636 events. This dataset was generated with the DAVIS346B sensor.

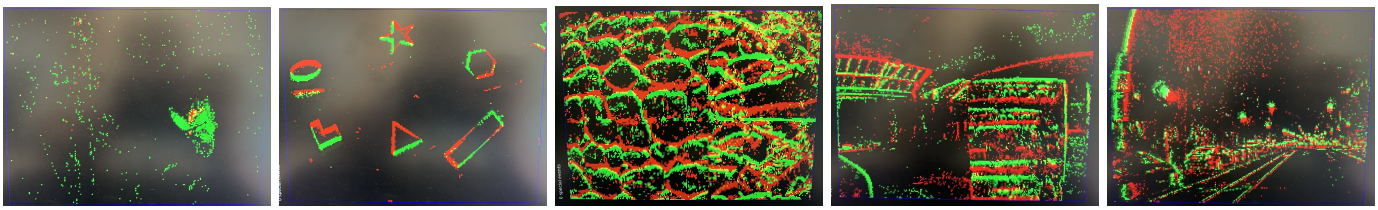
B. The Software

A number of software frameworks for event-based cameras are available [13]. The most popular and widely adopted is jAER, a Java-based framework[6] that allows to visualize and process recorded AER-compliant datasets or live recording through a set of pre-installed filters. In the latter case, event-based cameras can be connected via a USB interface and jAER reads the incoming stream of events. jAER, consists of around 300 classes, including a specific application called jAERViewer for the actual rendering, filters and visual annotations like cluster mass or identifier.

For remote communications, jAER employs a UDP socket. Fig. 3 shows the operational workflow. jAER implements a thread that reads events during a window W (for transmission or for display). At the transmitter side, during W, jAER counts the

[5]Available at: https://inilabs.com/

[6]Available online at: http://jaerproject.org



(a) Butterfly [21] (b) Shapes [22] (c) Poster [22], [23] (d) Indoor drone race [15] (e) Driving at night [24], [25]

Fig. 2. The employed datasets. The corresponding reference in the caption highlights the source of the image.

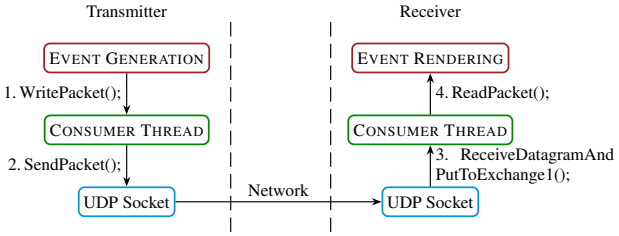


Fig. 3. The process of event streaming with jAER

number of events to determine the number of UDP packets to be passed to the network interface (*event generation*). These events are temporarily written into a buffer B that is implemented with an `ArrayBlockingQueue`, i.e., a bounded-buffer that ensures fairness with multi-threading by ordering producer and consumer threads. The maximum size of B is of 63 KBytes, slightly lower than the maximum UDP payload size. A specific *consumer tread* takes care of reading from B and sending the packets to the network interface and at the receiver side the reverse process takes place to extract events from the packets for playback.

C. The Testbed

Fig. 4 shows the testbed employed in the experiment and illustrates the equipment and the setting. Specifically, Fig. 4(a) shows the lab equipment, which in its basic setup includes two laptops for transmitting and receiving the event-based camera traffic and a WiFi Access Point (AP) that interconnects the two as per Fig. 4(b). For our study, we install jAER on both laptops that stream the flow of events. When studying the implication of accessing the medium with contention, we employ an additional laptop which generates background traffic. Specifically, the transmitter is a Dell Latitude E6430 with an Intel $i7-3720QM$ processor at 2.60 GHz, 16 GB RAM and Linux Ubuntu 18.04 LTS. The receiver is a Dell Latitude E7440 with an Intel $i7-4600U$ processor at 2.1 GHz, 8 GB RAM and Linux Ubuntu 16.04. Both are equipped with a 802.11ac compliant interface. The AP is a NETGEAR Nighthawk X10 R9000 supporting the 802.11ac protocol in the 5 GHz band (80 MHz wide channel).

For the network, we rely on 802.11ac because since its release in 2012 it became the dominant standard currently on the market, and only few devices equipped with the newer standard 802.11ax (WiFi 6) are available [26]. We choose not to perform experiments with LTE networks because their latencies are in the order of tens of milliseconds and are much higher than those achievable with 802.11ac. Furthermore, the latter operates in the unlicensed band and is easier to setup and operate. These are the main reasons for which nowadays most of factory floors rely on such communication technology [27], [28].

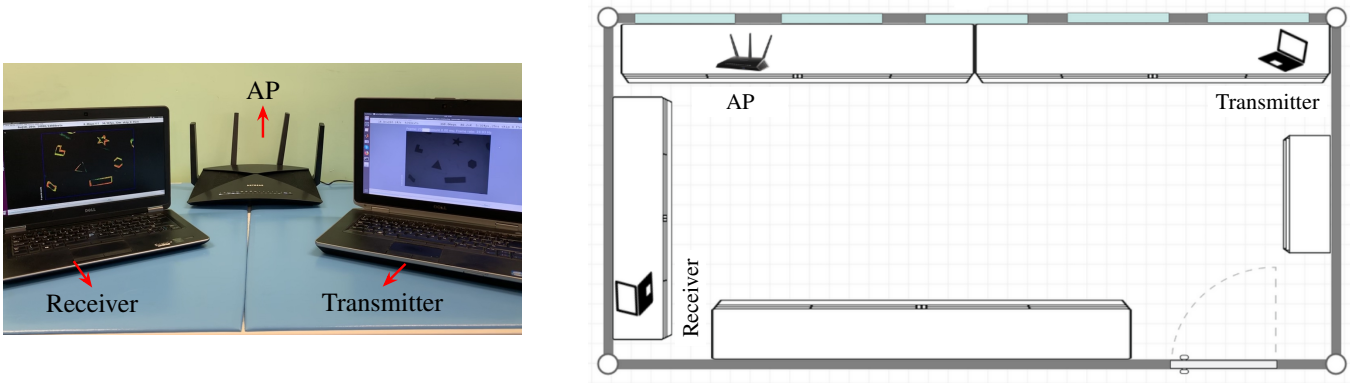
D. Evaluation Metrics

In the measurement study, we use a number of performance indicators that are specific to measure network performance and image quality. This section quickly overviews the former metrics that are well known to this community and mainly focuses on the latter metrics.

Concerning network-related metrics, we employ throughput to measure the amount of data transferred, packet inter-arrival times to determine the distribution of traffic at the receiver side and the retransmissions that correspond to the number of packets that the AP had to resend because they were not properly decoded at the receiver side.

Concerning image quality metrics, we resort to both reference-based and non-reference-based evaluation [29]. The former category features metrics that require comparison between a high-quality reference image and the image under evaluation. Metrics that fall into this category are Peak-Signal to Noise Ratio (PSNR) and Structural Similarity Index (SSIM) [30]. In non-reference-based scenarios where it is not possible to compare two images, image quality assessment employs algorithms that evaluate a single image at a time by first extracting relevant features and then finding patterns among the features. In this category we resort to the blind/referenceless image spatial quality evaluator (BRISQUE) [31]. Specifically:

- PSNR operates at per-pixel intensities and is expressed in dB as $10\log_{10}$ P/MSE where P is the maximum pixel value and MSE is the Mean Squared Error, i.e., averages of the squared difference of the pixel intensities of the reference image and the one under assessment. The PSNR metric can assume values up to infinite when the MSE approaches zero and this occurs when the images used for comparison are similar. Typical values are in the range of $20-50$ dB.
- SSIM is designed to evaluate the perceived change of structural information and assesses image distortion with respect to the reference image as a combination of loss of structural correlation, luminance and contrast distortion. The SSIM metrics can assume values in the range of $0:1$, where 1 implies a perfect structural similarity (i.e., the



(a) The equipment

(b) Lab setting

Fig. 4. The testbed setup for the experiments

images are equal), 0 no structural similarity and values below 0 difference.

- BRISQUE extracts the so-called natural scene statistics (NSS). It allows to measure the deviation of the distribution of pixel luminance of the image under assessment and that of the corresponding natural image for which the distribution is known to be always Gaussian. Typical values of the BRISQUE metric are in the range of $0-100$, where 0 is the best quality score and 100 is a soft upper bound that indicates a bad quality score.

IV. Network Traffic Characterization

This section presents the results of our study by first analyzing the characteristics of the datasets (§ IV-A) and then analyzing properties of the network traffic when only event-based camera traffic flows in the network (§ IV-B). Finally, we verify the findings obtained from tests in isolation to the more general case of a network with background traffic and we look specifically to the implications of accessing the medium under contention (§ IV-C).

A. Dataset Analysis

The objective of this subsection is to expose the correlation between scene complexity and the event generation process, which is essential to capture the dynamics of packet generation (see § IV-B). For the study, we analyze with Matlab the datasets *shapes*, *poster* and *drones* (see § III-A) that are in text file format, with *events.txt* containing one event <*timestamp, x , y, polarity*> per row.

Fig. 5 shows the event generation rate over time for the three datasets. Note that *shapes* and *poster* have the same recording time (60 s) while *drones* features a longer scene (72 s). For simplicity we cut the *drone* scene to 60 s as well. Recall that the *shapes* and *poster* datasets were recorded with increasing rotational motion (up to 730° and 884° for the *shapes* and the *poster* datasets respectively), but the magnitude of events generated by *poster* is significantly higher (by a factor of 10) because of the textured background. Thus, to provide a fair comparison of the event generation rate, we count the number of events in each second and normalize it with respect

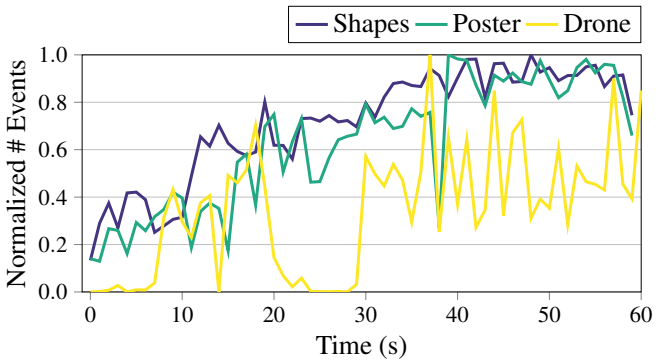


Fig. 5. Event generation rate over time for shapes and poster datasets

to the maximum. The plot confirms that the rate of event generation increases over time. Indeed, the increase of the rotational motion makes the scene much more dynamic and triggers many more events. We also note there is a drop in the rate near second 40 of the recording. As the recordings were captured by moving the event-camera with the hand [23], that drop should be attributed to the recording process (e.g., the recorder moved slowly the camera). Observe for comparison that the event rate generation of the *drones* dataset is radically different. For the initial part of the curve (before second 7) the rate is close to zero, which is attributed to the take off procedure and the moments in which the drone remains stable while being in-flight (e.g., around second 20 where the drone is hovering). In these moments, the mobility is almost zero and thus only a small number of events are captured.

B. Experiments without Background Traffic

The objective of this subsection is to characterize the nature of the event-based traffic, i.e., how the scene complexity and rate generation process determine throughput and inter-packet arrival times measured at the receiver side. We also evaluate the impact of losses on the performance of the object visualization. To this end, we modify jAER in order to be able to drop a certain fraction of events. Note that the objective of this section is preparatory to understand the more general scenario (see Section IV-C). Therefore, we perform these experiments with

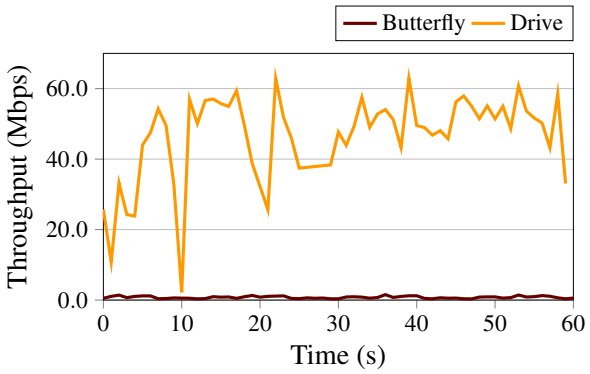


Fig. 6. Throughput comparison of datasets with radically different scene complexity

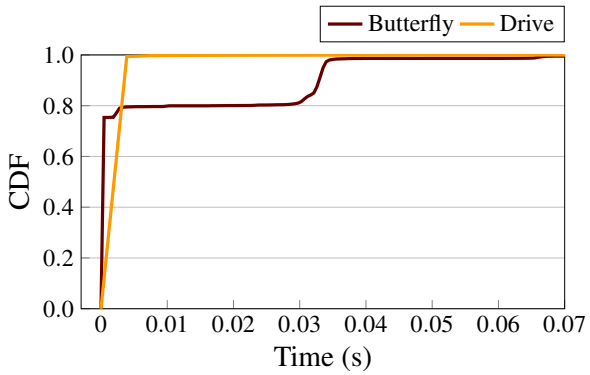


Fig. 7. Packet inter-arrival time

only event-based camera generated traffic and no other station is competing for the medium and transmitting background traffic in our private network.

1) The implications of scene complexity on network traffic: As shown in § IV-A, the complexity of a scene determines the rate at which events are generated. Next, we characterize how these events that are transformed into packets and streamed over our private network determine throughput and inter-packet arrival times.

During the preliminary analysis, we noted that jAER does not fragment UDP packets before transmission over the socket interface. Letting the IP layer deal with fragmentation is detrimental to the performance of the connection [32], since loosing one of the IP fragments makes the entire packet unrecoverable. Therefore, we modify jAER so that packet fragmentation (and re-assembly) occurs at the application layer and we set the maximum size of the payload that UDP carries to 1472 Bytes. This corresponds to a maximum of 184 events per packet.

Next, we measure throughput and packet inter-arrival times at the receiver side. Fig. 6 compares the observed throughput for datasets with highly different scene complexity, i.e., *butterfly* and *drive*. To make the two comparable given the short recording time of *butterfly*, we loop the recording of this dataset for a total duration of 60 s. Low complex scenes like butterfly generate a moderate throughput with peaks at 1.5 Mbps. In contrast, complex scenes like *drive* generate many more events and in turn traffic. Note that the throughput for *drive* exhibits a ramp-up phase when events start to accumulate at the transmitter while in the last period (from second 30 onward) the throughput is more stable. A slow down in the event generation rate results in a sudden drop in throughput (e.g., near second 10), where the moto driver is walking the moto out of the parking and suddenly moves slower and almost stops. In turn, the scene does not trigger further events until the driver resumes the movement.

Fig. 7 shows the packet inter-arrival time of the *butterfly* and *drive* datasets. Scenes with intermediate or high scene complexity generate events in such a way that the distribution of gaps between consecutive packets have little variation. This does not hold for low-complexity scenes like *butterfly*.

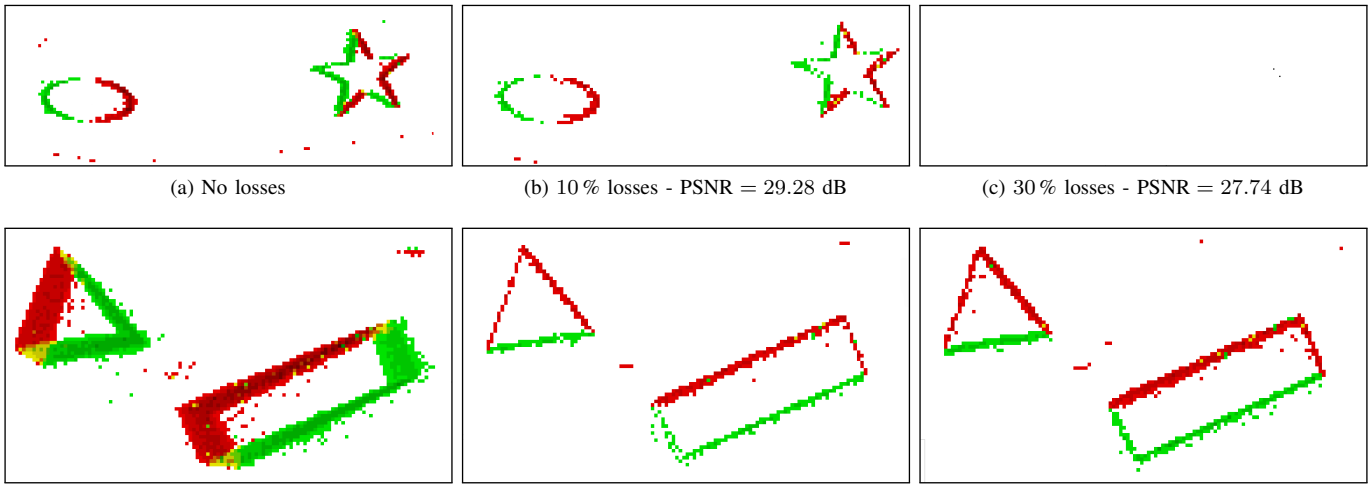
2) The impact of event loss on object visualization and tracking: The objective of the next set of experiments is to understand how the object visualization and tracking vary when some events are lost within a controlled scenario. To this end, we modify jAER at the transmitter side so that a fraction of events is dropped.

Fig. 8 shows for the *shapes* dataset the impact of different percentages of losses on the scene visualization, namely 10 % and 30 %. Recalling that the event generation rate for this dataset increases over time (see Fig. 5), we select two different time instants, i.e., second 1 and 40 of the recording when the normalized event generation rate is respectively around 20 % and 85 %. We compute PSNR and SSIM metrics by considering as reference images (Fig. 8(a)) and (Fig. 8(d)).

When the event generation is low (Fig. 8(a)), the loss of 10 % of events reduces the quality of the visualization, but the tracking algorithm still has a sufficient number of events that produce a cluster that allows to identify and follow the object (see Fig. 8(b)). With the loss of 30 % of events, this becomes impossible (see Fig. 8(c)). Note that a change from 10 % to 30 % of lost events generates a loss of image quality perceived with a PSNR reduction of more than 1.5 dB. We do not notice the same behavior during second 40 of the recording (Fig. 8(d)) and tracking remains possible with either 10 % or 30 % of event losses because the number of events received is sufficient to form a clear cluster that the tracking algorithm can identify (the PSNR metric actually improves by 0.18 dB with 30 % of event losses). Table II evaluates the BRISQUE metric for the six figures shown in Fig. 8. The values BRISQUE metric are high, which indicates that the level of blur and noise of the images is significant. This is expected as the images consist of events only. For Fig. 8(c) we obtain one value greater than 100. This is interesting as 100 is a soft bound for a bad score and it is due to the lack of received and rendered events which makes impossible to extract meaningful features.

C. Experiments with Background Traffic

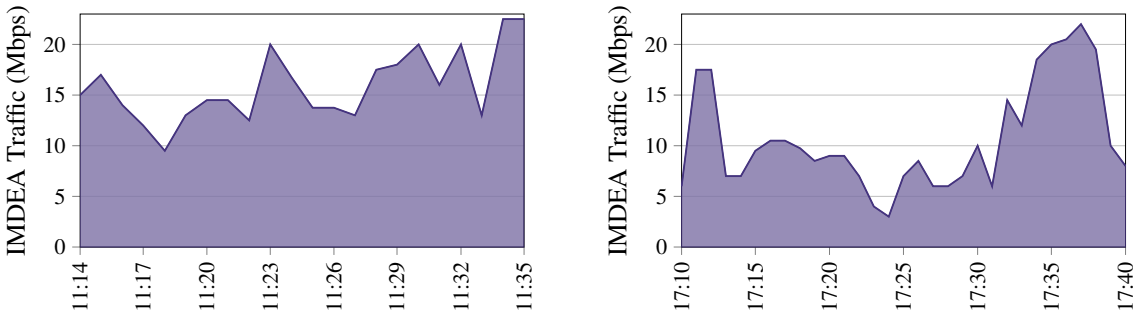
The objective of this section is to characterize the nature and the behavior of the event-based camera traffic in the presence of background traffic, i.e., when a station transmitting an event-based camera flow competes with other stations to access the wireless medium. As background traffic, both TCP and UDP are



(a) No losses (b) 10 % losses - PSNR = 29.28 dB (c) 30 % losses - PSNR = 27.74 dB

(d) No losses (e) 10 % losses - PSNR = 24.93 dB (f) 30 % losses - PSNR = 25.11 dB

Fig. 8. Impact of losses on video resolution at receiver and tracking - sec 1 of streaming (a), (b), (c) and sec 40 of streaming (d), (e), (f)



(a) Experiment that determines the amount of transmission retries (b) Experiment that evaluates the performance of tracking

Fig. 9. Bandwidth utilization of the WLAN network of IMDEA while performing different types of experiments

TABLE II
BRISQUE METRIC

STREAMING TIME	NO LOSSES	10% LOSSES	30% LOSSES
Second 1	78.87	81.23	127.82
Second 40	78.25	81.71	81.05

considered and we will evaluate the impact of the background traffic on transmission retries and over the tracking operation similarly to the experiment shown in Section IV-B. To compute the transmission retries, we use the `iw dump` command to output the MAC kernel module statistics of the network card. These statistics include inactive times, transmit and received bytes and packets, transmission retries (`tx retries`), signal strength and bitrate.

In this set of experiments, we include an additional laptop in our setup (see Fig. 4) that is the source of the background traffic. It runs both `iperf` and the D-IGT traffic generator [16]. D-IGT allows to configure the type of transport protocol, e.g., TCP, UDP or SCTP, and parameters like packet size and inter-departure times as random variable following a certain distribution, e.g., exponential, normal, Gamma, Weibull, etc. In our experiments, we consider UDP as transport protocol and use the following distributions:

- Poisson with *mean* inter-departure times (μ) of 600 and 1200 packets per millisecond.
- Uniform with *min* and *max* rate of $\rho = 300$ and $\omega = 400$ packets per millisecond.
- Weibull with *shape* parameter $\beta = 0.5$ and *scale* parameter $\eta = 300$. The *shape* parameter defines the slope of the distribution, while the *scale* parameter defines how much the distribution is stretched over the abscissa for a given value of *shape*.

Although the generation of background traffic with traffic generators allows to control the amount of injected traffic, this is not representative of real networks. For this reason, we also use the IMDEA Networks office WLAN, which provides 802.11ac connectivity to the entire building. In this case, we simply connect to this network the sender and receiver laptops that stream the event-based camera traffic. Fig. 9 shows the bandwidth utilization of the IMDEA Network WLAN during the time of the different experiments as provided by our IT department. Note that single experiments have a duration in the order of minutes and Fig. 9 reports a 30 min period where repeated measurements were taken.

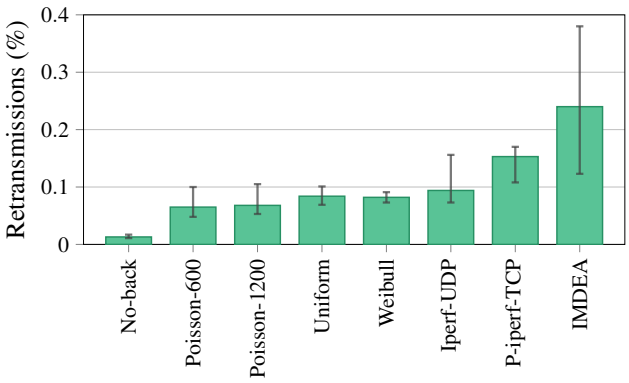


Fig. 10. Number of retransmissions of the event-based camera traffic with different distributions of background traffic

1) The impact of background traffic on transmission retries: Fig. 10 shows the number of transmission retries when the *shapes* dataset is transmitted for a period of 1 minute. We include, for the sake of comparison, the case when the same data transmission occurs in our private network without background traffic (No-back). As the `tx retries` parameter does not distinguish between re-transmissions due to collisions versus weak signal [33], the exercise of determining retransmissions in isolation allows to estimate the component due to weak signal solely (38 packets on average, over a total of 295 015 packets exchanged during the communication) and, in turn, to derive the component due to collisions.

First, note that the various traffic distributions generated by D-IGT lead to a similar amount of retransmissions that ranges from 192 of the Poisson distribution with $\mu = 600$ to 241 of the Weibull distribution. The variability, denoted as error bars, is higher for the experiments with the Poisson distribution and very small for the Weibull case. We also run `iperf` with TCP and UDP traffic and we set for the TCP case 5 parallel connections. Note that the highest amount of retransmissions occurs when the background traffic is that of the IMDEA WLAN network. The high variability is caused by the amount of background traffic present in the network during the various time instances when the single experiment runs were performed (see Fig. 9(a)).

2) The impact of background traffic on object tracking: Also for this experiment, we consider the *shapes* dataset because it easily allows to track objects because of the type of scene: *butterfly* only has one distinct element (the butterfly), *poster* because of the uniform texture does not have identifiable objects, and finally, *drone* and *driving* are very complex datasets with lots of identifiable objects with different size. The background traffic is that of the IMDEA WLAN network as per Fig. 9(b).

Fig. 11(a) compares the network throughput observed at the receiver side when the transmitter accesses the medium without and with contention of the IMDEA WLAN network. The latter case yields on average a throughput that is one order of magnitude lower than when the transmitter is alone in the network.

We instrument at the receiver side a procedure to capture screenshots of the jAER window with a granularity of 0.5 s. We then crop the obtained screenshots to limit the region of interest to the visualization window of jAER. This allows to obtain two sets of results: one with the transmitter contending for the medium and one where there is no contention. The evaluation uses reference-based metrics by comparing the two sets of images one by one with a common time reference. Fig. 11(b) and Fig. 11(c) show the obtained results for the PSNR and SSIM metrics respectively. In the first second, no events are rendered, hence the PSNR metric assumes the uncommon value of 361 dB and the SSIM metric equals to 1. Next, the PSNR metric oscillates in between $22 - 25$ dB, and the SSIM metric oscillates in between $0.77 - 0.87$, which indicate that in presence of contention much lower events are rendered.

Finally, in Fig. 11(d), we compute the times jAER is successful in correctly jointly visualize and track the objects contained in the *shapes* dataset during motion (note that visualization alone can be done more often). The tracking algorithm that jAER implements requires that a cluster of events is successfully received to identify the position of the object. However, this cluster might not have a sufficient number of events to *correctly* visualize the objects. When there is no contention (e.g., No Con. in Fig. 11(d)), in 59% of the cases the receiver can jointly track and visualize objects moving. Conversely, in case of contention (e.g., Con. in Fig. 11(d)) this figure sharply decreases to 29%.

V. Conclusion

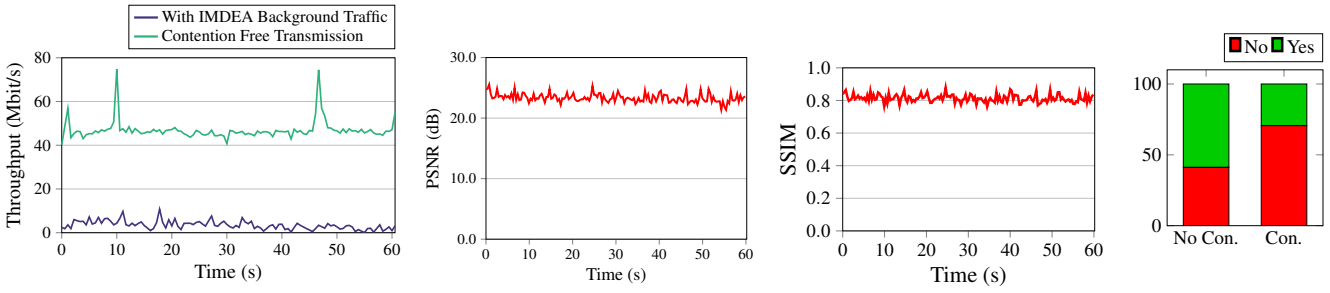
Event-based cameras overcome the blindness of conventional frame-based cameras between two consecutive frames, preventing loss of information on moving objects and rather capturing it as a continuous stream. In this paper, we take the first step in characterizing the network traffic generated by event-based cameras. Specifically, we look into the details of the relation between scene complexity and number of generated events and in turn packets, the impact of packets and events losses on computer vision operations like object tracking and the implications of accessing the wireless medium under contention. This study leverages a set of publicly available datasets recorded under different conditions, e.g., from a drone or car, and with different scene complexity, e.g., shapes over a white background or a poster with heavy textured background. Our results unveil that complex scenes that incur a high event generation rate are more robust against packet loss due to transmission errors or wireless contention. Conversely, packet loss or delay are more harmful to tracking and visualization operations when the event generation rate is small.

Acknowledgment

This work is partially supported by the the Madrid Regional Government through the TAPIR-CM program (S2018/TCS-4496) and the Juan de la Cierva grant from the Spanish Ministry of Science, Innovation and Universities (FJCI-2017-32309).

References

[1] G. Attanasio, "Event-based camera communications: a measurement-based analysis," M.Sc. Thesis, Politecnico di Torino, 2019.



(a) Throughput (b) PSNR (c) SSIM (d) Joint Tracking and Visualization

Fig. 11. Assessment of various metrics during the streaming of the *shapes* dataset with and without the presence of background traffic over the IMDEA WLAN network

[2] J. M. Chaquet, E. J. Carmona, and A. Fernández-Caballero, "A survey of video datasets for human action and activity recognition," *Computer Vision and Image Understanding*, vol. 117, no. 6, pp. 633 – 659, 2013.

[3] A. Mayberry, Y. Tun, P. Hu, D. Smith-Freedman *et al.*, "CIDER: Enabling robustness-power tradeoffs on a computational eyeglass," in *Proceedings of ACM MobiCom*, 2015, pp. 400–412.

[4] M. Daher, A. Diab, M. El Badaoui El Najjar, M. Ali Khalil *et al.*, "Elder tracking and fall detection system using smart tiles," *IEEE Sensors Journal*, vol. 17, no. 2, pp. 469–479, Jan 2017.

[5] M. Ángel Álvarez de la Concepción, L. M. S. Morillo, J. A. Álvarez García, and L. González-Abril, "Mobile activity recognition and fall detection system for elderly people using ameva algorithm," *Pervasive and Mobile Computing*, vol. 34, pp. 3 – 13, 2017, pervasive Computing for Gerontechnology.

[6] K. Y. Lam, L. Hang Lee, T. Braud, and P. Hui, "M2A: A framework for visualizing information from mobile web to mobile augmented reality," in *Proceedings of IEEE PerCom*, Mar 2019, pp. 1–10.

[7] V. Sze, Y. H. Chen, T. J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, Dec 2017.

[8] L. Liu, H. Li, and M. Gruteser, "Edge assisted real-time object detection for mobile augmented reality," in *Proc. ACM MobiCom*, 2019.

[9] M. Simsek, A. Aijaz, M. Dohler, J. Sachs *et al.*, "5G-enabled tactile internet," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 3, pp. 460–473, Mar 2016.

[10] Z. Xiang, F. Gabriel, E. Urbano, G. T. Nguyen *et al.*, "Reducing latency in virtual machines: Enabling tactile internet for human-machine co-working," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 5, pp. 1098–1116, May 2019.

[11] I. S. Kim, Y. Jeong, S. H. Kim, J. S. Jang *et al.*, "Deep learning based effective surveillance system for low-illumination environments," in *International Conference on Ubiquitous and Future Networks (ICUFN)*, July 2019, pp. 141–143.

[12] D. Gong, J. Yang, L. Liu, Y. Zhang *et al.*, "From motion blur to motion flow: A deep learning solution for removing heterogeneous motion blur," in *Proceedings of IEEE CVPR*, July 2017, pp. 3806–3815.

[13] G. Gallego, T. Delbrück, G. Orchard, C. Bartolozzi *et al.*, "Event-based vision: A survey," *CoRR*, vol. abs/1904.08405, 2019.

[14] C. Posch, T. Serrano-Gotarredona, B. Linares-Barranco, and T. Delbruck, "Retinomorphic event-based vision sensors: Bioinspired cameras with spiking output," *Proceedings of the IEEE*, vol. 102, no. 10, pp. 1470–1484, Oct 2014.

[15] J. Delmerico, T. Cieslewski, H. Rebecq, M. Faessler *et al.*, "Are we ready for autonomous drone racing? The UZH-FPV drone racing dataset," in *Proc. IEEE ICRA*, May 2019, pp. 6713–6719.

[16] A. Botta, A. Dainotti, and A. Pescapè, "A tool for the generation of realistic network workload for emerging networking scenarios," *Computer Networks*, vol. 56, no. 15, pp. 3531–3547, 2012.

[17] J. A. Pérez-Carrasco, B. Zhao, C. Serrano, B. Acha *et al.*, "Mapping from frame-driven to frame-free event-driven vision systems by low-rate rate coding and coincidence processing–application to feedforward ConvNets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 11, pp. 2706–2719, Nov 2013.

[18] T. Delbruck and M. Lang, "Robotic goalie with 3ms reaction time at 4 percent cpu load using event-based dynamic vision sensor," *Frontiers in neuroscience*, vol. 7, p. 223, 11 2013.

[19] R. Ghosh, A. Mishra, G. Orchard, and N. V. Thakor, "Real-time object recognition and orientation estimation using an event-based camera and CNN," in *Proc. IEEE BioCAS*, Oct 2014, pp. 544–547.

[20] D. Falanga, E. Mueggler, M. Faessler, and D. Scaramuzza, "Aggressive quadrotor flight through narrow gaps with onboard sensing and computing using active vision," in *Proc. IEEE ICRA*, 2017, pp. 5774–5781.

[21] "Vot2015 dataset," Accessed on Jan 31, 2020. [Online]. Available: http://www.votchallenge.net/vot2015/dataset.html

[22] C. Reinbacher, G. Munda, and T. Pock, "Real-time panoramic tracking for event cameras," in *IEEE International Conference on Computational Photography (ICCP)*, May 2017, pp. 1–9.

[23] E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck *et al.*, "The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and SLAM," *The International Journal of Robotics Research*, vol. 36, no. 2, pp. 142–149, 2017.

[24] A. Z. Zhu, D. Thakur, T. Özaslan, B. Pfrommer *et al.*, "The multivehicle stereo event camera dataset: An event camera dataset for 3D perception," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2032–2039, July 2018.

[25] A. Zhu, L. Yuan, K. Chaney, and K. Daniilidis, "EV-FlowNet: Self-supervised optical flow estimation for event-based cameras," in *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

[26] Q. Qu, B. Li, M. Yang, Z. Yan *et al.*, "Survey and performance evaluation of the upcoming next generation WLANs standard-IEEE 802.11 ax," *Mobile Networks and Applications*, pp. 1–14, 2019.

[27] M. Eisen, M. M. Rashid, K. Gatsis, D. Cavalcanti *et al.*, "Control aware radio resource allocation in low latency wireless control systems," *IEEE Internet of Things Journal*, pp. 1–13, Apr 2019.

[28] F. Tramarin, A. K. Mok, and S. Han, "Real-time and reliable industrial control over wireless LANs: Algorithms, protocols, and future directions," *Proceedings of the IEEE*, vol. 107, no. 6, pp. 1027–1052, June 2019.

[29] L. Liu, Y. Hua, Q. Zhao, H. Huang *et al.*, "Blind image quality assessment by relative gradient statistics and adaboosting neural network," *Signal Processing: Image Communication*, vol. 40, pp. 1 – 15, 2016.

[30] Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, April 2004.

[31] A. Mittal, G. S. Muralidhar, J. Ghosh, and A. C. Bovik, "Blind image quality assessment without human training using latent quality factors," *IEEE Signal Processing Letters*, vol. 19, no. 2, pp. 75–78, Feb 2012.

[32] L. Eggert and G. Fairhurst, "Unicast UDP Usage Guidelines for Application Designers," Internet Requests for Comments, RFC Editor, RFC 5405, Nov 2008. [Online]. Available: https://tools.ietf.org/html/rfc5405

[33] S. Rayanchu, A. Mishra, D. Agrawal, S. Saha *et al.*, "Diagnosing wireless packet losses in 802.11: Separating collision from weak signal," in *Proc. IEEE INFOCOM*, Apr 2008, pp. 735–743.