Cache and Delivery of VR Video over Named Data Networking

Yi Zhang*, Xiaoke Jiang†, Yi Wang‡, Kai Lei*§
* School of Electronic and Computer Engineering (SECE)
Peking University, Shenzhen, China, 518055
† Kandao Technology, Shenzhen, China
‡ Southern University of Science and Technology, Shenzhen, China, 518055
§Corresponding Author: leik@pkusz.edu.cn

***Abstract*—Virtual Reality (VR) may fundamentally change the way of human interaction with cyberspace, and VR video streaming is key to create an immersive VR experience. However, the delivery of VR video requires high spatial and temporal fidelity contents and strict low-latency, which causes that VR video streaming requires much larger bandwidth compared to traditional 2D videos. In this paper, we design and implement a VR video delivery system over Named Data Networking (NDN), which can be regarded as a future network architecture. We study the namespace design of video data, dynamic streaming switch of VR video, pipeline design of Interests. Besides, we also propose an integrating hotspot-based and popularity-based cache policy to cache the contents which are most likely to be requested. The experimental result shows that our proposed method can significantly reduce the transmission delay of VR video and enhance user experiences.**

***Index Terms*—Named Data Networking, virtual reality, pipeline, cache policy**

I. Introduction

Virtual Reality, known as VR, is becoming popular due to amusingness, future outlook and adaptive flexibility. According to a Goldman Sachs forecast, VR will grow to become a $100 billion market in 2025. Panoramic videos are widely used to build VR because they can immerse the users in an artificial environment. Compared to normal videos, a VR video has a larger Field of View (FoV), typically $360^o \times 180^o$ around a center point, and consists of two separate views to feed two eyes individually in order to construct a 3D panoramic scene. As a consequence, the delivery of VR video streaming requires much larger bandwidth compared to 2D video.

However, for VR videos, a watcher just focuses on a very small area of the $360^o \times 180^o$ video at a given time and it is a bandwidth waste to transmit the whole video. Based on this observation, Facebook proposed dynamic streaming technology to save bandwidth. Dynamic streaming produces several copies of videos, each of which contains only a fixed high-resolution area, while the other areas are low-resolution. It chooses the most appropriate copy and transmits it to users based on the users' current viewpoints. In this way, the bandwidth for transmission is significantly reduced. However, when the viewpoint of a user changes, the user will watch the low-resolution area, thus an appropriate copy in the server needs to be transmitted immediately, which is called copy switching. Obviously, Facebook's solution inevitably results in frequent copy switching in practice.

In this paper, we exploit Named Data Networking (NDN) [1] to facilitate VR streaming. NDN naturally supports viewpoint-dependent VR streaming by naming the data explicitly with additional bonuses: a) Smoothing dynamic streaming via "free-riding": a newly-sent request fetches data from "trace" (e.g., cached data) previously left by other nodes; b) Smoothing mobiles' handoff by simply re-issuing pending interest, noting that VR videos are usually consumed on mobile devices, such as smartphone and helmet-mounted device (HMD); c) Serving multiple devices simultaneously via broadcasting at mobile's last hop in 5G and WiFi network; d) Minimizing the transmission delay and saving network congestion with adaptive forwarding to leverage redundant links in real time [2].

In this paper, we design and implement a VR video delivery system over NDN. Note that cache is very important both in NDN and video distribution, so we propose a cache policy for VR video data, aiming to reduce transmission latency as much as possible and improve the user experience.

The rest of the paper is organized as follows. In Section II, we briefly discuss related work. Then we present an overview of our system in Section III. We describe our design details (including namespace design, smooth switch of viewpoint, and pipeline of interests) in Section IV. The design of cache policy for VR video data is presented in Section V. We evaluate our implementation and provide some results in Section VI. Finally, We discuss future works and conclude in Section VII.

II. Related Work

The Voice-over-CCN [3] project is an early exploration of the project's real-time communications in ICN. Compared with traditional VoIP [4], VoCCN has a shoulder-to-shoulder service quality, and its system architecture is more simple and flexible, and the potential scalability of the system is also better. The VoCCN design concept has a significant impact on the successive NDN applications. Many NDN real-time distribution systems draw on VoCCN's "pipelining Interests" solution for real-time scenarios. ACT [5] was an early NDN voice conference application proposed by Jacobson et al., Which mainly delineates the NDN network architecture. ACT

978-1-5386-5979-3/18/$31.00 ©2018 IEEE

enables efficient conversation and user discovery, which are problems that IP voice conference can not handle. However, subject to the construction of an early foundation platform for the NDN project, ACT is difficult to deploy on a large scale.

NDNVideo proposed by UCLA [6] supports two working modes of video-on-demand and live broadcasting, which is regarded as a typical example of network application development in content center. NDNVideo adopts the hierarchical naming data method and the sequential numbering mechanism based on the time line to name the video segment, so as to achieve random access. NDNVideo can be deployed on the NDN test network and can support multi-consumer, single-production mode of operation. From the experimental results: NDNVideo has good scalability, but suffers higher real-time streaming media distribution delay. After that, the NDN project team put forward two new application frameworks and related APIs for NDNLive and NDNtube [7] on their basis and dealt with two scenarios of live video and on-demand respectively. In summary, NDNVideo and NDNLive architectures are simple, flat, and has good scalability, but simple sequential request mode will lead to high latency. In addition, their main application scenario is a fixed terminal device.

The NDN-RTC [8] proposed by Peter Gusev et al. is another NDN real-time conferencing solution and aims to build an experimental research platform with low latency and real-time communication on NDN. The NDN-RTC aims to enable a demand-driven live broadcast system using NDN features such as named forwarding, data signing, caching, and request aggregation. NDN-RTC uses the WebRTC library for audio and video encoding and decoding, while potentially supporting the browser as a client. NDN-RTC uses a different naming scheme for the audio and video of live streams and enables the ”meta-information” concept and the quick access to the ”newest” data segment of the Interest filter. Currently, NDN-RTC is still in the architecture design and small-scale testing phase.

In addition, [9] proposed overall guidelines for the design of streaming media applications in the information center network. Reference [10] studied and discussed the NDN real-time TV service. The literature [11] describes the producer/subscriber service model in the content center network. It proposes the concept of ”ADU” mainly at the application layer. FileSync/NDN [12] discusses a solution for synchronizing data sets with iSync [13]. ChronoSync [14] is a synchronization protocol under NDN that provides a method of data synchronization and shows the architecture of an NDN chat and file sharing system based on synchronization protocols, ChronoChat [15] and ChronoShare [16].

Recently, with the booming development of Virtual Reality, the research on the transmission of VR video over NDN is put forward. The publication of [17] designs a VR video conference system over NDN. However, this system doesn't take dynamic streaming technology of Facebook into account. As a contrast, our work is based on the dynamic streaming technology and has more innovativeness.

III. System Overview

Our objective is to design a VR video delivery system over NDN, and this system should have the ability to address the following questions effectively:

1) *The naming of video data.* NDN is such a kind of network which use the name of content to fetch the corresponding data, so we should design a suitable namespace and decide what name should be given to each new data object.
2) *The smooth switch of viewpoint.* The dynamic streaming technology of Facebook can not guarantee the immediate handoff of streaming and smooth viewpoint switching. As a result, the scene will become blurred when user changes its viewpoint. We should take full advantage of the features of NDN to achieve a smooth switch of viewpoint.
3) *The design of Interest pipeline to minimize delay.* In order to decrease delay, multiple simultaneous Interests should be issued concurrently once video data are produced. Therefore, a smart design of Interest pipeline is essential to control the transmission of Interest.

With the above questions, we propose a framework shown in figure 1. From this figure we can see that the framework has two types of participants: publisher and consumers. The consumer requests video data from producer using Interests which contain the name of requested content. The producer or intermediate cache then respond with corresponding data. We describe the detailed system architecture respectively on the producer and consumer side as follows.

On the producer side, the VR application first captures 360-degree video streaming from a panoramic camera. Then the encoder encodes the VR video frames into different copies based on bitrate and viewpoint. After that, a module called *Segmenter* divides every frame into several segments and stores them in the cache waiting for the requests of Interests.

On the consumer side, the Interest pipeline issues Interests to the network for the needed video data. We use two modules to decide the information of requested data, *Helmet Sensor* and *Adaptive Bitrate(ABR)*. *Helmet sensor* can perceive the orientation of user's head and determine the viewpoint. *Adaptive Bitrate* can make an appropriate bitrate decision based on network throughput and playback buffer size. The returned video segments are stored in *Data Buffer*, from which the player reads data and takes a series of steps including decoding, rendering and playing.

IV. Design Details

A. Namespace Design

For a NDN application system, naming the data is the fundament of data transmission. An appropriate namespace design presents the data in a way that makes much sense to the system and makes this data fetched efficiently. The hierarchically structured name of streaming data in NDN is beneficial to the dynamic switching of VR video streaming. The namespace designed in this paper is illustrated in figure

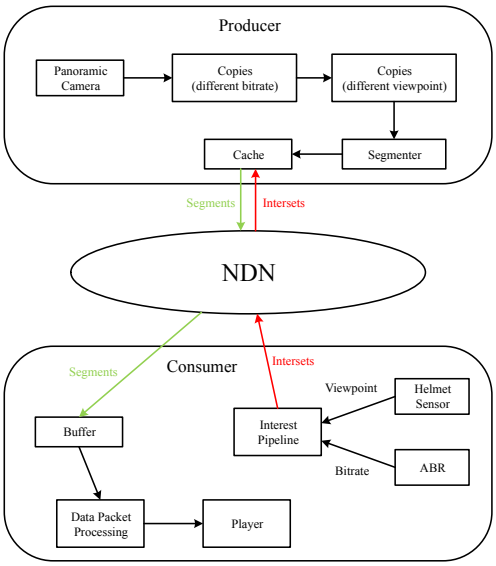

Fig. 1. System Architecture.

2. Here we use the following example in figure 3 to explain the semantic of each component in the namespace.

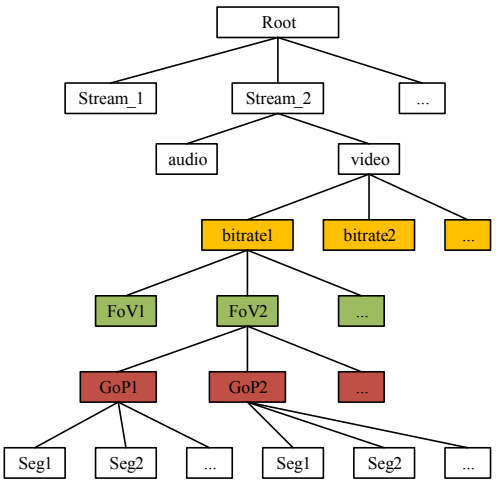

Fig. 2. Namespace.

1) *Root Prefix*. In figure 3, /kandaovr.com/VR-LIVE is the root prefix which describes the organization and application of requested data. Specifically, /kandaovr.com represents the organization of producer and /VR-LIVE represents the application.
2) *Streaming Number*. There are often many data streams in a streaming media delivery system, including video streams, audio streams or others, so we should number these streams to distinguish them. /video_101 represents that this is a video stream and the number is 101.
3) *Bitrate*. In the producer side, each stream is encoded into different bitrates representing the video quality levels, so consumer can request appropriate bitrate based on the network conditions. In this example, we set the bitrate as 18Mbps.
4) *Viewpoint*. The VR videos are panoramic and spherical, which contain $360° \times 180°$ scenes. Therefore, we use spherical coordinate (x, y) to describe the user's current viewpoint. x is the angle of horizontal direction, which ranges from $0°$ to $360°$ and y is angle of vertical direction ranging from $-90°$ to $90°$. In this example, (0, 90) represents that the user is requesting video segments whose viewpoint is right above the user.
5) *Group of Pictures*. In this paper we treat group of pictures (GoP) as the basic application-level data unit and /103 represents the number of current GoP. The first frame of a GoP is I-frame, so all frames in a GoP can be decoded independently of other GoPs. This makes GoP the basic unit of stream switch. If a GoP outsizes the upper bound of data packet, segmentation is adopted.
6) *Segment Number*. The size of video frames or GoPs is large and we need to split them into several segments to fit the size requirement of network transmission. In this example, /_s31 represents the segment number, from which we can obtain the order information of segments and assemble them in sequence in the consumer side.

Fig. 3. Naming Example.

B. Smooth Switch of Viewpoint

Compared to conventional video streaming, VR streaming is viewpoint-dependent, and the smooth switch of video streaming with different viewpoint is a crucial challenge. However, in IP network, the implementation of smooth streaming switch is difficult. When a user moves his head and changes the viewpoint, the client first informs the VR application server by sending control signal, then the server switches the streaming and returns the video data with current viewpoint. Therefore, when the head movement is frequent, the transmission of control signals will take up a lot of bandwidth, which will eventually increase the switch delay and cause lag phases and blurry scenes which severely degrade the user experience.

In this paper we exploit NDN to facilitate the smooth switch of viewpoint. With caching and interest aggregation, a newly-sent interest can leverage existing "trace" (be it cached data or forwarding record on routers) left by previously-sent interest without going to original data publisher itself. This free-riding feature smooths switch and mobile handoff in VR dynamic streaming.

As shown in figure 4, if M_1 switches from stream A to stream B, the buffered data D_{A1} is not the best choice any more. M_1 sends I_{B1} which then fetches data from $AP1$ without being forwarded further, since M_2 already sends I_{B1} before switch. As to mobile handoff, the returning data D_{C1}

is lost at last hop when M_3 moves to $M_3^{'}$. But the mobile can save packet loss by re-issuing the pending interest I_{C1} and the interest will meet desired data at router R. The free-riding works best for live streaming because the watchers' requests are almost synchronized.

Moreover, NDN over broadcast channel (e.g., 5G or WiFi) can deliver the same piece of data (D_{B2}) to multiple consumers (M_1 and M_2) simultaneously. This is of great importance given most VR playback devices are mobiles.

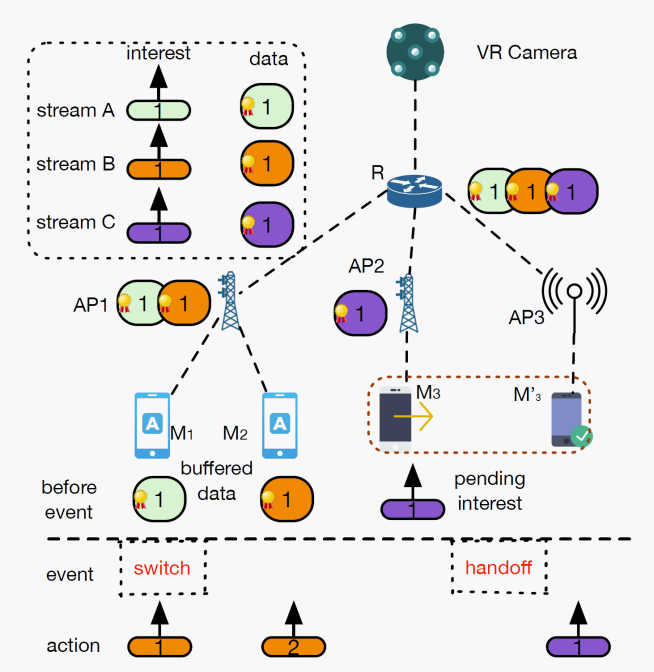

Fig. 4. Switch of Viewpoint.

C. Pipeline of interests

In NDN-VR, to fetch a data segment, consumer sends an interest packet which carries the name of requested data, then the video producer returns the corresponding data packet to the consumer. However, for a video application, it's impossible to send interest packets one by one and wait until the last data packet is received, which will cause a huge delay and seriously impact the QoE (Quality of Experience) of users. Therefore, in this system we design a pipeline of interests to concurrently request multiple segments. As shown in figure 5, consumer issues simultaneous interest packets to the network for data as needed. The data Buffer receives returned data packets and reassembles these segments in the correct order. video player reads data from Buffer and takes a series of steps including decoding, rendering and playing. The blank of Buffer in figure 5 represents issued but not answered Interests, which we call *in-flight Interests*. The consumer waits for *in-flight Interests* until the data arrives, otherwise these Interests time out and are reissued. There are some pointers in the Buffer that need to be described. The pointer *begin* indicates the start position when the player reads data while *end* indicates the end position. When the size of readable buffer reaches preset threshold, the player starts to read data from Buffer.

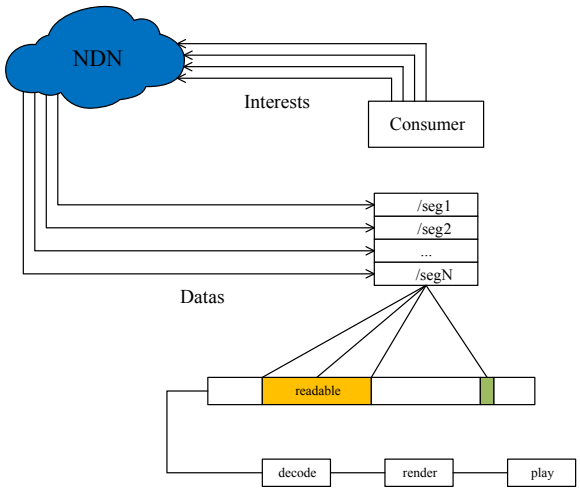

Fig. 5. Pipeline.

Then we show the timeline of data transmission between producer and consumer. In the beginning, consumer issues an estimated number of requests to fetch corresponding video data. As shown in figure 6, the consumer issues *M* Interests and waits for the responses. The time interval between Interest arriving and data publishing is called *production delay*, d_{prod}. Producer responses to the *M* Interests when the newly captured video frame is sliced into *N* segments and released. Consumer then knows that this frame has *N* segments through the information carried by returned data packet and issues $N-M$ Interests to request the remaining video segments. We can see that at this time there is no *production delay* because the remaining segments have been released.

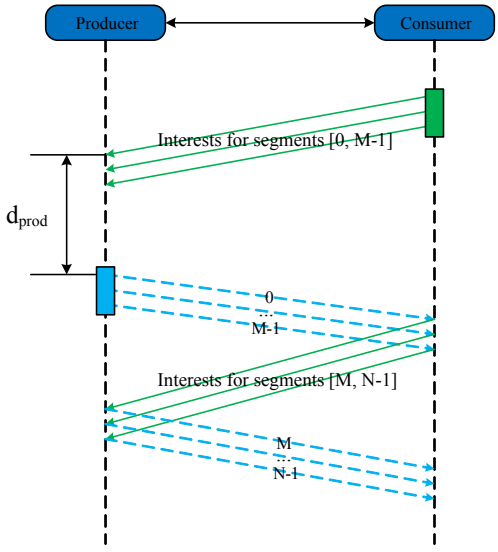

Fig. 6. Timeline.

V. Cache Policy

We have described our system design above from the perspective of producer and consumer. But beyond that, We should facilitate the transmission of video data from the Internet perspective. Note that content caching at intermediate nodes, also called in-network cache, is fundamentally important to support the basic concept of content-centric and peer-to-peer data delivery model of NDN at low cost, so it is essential to design an appropriate cache policy.

There are some differences between VR videos and traditional videos when caching their video data in the intermediate routers. Particularly, the dynamic streaming technology of VR videos has negative impact on NDNs in-network caching performance. The caching of VR videos need larger storage space because of their $360^o \times 180^o$ panoramic views. Furthermore, since segments of different FoVs have different names, variation in viewpoint of users causes lower cache hit ratio. Therefore, in this paper we propose an integrating hotspot-based and popularity-based cache policy to cache the contents which are most likely to be requested.

A. Hotspot-Based

Hotspot represents the region which contains the most information or can attract the most attention in a $360^o \times 180^o$ panoramic view. For example, in a large-scale concert, the hotspot is the scene on the stage. Most of the attention of the audience will focus on the performance of stars on the stage, and it's almost impossible to look right and left all the time. The routers will preferentially cache the segments whose viewpoints are the closest to hotspot. However, sometimes the hotspot of a panoramic view is not only one, so in this paper we use h_{ij} to accurately describe the intrinsic value of video segments. Specifically, h_{ij} represents the intrinsic value of ith video segment which belongs to the jth viewpoint. We set the value range of h_{ij} as $(0, 1]$, that is, if the jth viewpoint of ith video segment is just a hotspot, then the value of h_{ij} is 1. Otherwise, the value of h_{ij} decreases with the distance between viewpoint and the nearest hotspot increasing. Assuming that H is the set of all the hotspots in a panoramic view, dis_{jh} represents the distance between jth viewpoint and hotspot h, and μ is a adjustable coefficient, then Eq. (1) shows the formula of h_{ij}.

$$h_{ij} = \begin{cases} 1 & j \in H \\ \mu \dfrac{1}{\min\limits_{h \in H} dis_{jh}} & j \notin H \end{cases} \tag{1}$$

B. Popularity-Based

In addition to hotspot, we also use the popularity of video segments to determine the cache policy. Popularity means how often a data segment is requested, which can be represented as the request frequency of a video segment over a period of time. It is worth noting that the popularity of every video segment keeps continuously changing with the transmission of video, so we design a data structure to record the popularity information in every router. Besides, each router needs to communicate with each other to count the global popularity [18]. In this paper we use p_{ij} to represent the global popularity of ith video segment which belongs to the jth viewpoint. The large value of p_{ij} means that this video segment is requested many times by users, so the caching of this segment is of more significance.

C. Cache Policy of VR Video

In this paper we propose an integrating hotspot-based and popularity-based cache policy to cache the contents which are most likely to be requested. We can get the value of h_{ij} and p_{ij} from above mentioned methods and we should integrate these two measures to make the final caching decisions. As shown in Eq. (2), when router receives a new data packet, it will replace the data with the minimal value of c_{ij} out. Specifically, c_{ij} is the weighted value of h_{ij} (hotspot-based) and p_{ij} (popularity-based). In this way, we can guarantee that the cached video data will be requested frequently and thus improving cache hit ratio.

$$\min_{i,j} c_{ij} = \min_{i,j} (\alpha h_{ij} + \beta p_{ij}) \tag{2}$$

VI. Evaluation

We simulate a VR live streaming scenario to verify how NDN smooths dynamic streaming. The topology is a tree structure and the tree depth is 4. We set the root of tree as video producer. The live video is transcoded to four streams at producer and 20 consumers request video data at the leaves. We assume players reserve data of 1 second. Figure 7 shows the retrieval delay of data after switch. As we can see, the retrieval delays of NDN distribute in three areas (20ms, 120ms, 200ms), indicating the three routers from consumers to publisher. In fact, most data is fetched from consumer's directly connected router. Compared to the delay of IP, that of NDN is much smaller. This means video streaming becomes smooth with the help of free-riding feature.

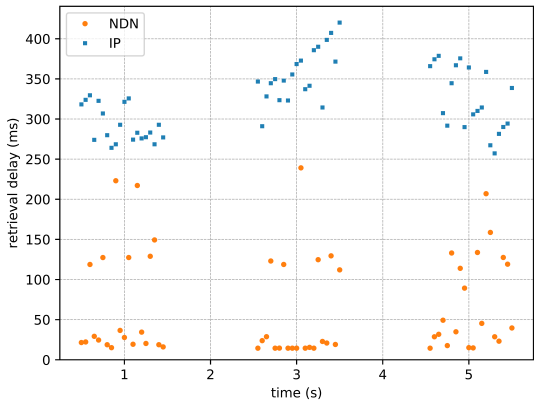

Fig. 7. Evaluation.

VII. Conclusion

In this paper, we exploit Named Data Networking to facilitate VR streaming. NDN naturally supports viewpoint-dependent VR streaming by naming the data explicitly with additional bonuses. In this paper, we design and implement a VR video delivery system over NDN. Note that cache is very important both in NDN and video distribution, so we propose a cache policy for VR video data, aiming to reduce transmission latency as much as possible and improve the user experience. Consequently, our system helps to avoid discomfort during VR video playback, and provides more immersive VR experience. The experimental results demonstrate that the transmission of VR video over NDN has a better performance than IP. What's more, our proposed cache policy for VR video is also more suitable for VR. In the future, we will extend our experiment in a larger network topology to evaluate the scalability of our system. We also plan to employ reinforcement learning to optimize our cache policy in a unstable network environment in the future.

VIII. Acknowledgement

This work has been financially supported by Shenzhen Key Fundamental Research Projects (Grant No.: JCYJ20170412150946024, JCYJ20170412151008290)

References

[1] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, K. Claffy, P. Crowley, L. Wang, L. Wang, and B. Zhang, "Named data networking," *Acm Sigcomm Computer Communication Review*, vol. 44, no. 3, pp. 66–73, 2014.

[2] K. Lei, C. Hou, L. Li, and K. Xu, "A rcp-based congestion control protocol in named data networking," in *International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, pp. 538–541, 2015.

[3] V. Jacobson, D. K. Smetters, N. H. Briggs, M. F. Plass, P. Stewart, J. D. Thornton, and R. L. Braynard, "Voccn: voice-over content-centric networks," in *The Workshop on Re-Architecting the Internet*, pp. 1–6, 2009.

[4] B. Goode, "Voice over internet protocol (voip)," *Proceedings of the IEEE*, vol. 90, no. 9, pp. 1495–1517, 2002.

[5] Z. Zhu, S. Wang, X. Yang, V. Jacobson, and L. Zhang, "Act: audio conference tool over named data networking," in *ICN*, 2011.

[6] D. Kulinski and J. Burke, "Ndnvideo: Random-access live and pre-recorded streaming using ndn," 2012.

[7] L. Wang, I. Moiseenko, and L. Zhang, "Ndnlive and ndntube: Live and prerecorded video streaming over ndn," 2015.

[8] P. Gusev and J. Burke, "Ndn-rtc: Real-time videoconferencing over named data networking," in *International Conference on Information-Centric NETWORKING*, pp. 117–126, 2015.

[9] G. Rossini and D. Rossi, "Exploiting topology knowledge in information centric networks: Guidelines and challenges," *IEEE Multimedia Communications Letter*, 2013.

[10] G. Piro and V. Ciancaglini, "Enabling real-time tv services in ccn networks," *IEEE COMSOC MMTC E-Letter*, vol. 8, no. 4, 2013.

[11] I. Moiseenko, L. Wang, and L. Zhang, "Consumer / producer communication with application level framing in named data networking," in *International Conference on Information-Centric NETWORKING*, pp. 99–108, 2015.

[12] J. Lindblm, M. Huang, J. Burke, and L. Zhang, "Filesync / ndn: Peer-to-peer file sync over named data networking," *Cs.ucla.edu*, 2013.

[13] W. Fu, H. B. Abraham, and P. Crowley, "isync: a high performance and scalable data synchronization protocol for named data networking," in *ACM Conference on Information-Centric NETWORKING*, pp. 181–182, 2014.

[14] Z. Zhu and A. Afanasyev, "Let's chronosync: Decentralized dataset state synchronization in named data networking," in *IEEE International Conference on Network Protocols*, pp. 1–10, 2014.

[15] Z. Zhu, C. Bian, A. Afanasyev, V. Jacobson, and L. Zhang, "Chronos: Serverless multi-user chat over ndn," 2012.

[16] Z. Zhu, A. Afanasyev, and L. Zhang, "Chronoshare: a new perspective on effective collaborations in the future internet," 2013.

[17] C. Westphal, C. Westphal, and C. Westphal, "Vr video conferencing over named data networks," in *The Workshop on Virtual Reality and Augmented Reality Network*, pp. 7–12, 2017.

[18] C. Bernardini, T. Silverston, and O. Festor, "Mpc: Popularity-based caching strategy for content centric networks," in *IEEE International Conference on Communications*, pp. 3619–3623, 2014.