Design and Evaluation of Correlation-Aware Scheduling for Wireless Surveillance Cameras

Chang-Yu Song and Hung-Yun Hsieh
Graduate Institute of Communication Engineering &
Department of Electrical Engineering
National Taiwan University, Taipei, Taiwan 106

I. Introduction

Many machine-to-machine (M2M) applications are characterized by a large amount of data to transport over a rather limited amount of wireless resources. While for some applications the amount of data produced by each machine is not huge and the delay sensitivity is rather low, for applications such as multimedia surveillance networks the requirement on communication is demanding. Fortunately, since such wireless surveillance cameras are typically deployed with overlapping view angles, the images or videos captured by individual cameras exhibit correlation that can potentially be leveraged for bandwidth-efficient reporting of the collected data.

In this paper, we investigate the problem of correlated data gathering from a set of cameras deployed in a city. It is required that cameras periodically send back the collected images back to the aggregator (e.g. base station) through direct wireless communications (e.g. LTE or WiMAX). Since there might be multiple cameras deployed in a neighborhood area to provide different perspectives of the area, we exploit the capability of *transmission overhearing* among cameras. If a camera can overhear transmissions from nearby cameras, it can reference the image (e.g. as an I-frame) and perform *dependent coding* to reduce the amount of bits required to encode its image (e.g. as a P-frame). Clearly, if the reference image is highly correlated with the target image, the compression ratio will be high. We propose a *correlation-aware scheduling algorithm* to determine the order of transmissions for all cameras based on their locations and the correlation of collected images. To evaluate the proposed algorithm, we resort to a 3D modeling software to generate quasi-realistic city views for all cameras and use H.264 MVC reference software to encode collected images. Evaluation results show the proposed scheduling algorithm to outperform baseline approaches, motivating further investigation along this direction.

II. Formulation and Proposed Algorithm

A. Problem Formulation

Let $V = \{v_1, v_2, \cdots v_N\}$ be the set of N cameras under consideration and X_i be the snapshot of image produced by camera v_i. Denote $H(X_i)$ as the amount of bits required to encode X_i independently (entropy), and $H(X_i|X_j)$ as the amount of bits required if X_j is used as reference for encoding X_i (conditional entropy). Clearly, for camera v_i to reference the image captured by camera v_j, it is required that v_j is scheduled before v_i and the transmission range of camera v_j covers camera v_i. Based on such a notation, the total amount of encoded bits for transmission can be written as follows:

$$\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_{ij}H(X_i|X_j), \tag{1}$$

where $H(X_i|X_i) = H(X_i)$ for sake of notation simplicity and $\alpha_{ij} \in \{0, 1\}$ is an indicator variable such that $\alpha_{ij} = 1$ when camera v_i overhears camera v_j and $\alpha_{ii} = 1$ when camera v_i performs independent encoding. We require that

$$\sum_{j=1}^{N}\alpha_{ij} = 1,\ \forall i \in V, \tag{2}$$

such that each camera in V is allowed to overhear *only one* camera in its neighborhood.

B. Scheduling Algorithm

To schedule the given set of cameras for minimizing the total encoded bits, denote $\Phi \subset V$ as the subset of cameras already scheduled and ϕ_l as the last scheduled camera in Φ. Now consider two different schedules: *Schedule* 1: $\phi_l \leftarrow v_i \leftarrow v_j$ (v_i is scheduled immediately after ϕ_l) and *Schedule* 2: $\phi_l \leftarrow v_j \leftarrow v_i$ (v_j is scheduled before v_i). If the transmission order is changed from *Schedule* 1 to *Schedule* 2, the reference frame of camera v_i will change from ϕ_l to v_j, resulting in a change in the amount of encoded bits for camera v_i as $H(X_i|X_{\phi_l}) - H(X_i|X_j)$. For camera v_j, the difference in the amount of encoded bits is $H(X_j|X_i) - H(X_j|X_{\phi_l})$. Therefore, the total amount of change in the amount of encoded bits by changing from *Schedule* 1 to *Schedule* 2 is:

$$\begin{aligned}\Delta R(v_i, v_j) = &\{H(X_i|X_{\phi_l}) - H(X_i|X_j)\}\\ &+ \{H(X_j|X_i) - H(X_j|X_{\phi_l})\}.\end{aligned} \tag{3}$$

Clearly, if the amount of encoded bits can be reduced by changing from *Schedule* 1 to *Schedule* 2, then camera v_i should be scheduled after camera v_j.

Based on this concept, let v_i and v_j be two different unscheduled cameras. $\Delta R(v_i, v_j)$ as defined in Equation (3) is the difference in the amount of encoded bits if camera v_i is not the first camera to schedule after ϕ_l but deferred to the next

Algorithm 1 Proposed scheduling algorithm

1: $\Phi \leftarrow \emptyset$, $\Phi^c \leftarrow V$
2: **while** $\Phi^c \neq \emptyset$ **do** //loop until all cameras have been scheduled
3: $\quad \omega_i \leftarrow \max\limits_{v_j \neq v_i, v_j \in \Phi^c} \Delta R(v_i, v_j), \forall v_i \in \Phi^c$ //calculate the scheduling metric for all unscheduled cameras
4: $\quad v_k \leftarrow \arg\min\limits_{v_i \in \Phi^c} \omega_i$ //choose camera with the smallest scheduling metric as the next
5: $\quad \Phi \leftarrow \Phi \cup \{v_k\}$ //record v_k as a scheduled camera
6: $\quad \Phi^c \leftarrow \Phi^c \setminus \{v_k\}$ //remove v_k from the unscheduled cameras set
7: $\quad \phi_l \leftarrow v_k$ //update the last scheduled camera in Φ
8: **end while**

scheduling position after camera v_j. The proposed scheduling metric for each unscheduled camera v_i can be written as:

$$\omega_i = \max_{v_j \in \Phi^c, v_j \neq v_i} \Delta R(v_i, v_j), \tag{4}$$

where $\Phi^c = V \setminus \Phi$ is the subset of all unscheduled cameras. The proposed scheduling algorithm thus is to choose camera

$$v_k = \arg\min_{v_i \in \Phi^c} \omega_i \tag{5}$$

as the next camera to be scheduled for reducing the largest amount of encoded bits. As Algorithm 1 shows, the algorithm starts with $\Phi = \emptyset$ and iteratively chooses a camera to schedule based on Equation (5).

III. Performance Evaluation

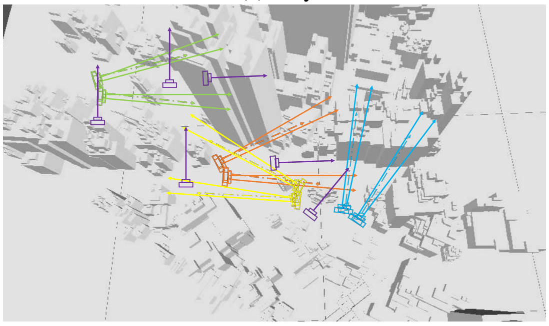
A. Experiment Settings

To create quasi-realistic 3D city views, we make use of an open-source 3D modeling software [1] and a 3D city generator [2]. A total of 30 cameras are then deployed at different locations (crossroads) inside the city of size $500m^2$ (limited by the capacity of the modeling software) for collecting the desired city snapshots (1280×720 HD images). Then, H.264 [3] is used to encode the images collected by individual cameras with or without reference frames. Figure 1 shows the 3D city view and the deployment locations for 30 cameras. With reference to real-world applications, multiple cameras are deployments at one crossroad for capturing views from different angles. The arrow in Figure 1b is the sensing direction of each camera while different groups of cameras are shown by different colors.

B. Experiment Results

To evaluate the performance of the proposed algorithm, we compare against the MLS algorithm proposed in [4]. The authors in [4] solve a *relaxed integer programming* problem to obtain the probability that a camera should overhear transmissions of other cameras. Since the binary decision for each camera is made by approximating the probability thus solved, there is performance loss during the transformation. Figure 2 shows that MLS can improve the baseline performance (all cameras perform independent encoding) for 6.3%, whereas the proposed algorithm can achieve a 16.2% improvement. The result substantiates the benefits of the proposed scheduling algorithm and motives further investigation along this direction.

(a) City view



(b) Camera deployment

Figure 1: City view and camera deployment

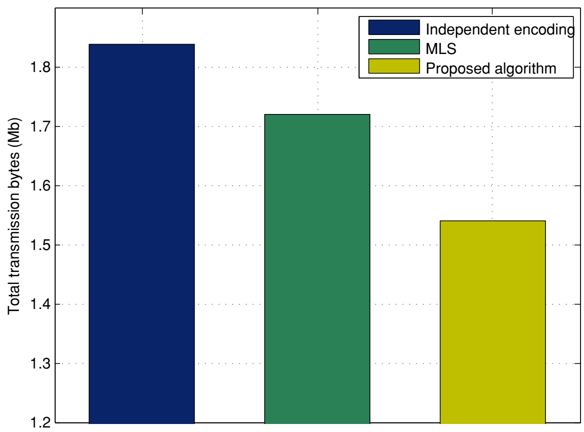


Figure 2: Performance comparison

References

[1] Blender Online Community, *Blender - a 3D modelling and rendering package*, Blender Foundation, Blender Institute, Amsterdam. [Online]. Available: http://www.blender.org
[2] A. Couturier, *Suicidator City Generator - Create 3D cities*. [Online]. Available: http://cgchan.com/suicidator
[3] *JMVC Reference Software*. [Online]. Available: http://ftp3.itu.int/av-arch/jvt-site/2008_10_Busan/JVT-AC207.zip.
[4] P. Wang, R. Dai, and I. F.Akyildiz, "Visual correlation-based image gathering for wireless multimedia sensor networks," in *Proceedings of IEEE INFOCOM*, April 2011, pp. 2489–2497.