Video Streaming using Scalable Video Coding over Opportunistic Networks

Abhishek Thakur
Institute of Development and Research in Banking Technologies
Hyderabad, India
abhishekt@idrbt.ac.in

Vaibhav Balloli
Dept. of Electrical and Electronics Engineering
BITS Pilani Hyderabad Campus
Hyderabad, India
balloli.vb@gmail.com

Arnav Dhamija
Dept. of Computer Science
BITS Pilani Hyderabad Campus
Hyderabad, India
arnav.dhamija@gmail.com

***Abstract*—Opportunistic networks provide delay-tolerant communication in the absence of network infrastructure. Video streaming is challenging for such networks since real-time feedback is not available and the network is highly distributed and disconnected. This paper proposes an algorithm to adaptively transfer video over such networks according to available network resources. Scalable Video Coding is used to ensure that video of minimal quality gets delivered. If resources allow, the algorithm transmits the video at higher quality. We present a simple implementation for Android devices as a proof-of-concept. For evaluating the performance impact of variants in the algorithm, we also present the simulation results.**

Keywords—user-generated content, video streaming, scalable encoding, opportunistic networks, delay and disruption tolerant networks

I. Introduction

Low node density and high node mobility can cause partitions in ad-hoc networks. Delay and disruption tolerant networks (DTN's) [1] use the concept of store-carry-forward to provide communication between nodes. Opportunistic Networks (OppNets) are a sub-category of DTN where the node contact patterns are not predictable. Such networks rely on the creation of multiple replicas of each payload. Generally, higher number of replicas increase overheads while reducing delay and increasing the delivery rate.

Note that opportunistic communication in the last hop or peer-to-peer mesh networks for video transmission like [2] differs from our work since they either distribute prior generated content or have simultaneous connectivity between most nodes. Our work primarily assumes a delay and disruption tolerant network, with delays of the order of hours between source and destination, with no long-term contacts between nodes.

Continuous video streaming over such networks is challenging since it is difficult to predict the capability of the network. Underestimating available network resources can cause under-utilization, while overestimation can lead to congestion, increased delays and the dropping of payloads on intermediate nodes.

In this work, we use the Scalable Video Coding (SVC) [3] codec and adapt the simple Spray-And-Wait (SNW) [4] routing protocol to implement adaptive video transmission. Simulation results show that the proposed scheme provides significantly better results than non-adaptive transmission. Regarding prior work, Klaghstan [5] has attempted SVC transmission in OppNets but studied only one segment of video transmission. Moreover [5] did not have any real-world implementation. Our work uses information from prior segments, to adapt the quality of video for subsequent transmissions. We also present a proof-of-concept for the same with an Android application built on top of Nearby Connections API.

In section II we provide an overview of SVC and DTN routing. Section III provides the algorithm details for adaptive transmission by the video source. Section IV provides simulation results. Section V provides the details of proof-of-concept for video streaming using Android devices. Section VI concludes the paper and also discusses the proposed future work.

II. Background Details

To stream video content over OppNets, the captured raw video needs to be compressed and packaged into payloads. Intermediate nodes relay these payloads to the destination. We can use different DTN routing approaches to relay the payload. In this work, we use SNW routing as it allows setting the maximum replica count of a payload at the source.

The primary contribution of our work is continuous adaptive streaming of video over OppNets, using end-to-end mechanisms only. Similar to additive-increase and multiplicative decrease of TCP, only the source and destination nodes implement the adaptation, without requiring any changes on intermediate nodes.

A. Scalable video coding and payload generation

In reliable networks with round-trip delays below a second, the source can get feedback from the receiver and adapt its encoder settings to target an optimal bit-rate. For OppNets, the acknowledgment may come after several hours of delay, or it may be lost entirely. Hence, we cannot utilize such schemes. Scalable Video Coding (SVC) and Multi-Description Coding (MDC)[6] provide an alternative for such scenarios. The source can create payloads of varying bit-rate and quality. In MDC, the payloads carry mutually overlapping information for redundancy. This information overlap cannot be controlled externally, making it unsuitable for adaptive transmission. For our work, SVC is a better fit since it allows external redundancy control. An in-depth discussion of SVC is out of the scope of this paper and [3] provides a much more comprehensive overview of the same.

Payload can be mapped at each network adaptation layer unit (NALU) or SVC layers of video chunks of few seconds each can be mapped to payload. NALU level mapping of payload requires the network elements to be aware of the media flow[5]. In our work, we do not expect any support from intermediate nodes, hence the source (SVC encoder) compresses a chunk of captured video (of a few seconds each) into a base layer (BL) and multiple enhancement layers (EL). Only after receiving the base layer, the content from the enhancement layer can be used to improve the quality of the video further. The receiver necessarily requires the BL to decode the video. On receiving subsequent enhancement layers, the decoder can increase resolution, improve frame rate, increase colour depth and reduce quantization loss. For a particular chunk, if an intermediate EL is missing, higher ELs cannot be decoded. Let us consider the example, where the source encodes and transmits a segment of video using SVC layers as BL and three ELs (labelled as EL1, EL2, and EL3). If BL is not received, the complete segment cannot be decoded. In another instance, if only BL, EL2 and EL3 are received, the decoder can only provide the output to the quality carried in BL. In this case, EL1 is missing, and hence, we cannot use EL2 and EL3.

B. *Routing in OppNets*

Reference [7] provides a detailed description of various routing protocols for routing in Delay Tolerant Networks. At one extreme, we have direct-delivery where the source should meet the destination to deliver the payload. Direct-delivery has large delays, and the network may suffer from delivery failures. Nevertheless, this approach has the lowest network utilization and the least demand on storage space. On the other extreme, we have the epidemic routing protocol where the payload is shared with all nodes. When the network is lightly loaded, epidemic routing will provide the best performance on delivery rates with minimal delay. However, epidemic routing has the highest network resource utilization as it shares a copy with all the nodes in the network. It also has high overheads on storage of bundles. Network congestion may occur when large or frequently generated payloads added.

Spray-and-Wait [4] provides a balance between direct delivery and epidemic routing. The source node can set the max upper bound to the number of replicas (L) that can be created. If L equals one, SNW behaves similarly to the direct delivery protocol. If the initial value of L is more than the number of nodes, it behaves similarly to epidemic routing. The Spray phase is the initial phase where the payload is relayed to the other nodes. In this phase, the copy counts are adjusted on both nodes when a payload is transferred. For Binary-SNW routing protocol used in this work, the relaying node drops the copy count of a payload by half after transferring it. A payload on a node enters the Wait phase when it has a copy count of one. In the Wait phase, the payload cannot be relayed to any node except for the destination.

III. Adaptive Video Transmission In OppNets

In the beginning, the source transmits all the SVC layers. The BL is given the highest value for L compared to all other layers. Similar to Klaghstan [5], higher enhancement layers have a lower value of L. The destination sends acknowledgments for all the payloads that it receives.

In our experiments, we have first implemented temporal scaling of five layers, before bringing in spatial and quality scaling. This implies we have first layer as BL, followed by EL1-EL4 as temporal enhancements. EL5-EL9 improve the spatial quality at different temporal layers. For creating a video of consistent spatial quality, we should have either all of the layers between EL5 to EL9 or none of them [5]. We have also experimented with other sequence of SVC layers (e.g. BL followed by EL1 at higher resolution, and EL2-EL6 bringing in temporal scalability). Because of space constraints, we are only discussing adaptation for ten layers BL, EL1, EL2, EL3, EL4 and EL5-EL9 scenario.

Algorithm 1 implements adaptive transmission. Delay-Target (*DT*) is a deployment specific setting. It is the intended delay from the time of the capture of a video segment at the source, to the time for decode and playback of the segment at the destination. Burst-Gap is the time gap between each segment capture. Note that AdaptiveRecordAndTransmit is invoked at the source for each segment. *MxL* is the maximum number of SVC layers that the encoder produces.

Algorithm 1: AdaptiveRecordAndTransmit

Initialize: burstId←0, MxL←10, factor ←1

```
1.  begin
2.  burstId← burstId +1
3.  dTAwayId ← burstId – dtMult × DT/burstGap
4.  if (ackReceived(dTAwayId)) then
5.          factor ← factor + numAcksAddIncr
6.  else
7.          factor ← factor × MultDecr
8.  if (factor > 1) then
9.          factor ← 1
10. if (factor < 1/MxL) then
11.         factor ← 1/MxL
12. numLyrs2Send← MxL * factor
13. if (numLyrs2Send > 5 AND numLyrs2Send < 10) then
14.         numLyrs2Send ←5
15. SVCEncodeAndTransmit(numLyrs2Send)
16. end
```

For instance, in a sparse network with an average delivery delay is of 8 hours, DT may be set to 12-24 hours and burst-gap to 5 minutes. Since the network reacts with a significant lag, we also experimented by changing dtMult in line 3 of Algorithm 1 for values of 1, 2 and 4 during simulation runs. Values for Additive Increase (AI) as 0.003 and Multiplicative Decrease (MD) as 0.01 are determined through simulations.

SVCEncodeAndTransmit encodes the raw video and only transmits the SVC layers up to numLyrs2Send. Each layer is transmitted as a separate payload. The time-to-live for each payload is set to twice the delay-target to ensure that the payloads do not continue to occupy buffer forever.

IV. Simulation Results

Opportunistic Network Environment (ONE) [8] was used to simulate the video transmission. We simulated fifty pedestrian nodes based on the Random Waypoint [8] movement model of ONE. DT is set to three hours and burst-gap to 5 minutes. To simulate the specifications of a smartphone's Bluetooth radio, we used a radio range of 10m with a network speed of 250 KB/sec and a buffer space of 50 MB in these simulations. The video source had 500 MB buffer space. The time-to-live for payloads is set to six hours and payload size is determined using SHM (12.4 from https://hevc.hhi.fraunhofer.de/svn/).

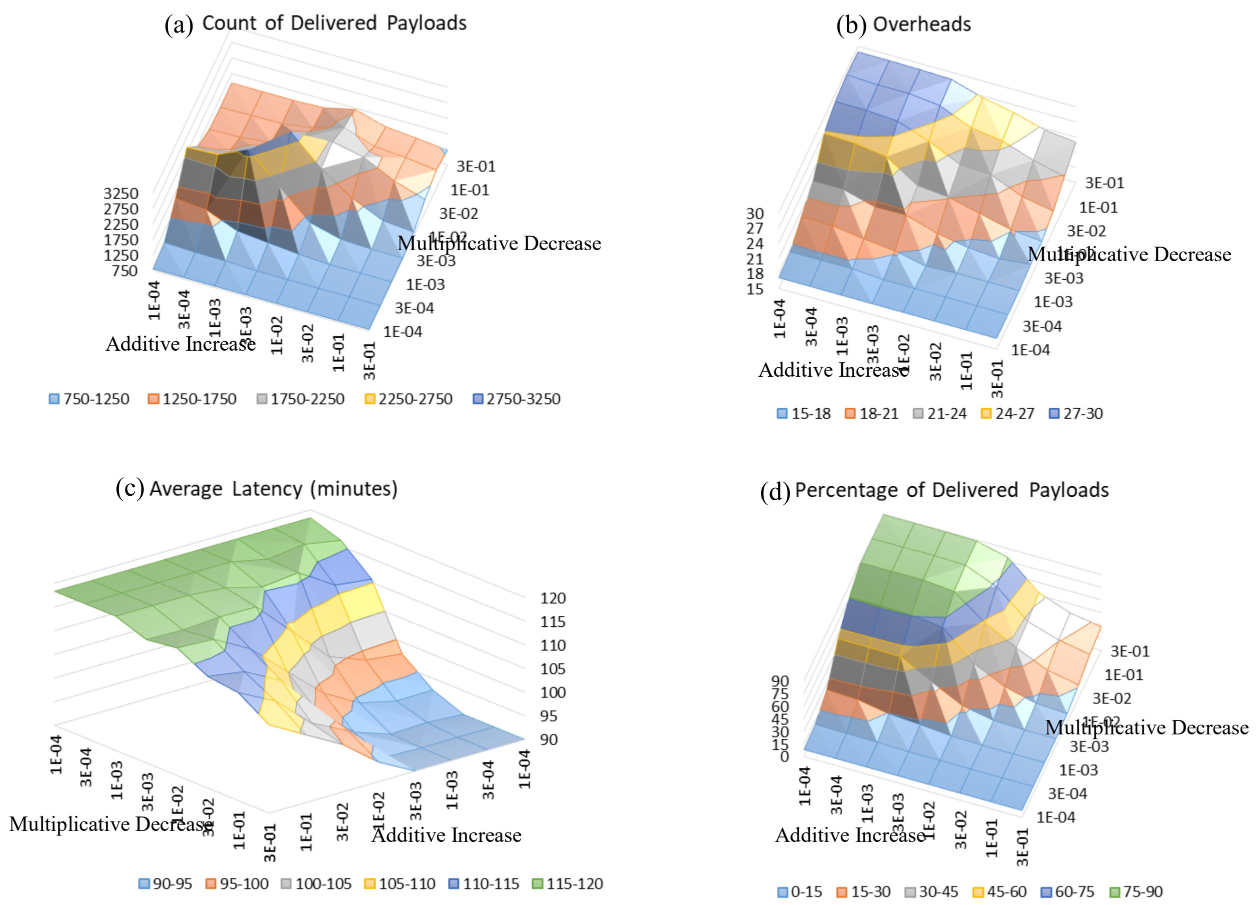


Fig. 1 Simulation results for dtMult as two

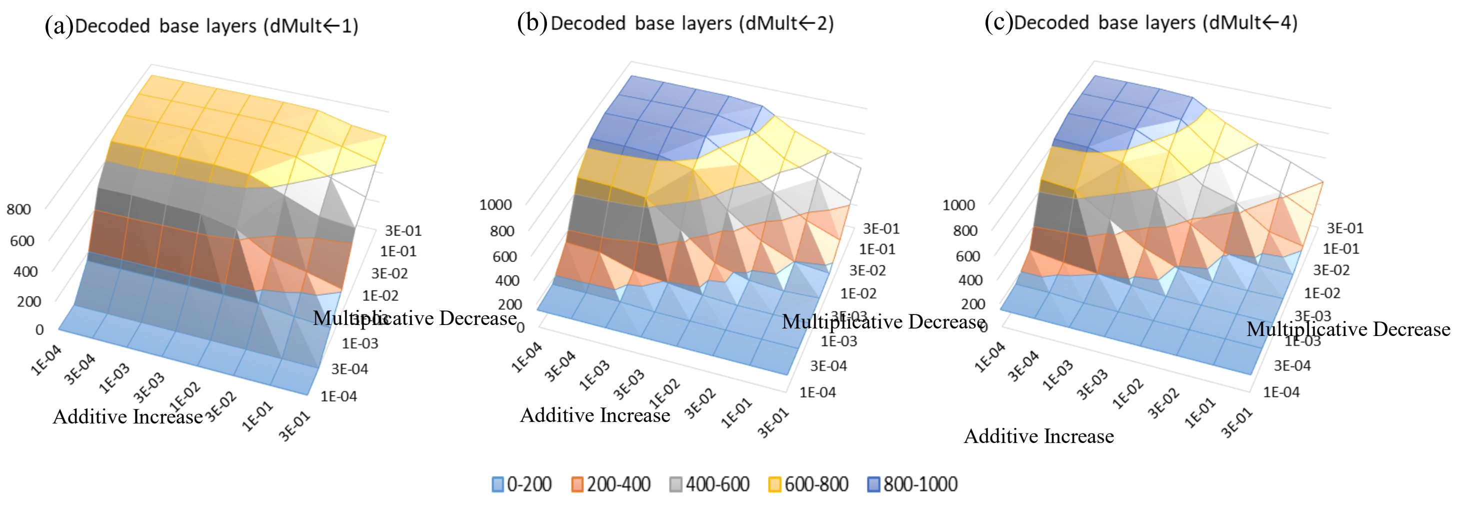


Fig. 2 Simulation results for varying dtMult

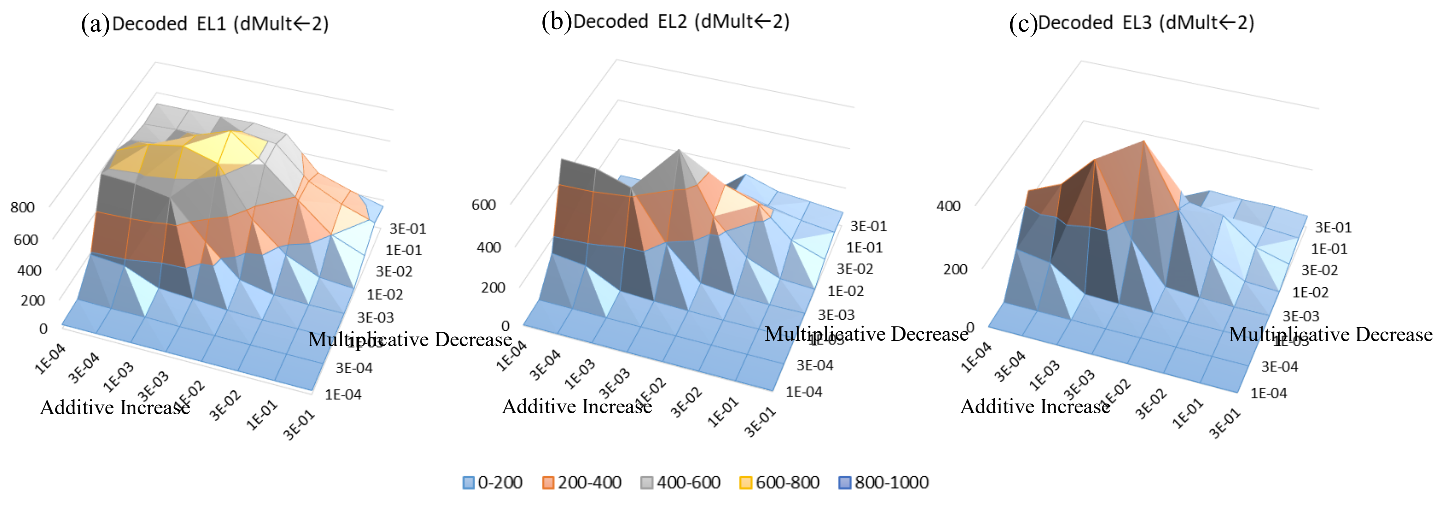


Fig. 3 Enhancement layers for dtMult as two

The video payload used in the simulation is compressed using SHM (ver. 12.4) into payloads corresponding to BL, EL1, EL2, till EL9 for a total of ten payloads for each video segment captured. The simulation involved five runs with different seeds to ensure that there is no specific bias in the simulation.

Fig. 1 presents the results for various metrics as multiplicative decrease and additive increase values are varied for adaptation, using acknowledgment for payload generated two DT ago. For all the metrics, it is observed that the MD values above 0.003 and AI values below 0.01 provide relatively good results. Fig. 1 (a) shows that count of delivered payloads has two pronounce peaks at AI=0.003 / MD=0.01 and AI=0.001 / MD = 0.003 respectively. The delays (Fig. 1 (c)) and percentage of delivered payloads (Fig. 1 (d)) have best values for MD greater than 0.003 and AI lower than 0.003. In this range the overheads peak (Fig. 1 (b)). Despite the high overheads, these results show that AI around 0.003 and MD as 0.01 and its vicinity are good area to operate in these scenarios.

Note that when SVC is not used only 82 segments of video (of the 1000 that were transmitted) reached the destination. For low values of MD, SVC adaptation is poorer than that of non-SVC transmission. As MD value increases to 0.003 and higher, adaptive transmission does better than non-SVC transmission. Detailed analysis showed that at MD of 0.3, almost 600 bursts are decoded, though most of them are at base layer.

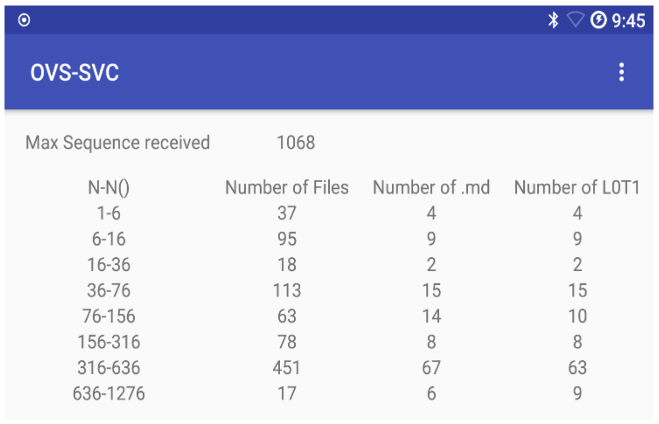
Fig. 2 a-c shows the number of delivered bursts with dtMult of one, two and four. It shows that for the simulation scenario, dtMult of four (peak of 882) does not provide any appreciably improvement when compared with dtMult of two (peak of 885). dtMult of one had a significantly poor result (peak of 728). The best result of dtMult as two corresponds to round trip time for the communication. The mobility model is synthetic in nature with nodes randomly walking along predetermined routes. The randomness brings in symmetry for round-trip time computation.

Fig. 3 a-c shows the first three enhancement layers for dtMult as two. The peaks for first enhancement layer corresponds to 736 segments of video, second enhancement layer as 553 segments and third enhancement layer as 400 segments. These values are gain for AI as 0.003 and MD as 0.03/0.01.

Note that the simulation matches with a disaster response scenario where the activity is continued on a 24x7 basis. In case other mobility model like a working day scenario or traces from real world are used, we observe that the dtMult values closer to twenty-four hours provide the best results. For brevity, this discussion is not included in the current work.

V. Proof of Concept

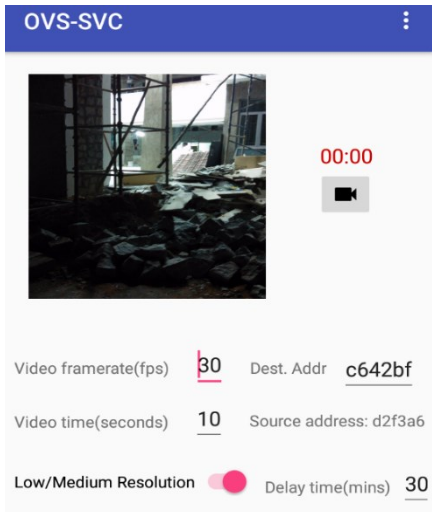
VECTORS [9], an app developed by the author's team in the past that creates a DTN on Android devices is extended for this work. Prior release implemented the video capture and encoding logic to create the payload using an SVC encoding library, SHM, on Linux based machines. SHM is cross-compiled to run on Android as a native application that generates a total of 10 spatial and temporal layers for the payload. When compared to [9], present work simplifies the adaptation logic and runs entirely on Android devices, eliminating the use of Linux computers.



Max Sequence received 1068

N-N()	Number of Files	Number of .md	Number of L0T1
1-6	37	4	4
6-16	95	9	9
16-36	18	2	2
36-76	113	15	15
76-156	63	14	10
156-316	78	8	8
316-636	451	67	63
636-1276	17	6	9

Fig. 4 Screen capture for the destination node



Video framerate(fps) 30 Dest. Addr c642bf

Video time(seconds) 10 Source address: d2f3a6

Low/Medium Resolution Delay time(mins) 30

Fig. 5 Screen capture for the source node

To simulate a construction site with no infrastructural support scenario, an experimental deployment was set up on a construction site in BITS Pilani, Hyderabad Campus consisting of four Android devices. One device was kept static, acting as the source node to record the area of construction and the destination node was placed at the site-management office, both of which were running the OVS-SVC app. First device captured video and generated the payload. The second device, received the payload and tabulated the received payload statistics. The construction site employees carried the remaining two devices running VECTORS [9] that relayed the payload. Since the employees at the construction site do not work in the nights, we used DT as 12 hours and dtMult of 2. TTL was set to 24 hours.

Fig. 4 shows the screenshots of the OVS-SVC app for the destination, and Fig. 5 for the source. L0T1 in Fig. 4 is the equivalent of the BL in the payload generated.

VI. Conclusion and Future Work

Through simulations we experimented with AIMD based adaptation with varying periods to check for acknowledgments. As compared to non-adaptive and non-SVC based transmissions, Adaptive SVC provides significant improvements with correct choice of AI and MD values. The decodable bursts were found to be best when acknowledgement for payloads generated with twice the delay-target are considered. In future, we plan to experiment with other adaptation algorithms and different mobility scenarios, including real-world traces. We are also extending this work to measure fairness for multiple such video streams. Priority of video streams is another area to be explored.

We also demonstrated the adaptive streaming of video chunks using scalable video coding. As compared to our prior work (VECTORS) which utilized both Linux computers and Android devices, this solution only utilizes Android devices. Since Android devices are much more affordable and portable, our work would make opportunistic video streaming possible for a greater number of applications.

As SHM is a software encoder, the present system can only transmit 30-second videos at four frames per second. This is compressed into a set of temporal enhancement layers followed by spatial enhancement layers. Future extension of this work will explore hardware-compression and dynamic adaptation using variable frame rates. The team also intends to create an Android application ecosystem for deploying such apps to domain-specific requirements – e.g., event management, policing, disaster response, wildlife monitoring, etc.

References

[1] Fall, Kevin. "A delay-tolerant network architecture for challenged internets." In Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications, pp. 27-34. ACM, 2003.

[2] Mavromoustakis, Constandinos X., et al. "On the perceived quality evaluation of opportunistic mobile P2P scalable video streaming." Wireless Communications and Mobile Computing Conference (IWCMC), 2015 International. IEEE, 2015.

[3] Boyce, Jill M., Yan Ye, Jianle Chen, and Adarsh K. Ramasubramonian. "Overview of SHVC: Scalable extensions of the high efficiency video coding standard." IEEE Transactions on Circuits and Systems for Video Technology26, no. 1 (2016): 20-34.

[4] Spyropoulos, Thrasyvoulos, Konstantinos Psounis, and Cauligi S. Raghavendra. "Spray and wait: an efficient routing scheme for intermittently connected mobile networks." In Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking, pp. 252-259. ACM, 2005.

[5] Klaghstan, Merza. "Multimedia data dissemination in opportunistic systems." PhD diss., Université de Lyon, 2016.

[6] Kazemi, Mohammad, Shervin Shirmohammadi, and Khosrow Haj Sadeghi. "A review of multiple description coding techniques for error-resilient video delivery." Multimedia Systems 20, no. 3 (2014): 283-309.

[7] Cao, Yue, and Zhili Sun. "Routing in delay/disruption tolerant networks: A taxonomy, survey and challenges." IEEE Communications surveys & tutorials 15, no. 2 (2013): 654-677.

[8] Keränen, Ari, Jörg Ott, and Teemu Kärkkäinen. "The ONE simulator for DTN protocol evaluation." In Proceedings of the 2nd international conference on simulation tools and techniques, p. 55. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2009.

[9] Thakur, Abhishek, Arnav Dhamija, Tejeshwar Reddy G "VECTORS – Video communication through opportunistic relays and scalable video coding" SoftwareX volume 9: p.55-60.