UAV-IoT for Next Generation Virtual Reality

Jacob Chakareski

***Abstract*— We investigate UAV-IoT data capture and networking for remote scene virtual reality (VR) immersion. We characterize the delivered immersion fidelity as a function of the assigned UAV-IoT capture/network rates and study the optimization problem of maximizing it, for given system/application constraints. We explore fast reinforcement learning to discover the best dynamic UAV-IoT network placement over the scene of interest to maximize the expected remote immersion fidelity. We design scalable source-channel viewpoint coding to maximize the expected reconstruction fidelity of the data captured at every UAV location at the ground-based aggregation point. Finally, we explore layered directional networking and rate-distortion-power optimized embedded scheduling methods to effectively transmit the encoded data and overcome network transients that lead to packet buffering, which represent the fourth system component of our framework. Experimental results demonstrate considerable performance efficiency gains enabled by each system component over the respective state-of-the-art reference methods, in delivered VR immersion fidelity, application interactivity/playout latency, and transmission power consumption.**

***Index Terms*— UAV-IoT data capture and networking, virtual reality, reinforcement learning, remote immersion.**

I. Introduction

CYBER-PHYSICAL systems (CPS) and the Internet-of-Things (IoT) are set to play an increasingly prominent role in our society, advancing research and technology across diverse disciplines, at the same time. Virtual Reality (VR) and Unmanned Aerial Vehicles (UAV) are two emerging CPS technologies of prospectively broad societal impact. VR suspends our disbelief of being at a remote location (virtual or actual), akin to *virtual human teleportation* [1]. The flurry of related devices, services, and platforms will play a major role in charting the emerging global IoT framework [2] that aims to integrate online real-time sensor measurements and device control in diverse industrial, commercial, and societal application domains. Networked VR applications are expected to represent the bedrock of the anticipated 5G tactile Internet ecosystem [3]. UAV-IoT can have a similar transformative impact on remote monitoring/data acquisition applications, by lowering their cost and extending their scope [4].

The paper investigates a system framework for UAV-IoT networking for remote VR immersion, as illustrated in Figure 1. The UAV-IoT network is spatially distributed over the remote scene of interest, capturing different viewpoints V_i of it. The UAVs are linked to a ground/air-based station over direct network links of shared aggregate data rate C. The links relay the captured data towards the aggregation point and control information towards the drones. A powerful server computer located at the aggregation point constructs a viewport-driven immersive representation of the remote scene for a VR user, from the incoming data, and streams select VR views from it to the user, according to his viewport navigation actions. The immersion fidelity delivered to the user will depend on the space-time position of the UAV-IoT network, the network/acquisition rates R_i at which the data is captured/transmitted, and the user navigation actions. Maximizing it for the given system and application constraints raises multiple technical challenges due to present technology limitations [5] and requires devising novel holistic approaches to *capture, coding, networking, and reconstruction* of VR data.

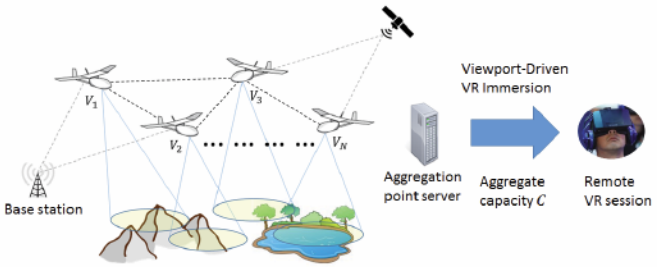

Fig. 1. UAV-IoT networking for next generation VR applications.

The proposed system framework comprises four integrated components that advance the state-of-the-art in UAV networking in their domains of operation and collectively extend its scope towards next generation applications, as follows:

(1) Formulation of the user viewport-driven remote scene VR immersion fidelity as a function of the assigned UAV-IoT capture/network rates, for given spatial UAV positions, and the optimization problem of maximizing it, given an aggregate network capacity C and VR application latency/interactivity constraints.
- Analysis that shows how the latter can be reformulated and solved efficiently as a convex optimization problem.

(2) Exploration of efficient reinforcement learning that discovers the best dynamic UAV-IoT network placement over the scene of interest, to maximize the expected VR immersion fidelity delivered over time.
- Analysis that enables the learning to balance effectively the exploration of new UAV network positions and the exploitation of already examined positions, in the search for an optimal dynamic UAV placement policy, to ensure its rapid convergence and maximization of the immersion fidelity reward it pursues.

Manuscript received October 8, 2018; revised April 21, 2019; accepted May 30, 2019. Date of publication June 14, 2019; date of current version August 30, 2019. This work was supported in part by the National Science Foundation (NSF) under Award CCF-1528030, Award ECCS-1711592, Award CNS-1836909, and Award CNS-1821875, and by research gifts and an Adobe Data Science Award from Adobe Systems. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Marta Mrak.

The author is with the Ying Wu College of Computing, New Jersey Institute of Technology, Newark, NJ 07103 USA (e-mail: jakov@jakov.org).

Digital Object Identifier 10.1109/TIP.2019.2921869

(3) Scalable joint source-channel viewpoint coding that maximizes the expected reconstruction fidelity of the data captured by each UAV, at the aggregation point.

(4) Layered directional networking and rate-distortion-power optimized embedded scheduling for effective transmission of the encoded data.

The remainder of the paper is organized as follows. First, we review related work in Section II. Then, we outline our system framework in Section III and describe in detail its four major components in Sections IV-VIII. Finally, we carry out an experimental evaluation in Section IX, outline future work in Section X, and conclude in Section XI.

II. Related Work

UAV IoT networking for remote VR immersion is a new topic. Related areas include ground-based multi-view sensing [6], immersive telecollaboration [7], multi-view video coding/communication [8]–[11], 360° on-demand video streaming [12]–[14], and online gaming [15], [16].

In particular, in ground-based multi-view sensing a scene of interest is captured from multiple collocated static cameras, to enable a remote user to dynamically switch between the different camera viewpoints and enhance his quality of experience. In immersive telecollaboration, a user is being sent a 3D view of the person(s) on the other end, captured by a static stereo-camera. In multi-view coding, multi-perspective data is compressed by leveraging the spatial correlation existing between the different capture locations. In 360° on-demand video streaming, a remote scene is captured off-line by an omnidirectional camera. The content is then streamed on-demand to provide a user with a 360° look-around of the scene on his VR headset. In online gaming, synthetic computer generated content is delivered to the user, who can experience it on a traditional 2D display or a VR device.

Our setting involves aerial multi-camera capture enabled by a dynamic UAV IoT network, to synthesize online a remote scene VR immersion for a user. Thus, our system setting and application constraints are different and more restrictive, involving more limited resources (e.g., network bandwidth) and higher application requirements (e.g., latency). To highlight this, we note that conversational video requires a 150ms round-trip-time interactive latency, while online VR requires a 20-30ms interactive delay. In our approach, we partially offset this stringent requirement by streaming from the ground-based server a temporal segment of data that integrates select VR views most likely navigated by the user over that segment.

Ensuring area coverage via aerial sensors has been studied for surveillance/exploration in the control/robotics community, e.g., in [17], aiming to have a sensor at every location of interest or minimize the sensor travel time across them. UAV-enabled network access over an area represents another instance of such a coverage problem [18]. More recent coverage work considers imaging aspects, e.g., the number of acquired pixels per unit area covered [19]. In contrast, *our objective is different and more challenging*: To select, capture, and network over space and time the best scene viewpoints for remote VR immersion. Similarly, weighted random walks have been used in robotics before [20]–[22]. However, the present study is the first to formally characterize/control the trade-off between achieved expected reward and policy convergence time of reinforcement learning decision policies in this context.

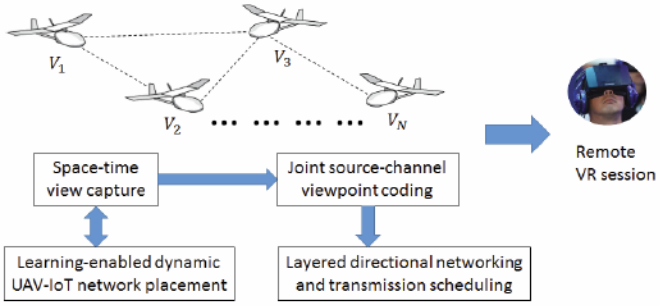

Fig. 2. System architecture: Main component blocks.

Advances have been made in video streaming in the areas of packet scheduling, multi-path routing, error resilience, proxy caching, and network multi-cast/broadcast [23]–[31] with more recent attention on buffer management for MPEG DASH [32], [33], the current streaming standard. Our paper leverages video streaming concepts, however, under new and more challenging operation settings and application objectives, as noted earlier.

Finally, the present paper is motivated by an earlier preliminary study we carried out in [34]. Much more extensive analysis, modeling, results/insights, and experimentation, as well as new system components of scalable joint source-channel viewpoint coding, layered directional networking, and rate-distortion-power optimized embedded scheduling, introduced for further performance enhancement, are some of the advances that the present paper introduces relative to [34].

III. System Framework Outline

Our framework comprises four integrated components, as illustrated in Figure 2. Efficient online learning decides on the optimal dynamic placement of the UAV-IoT network over the scene of interest that maximizes the immersion fidelity delivered to the VR user, for the given system resources and application constraints. The learning interacts closely with another system component that selects the optimal UAV data capture rate over time and considered spatial locations. The acquired data is efficiently represented using joint source-channel viewpoint coding to maximize its reconstruction fidelity at the aggregation point. Finally, the encoded data is effectively transmitted using layered directional networking and rate-distortion-power optimized transmission scheduling, which represent the fourth system component of our architecture, as illustrated in Figure 2.

In the following, we describe in detail each system component and the related analysis and optimization techniques that we formulate therein.

IV. Space-Time View Capture

Let $V = \{V_1, \ldots, V_N\}$ be the collection of aerial viewpoints captured by the UAV IoT network. For every UAV i (the association between a captured viewpoint V_i and UAV i is

unique), let R_i denote its assigned network rate, i.e., the data rate at which UAV i can transmit its data. Note that R_i also represents the temporal (data) sampling rate that UAV i uses to capture the respective viewpoint V_i. The aggregate transmission capacity the UAV network can use to send the captured data is C. Similarly, there are transmission rate constraints R_i^c, i.e., $R_i \leq R_i^c, \forall i$, which stem from the individual UAV locations relative to the back-end station towards which the captured data is relayed. A powerful server computer located at the back-end station aggregates the incoming data to construct a VR representation of the scene, denoted as $\mathcal{V}$, from which select viewpoints are streamed to the user, according to his navigation patterns. $\mathcal{V}$ comprises V, i.e., $V \subset \mathcal{V}$, as well as additional virtual (non-captured) viewpoints of the remote scene that are synthesized for the user from the captured data, i.e., V. The synthesis process is carried out by the server using geometric signal processing,[1] i.e., a procedure known as depth-image-based rendering (DIBR) [37]. Finally, let $R = (R_1, \ldots, R_N)$ denote the aggregate capture rate vector for the UAV-IoT network.

Next, we formulate the expected viewport-driven immersion fidelity delivered to the user, as our optimization objective. In particular, let Q_v denote the reconstruction fidelity/quality of an arbitrary viewpoint $v \in \mathcal{V}$ (virtual or captured), and let γ_v denote the respective likelihood of the user accessing/requesting this viewpoint during his/her navigation of the remote scene.[2] We can then characterize the expected VR immersion fidelity of the remote scene, as experienced by the user, as $Q_I = \int_{v \in \mathcal{V}} \gamma_v Q_v(R)$.

The problem of interest can then be formulated as

$$\max_R Q_I, \text{ subject to: } \sum_i R_i \leq C, R_i \leq R_i^c, \quad \forall i, \mathcal{A}, \quad (1)$$

where $\mathcal{A}$ captures the VR application play-out and interactivity delay constraints. Note that R in essence captures the viewpoint locations V across time and space, simultaneously. In particular, $R_i = 0$ signifies that signal/location V_i is not captured (at all).

V. Analysis and Optimization

Let v denote a virtual viewpoint that the back-end server can synthesize for the user and let V_i and V_j denote the two captured viewpoints closest to v in the aggregate view space $\mathcal{V}$. The application server will synthesize v from $V_i, V_j \in V$ using DIBR, as explained earlier.[3] Leveraging our recent work [9], we characterize the inverse of $Q_v(R)$, i.e., the reconstruction error/distortion $D_v(R)$[4] for virtual viewpoints $v \in \mathcal{V}$ and a given capture/network rate vector R, as a linear function of $D_{V_i}(R_i)$ and $D_{V_j}(R_j)$, the reconstruction errors/distortions of viewpoints V_i and V_j, each multiplied by polynomial powers of x, the relative position of v with respect to V_i and V_j in the view space $\mathcal{V}$, as follows:

$$D_v(R) = \sum_{k \in \{i,j\}} \left(\sum_{n=0}^{3} a_{k,n} x^n \right) D_{V_k}(R), \quad (2)$$

where $a_{k,n}$, for $k \in \{i, j\}$, denote the respective polynomial model coefficients for the captured viewpoints V_i and V_j.

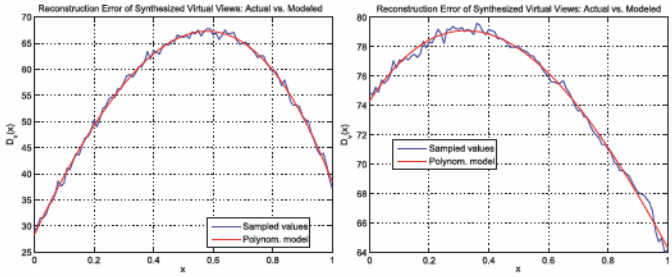

Fig. 3. Reconstruction error of virtual viewpoints $D_v(x)$: Actual and modeled values (according to the polynomial model).

In Figure 3, we show two examples of the reconstruction error $D_v(R)$, for virtual views v synthesized at different relative locations x, in the case of two different pairs of captured views V_i and V_j, selected to be used in the synthesis process via DIBR. We can see a close agreement between the actual values and the values predicted by the polynomial model. The coefficients $a_{k,n}$ are computed such that the prediction error (the squared error between the actual and modeled values) is minimized, using linear least-squares estimation.

Leveraging this advance, we can then reformulate (1) into an equivalent minimization problem, as follows

$$\min_R D_I = \min_R \int_{v \in \mathcal{V}} \gamma_v D_v(R) = \min_R \sum_i \alpha_i D_i(R_i), \quad (3)$$
$$\text{subject to: } \sum_i R_i \leq C, R_i \leq R_i^c, \forall i, \mathcal{A},$$

where to simplify the notation, we used $D_i(R_i) = D_{V_i}(R_i)$ (the reconstruction error of captured viewpoint V_i). The last equality in the objective in (3) is enabled by our polynomial model for $D_v(R)$ in the case of virtual viewpoints $v \in \mathcal{V}$. In particular, the coefficients $\alpha_i > 0$ in the resulting expression therein are obtained by grouping terms multiplying each factor $D_{V_i}(R_i)$, when the model for $D_v(R)$ is introduced into the expression preceding that equality.

Finally, our last analytical advance here is to characterize the functions $D_i(R_i)$ as exponential functions, i.e., $D_i(R_i) = a_i e^{b_i R_i}$. The accuracy of this approximation is shown in Figure 4 for three different captured viewpoints $V_{i_1}, V_{i_2}, V_{i_3} \in V$. Modeling $D_i(R_i)$ via exponential dependencies is intuitive and meets fundamental coding theory rate-distortion postulates.

The model parameters a_i and b_i are computed such that the squared fitting error is minimized. It can be seen from Figure 4 that this analytical model for $D_i(R_i)$ is quite accurate.

Finally, armed with this last advance, we can solve (3) using efficient convex optimization methods [39]. In particular, we introduce the model $D_i(R_i)$ into the last objective function in (3) to transform it into a convex function, since the coefficients α_i are non-negative and a sum of non-negative weighted

[1]This operation requires scene depth signals that can be captured via IR sensors [35] or estimated from the captured viewpoints (color signals) [36].

[2]This quantity stems from the user view navigation patterns.

[3]A higher number of captured views can be used to synthesize v, however, the synthesis gains rapidly diminish after two, as shown in [38].

[4]There is a one-to-one mapping between $Q_v(R)$ and $D_v(R)$.

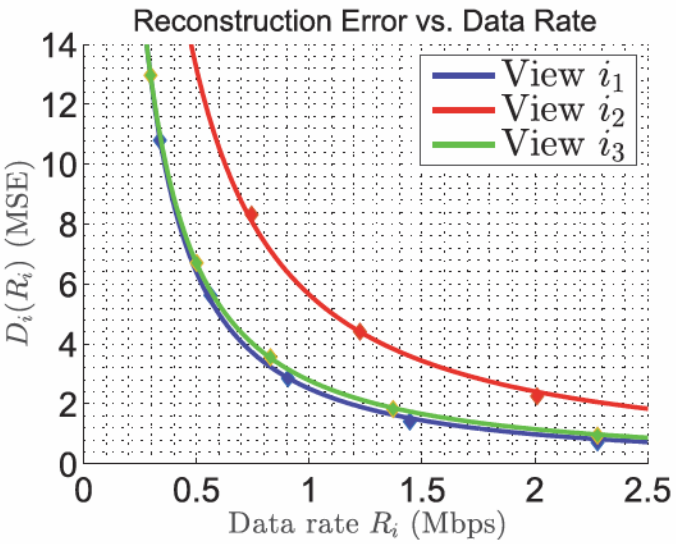

Fig. 4. Reconstruction error $D_i(R_i)$: Sampled vs. analytical values.

convex functions is convex [39]. Moreover, the constraints in (3) are linear functions,[5] hence they are also convex. Thus, (3) becomes a convex optimization problem.

VI. Dynamic UAV IoT Network Placement

We consider that the UAVs can capture viewpoints of the remote scene of interest from a collection of locations (way-points) $\{p_i\}$ that we model as a graph, as illustrated in Figure 5. Due to the scene characteristics and topology, we assume there is a non-uniform cost c_{ij} of traversing the link between every two adjacent locations p_i and p_j.

We discretize time and consider that at every time slot t, a drone can either stay at its current position (hover) or move to one of the neighboring locations (transition). Let S_t denote the current configuration (state) of the UAV network across the graph. At the onset of the next time slot $t+1$, the UAVs can collectively transition into another state denoted as S_{t+1}. The values that S_{t+1} can attain depend on S_t. Let $a : S_t \mapsto S_{t+1}$ denote the state transition action the UAV network will take at time $t+1$. Furthermore, let $r(S_t, a)$ denote the reward the network will earn at time $t+1$ by carrying out a, given S_t. We formulate $r(S_t, a)$ as $u(S_t, a) - \beta c(S_t, a)$, where $u(S_t, a)$ represents the utility the network will achieve by taking action a, given the current state S_t, and $c(S_t, a)$ denotes the respective cost, accounted for as the sum of all cost factors c_{ij} activated by executing a, given S_t. Finally, the parameter β trades achieved utility for induced cost.

In our case, $c(S_t, a)$ captures the energy the network will expend to move from state S_t to state S_{t+1} over the scene. The utility $u(S_t, a)$ represents the resulting VR immersion fidelity delivered to the user, as introduced earlier, i.e., $u(S_t, a) = \int_{v \in \mathcal{V}} \gamma_v Q_v(S_{t+1}, R)$. The state-space comprising every possible UAV network configuration can be designed such that it excludes undesirable or unfeasible states S_t, e.g., featuring multiple UAVs occupying the same spatial position.

We are interested in finding the dynamic network placement policy $\pi : S_t \mapsto a$ that maximizes the cumulative reward

[5] Inclusive of $\mathcal{A}$, which can be represented as a set of linear constraints.

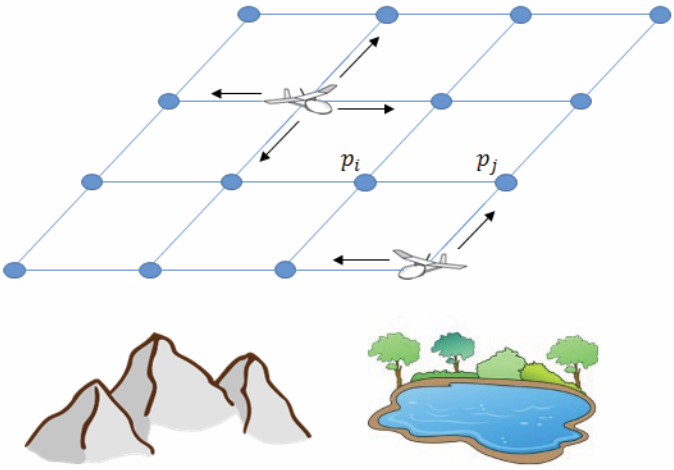

Fig. 5. Dynamic UAV IoT network placement.

$\rho(\pi) = \sum_t r(S_t, a)$. One major challenge in identifying such a policy is the necessarily online nature of π, which will have to explore unvisited network configurations for new information (reward values) and exploit already gathered information (known reward values), at the same time. In particular, leveraging only already-visited states may give us a certain level of guaranteed reward and performance, however, at the prospective penalty of missing out on unvisited states that may lead to even higher rewards. Simultaneously, exploring new states also carries out the risk of encountering lower reward values, which would degrade performance instead. Thus, *an effective balance between information exploitation and exploration* will need to be integrated into the search for π. This necessitates leveraging online machine learning denoted as reinforcement learning [40], which provides a natural paradigm for addressing problems of this nature.

Another major challenge that arises here is the discrete complex nature of this problem, characterized by *an excessive search space*. To address these challenges more effectively, we design π as a probabilistic policy, where every prospective action a is selected with probability $p_a \triangleq p(a|S_t)$ (having an expected reward objective $E[\rho(\pi)] = \sum_t \sum_a p_a r(S_t, a)$). This will enable us to explore a richer policy space for π and pursue an effective solution that leverages concepts from reinforcement learning, random walks, spectral graph theory, and convex optimization, as follows.

Let $G = (\mathcal{S}, \mathcal{E})$ denote the state-space graph of the problem, where $\mathcal{S}$ denotes the set of all (possible) UAV network states over the scene and $\mathcal{E}$ denotes the state-space graph edges that indicate the prospective transitions between any two states s_i and s_j in $\mathcal{S}$. Stochastic control problems of this nature are typically investigated within the framework of Markov Decision Processes (MDP), where one formulates a state-value function $V^\pi(s), \forall s \in \mathcal{S}$, and a decision policy π. In our case, $V^\pi(s) = E[\rho(\pi)] = \sum_t \sum_a p_a r(S_t, a)$. The optimal state-value function $V^*(s)$ needs to satisfy the Bellman equation [41], which in our case, we formulate as

$$V^*(s) = \max_{p(\cdot|s)} \left(\sum_a p(a|s)(r(s,a) + V^*(a)) \right), \qquad (4)$$

where the optimal stochastic policy $\pi : s \triangleq p(\cdot|s)$ is the argument that maximizes the right hand side in (4). There are algorithms to solve (4) such as Policy Iteration or Value Iteration [42]. However, they assume a priori knowledge of the reward function $r(s, a), \forall s, a$, which in our case does not hold. Thus, we need to solve (4) in an *online fashion*, taking actions in parallel as we learn the reward function and the optimal decision policy. We carry this out using the paradigm of reinforcement learning (RL) [40], as follows.

First, we can show that π is equivalent to a non-negative symmetric weight matrix W that defines a weighted random walk on G. We can also show that W in turn induces a corresponding reversible Markov chain on G. To conserve space, we omit the derivation of these results here and refer the reader to [43] for its details.[6] Thus, finding the optimal π is equivalent to finding the corresponding W. To avoid the challenges of intractability and lack of performance guarantees, arising if solving for W directly, we leverage the approach of parameterized policy gradient [40], to parameterize W using a vector $\theta = (\theta_1, \ldots, \theta_m)$ and a dictionary of symmetric weight matrices W_i, such that $W = \sum_i \theta_i W_i$, where $\sum_i \theta_i = 1$. This will then enable us to solve the problem under consideration exactly, by utilizing derivatives and a reformulation, as described in the following. Let the expected reward associated with the decision policy induced by θ be denoted as $\rho(\theta)$.

Conventional MDP theory requires an infinite execution time for an optimal policy so that an expected reward maximization is ensured [42]. Similarly, state-of-the-art RL techniques are typically very slow to converge to the optimal policy of interest [40], [46], [47]. In our case, we want to compute θ that maximizes $\rho(\theta)$ as quickly as possible, due to the interactive low-latency nature of the VR application and for faster scene dynamics adaptation. Moreover, we want to be able to formally characterize/control the trade-off between tolerable policy convergence time and approximate maximum expected reward earned, which is induced thereby. Let P_θ denote the transition probability matrix of the equivalent Markov chain induced by W. How quickly our policy achieves the maximum expected reward is equivalent to how quickly the chain converges to its stationary distribution π_θ over $\mathcal{S}$.

We quantify the latter precisely via the mixing time t_{mix} of the chain, which represents the smallest number of steps n to run the chain over G to achieve an ϵ-approximation of π_θ [43]. In turn, t_{mix} is upper bounded by the second largest magnitude eigenvalue of the transition matrix P_θ denoted as $\mu(P_\theta)$, as $t_{\text{mix}}(\epsilon) \leq \frac{\kappa(\epsilon)}{1-\mu(P_\theta)}$ [43], where $\kappa(\epsilon)$ is a constant. For simpler notation, we define $\mu(\theta) \triangleq \mu(P_\theta)$ and use it going forward, to denote this quantity. Thus, $\mu(\theta)$ governs for how long our policy needs to run to achieve the maximum reward. Hence, we can effectively control the latter via the former. Given these advances, we can formulate in two ways our decision policy optimization:

$$\textbf{P1}\text{: Maximize: } \rho(\theta), \tag{5}$$

$$\text{subject to: } \mu(\theta) \leq \delta, \quad \theta_i \geq 0, \sum_{i=1}^{m} \theta_i = 1. \tag{6}$$

$$\textbf{P2}\text{: Maximize: } \rho(\theta) - \lambda\mu(\theta), \quad \text{subject to: } \theta_i \geq 0, \sum_{i=1}^{m} \theta_i = 1, \tag{7}$$

where in **P1**, (6) ensures that the computed policy is sufficiently fast. Our objective in **P2** is to use λ to control the trade-off between the achieved expected reward and policy convergence rate, for effective adaptation to scene dynamics and meeting tight application constraints.

Next, we derive expressions for $\rho(\theta)$ and $\mu(\theta)$. Let $p_\theta(s, i)$ be the probability that we select policy W_i at state s, given θ. Then, $p_\theta(s, i)$ can be computed as

$$p_\theta(s, i) = \frac{\theta_i H_i(s)}{H(s)}, \forall s, i, \quad \text{where } H_i(s) = \sum_j \mathbf{W}_i(s, j), H(s) = \sum_j \mathbf{W}(s, j). \tag{8}$$

Since an action a may appear in multiple policies W_i, given state s, the probability that it is selected can be computed as $d_\theta(s, a) = \sum_{i:s\to a} p_\theta(s, i)$. Similarly, we formulate $\pi_\theta(s) = H(s)/\sum_{s\in\mathcal{S}} H(s)$. Consequently, the expected reward can be computed as:

$$\rho(\theta) = \sum_s \pi_\theta(s) \sum_a d_\theta(s, a) r(s, a). \tag{9}$$

Furthermore, we formulate $\mu(\theta)$ leveraging the following properties from matrix theory [48]. $\mu(\theta)$ is the second largest magnitude eigenvalue of the matrix P_θ, whose (first) largest magnitude eigenvalue is $\lambda_1(P_\theta) = 1$, with a corresponding eigenvector $\mathbf{1}$. We construct the symmetric matrix $A = D_\theta^{-1/2} W D_\theta^{-1/2} - \sqrt{\pi_\theta}\sqrt{\pi_\theta}^T$, where D_θ is a diagonal matrix with diagonal elements taken from the stationary distribution π_θ. Then, $\mu(\theta)$ also represents the (first) largest magnitude eigenvalue of A and we can compute it as the spectral norm $\|\cdot\|_2$ or maximum singular value of A, since A is a symmetric matrix [49]. Thus, we write

$$\mu(\theta) = ||D_\theta^{-1/2} W D_\theta^{-1/2} - \sqrt{\pi_\theta}\sqrt{\pi_\theta}^T||_2. \tag{10}$$

Finally, we can show that $\rho(\theta)$ and $\mu(\theta)$ are convex functions of θ. In particular, $\rho(\theta)$ is a linear function of θ, hence, it is convex. Similarly, from (10), it follows that $\mu(\theta)$ is the norm of an affine function of θ, hence, it is also convex [39]. Thus, we can solve **P1** and **P2** efficiently using tools from convex optimization [39].

We integrated our analytical advances into an effective reinforcement learning framework that updates the policy θ as new knowledge (reward values $r(s, a)$) is acquired via exploration. The learning is carried out in batches (learning epochs) comprising N policy execution steps followed by one policy update step, which are repeated until convergence. In particular, we designed a gradient descent with projection onto convex sets algorithm (FEEL) to solve **P2**, shown in Algorithm 1. As the gradients $\nabla\rho(\theta)$ and $\nabla\mu(\theta)$ can be computed in closed form, FEEL is implemented effectively.

[6] When the optimal policy π does not fall into the class of reversible policies, matrix reversibilization can be leveraged to approximate π with strong probabilistic bounds [44], [45].

Algorithm 1 Fast Explore and Exploit Learning (FEEL)

Require: $\theta^0, \alpha, \lambda, k = 0, \epsilon$, initial state s_0
1: **repeat**
2: Execute the parameterized policy W for N epochs.
3: Observe and update the earned reward $r(s, a)$.
4: Compute $\nabla\rho(\theta)|_{\theta^k}$ and $\nabla\mu(\theta)|_{\theta^k}$
5: Update $\theta^{k+1} = \theta^k - \alpha_k \left(\nabla\rho(\theta)|_{\theta^k} - \lambda_k \nabla\mu(\theta)|_{\theta^k}\right)$
6: Increment iteration counter $k = k + 1$
7: Project θ^k back to feasible set (if necessary).
8: **until** *convergence* (Change in objective function $< \epsilon$)

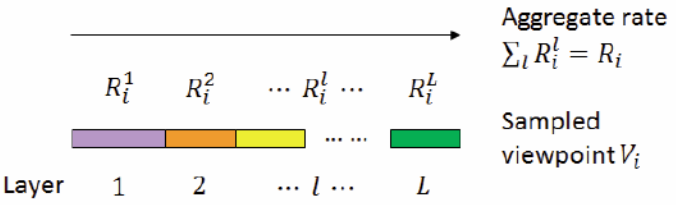

Fig. 6. Scalable captured viewpoint V_i data coding.

α_k in Line 5 represents the learning rate of the algorithm that can be adapted over time (iterations).

Moreover, the regularizer λ_k is systematically updated at every iteration k such that it emphasizes reward structure exploration initially and gradual transition to reward maximization subsequently, for fast policy convergence. In particular, λ_k is set to large values early on, to encourage exploration, and is then decreased over time to ensure Algorithm 1 returns an optimal policy θ that maximizes the expected reward $\rho(\theta)$.

We rigorously derive the necessary conditions on α_k and λ_k to ensure convergence of Algorithm 1. We omit the details of the derivation here to conserve space. In brief, they need to meet the following conditions: $0 \leq \alpha_k = \alpha < 1/L$ (a constant learning rate is sufficient) and $\lambda_k \geq 0, \sum_k^\infty \lambda_k < \infty$. L here denotes the Lipschitz constant for the gradient $\nabla\rho(\theta)$ [50]. One option for the regularizer that we have examined in our experimentation is a/k^2, for a small constant a.

VII. Scalable Source-Channel Viewpoint Coding

To address the impact of unreliable/dynamic aerial network links, we design an efficient scalable source-channel viewpoint coding for the collection of viewpoints $\{V_i\}$ captured by the UAV IoT network, as follows. First, the captured V_i data is encoded by the respective UAV into a set of embedded layers, illustrated in Figure 6, featuring incrementally increasing levels of data fidelity with the layer index l [51]. This will allow for inherent adaptation of the transmitted data to prospective bandwidth variations on the aerial network links, as the former is communicated towards the aggregation point.

Second, to protect against prospective transmission errors (packet loss) on the aerial links, each UAV applies efficient rateless random linear coding [52] to its encoded viewpoint data, using the construction procedure from Figure 7.[7]

In particular, L transmission windows are constructed, where window l represents the aggregate collection of the first

[7]Developed using a related illustration from [11], courtesy of the authors.

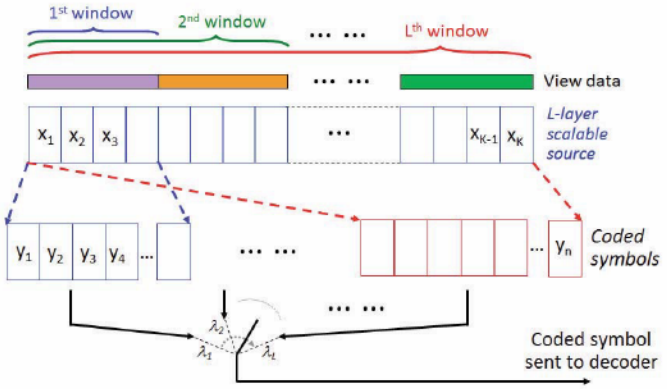

Fig. 7. Rateless random linear source-channel view coding.

l layers of the encoded captured viewpoint. Coded symbols are constructed from each window using random linear combinations of the source symbols. Finally, a coded symbol is selected for transmission from window l with probability λ_l.

Our embedded signal representation offers three advantages: inherent adaptability to dynamic network links, minimum required redundancy (to overcome transmission loss),[8] and inherent unequal error protection, all due to its design. The optimization problem that each UAV i will consider here is

$$\min_{\{R_i^l\},\{\lambda_l\}} \sum_l E[D_i(\sum_{k=1}^{l} R_i^k), \{\lambda_l\}], \quad (11)$$
$$\text{subject to: } \sum_l R_i^l \leq R_i \text{ and } \sum_l \lambda_l = 1, \quad \forall k,$$

where R_i^l is the data rate of layer l and $E[D_i(\sum_{k=1}^{l} R_i^k)|\{\lambda_l\}]$ is the expected reconstruction error of viewpoint V_i, when the first l layers has been received/decoded at the destination.[9]

Solving (11) is complex due to its nature. We design a coordinate descent algorithm to solve it iteratively, at low complexity. In particular, for fixed $\{\lambda_l\}$, (11) transforms into

$$\min_{\{R_i^l\}} \sum_l E[D_i(\sum_{k=1}^{l} R_i^k), \{\lambda_l\}], \text{ subject to: } \sum_l R_i^l \leq R_i, \quad (12)$$

which again is a convex optimization problem that can be solved efficiently. The algorithm operates by adjusting $\{\lambda_l\}$ iteratively, in between every solution of (12), until convergence. The adjustment is carried out such that the objective in (12) is minimized, every time $\{\lambda_l\}$ is updated. A formal description of our algorithm is provided in Algorithm 2.

Convergence of the algorithm is ensured, as the objective in (11) is positive, bounded from below by zero, and monotonically decreasing at every iteration. We have empirically verified that convergence is typically very quick. Algorithm 2 carries out a coordinate descent search to find the optimal probability values $\{\lambda_l\}$, while computing the respective optimal source coding rates R_i^l. In comparison to a full-search approach, it offers practically the same performance,

[8]Our channel coding approach will allow UAV i to maintain its aggregate source-channel data rate close to R_i, i.e., it will result in very little overhead.

[9]The probability of this event can be denoted as $P_l(\{\lambda_l\}, R_i)$ [52] and thus $E[D_i^l(\sum_{k=1}^{l} R_i^k)|\{\lambda_l\}] = P_l(\{\lambda_l\}, R_i) D_i(\sum_{k=1}^{l} R_i^k)$.

Algorithm 2 Low-Complexity Joint Source-Channel Viewpoint Coding

Initialize $\{\lambda_l\} = \{1, 0, \ldots, 0\}$; Compute $\{R_i^l\}$ from (12)
Set $\Delta\lambda$; Assign D_{max} = objective in (11)
for $i = 1$ to $L - 1$ **do**
  FLAG1 = 0
  repeat
    $\lambda_i = \lambda_i - \Delta\lambda; \lambda_{i+1} = \lambda_{i+1} + \Delta\lambda$
    Compute $\{R_i^l\}$ from (12)
    Assign D = objective in (12)
    if $D < D_{max}$ **then**
      $D_{max} = D$
      FLAG1 = 1
    else
      $\lambda_i = \lambda_i + \Delta\lambda; \lambda_{i+1} = \lambda_{i+1} - \Delta\lambda$
      break
    end if
  until $\lambda_i \leq \Delta\lambda$
  if FLAG1 = 0 **then**
    break
  end if
end for
Return $\{\lambda_l\}, \{R_i^l\}$

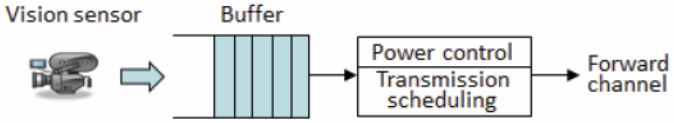

Fig. 8. Drone vision sensor transmission system.

for the number of source coding layers L examined in our experiments. Similar observations have been made in the past for analogous coordinate descent search algorithms applied to IPTV streaming and multi-view video coding [9], [28].

VIII. Layered Directional Networking and Packet Scheduling

We enhance the UAV rate-distortion-power transmission efficiency via smart multi-beam directional antennas [53], [54] and layered scheduling. Simultaneously, this will enhance further the interactivity and quality of experience of the enabled VR application, as it will reduce its data delivery latency. Moreover, prospective packet buffering due to network transients will be overcome more effectively.

We first model the transmission system of a UAV, as illustrated in Figure 8. Data captured by its vision sensor is scalable source-channel encoded, as explained earlier, and queued for transmission. Power control and transmission scheduling actions are carried out to decide the scalable packets to be transmitted next and the amount of transmit power to be used to carry that out.[10] Via receiver feedback, the transmitter is aware of the current forward channel quality.

The scalable source-encoded packets exhibit prediction dependencies, as illustrated in Figure 9, which are induced on them by the sensor encoder, to increase its compression efficiency. In particular, let there be K data units currently enqueued in the transmission buffer. Each data unit k is

[10]Simultaneously, this will help overcome network transients that may lead to unwanted packet buffering.

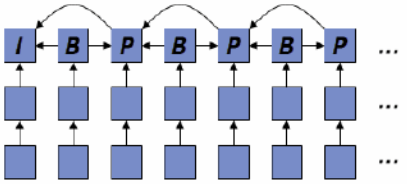

Fig. 9. Scalable data unit dependencies of an encoded viewpoint.

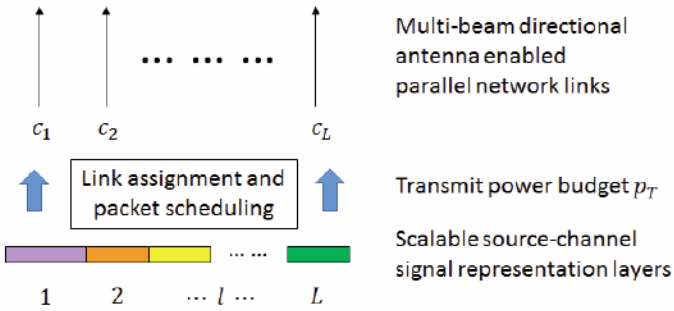

Fig. 10. Link assignment and transmit power/packet scheduling.

characterized with its delivery deadline $t_{k,d}$, size B_k in bytes, reduction in reconstruction error ΔD_k of the respective viewpoint that k will contribute to, if received and decoded on time, and two sets of ancestor and descendant data units $\{j \preceq k\}$ and $\{j \succeq k\}$, induced by the source encoding prediction dependency graph. $t_{k,d}$ represents the latest time by which k needs to be received, in order to be usefully decoded.

Now, let L denote the number of parallel beams/network links that the UAV antenna can establish towards its destination. Recall that L also denotes the number of scalable signal representation layers[11] constructed in Section VII. Let c_l denote the transmit power allocated to send a packet over link l. Coded packets from transmission window l in Section VII are assigned to the corresponding network link for transmission, as illustrated in Figure 10. Let p_T denote the available transmit power budget of UAV i. We consider that p_T is much larger than the minimum required power to transmit the aggregate capture rate R_i of UAV i. This is a reasonable assumption that any UAV battery can easily meet, given its much higher energy volume. In particular, the energy consumption of UAV propulsion is a few orders of magnitude greater than the energy a UAV consumes for data capture/communication [55].

We model the forward channel on each network link l as a packet erasure channel, with loss probability $\epsilon_l \triangleq \epsilon_l(h_l, c_l)$, where h_l denotes the current channel state/quality, as informed by receiver feedback, and transmission delay τ_l, which may be non-deterministic, with density $p_{\tau_l} \triangleq p_{\tau_l}(h_l, c_l)$. We consider that c_l can be selected from a finite set of power levels C.

Now, let $\pi = (\pi_1, \ldots, \pi_K)$ denote the packet transmission policy of the UAV, where $\pi_k \in \{\{l, 0\}, \{l, 1\}\}$ indicate the two possible choices of not sending or sending packet k from scalable layer l on network link l at present (current time t). Similarly, let $c = (c_1, \ldots, c_K)$ denote its per-packet transmit power control policy, where $c_k = 0$, for $k : \pi_k = \{l, 0\}$. Moreover, let $\epsilon(\pi_k) \triangleq P\{\tau_k > t_{k,d} + \alpha - t\}$ be the expected

[11]Selected to match the physical capabilities of the UAVs.

error of transmitting packet k under policy π_k on link l, where α captures the latency/interactivity requirements of the VR application. It can be computed as

$$\epsilon(\pi_k) = \begin{cases} 1, & \text{if } \pi_k = \{l, 0\}, \\ \epsilon_l + (1-\epsilon_l)\int_{t_{k,d}+\alpha-t}^{\infty} p_{\tau_l}, & \text{if } \pi_k = \{l, 1\}. \end{cases} \quad (13)$$

The expected reduction in viewpoint V_i reconstruction error under policy π can then be formulated as

$$D(\pi, c) = \sum_k \Delta D_k \prod_{j \preceq k} \left(1 - \epsilon(\pi_j)\right). \quad (14)$$

The corresponding transmission rate on each link l can be formulated as $R_{t,l}(\pi) = \sum_{k:\pi_k=\{l,1\}} B_k$. Finally, let $E_l(\pi, c) = \sum_{k:\pi_k=\{l,1\}} c_k B_k$ denote the consumed transmit power on network link l. The sender is interested in solving the following optimization problem

$$\max_{\pi,c} D(\pi, c), \quad (15)$$
$$\text{subject to: } R_{t,l}(\pi) \le R_i^l, \quad \forall l, \quad \sum_l E_l(\pi, c) \le p_T.$$

We study exact and approximate/lower-complexity solution strategies to solve (15). First, using the Generalized Lagrange multiplier method [56], we recast the problem as

$$J(\pi, c) = \min_{\pi,c} \Bigg(D_0 - D(\pi, c) + \sum_l \lambda_l R_{t,l}(\pi, c) + \lambda_{L+1} \sum_l E_l(\pi, c) \Bigg), \quad (16)$$

where $D_0 = \sum_k \Delta D_k$, $\lambda_l > 0$, for $l = 1, \ldots, L+1$, denote the corresponding Lagrange multipliers, and for mathematical convenience, we have introduced a minus sign in front of $D(\pi, c)$ to replace the max operator from (15) with a min operator. We can then compute the optimal policy via the Bellman iteration

$$(\pi^*, c^*)(q_i) = \arg\min_{\pi_i, c_i} \sum_{q_{i+1}} P_{\pi,c}(q_{i+1}|q_i) J_{\pi^*,c^*}(q_{i+1}), \quad (17)$$

where q_i is a state in the joint policy space of (π, c), uniquely described by the actions $\pi_1^*, \ldots, \pi_i^*$ and $c_1^*, \ldots, c_i^*$. $P_{\pi,c}(q_{i+1}|q_i)$ are state transition probabilities induced by (π, c), and $J_{\pi^*,c^*}(q_{i+1})$ is the optimal Lagrange cost that can be backtracked with an equivalent equation.

Second, we explore minimizing $J(\pi, c)$ iteratively, one policy pair (π_k, c_k) at a time. In particular, note that (15) and (16) represent discrete optimization problems that are complex to solve, due to their large state-space that requires an enumeration of the $2^K \times |C|^K$ choices that (π, c) can take on jointly. Thus, we also design a faster iterative algorithm that computes an approximate solution at lower complexity, as follows. Starting from an initial solution for (π, c), we iteratively solve (16) one variable pair (π_k, c_k) at a time, while keeping the others $((\pi_j, c_j)$, for $j \neq k)$ fixed, until convergence.

Precisely, let $(\pi^{(0)}, c^{(0)})$ denote an initial choice for the joint transmission scheduling/power control policy. Moreover, let $n = 1, 2, \ldots$ denote an iteration count. We select one policy pair $(\pi_k^{(n)}, c_k^{(n)})$ to optimize at iteration n, e.g., in a round-robin fashion, for $k = 1, \ldots, K$. We set (fix) the remaining policy pairs $(\pi_j^{(n)}, c_j^{(n)}) = (\pi_j^{(n-1)}, c_j^{(n-1)})$, for $j \neq k$, and compute the values of $(\pi_k^{(n)}, c_k^{(n)})$ that solve (16). We then increment n and move on to the next k, until $J(\pi^{(n)}, c^{(n)}) = J(\pi^{(n-1)}, c^{(n-1)})$.

Algorithm 3 Optimal Transmission Scheduling Policy

Require: $\pi_k^{(0)}, c_k^{(0)}, J^{(0)}, \lambda_l, n = 0, \theta$
1: **repeat**
2: $n = n + 1$; $k = (n \mod K)$
3: Solve (18) $\Rightarrow \pi_k^{(n)}, c_k^{(n)}$
4: Compute new objective $J^{(n)}$ from (16)
5: **until** *convergence* $(J^{(n-1)} - J^{(n)} < \theta)$

By grouping terms in (16) associated with (π_k, c_k), we can formulate the key step of the iterative optimization as

$$\pi_k^{(n)}, c_k^{(n)} = \arg\min_{\pi_k, c_k} S_k^{(n)} \epsilon(\pi_k, c_k) + \lambda \pi_k(2) B_k, \quad (18)$$

where $\lambda = \lambda_l + \lambda_{L+1} c_k$ and $S_k^{(n)}$ captures the overall impact of packet l on the reconstruction error of viewpoint V_i. Recall that l denotes the scalable layer/network link associated with packet k and $\pi_k(2) \in \{0, 1\}$ indicates the decision to transmit (or not) this packet over the link, as induced by its policy π_k. We derive $S_k^{(n)}$ from (16) as

$$S_k^{(n)} = \sum_{j \succeq k} \Delta D_k \prod_{\substack{m \preceq j \\ m \neq k}} (1 - \epsilon(\pi_m^{(n)})).$$

The optimization is formally summarized in Algorithm 3. Its convergence is guaranteed, as the objective function $J(\pi, c)$ is bounded from below and is monotonically non-increasing at every iteration.

IX. Experiments

A. Experimentation Methodology and Data Characteristics

We carry out experimental analysis to evaluate the effectiveness of each system component comprising our framework. The analysis leverages aerial viewpoint data captured in a controlled laboratory setting. We have also densely sampled the scene underneath using fixed ceiling-mounted vision sensors and used this data to verify the UAV captured data. In addition, we have used in our experimentation an aerial viewpoint data set captured in an outdoor environment [57]. To overcome the challenge of having remotely collocated camera locations in this data set, we only used a subset where multiple closely collocated viewpoints can be identified. Moreover, we have remedied this further by selecting temporally collocated viewpoints from the dataset as spatially adjacent viewpoints to be captured by the UAVs, as they move from one location to another, to facilitate view synthesis, assuming a fairly static remote sensing scene.

For reproducibility and rigorous analysis, our experimentation methodology was designed as follows. In the indoor setting, viewpoint data was acquired using DJI Matrice 100 UAVs [58] equipped with wide-panorama 4K cameras

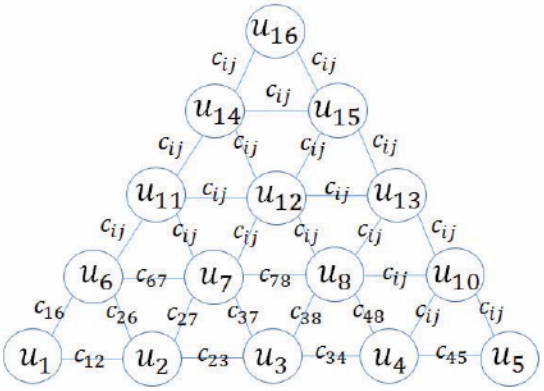

Fig. 11. MDP state transition graph with reward and cost values u_i, c_{ij} associated with states and state transitions indicated.

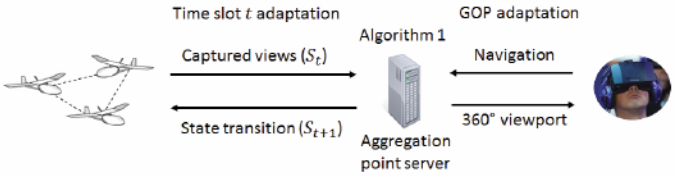

Fig. 12. Exchange of data and control information between the UAV network and the ground-based server that executes Algorithm 1.

and Wi-Fi dongles, communicating with a ground based Wi-Fi access point. In the outdoor setting, the UAVs are equipped with LTE 4G dongles instead, communicating with a ground-based base station. As this is a proof-of-concept study of an emerging application, we focused on a one-hop aerial network in our experiments. In both settings, we also collected measurements of the aerial wireless link to the ground-based access point or base station, associated with each prospective UAV location. These traces and the UAV captured data were then used to assess our system framework components via numerical simulations. The viewpoint popularity distribution γ_v was compiled based on traces of VR user head movements that we collected [59], [60].

Our channel coding method, like digital fountain codes, as well, represents a universal channel coding scheme for erasure channels [52], [61], [62]. Thus, its performance is only affected by the long-term average packet loss rate. This makes it sufficient to examine only the number of received packets at the receiver for each coding window, and hence a traditional packet erasure channel model has been used in our experiments.

In the implementation of Algorithm 2 that solves the optimal source-channel view coding rate allocation, we set $\Delta\lambda = 0.1$ as the step size used to discretize the probabilities λ_i. This was empirically proven to provide a good trade-off between complexity and performance. The number of scalable content layers L for an encoded view has been set to three. We also implemented a full-search method that examines exhaustively the solution to (11), for every possible triplet $\{\lambda_1, \lambda_2, \lambda_3\}$, and selects the optimal one. We established that the iterative local-search low-complexity solution provided by Algorithm 2 always coincides with the optimal solution.

The state transition graph that we used to evaluate the online learning-based UAV network placement strategy from Section VI is shown in Figure 11. We developed this graph as follows. We considered a 2D uniform grid of spatial positions in which a UAV can be placed, at a given height above the scene (see Figure 5 for an illustration). Two adjacent grid points are spaced apart for 50cm along either the x or y coordinates of the 2D grid. The grid size is 6×6. There are three UAVs moving collectively across the grid points. We selected a subset of 16 distinct configurations (states) that the UAV network can exhibit over the grid. They represent the states s_i of the MDP state transition graph indicated in Figure 11. To each s_i, we assigned the utility value $u_i := u(s_i)$, as defined in Section VI.

The utility values of states not visited earlier are learned by the UAV network through exploration, as explained in Section VI. We also captured the cost of transition $c_{ij} \triangleq c(s_i, s_j)$ between states s_i and s_j, as defined in Section VI, facilitating data on propulsion energy consumption for the specific UAVs we used [55]. Similarly, the transition costs of graph edges not traversed before are learned by the UAV network during exploration. To facilitate the numerical evaluation, we normalize the utility and cost values with their respective maximum values in the state graph. Together, the utility and cost values comprise the reward values $r(s_i, s_i \rightarrow s_j)$ defined in Section VI. The online learning optimization is carried out at the aggregation point server computer attached to the ground-based access point or base station, as illustrated in Figure 1.

We generate the dictionary weight matrices W_i as binary matrices comprising nonzero entries only at a random subset of indices ij associated with neighboring states s_i and s_j in the state transition graph. We empirically selected the dictionary size m to identify the best trade-off between performance and computational complexity. We noted that beyond $m = 6$, learning performance did not gain further improvement, while computing complexity was still fairly modest, as expected, given the small scale of the state graph.

Given the speed at which our UAVs can move and capture data (50mph [63], [64]), and the speed at which an update of the optimal navigation policy θ can be computed by Algorithm 1 (<2ms), we selected 200ms as the time slot duration. The learning is implemented at the aggregation server that receives captured data from the UAVs at each time slot t and sends back control information in return that indicates the next state S_{t+1} the UAV network should transition to. The number of epochs N for which the policy is executed, between every two policy updates, as described in Algorithm 1, is set to ten.

Figure 12 shows a high level illustration of the deployment of Algorithm 1 in our setting. The UAV network sends data (captured views) associated with its present state S_t. The server in return sends back control information – the new state S_{t+1} that the network should take on at the next time slot. For illustration, Figure 12 also highlights the interaction between the server and the VR client, where navigation data is sent by the client, while encoded 360° viewport data is sent by the server. Here, adaptation is carried out at the GOP-level.

In particular, the server integrates the captured data over multiple consecutive states S_t to construct a viewport-driven

360° video representation of the scene streamed to the user. The 360° content is encoded such that it maximizes the immersion fidelity experienced by the user over each delivered GOP, given his navigation actions. This is implemented by leveraging our recent advances in efficient 360° video coding [59], [65]. The GOP size is set to two seconds.

Indoor vs. outdoor. We generally did not observe notable differences in the performance of the various system components of our framework between the indoor and outdoor settings, save for a little lower delivered immersion fidelity per unit of available transmission power, in the latter case, when the available transmission power is kept uniform across the two settings. We believe this can be attributed to two factors, the known higher transmission power efficiency of Wi-Fi vs. LTE [66], and the slightly lower fidelity of the content that has been pre-captured outside. The performance analogies between the two settings are not surprising, given the small scale of our setups and the generally similar static nature of the content captured in each case. We also note that it may be misleading to compare directly (in the absolute sense) some performance metrics across the two settings, e.g., the delivered immersion fidelity, as the content is different in each case, and recorded at slightly different quality levels. Still, to illustrate the observed parallels, we have included one set of representative results that places the two settings on one graph.

B. Reference Methods

We have designed a number of reference methods, denoted henceforth as *Reference*, to compare against in the evaluation of the individual system components comprising our framework. In particular, in the assessment of the capture efficiency of the optimization in (1), we have introduced a reference method that allocates the available network bandwidth C uniformly across all capture locations, as commonly carried out. Here, we have also implemented another competing strategy introduced in [67], which employs an information theoretic metric to allocate C across the capture allocations. We denote this strategy as *Wang2011* in the evaluation. In the assessment of the reinforcement learning based dynamic UAV network placement, formulated in Section VI, we compare against three other competitive methods, denoted as ADP, USP, and Offline, introduced in detail therein. Finally, in the assessment of the scalable source-channel viewpoint coding and layered transmission packet scheduling formulated in Section VII and Section VIII, we have introduced a reference method that leverages the latest state-of-the-art High Efficiency Video Coding (HEVC) standard [68] to compress the captured viewpoint data at each drone and hybrid ARQ [69] that integrates competitive Reed-Solomon channel coding and Automatic Repeat reQuest (ARQ) scheduling, to stream the data with protection against packet erasures. Here, we have also implemented another competing strategy introduced in [70], which studies state-of-the-art network coding for error-resilient video streaming. Concretely, data packets are network coded before transmission at the sender, which makes the streaming error-rate adaptive, as more network coded packets can be transmitted if the error rate increases, and vice versa. We denote this strategy as *Thomos2011* in the evaluation. Henceforth, we denote as *Optimal*, the individual system component of our framework assessed in a specific evaluation.

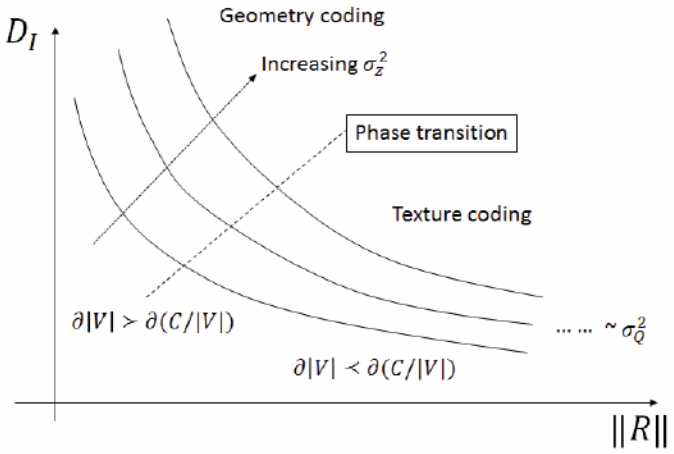

Fig. 13. Scaling behavior of the VR remote immersion reconstruction error D_I vs. the optimal UAV network/capture rate vector norm $\|R\|$.

C. Ablation Evaluation

In the following, we carry out a number of experiments that examine a certain analytical aspect of our framework or compare the performance of one of its system components relative to its alternative (the respective reference method), in their specific application domain. These latter experiments serve to provide understanding of the benefits of using that specific component proposed as part of the overall framework.

Scaling behavior. We first investigate how our characterization of the VR immersion fidelity delivered to the remote user as a function of the capture rate R_i at every UAV viewpoint location V_i, and the related optimization in (3) that aims to maximize it, behave as the available network capacity C is scaled. These results are summarized in Figure 13. In particular, we record the values of D_I, the objective in (3), for the optimal capture rate vector R, as we vary C. We plot the resulting dependency D_I versus $\|R\|$ (where $\|R\| = \sum_i R_i \sim C$), parameterized by σ_z, the standard deviation of the surface $f(x, y)$ representing the remote scene 3D geometry of the area under the UAV network (we can control this quantity in laboratory settings). We can observe two distinct operating regimes (modes). For smaller $\|R\|$, D_I scales as $c_1(\sigma_z^2)|R|^{-c_2(1/\sigma_z^2)}$, where c_1 and c_2 are two constants that depend on σ_z. We denote this mode *geometry coding*, since the optimal capture policy emphasizes the fidelity of the representation of $f(x, y)$ here, due to its impact on D_I. This is accomplished by prioritizing capturing more viewpoints $v \in V$ at the expense of the data/temporal capture rate assigned to each such v. Note that the higher the information rate (or entropy) of $f(x, y)$ across the multi-view captured area, as embodied by σ_z, the higher D_I is, as seen from Figure 13. Once $\|R\|$ is sufficiently large and $f(x, y)$ can be represented at sufficiently high fidelity, we observe a transition to another operating regime, where D_I now scales as $c_3(\sigma_c^2)|R|^{-c_4(1/\sigma_c^2)}$, and c_3/c_4 are again constants that depend on σ_c (the standard

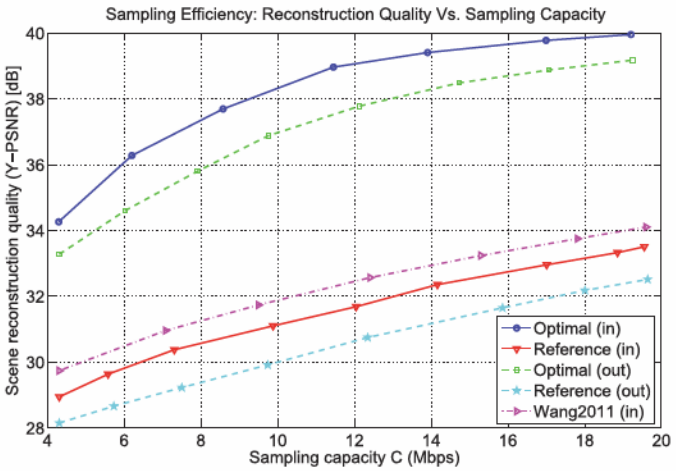

Fig. 14. Immersion fidelity vs. network capacity (indoor/outdoor).

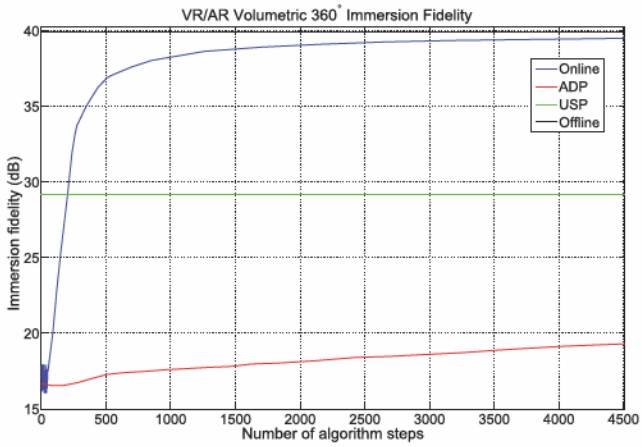

Fig. 15. VR immersion fidelity vs. iteration count.

deviation of the scene pixel color values). We denote this mode *texture coding*, as the optimization emphasizes here the color value representation fidelity for every triple $(x, y, f(x, y))$. This is accomplished by emphasizing temporal acquisition at the expense of the number of captured viewpoints. Still, the impact of σ_c is much smaller here and all three D_I versus $\|R\|$ graphs eventually converge.[12]

Capture efficiency. In Figure 14, we examine the capture efficiency of our framework relative to *Reference*, in each setting, indoor and outdoor. In particular, we graph the VR immersion fidelity Q_I[13] versus the available network capacity C, delivered by the two methods under comparison. It can be seen that *Optimal* is able to leverage the available network resources much more effectively, enabling considerable immersion fidelity gains over *Reference*, for the same C. Furthermore, we can see that *Optimal* improves its performance at a much faster rate as C is increased, relative to *Reference*. The same observations regarding the relative performance of the two methods can be made for both settings, which is expected. One expected difference is that each method performs better in the indoor setting, which stems from these two reasons. The content is slightly more complex in the outdoor setting, which makes it harder to capture/record at the same quality. Since most cross-setting results share the same parallels, to conserve space and focus the discussion, we only include performance results for the indoor setting in the remainder.

Lastly, we can observe that the second reference method *Wang2011*, evaluated in the indoor setting, outperforms *Reference*, as expected, since it takes into account scene information when allocating capture resources across each sensor location. However, both reference methods considerably underperform relative to *Optimal*, as seen from Figure 14, since the proposed method integrates the rate-distortion importance of each capture location and the impact of the user navigation patterns, on the delivered immersion fidelity to the user.

Online policy learning. To examine the efficiency of our online machine learning for dynamic UAV network placement from Section VI, we implemented Algorithm 1, denoted here as *Online*, and compared its performance relative to those of ADP (approximate dynamic programming) [72], a state-of-the-art learning method, and USP (uniform spatial placement), a traditional UAV coverage method [73]. For further reference, we implemented an off-line solution that computes the optimal UAV IoT network placement with a priori reward function knowledge, denoted here as *Offline*. The VR immersion fidelity enabled for the end user vs. number of elapsed algorithm steps, in the case of each of these methods, is shown in Figure 15. Due to its ability to effectively control the trade-off between exploration and exploitation over time, as introduced in its design in Section VI, *Online* delivers considerable performance gains over ADP. Similarly, once it quickly explores the UAV network placement decision policy space, *Online* leverages that capability to enable more than 10 dB improvement in immersion fidelity over the conventional USP. This outcome is particularly noteworthy, as USP represents a state-of-the-art UAV coverage method introduced in the literature. Moreover, the relative initial inefficiency of *Online* with respect to the off-line performance bound represented by *Offline* becomes only marginal, as Online rapidly converges towards its long-term operational point, as seen from Figure 15.

Note that implementing Algorithm 1 is computationally very efficient, as it leverages closed form expressions and efficient convex optimization. Depending on the server machine where it is implemented, one iteration count of the algorithm consumes as little as 2-3ms. In an ideal prototype development, which lies beyond the scope of the present study, the speed factor can be increased further. Still, we note that the selection of the time slot duration at which Algorithm 1 is paced, in a decentralized setting as considered here, will be impacted in major part by the spatial density at which scene viewpoints are captured, the velocity at which the UAV IoT sensors move, and the aerial network's transmission speed.

Viewpoint coding/transmission scheduling efficiency. We carry out two experiments to evaluate our scalable source-channel viewpoint coding and transmission packet scheduling in terms of adaptivity to network/capture rate R_i mismatch and robustness to unreliable transmission over the respective downlink. In particular, in the first experiment, we consider that viewpoint $\{V_i\}$ data has been acquired at an assigned target rate R_i (12 MBps here), however, the actual

[12] σ_Q here is the quantization step size used to encode the captured viewpoints $\{V_i\}$ [71].

[13] Recall this is the objective in (1) and the inverse of the objective in (3).

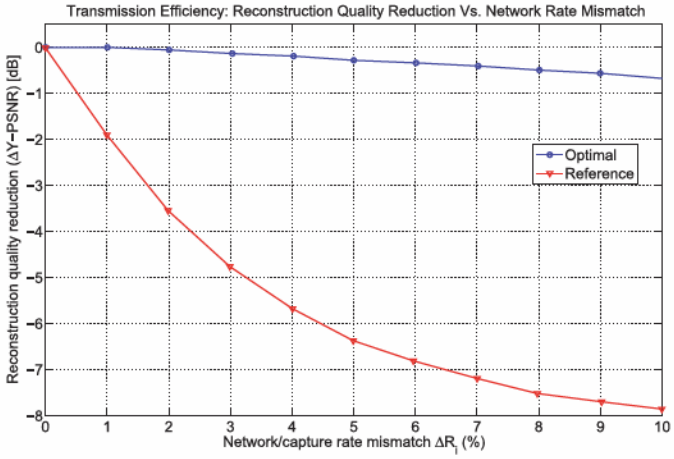

Fig. 16. Transmission efficiency vs. network rate mismatch ΔR_i (%).

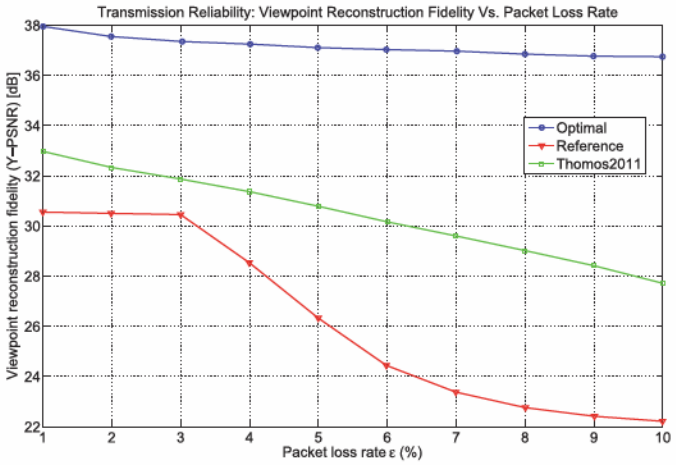

Fig. 17. Transmission robustness vs. packet loss rate ϵ (%).

available transmission rate R_i' at which it can be sent to the ground-based aggregation point is smaller, due to temporal network fluctuations. We measure the expected reduction in reconstruction quality of viewpoint V_i for different values of the mismatch ΔR_i, expressed in percent of R_i. These results are shown in Figure 16. We can see that our approach enables a graceful reconstruction quality degradation as ΔR_i is increased. In particular, due to their rate-distortion optimized design, our viewpoint coding and transmission scheduling are able to encode the data and select the most important data units to transmit such that their aggregate transmission efficiency is consistently maximized across the whole range of network/capture rate mismatch values ΔR_i we examined. This in turn minimizes the resulting reduction in reconstruction quality of viewpoint V_i induced by ΔR_i, as evident from Figure 16. On the other hand, even a small network rate mismatch can degrade considerably the viewpoint reconstruction fidelity delivered by *Reference*, as Figure 16 shows, due to its non-scalable and non-adaptive data-agnostic design and operation.

In the second experiment, we measure the achieved reconstruction fidelity of V_i as the average packet loss rate ϵ on the respective network downlink is increased. These results are shown in Figure 17. We can see that our viewpoint coding and transmission packet scheduling offer robustness and graceful performance degradation to increasing packet loss experienced on the downlink, thereby preventing dramatic degradation and variations in the reconstruction fidelity of viewpoint V_i delivered to the end user. On the other hand, *Reference* exhibits considerable degradation in performance, featuring a cliff-effect typical for conventional error protection (FEC) [74], where stable performance is maintained up to the target error protection rate of the FEC code, beyond which performance increasingly degrades, as the packet loss rate is increased, as seen from Figure 17. Moreover, *Reference* does not take into account the unequal importance of different data packets in its operation, which is another shortcoming and contributing factor. On the other hand, our approach integrates scalable signal design and error protection rate, and rate-distortion optimized packet scheduling, which synergistically enable a much higher performance efficiency. We note that implementing our transmission scheduling on the UAV platform can be computationally efficient, due to the nature of the iterative optimization it carries out. In particular, we observed that the scheduling algorithm consistently converges to the optimal policy within a few iterations only.

Finally, to compare against non-conventional state-of-the-art error protection methods, in Figure 17, we also evaluate the performance of *Thomos2011*. We can see that this reference methods enables improvement and more graceful degradation in performance over *Reference*, due to its error-adaptive design and operation. However, as *Thomos2011* does not integrate any data-related information, it still considerably under performs relative to the proposed approach, as seen from Figure 17.

Transmission power/latency reduction. Finally, in our implementation of the proposed layered directional networking and packet scheduling from Section VIII, we leveraged available power consumption data associated with the two types (Wi-Fi/cellular) of wireless transceivers we considered [66], [75], to measure the transmission power consumption of our approach and compare it to that of the reference method. In particular, we assessed through our experiments that our approach consumes approximately 30% less transmission power relative to *Reference*, due to its optimized data-aware design, in each setting we considered (outdoor or indoor). Our findings overlap with earlier results reported in prior studies involving directional networking [76]. Simultaneously, we also measured that the VR application interactivity and play-out latencies have been reduced by 25% due to the introduced layered directional networking and packet scheduling, as the client can have VR data associated with the lower layers l available to decode and reconstruct sooner. These advances signify that our system framework can operate on a considerably smaller energy budget, while still enabling considerable quality of experience gains for the end user, in terms of delivered VR immersion fidelity and application interactivity/play-out latency, at the same time. This outcome makes our framework very promising for future practical deployment of the envisioned societal VR applications of the future.

X. Practical Considerations and Follow-up Work

The paper's main aim is to explore several high-level system challenges in remote VR immersion via UAV-IoT networking and reveal novel insights and trade-offs that arise therein. The proposed system framework represents a proof-of-concept and bringing it to a practical prototype requires further development work that lies beyond the scope of the present paper. Still, the introduced models and analyses are general and permit follow-up work that can extend them towards this objective. For instance, the state-space modeling used in deciding on the dynamic UAV-IoT network placement can integrate further states that capture the possibility of different elevation levels and camera angles for the UAVs. Similarly, in our case, we assumed fixed closely collocated prospective UAV positions over the scene. A follow-up study could explore how the viewpoint sampling locations and data rates could be decided adaptively based on the dynamics of the scene and the user navigation actions, and what would be the minimum space-time sampling that could be employed.

XI. Conclusion

We investigated UAV-IoT data capture and networking for remote scene VR immersion, pursuing multiple objectives in this context. We characterized the delivered viewport-driven VR immersion fidelity as a function of the assigned UAV-IoT network/capture rates, and studied the optimization problem of maximizing it, for the given system/application constraints. We explored efficient online learning to discover the best dynamic UAV-IoT network placement over the scene of interest, to maximize the delivered expected immersion fidelity. We designed novel scalable joint source-channel viewpoint coding to maximize the reconstruction fidelity of the data captured at every UAV location, at the ground-based aggregation point. Finally, we explored layered directional networking and rate-distortion-power optimized embedded scheduling to effectively transmit the encoded data. Through experiments and analysis, we demonstrated considerable networked VR performance efficiency gains enabled by each system component over state-of-the-art reference methods, in delivered VR immersion fidelity, application interactivity/play-out latency, and transmission power consumption.

Acknowledgment

The author is very grateful to the Associate Editor and anonymous reviewers for the constructive comments and guidance that helped considerably improve the quality of the manuscript. Chakareski is equally grateful to Vladan Velisavljević and Vladimir Stanković, for their exemplary collegiality, and to Dese P (la mushata ali tato, parumba mea), for the kind support and continuous encouragement.

References

[1] J. G. Apostolopoulos, P. A. Chou, B. Culbertson, T. Kalker, M. D. Trott, and S. Wee, "The road to immersive communication," *Proc. IEEE*, vol. 100, no. 4, pp. 974–990, Apr. 2012.

[2] P. F. Drucker. (Jan. 2015). *Internet of Things Position Paper on Standardization for IoT Technologies*. [Online]. Available: http://www.internet-of-things-research.eu/pdf/IERC_Position_Paper_IoT_Standardization_Final.pdf

[3] G. P. Fettweis, "The tactile Internet: Applications and challenges," *IEEE Veh. Technol. Mag.*, vol. 9, no. 1, pp. 64–70, Mar. 2014.

[4] *Agricultural Drones: MIT Technology Review: 10 Breakthrough Technologies*. (2014). [Online]. Available: https://www.technologyreview.com/lists/technologies/2014/

[5] B. Begole, "Why the Internet pipes will burst when virtual reality takes off," Forbes Magazine, Feb. 2016.

[6] J. Chakareski, "Uplink scheduling of visual sensors: When view popularity matters," *IEEE Trans. Commun.*, vol. 63, no. 2, pp. 510–519, Feb. 2015.

[7] R. Vasudevan, Z. Zhou, G. Kurillo, E. Lobaton, R. Bajcsy, and K. Nahrstedt, "Real-time stereo-vision system for 3D teleimmersive collaboration," in *Proc. IEEE Int. Conf. Multimedia Expo*, Suntec City, Singapore, Jul. 2010, pp. 1208–1213.

[8] G. Cheung, A. Ortega, and N.-M. Cheung, "Interactive streaming of stored multiview video using redundant frame structures," *IEEE Trans. Image Process.*, vol. 20, no. 3, pp. 744–761, Mar. 2011.

[9] J. Chakareski, V. Velisavljević, and V. Stanković, "User-action-driven view and rate scalable multiview video coding," *IEEE Trans. Image Process.*, vol. 22, no. 9, pp. 3473–3484, Sep. 2013.

[10] J. Chakareski, "Wireless streaming of interactive multi-view video via network compression and path diversity," *IEEE Trans. Commun.*, vol. 62, no. 4, pp. 1350–1357, Apr. 2014.

[11] J. Chakareski, V. Velisavljević, and V. Stanković, "View-popularity-driven joint source and channel coding of view and rate scalable multi-view video," *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 3, pp. 474–486, Apr. 2015.

[12] X. Corbillon, A. Devlic, G. Simon, and J. Chakareski, "Optimal set of 360-degree videos for viewport-adaptive streaming," in *Proc. Int. Conf. Multimedia*, Mountain View, CA, USA, Oct. 2017, pp. 934–951.

[13] M. Hosseini and V. Swaminathan, "Adaptive 360 VR video streaming: Divide and conquer," in *Proc. IEEE Int. Symp. Multimedia*, San Jose, CA, USA, Dec. 2016, pp. 107–110.

[14] J. Chakareski, "VR/AR immersive communication: Caching, edge computing, and transmission trade-offs," in *Proc. ACM SIGCOMM Workshop Virtual Reality Augmented Reality Netw.*, Los Angeles, CA, USA, Aug. 2017, pp. 36–41.

[15] K. Pires and G. Simon, "DASH in twitch: Adaptive bitrate streaming in live game streaming platforms," in *Proc. ACM Conext Workshop VideoNext*, Sydney, Australia, Dec. 2014, pp. 13–18.

[16] S. Ahmad *et al.*, "Peer-to-peer live video streaming with rateless codes for massively multiplayer online games," *Springer Peer-to-Peer Netw. Appl.*, vol. 11, no. 1, pp. 44–62, Jan. 2018.

[17] G. A. Hollinger and G. S. Sukhatme, "Sampling-based robotic information gathering algorithms," *Int. J. Robot. Res.*, vol. 33, no. 9, pp. 1271–1287, Aug. 2014.

[18] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs," *IEEE Trans. Wireless Commun.*, vol. 15, no. 6, pp. 3949–3963, Jun. 2016.

[19] D. Panagou, D. M. Stipanović, and P. G. Voulgaris, "Dynamic coverage control in unicycle multi-robot networks under anisotropic sensing," *Frontiers Robot. AI*, vol. 2, no. 3, Mar. 2015.

[20] B. Pang, Y. Song, C. Zhang, H. Wang, and R. Yang, "A swarm robotic exploration strategy based on an improved random walk method," *J. Robot.*, Mar. 2019.

[21] L. Lima and J. Barros, "Random walks on sensor networks," in *Proc. 5th Int. Symp. Modeling Optim. Mobile, Ad Hoc Wireless Netw. Workshops*, Apr. 2007, pp. 1–5.

[22] M. A. Batalin and G. S. Sukhatme, "The analysis of an efficient algorithm for robot coverage and exploration based on sensor network deployment," in *Proc. IEEE Int. Conf. Robot. Automat.*, Apr. 2005, pp. 3478–3485.

[23] J. Chakareski, "Cost and profit driven cloud-P2P interaction," *Springer Peer-to-Peer Netw. Appl.*, vol. 8, no. 2, pp. 244–259, Mar. 2015.

[24] J. Chakareski, "Social video caching," *Signal Process., Image Commun.*, vol. 29, no. 4, pp. 462–471, Apr. 2014.

[25] D. Jurca and P. Frossard, "Distributed media rate allocation in multipath networks," *Signal Process., Image Commun.*, vol. 23, no. 10, pp. 754–768, Nov. 2008.

[26] T. Nguyen and A. Zakhor, "Multiple sender distributed video streaming," *IEEE Trans. Multimedia*, vol. 6, no. 2, pp. 315–326, Apr. 2004.

[27] P. A. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *IEEE Trans. Multimedia*, vol. 8, no. 2, pp. 390–404, Apr. 2006.

[28] J. Chakareski, "Joint source-channel rate allocation and client clustering for scalable multistream IPTV," *IEEE Trans. Image Process.*, vol. 24, no. 8, pp. 2429–2439, Aug. 2015.

[29] J. Chakareski, "Informative state-based video communication," *IEEE Trans. Image Process.*, vol. 22, no. 6, pp. 2115–2127, Jun. 2013.

[30] J. Chakareski, "Cost-performance optimization of network flow allocation and data center placement in online multi-service provider networks," in *Proc. IEEE Int. Packet Video Workshop*, Munich, Germany, May 2012, pp. 101–106.

[31] J. Chakareski, "In-network packet scheduling and rate allocation: A content delivery perspective," *IEEE Trans. Multimedia*, vol. 13, no. 5, pp. 1092–1102, Oct. 2011.

[32] I. Sodagar, "The MPEG-DASH standard for multimedia streaming over the Internet," *IEEE Multimedia*, vol. 18, no. 4, pp. 62–67, Apr. 2011.

[33] S. Chen, J. Yang, Y. Ran, and E. Yang, "Adaptive layer switching algorithm based on buffer underflow probability for scalable video streaming over wireless networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 6, pp. 1146–1160, Jun. 2016.

[34] J. Chakareski, "Drone networks for virtual human teleportation," in *Proc. 3rd ACM MobiSys Workshop Micro Aerial Vehicle Netw., Syst., Appl.*, Niagara Falls, NY, USA, Jun. 2017, pp. 21–26.

[35] S. B. Gokturk, H. Yalcin, and C. Bamji, "A time-of-flight depth sensor—System description, issues and solutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshop*, Washington, DC, USA, Jun./Jul. 2004, p. 35.

[36] M. Tanimoto, T. Fujii, and K. Suzuki, *Multi-View Depth Map of Rena and Akko & Kayo*, Standard M14888, ISO/IEC JTC1/SC29/WG11, MPEG, Oct. 2007.

[37] H.-Y. Shum, S.-C. Chan, and S.-B. Kang, *Image-Based Rendering*, 1st ed. New York, NY, USA: Springer, 2006.

[38] J. Chakareski, "Transmission policy selection for multi-view content delivery over bandwidth constrained channels," *IEEE Trans. Image Process.*, vol. 23, no. 2, pp. 931–942, Feb. 2014.

[39] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[40] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambride, MA, USA: MIT Press, 1998.

[41] R. Bellman, *Dynamic Programming*. Princeton, NJ, USA: Princeton Univ. Press, 1957.

[42] M. L. Puterman, *Markov Decision Processes*. Hoboken, NJ, USA: Wiley, 1994.

[43] D. Aldous and J. A. Fill. (2002). *Reversible Markov Chains and Random Walks on Graphs*. [Online]. Available: http://www.stat.berkeley.edu/~aldous/RWG/book.html

[44] D. A. Levin and Y. Peres, *Markov Chains Mixing Times*, 2nd ed. Providence, RI, USA: American Mathematical Society, Oct. 2017.

[45] P. Brémaud, *Markov Chains: Gibbs Fields, Monte Carlo Simulation, and Queues*, 1st ed. New York, NY, USA: Springer-Verlag, 2008.

[46] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 279–292, 1992.

[47] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *J. Artif. Intell. Res.*, vol. 4, pp. 237–285, May 1996.

[48] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 4th ed. Baltimore, MD, USA: The Johns Hopkins Univ. Press, 2012.

[49] S. Boyd, P. Diaconis, and L. Xiao, "Fastest mixing Markov chain on a graph," *SIAM Rev.*, vol. 46, pp. 667–689, Jan. 2004.

[50] *Lipschitz constant, Encyclopedia of Mathematics*. (2010). [Online]. Available: https://www.encyclopediaofmath.org/index.php/Lipschitz_constant

[51] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 9, pp. 1103–1120, Sep. 2007.

[52] D. Vukobratovic and V. Stanković, "Unequal error protection random linear coding strategies for erasure channels," *IEEE Trans. Commun.*, vol. 60, no. 5, pp. 1243–1252, May 2012.

[53] S. Atmaca, C. Ceken, and I. Erturk, "A new QoS-aware TDMA/FDD MAC protocol with multi-beam directional antennas," *Comput. Standards Interfaces*, vol. 31, no. 4, pp. 816–829, Jun. 2009.

[54] *Cubic | Global. Innovative. Trusted. | Gigabit Multi-Beam Directional Antennas*. [Online]. Available: https://www.cubic.com/

[55] L. Gupta, R. Jain, and G. Vaszkun, "Survey of important issues in UAV communication networks," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1123–1152, 2nd Quart., 2016.

[56] H. Everett, "Generalized Lagrange multiplier method for solving problems of optimum allocation of resources," *Oper. Res.*, vol. 11, no. 3, pp. 399–417, 1963.

[57] *VIRAT Video Dataset*. (2016). [Online]. Available: http://www.viratdata.org/

[58] *DJI Matrice 100: The Quadcopter for Developers*. [Online]. Available: http://www.dji.com/matrice100/

[59] J. Chakareski, R. Aksu, X. Corbillon, G. Simon, and V. Swaminathan, "Viewport-driven rate-distortion optimized 360° video streaming," in *Proc. IEEE Int. Conf. Commun.*, May 2018.

[60] X. Corbillon, F. De Simone, and G. Simon, "360-degree video head movement dataset," in *Proc. ACM Multimedia Syst. Conf.*, Taipei, Taiwan, Jun. 2017, pp. 199–204.

[61] J. W. Byers, M. Luby, and L. Mitzenmacher, "A digital fountain approach to asynchronous reliable multicast," *IEEE J. Sel. Areas Commun.*, vol. 20, no. 8, Oct. 2002, pp. 1528–1540.

[62] D. Vukobratovic, V. Stanković, D. Sejdinovic, L. Stankovic, and Z. Xiong, "Expanding window fountain codes for scalable video multicast," in *Proc. IEEE Int. Conf. Multimedia Expo*, vol. 11, Jun./Apr. 2009, pp. 77–80.

[63] N. Heath. (May 2017). *The Long-Range Drone That can Keep up With a Car and Fly for an Hour*. [Online]. Available: https://www.techrepublic.com/blog/european-technology/the-long-range-drone-that-can-keep-up-with-a-car-and-fly-for-an-hour/

[64] J. Flynt. (Oct. 2018). *How Fast Can Drones Fly?, 3D Insider*. [Online]. Available: https://3dinsider.com/drone-speed/

[65] X. Corbillon, A. Devlic, G. Simon, and J. Chakareski, "Viewport-adaptive navigable 360-degree video delivery," in *Proc. IEEE Int. Conf. Commun.*, Paris, France, May 2017.

[66] *Wireless Tranmission Power Consumption for Different Transceivers*. [Online]. Available: https://en.wikipedia.org/wiki/DBm

[67] P. Wang, R. Dai, and I. F. Akyildiz, "A spatial correlation-based image compression framework for wireless multimedia sensor networks," *IEEE Trans. Multimedia*, vol. 13, no. 2, pp. 388–401, Apr. 2011.

[68] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[69] R. Comroe and D. J. Costello, Jr., "ARQ schemes for data transmission in mobile radio systems," *IEEE J. Sel. Areas Commun.*, vol. SAC-2, no. 4, pp. 472–481, Jul. 1984.

[70] N. Thomos, J. Chakareski, and P. Frossard, "Prioritized distributed video delivery with randomized network coding," *IEEE Trans. Multimedia*, vol. 13, no. 4, pp. 776–787, Aug. 2011.

[71] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*, 1st ed. New York, NY, USA: Springer, 1991.

[72] F. L. Lewis and D. Liu, *Reinforcement Learning and Approximate Dynamic Programming for Feedback Control*. Hoboken, NJ, USA: Wiley, 2013.

[73] N. Nigam, S. Bieniawski, I. Kroo, and J. Vian, "Control of multiple UAVs for persistent surveillance: Algorithm and flight test results," *IEEE Trans. Control Syst. Technol.*, vol. 20, no. 5, pp. 1236–1251, Sep. 2012.

[74] S. Rane, P. Baccichet, and B. Girod, "Systematic lossy error protection of video signals," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 10, pp. 1347–1360, Oct. 2008.

[75] Cisco. (2019). *Cisco Aironet 1815i Access Point Data Sheet*. [Online]. Available: https://www.cisco.com/c/en/us/products/collateral/wireless/aironet-1815-series-access-points/datasheet-c78-738243.html

[76] A. Spyropoulos and C. S. Raghavendra, "Energy efficient communications in ad hoc networks using directional antennas," in *Proc. IEEE INFOCOM*, Jun. 2002, pp. 220–228.

Jacob Chakareski is currently Associate Professor in the Ying Wu College of Computing, NJIT, where he leads the Laboratory for VR/AR Immersive Communication (LION). His research interests span networked virtual and augmented reality, UAV IoT sensing and networking, fast online machine learning, 5G wireless edge computing/caching, ubiquitous immersive communication, and societal applications. He received the Adobe Data Science Faculty Research Award in 2017 and 2018, the Swiss NSF Career Award Ambizione (2009), the AFOSR Faculty Fellowship in 2016 and 2017, and Best/Fast-Track Paper Awards at IEEE ICC 2017 and IEEE Globecom 2016. He is the organizer of the first NSF Visioning Workshop on networked VR/AR communications. He trained as a Ph.D. student at Rice and Stanford, held research appointments with Microsoft, HP Labs, and EPFL, and sits on the advisory board of Frame, Inc. His research is supported by the NSF, AFOSR, Adobe, Tencent Research, NVIDIA, and Microsoft.