Stabilized Adaptive Sampling Control for Reliable Real-Time Learning-based Surveillance Systems

Dohyun Kim, Soohyun Park, Joongheon Kim, Jae young Bang, and Soyi Jung

Abstract: **In modern security systems such as CCTV-based surveillance applications, real-time deep-learning based computer vision algorithms are actively utilized for always-on automated execution. The real-time computer vision system for surveillance applications is highly computation-intensive and exhausts computation resources when it performed on the device with a limited amount of resources. Based on the nature of Internet-of-Things networks, the device is connected to main computing platforms with offloading techniques. In addition, the real-time computer vision system such as the CCTV system with image recognition functionality performs better when arrival images are sampled at a higher rate because it minimizes missing video frame feeds. However, performing it at overwhelmingly high rates exposes the system to the risk of a queue overflow that hampers the reliability of the system. In order to deal with this issue, this paper proposes a novel queue-aware dynamic sampling rate adaptation algorithm that optimizes the sampling rates to maximize the computer vision performance (i.e., recognition ratio) while avoiding queue overflow under the concept of Lyapunov optimization framework. Through extensive system simulations, the proposed approaches are shown to provide remarkable gains.**

Index Terms: **Lyapunov optimization, real-time computer vision system, reliable system, sampling rate optimization, surveillance applications**

I. INTRODUCTION

WE are witnessing today the rapid adoption of deep-learning based computer vision technologies into our daily lives. One of the major applications of the technologies is real-time surveillance systems can be implemented as the mixture of various techniques such as object detection, face recognition, and face identification [1]–[5], in order to monitor target areas and detect intruders. The increased demand of the technologies in surveillance systems has widened the variety of host devices on which the systems are deployed. These systems incorporate not just the surveillance cameras that provide real-time video feeds into the system via the concept of offloading; but also power-hungry smartphones and even Internet-of-Things (IoT) devices, both of which often have limitations in terms of available computation resources [6].

One of major challenges in order to build a reliable, real-time surveillance system in practice is that the required amount of computation resources to process the surveillance video feeds may be much larger than the amount of available resources in the system. The real-time surveillance system in our context consists of (1) a set of image *acquirer* components, each of which samples images from a real-time video feed (i.e., a stream of images) and (2) a set of image *classifier* components, each of which detects and identifies human faces from the acquired surveillance images by applying computer vision technologies. A conventional design of the system embeds an image buffer (or queue) to temporarily store the images from the acquirers until they are processed by the built-in classifiers. In this case, if the rate of acquirers' image insertion into the queue (i.e., arrival rate) is larger than the rate of classifiers' image process from the queue (i.e., departure rate), the queue be in overflow situations and thus cause unexpected system malfunctions and unstability. Therefore, it is essential to find an "optimal sweet spot" of the image sampling rate for building a reliable face identification-based automated surveillance system. The system may only identify faces that appear on the CCTV-recorded sampled images that the acquirers generate.

The optimal image sampling rates at the surveillance system can be dynamically computed depending on queue-backlog (i.e., delay situations) in order to achieve the maximum face identification performance while preserving system stability. The dynamic optimal image sampling rate control depends on the following two factors. The first factor is the variance in the volume of images to process. For example, the acquirers (e.g., cameras) can be configured to be activated and begin video recording only when a motion sensor detects a moving object. The second factor is the variance in the image processing speed. One of popular deep-learning based face recognition libraries such as `OpenFace` [7], [8] often requires longer processing times when there is a higher number of faces in an image [9]. In both cases, the necessary amount of computation resources for face detection fluctuate as objects appear and disappear. A real-time surveillance system needs to be able to adjust the sampling rate on-the-fly to achieve the maximum performance and system stability at the same time.

In modern surveillance systems with computer vision functionalities, several research results have been actively proposed for various applications [10]–[15]. In the previous research, efficient algorithms for vision-based surveillance platforms are proposed, however the algorithms are focusing on computer vi-

Manuscript received July 17, 2020; revised February 24, 2021; approved for publication by Patrick Seeling, Division II Editor, March 6, 2021.

This research was supported by National Research Foundation of Korea (2019R1A2C4070663) and Korea University Future Research Grant (KU-FRG).

D. Kim is with Naver Corporation, Seongnam, Republic of Korea, email: ppooiiuuyh@naver.com.

S. Park, J. Kim, and S. Jung are with the School of Electrical Engineering, Korea University, Seoul, Republic of Korea, emails: {soohyun828, joongheon, jungsoyi}@korea.ac.kr.

J. y. Bang is with Quandary Peak Research, Los Angeles, CA 90012, USA, email: jae@outlook.com.

J. Kim and S. Jung are also with Artificial Intelligence Engineering Research Center, College of Engineering, Korea University, Seoul, Republic of Korea.

S. Jung is a corresponding author of this paper.

Digital Object Identifier 10.23919/JCN.2021.000009

1229-2370/21/$10.00 © 2021 KICS

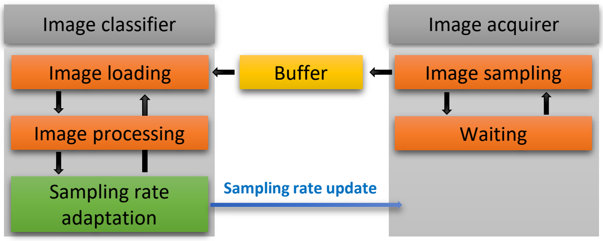


Fig. 1. Improved pipeline of the real-time computer vision system with the additional procedure that computes optimal sampling rates and adjusts the values.

sion based real-world implementation whereas the proposed algorithm in this paper is focusing on system-specific algorithm design and theoretical analysis.

In this paper, we propose a dynamic sampling rate optimization algorithm for real-time surveillance systems that automatically adapts the image sampling rate to the computation resource availability via Lyapunov optimization. A real-time surveillance system that implements our algorithm dynamically computes the optimal sampling rate during run-time and controls the sampling rate in a way that maximizes the face identification performance of the system while no overflow occurs at the image queue and the system remains reliable.

This proposed sampling control algorithm for stabilized face identification is definitely worthy to consider in many surveillance applications. In particular, this can be helpful for CCTV-based automated surveillance systems especially in large-scale department stores. In large-scale department stores, many people walk around. Thus, detecting target people in real-time without automated systems is not possible. Thus, the proposed algorithm is obviously useful in large-scale department store surveillance applications.

The main contributions of this work can be summarized as follows.

- First of all, this is the first attempt to utilize queue-based sampling rate control mechanisms for face identification automated surveillance systems, to the best of our knowledge. This is essentially required for practical solutions because the arrivals by CCTV-cameras and the departures by face-identification functionalities can be with unexpected random events in real-world situations.
- This paper adds novel contributions on top of our earlier research contributions. We previously presented an algorithm that dynamically selects learning architectures for surveillance systems [16]. While theoretically sound, we found that the proposed algorithm would need a revision to be implemented in a real-world application. Furthermore, the earlier work is selecting learning architectures in order to control the departure process. In this case, various learning architectures should be implemented in a single platform which is burden in terms of training and management. In our proposed work in this paper, it controls arrival process instead of departure process. Thus, only a single learning architecture exists which is obviously easier to implement in real-world applications.
- Then, we propose an initial version of the dynamic sampling rate adaptation [17], [18]. In this paper, we add more systems-specific mechanisms including max-queue setting and overflow handling. Furthermore, more detailed evaluation-based performance evaluation results are included with more detailed descriptions in terms of overall system architectures.

The remainder of this paper is organized as follows: In Section II, we introduce the theory and the applications of the Lyapunov optimization. We also describe in detail about the proposed surveillance system. We evaluate the performance of the proposed algorithm in Section III, and we present the limitations of the proposed algorithm and corresponding discussions in Section IV. Lastly, Section V concludes this paper and presents future research directions.

II. STOCHASTIC FRAME RATE ADAPTATION

In this section, basic algorithm design rationale is explained in Section II.A and the fundamental theory behind this system, i.e., Lyapunov optimization framework, is briefly introduced in Section II.B. In Section II.C, the proposed sampling rate adaptation algorithm is presented, and then the practical modification is further discussed in Section II.D.

A. Algorithm Design Rationale

To adapt sampling rate dynamically, an algorithm for computing optimal sampling rates at each processing is desired. For this purpose, Fig. 1 shows the improved pipeline of the real-time computer vision system for CCTV-capable surveillance applications with the additional procedure in terms of optimal sampling rate computation.

The ideal method is to predict the image processing rate p accurately and set the sampling rate according to this. However, as mentioned earlier, no matter how used algorithms are sophisticated, it is generally not possible to predict the exact processing time, and thus the queue can be unstable due to unexpected slight measurement errors. Even if it can be predicted without errors, there still exist some problems to take account the time for reflecting the calculated results to the acquired images. Therefore, it is necessary to consider the prediction of the processing rate p as well as the recovery of the increased queue because the queue backlog can be longer in any cases. To do this, we can consider the other approach for setting the sampling rate to be smaller than the predicted processing rates according to the queue length such as $x(t) = \lfloor p \cdot (1 - (|Q(t)|)/(Q_{\max}))\rfloor$ when $x(t)$, p, $|Q(t)|$, and $Q_{\max}$ are calculated sampling rate at time t, processing rate at time t, current queue backlog at time t, and maximum queue size (i.e., queue capacity), respectively. It can reasonably work well in terms of queue stability when p is higher than real. However, there may exist certain amounts of wastes due to the fact that the sampling rate $x(t)$ can not be over than p if predicted p is lower than measured real values.

As discussed, the approach of setting the sampling rate around the predicted processing rate has potential problems when processing time can not be accurately predicted. Therefore, it is necessary to take a new approach that uses (i) the processing time as side information as well as (ii) the queue-backlog as an additional consideration factor to recover system stability (i.e., queue stability). To achieve this goal, *Lya-*

punov optimization framework is considered in this paper which is an optimization technique that maximizes the time-average objective function subject to queue stability. Note that the Lyapunov optimization based algorithm design theoretically guarantees the time-average optimality under stabilization conditions [19]. In this CCTV-enabled surveillance application, the objective function is the security levels which is a monotonously function depending on the given sampling rate. Note that higher sampling rates achieve faster image frame generation and thus there has little chance to loose information (i.e., fast moving face detection).

In summary, a dynamic sampling rate optimization algorithm is proposed in this paper based on Lyapunov optimization framework, which designs a system that is able to compute time-average security level maximization (which can be achieved by taking time-average optimal sampling rates) subject to system stability which will be applied on next sampling by itself.

B. Lyapunov Optimization Framework

The theory of stochastic optimization [19] aims at optimizing a time-average utility subject to queue stability when the objective function and the queue stability constraints are in a trade-off relationship. As clearly discussed in [19], our considering Lyapunov control theory based stochastic optimization guarantees time-average utility optimality subject to queue stability with constant gap approximation. More details about the theory are in [19], [21].

The stochastic optimization models the queue stability using the Lyapunov drift [19]. In each unit time t, the stochastic model takes actions that minimizes the Lyapunov drift (i.e., minimization of system drifts) while pursuing the maximization of time-average objective function with the gap of $O(1/V)$ under queue stability with the bound of $O(V)$ where V is defined as a trade-off between utility (i.e., objective function) and stability (queue stability).

Our mathematical program for time-average utility maximization subject to queue stability can be modeled as follows:

$$\max : \lim_{t\to\infty} \sum_{\tau=0}^{t-1} U(x(\tau),\tau) \tag{1}$$

subject to

$$\lim_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} Q(\tau) \leq \infty, \tag{2}$$

where $U(x(\tau),\tau)$ is utility with decision $x(\tau)$ and time τ, and $Q(\tau)$ is queue-backlog size at τ.

According to the theories in [19], the general form of the time-average stochastic optimization subject to queue stability can be expressed as follows:

$$x^* \leftarrow \arg\max_{x(t)\in\mathcal{X}} \{V \cdot U(x(t),t) \\ -Q(t) \cdot (\mathsf{In}(x(t),t) - \mathsf{Out}(x(t),t))\}, \tag{3}$$

where $\mathcal{X}$ is a decision set, $\mathsf{In}(x(t),t)$ is queue arrival procedure with decision $x(t)$ and time t, $\mathsf{Out}(x(t),t)$ is queue departure process with decision $x(t)$ and time t, V is a trade-off factor, and x^* is optimal decision, respectively. More details about this theory are discussed in [19]; and the various applications that implement the theory are in [20]–[28].

C. Dynamic Sampling Rate Adaptation via Lyapunov Optimization Framework

Our goal is to design a control algorithm that yields a time-average optimal sampling rate that maximizes surveillance performance while ensuring queue stability.

Since the hit-rate which means the ratio of frames processed per time is our surveillance performance consideration, in this case, the objective function of this equation is $h(x(t))$ which stands for the hit rate function of sampling rate during time t, i.e., $x(t)$. In addition, the hit-rate function can be expressed as $h(x(t)) = c \cdot x(t)$ due to the fact that the hit rate is simply proportional to the sampling rate; and note that the c (i.e., a scaling constant) can be merged with objective function weight V (i.e., trade-off coefficient which is a constant). Eventually, $V \cdot U(x(t),t)$ equals to

$$V \cdot U(x(t),t) = V \cdot h(x(t)) = V \cdot c \cdot x(t) = V' \cdot x(t), \tag{4}$$

where $V' = V \cdot c$.

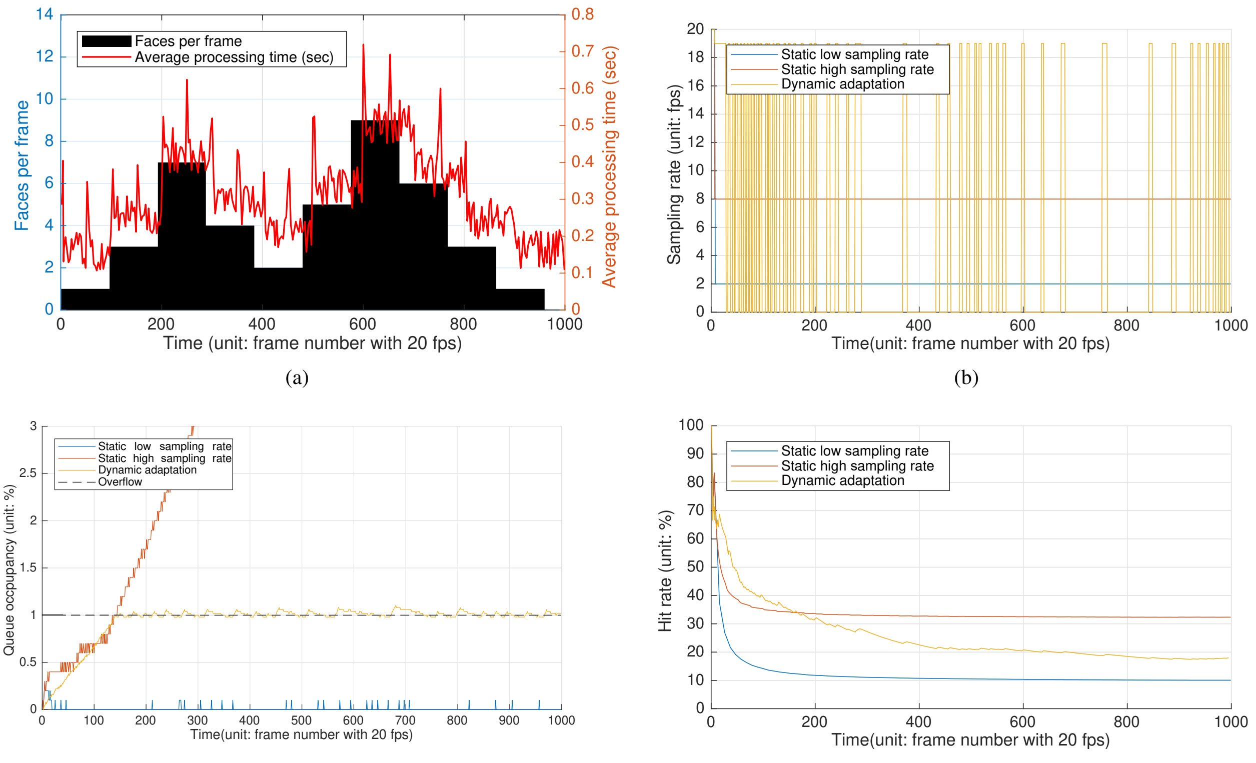
In the expression $\mathsf{In}(x(t),t) - \mathsf{Out}(x(t),t)$ which represents the change in the length of the queue, and the $\mathsf{In}(x(t))$ which means the amount of arrival items at queue during time t with the sampling rate $x(t)$ equals to the $x(t)$ due to the fact that the sampling rate stands for the arrival rate of the queue.

Unlike $\mathsf{In}(x(t),t)$, $\mathsf{Out}(x(t),t)$ which stands for the amount of the items processed in the queue during time t is independent to the sampling rate $x(t)$, and it is a constant that presents the number of processed images during time t and it will be represented by p in this paper. Eventually, $\mathsf{Out}(x(t),t)$ and $Q(t)$ in our Lyapunov optimization framework are constants which are independent to sampling rate $x(t)$. Finally, $Q(t) \cdot \mathsf{Out}(x(t),t)$ does not affect on our closed-form control formula in (3), thus $\mathsf{Out}(x(t),t)$ can be ignored, as well-discussed in Appendix A.

As a result, the updated closed-form time-average optimization equation subject to queue stability from (3) can be redesigned as follows:

$$x^* \leftarrow \arg\max_{x(t)\in\mathcal{X}} \{V' \cdot x(t) - Q(t) \cdot \mathsf{In}(x(t),t)\}, \tag{5}$$

In (5), V' should be determined according to the system design consideration, i.e., queue should be stabilized. Therefore, our decision should be made under the assumption that the queue backlog with the decision should be less than $Q_{\max}$ which is the user-defined limit length of the queue. According to the fact that this control formula is a kind of decreasing function that selects sampling rates start from the highest sampling rate at the beginning of the queue; and then it selects a lower sampling rate as the queue length increases, 0 sampling rate means there will be no inputs at next time, and means there will be no queue length increasing. Therefore, the stability of the queue can be guaranteed when the queue length is maximum $Q_{\max}$ by setting V' to be negative which is a time-average optimal sampling rate. According to the proof in Appendix B, $V' = Q_{\max}$ is derived to satisfy this requirement.



(a)

(b)

(c)

(d)

Fig. 2. Data-intensive performance evaluation results of sampling rate adaptation with (7). Fig. 2(a)–(d) show the performance evaluation results depending on various criteria such as sampling rate dynamics, queue stability, and hit ratio which is represented by the percentage of cumulative sampled image over total frames respectively as time elapsed: (a) Scenario, (b) changes in sampling rate, (c) stability of the queue, and (d) hit rate.

In summary, our time-average optimal sample rate adaptation via Lyapunov optimization framework works as follows: We can obtain the basic control equation as follows. First of all, V' should be determined and it can be derived from $Q_{\max}$, which is user defining limit length of the queue. Because this control equation is a monotonic decreasing expression that selects the highest sampling rate at the beginning of the queue and then gradually selects a lower sampling rate as the queue length increases, the length of the queue can not increase over than the length when the sampling rate is chosen to be zero. Therefore, if the sampling rate is set to 0 within the limit length $Q_{\max}$, it can be considered that the stability of the queue is guaranteed. To satisfy this condition, the control equation need to satisfy

$$\frac{\partial}{\partial x(t)}\left(V'\cdot x(t) - Q_{\max}\cdot x(t)\right) \le 0 \qquad (6)$$

when $x(t)$ is 0, and thus it can be shown that $V' = Q_{\max}$. Finally, our closed-form time-average sampling rate optimization subjec to queue stability can be re-formulated as follows from (5):

$$x^* \leftarrow \arg\max_{x(t)\in\mathcal{X}} \left\{Q_{\max}\cdot x(t) - Q(t)\cdot x(t)\right\}. \qquad (7)$$

D. Queue Exhaustion Consideration and Re-formulation

Even though the dynamic sampling rate adaptation with (7) (refer to Section II.C) guarantees the queue stability and maximizes the hit rate quite well, we have some chances to improve the basic algorithm because it selects only the maximum value, no matter how many sampling rates in this system have as candidates, until the queue length reaches the limit length as shown in Fig. 2(b). Therefore, this system adjusts sampling rate only after the queue has exhausted and it makes the system work as a non-queuing system, i.e., a system that waits for arrivals until processing is completed. In this situation, the algorithm will lose the advantages of the utilization of the queue. If the candidate contains a sampling rate that is too high to be processed in time, or if the performance of the system is kept lower than initially calculated, the queue will always be exhausted and will operate as a system without queues. When it happens, the system can not separate the image classifier from image acquirer any more, in addition, it can not be longer responded to the busty situation. In order to deal with this situation, the improvement of the sampling rate adjustment over several stages according to queue-backlog is desired.

The main issues in (7) happen because we ignored the fact that higher the sampling rate can also make the queue exhaust quicker. In order to consider this, we can multiply the objective function $U(x(t), t)$ with the penalty function which represents queue exhaustion rate, $E(x(t))$. The queue exhaustion rate $E(x(t))$ means the rate of the amount of items that left unprocessed (i.e., the amount of difference between arrived items and processed items) to the amount of processed times and the penalty function of this is presented as $1-E(x(t))$. The $E(x(t))$ can be expressed as $\frac{x(t)-p(t+1)}{p(t+1)}$ because $x(t) - p(t+1)$ means the amount of the surplus left unprocessed and exhaustion rate can be obtained by dividing $x(t) - p(t+1)$ by $p(t+1)$. Therefore, we can obtain the equation $\frac{2p(t+1)-x(t)}{p(t+1)}$ as the penalty function of queue exhaustion rate.

In addition to $\frac{2p(t+1)-x(t)}{p(t+1)}$, it should be noted that more attention must be paid to penalty of queue exhaustion rate as the queue is exhausted. It can be considered by adding an exponential weight $Q(t)/Q_{\max}$ that means the exhaustion rate of current queue to $E(x(t))$. Therefore, the penalty function of the queue exhaustion rate can be expressed as $\left(\frac{2p(t+1)-x(t)}{p(t+1)}\right)^{\frac{Q(t)}{Q_{\max}}}$.

There is another issue where $p(t+1)$ is hard to be predicted accurately in advance. As a simple solution to this problem, we can use $p(t)$ instead of $p(t+1)$ because we only need current or previous processing time just as the indicator to predict the next processing time to some extent, however there is no need to predict the exact processing time. In addition, V also can be simply derived similar to (7), and also due to Appendix C, we can derive the weight of objective function as $Q_{\max}/2$. To sum up, the final updated practical time-average optimal sampling rate adaptation with Lyapunov optimization framework can be described as follows:

$$x^* \leftarrow \arg \max_{x(t)\in\mathcal{X}} \left\{ \frac{Q_{\max}}{2} \cdot x(t) \cdot \left(\frac{2p(t+1)-x(t)}{p(t+1)}\right)^{\frac{Q(t)}{Q_{\max}}} - Q(t)\cdot x(t) \right\}, \quad (8)$$

where $x(t)$ is a decision action (i.e., sampling rate candidate) which is selected from $\mathcal{X}$ which a decision set, $Q_{\max}$ is maximum queue-backlog size (user-defined queue length limit), $p(t)$ is the amount of processed items during last processing cycle, $Q(t)$ is queue-backlog size at t, x^* is optimal decision, respectively.

In conclusion, the practical and updated time-average optimal sampling rate adaptation with (8) works as follows: It begins to recover queue stability all queues are exhausted and the queues do not exceed the queue-backlog limits what we specified. In Section III, more detailed experiments and data-intensive performance evaluation results are presented.

Algorithm 1: Frame rate control via Lyapunov optimization

```
1  Initialize:
2  $Q(t) \leftarrow 0$
3  $t \leftarrow 0$
4  Lyapunov-based dynamic sampling rate control:
5  while $t \leq T$ do
6      // $T$: operation time
7      $\mathcal{K}^* \leftarrow -\infty$
8      // $\mathcal{K}^*$ : index for obtaining maximum value calculation
9      Observe $Q(t)$
10     for $x(t) \in \mathcal{X}$ do
11         // $\mathcal{X}$: set of possible options
12         Compute $\mathcal{K}$ with (8)
13         if $\mathcal{K} \geq \mathcal{K}^*$ then
14             $\mathcal{K}^* \leftarrow \frac{Q_{\max}}{2} \cdot x(t) \cdot \left(\frac{2p(t+1)-x(t)}{p(t+1)}\right)^{\frac{Q(t)}{Q_{\max}}} - Q(t)\cdot x(t)$
                 $x^* \leftarrow x(t)$
15         end
16     end
17 end
```

Table 1. Specification of computing platforms.

System	Specification
CPU	• Intel(R) Core(TM) i5-2500 CPU @3.3GHz RAM: 8GB
GPU	• NVIDIA Quadro P4000 The number of cores: $1,792$ Memory: 8 GB GDDR5
Platform (PC)	CPU: Intel i5-6500 (3.2 GHz) Memory: DDR4 16GB SSD: 500GB (NVMe), 128GB HDD: 1TB VGA: RTX 2080 Ti

III. IMPLEMENTATION AND PERFORMANCE EVALUATION

In this section, the experimental setup for data-intensive performance evaluation is presented in Section III.A; and the details results are discussed in Section III.B, respectively.

A. Experimental Setup and Basic Results

For experiment-based performance evaluation, we utilize the existing real-time computer vision framework and improve it by adding a dynamic sampling rate adaptation algorithm which is proposed in this paper. The pseudo-code of the proposed algorithm is in Algorithm 1. All parameters are initialized in (lines 1–3). In (line 9), current $Q(t)$ is observed to be used in our proposed Lyapunov optimization framework. In (lines 10–16), the main computation procedure is described. The computation results are used for determining optimal sampling rate x^*, as shown in (line 14). According to the fact that our proposed algorithm solves a closed-form equation with the number of decision actions, its complexity is only $O(N)$ (low computational complexity). Our reference baseline software is an open source real-time facial recognition library `OpenFace` which is implemented with Google FaceNet [7]. Based on the `OpenFace`, our proposed dynamic real-time computer vision software is designed and implemented.

Fig. 2 shows the result of data-intensive performance evaluation for basic dynamic sampling rate adaptation with (7), and Figs. 2(a)–2(d) show the various performance evaluation results depending on various scenarios such as sampling rate dynamics, queue stability, and hit ratio which is represented by the percentage of cumulative sampled image over total frames respectively as time elapsed. In addition, static low or high sampling rates are used for performance comparison with this basic adaption algorithm, which means that the sampling rate is set at a low level by concentrating on the queue stability; and the sampling rate is set to a moderately high level by abandoning the full guarantee of stability respectively. In Figs. 2(c) and 2(d), it can be seen that the proposed dynamic sampling rate adaptation algorithm maximizes the hit-rate as high as possible while guaranteeing queue stability when (i) the static high sampling rate can not prevent queue overflow and (ii) the static low sampling rate shows low performance evaluation results.

In addition, for our realistically and practically modified proposed algorithm, we set the case where the performance changes every moment as shown in Fig. 3(a) as a test scenario. To emphasize the performance fluctuation of the device, we set the number of faces per image as a major factor of variation processing time and impose some arbitrary delays within several tens of milliseconds at every moment as a minor factor. In Fig. 3(a), the black bars represent the number of faces per image at each moment, and the red line represents the one of pre-calculated processing times under this test scenario on our test device. To emphasize the performance fluctuation, we also use CPU rather than GPU due to the fact that we can observe more unstable hazards on resource-limited environments. More details of our computing platform which is used in this real-world prototype-based performance evaluation results are summarized in Table 1. Note that one of the beauty of the proposed algorithm is that it adaptively works depending on queue-backlog. If the hardware/system specification is not good, the processing will be delayed, thus queue-backlog will be increased, i.e., the control action in the next time slot will be affected. Thus, our proposed algorithm is self-adaptive and it is independent to hardware/system specification in this local environment.

Although the proposed algorithm can be applied to real environments, we have experimented with extremely controlled pre-recorded video to perform experiments and ensure fair comparisons in various situations. In this experiment, we assume that the real frame flows at 20 frames per second, thus we have set the pre-built video process at 20 frames per second. To reflect the real-time flow of reality, we convert the progress of elapsed time into a frame number, then capture and discard the missed frames instead of capturing every frame in a sequential manner. For example, when capturing an image at a sampling rate of 10 frames per second, the next frame to be captured after processing the first frame of the test video is the 11th frame of the video. In addition, the frame to be processed 30 seconds after starting the video is the 300th frame. Lastly, the frame to be processed after 30 seconds from start will be the 300th frame of the video.

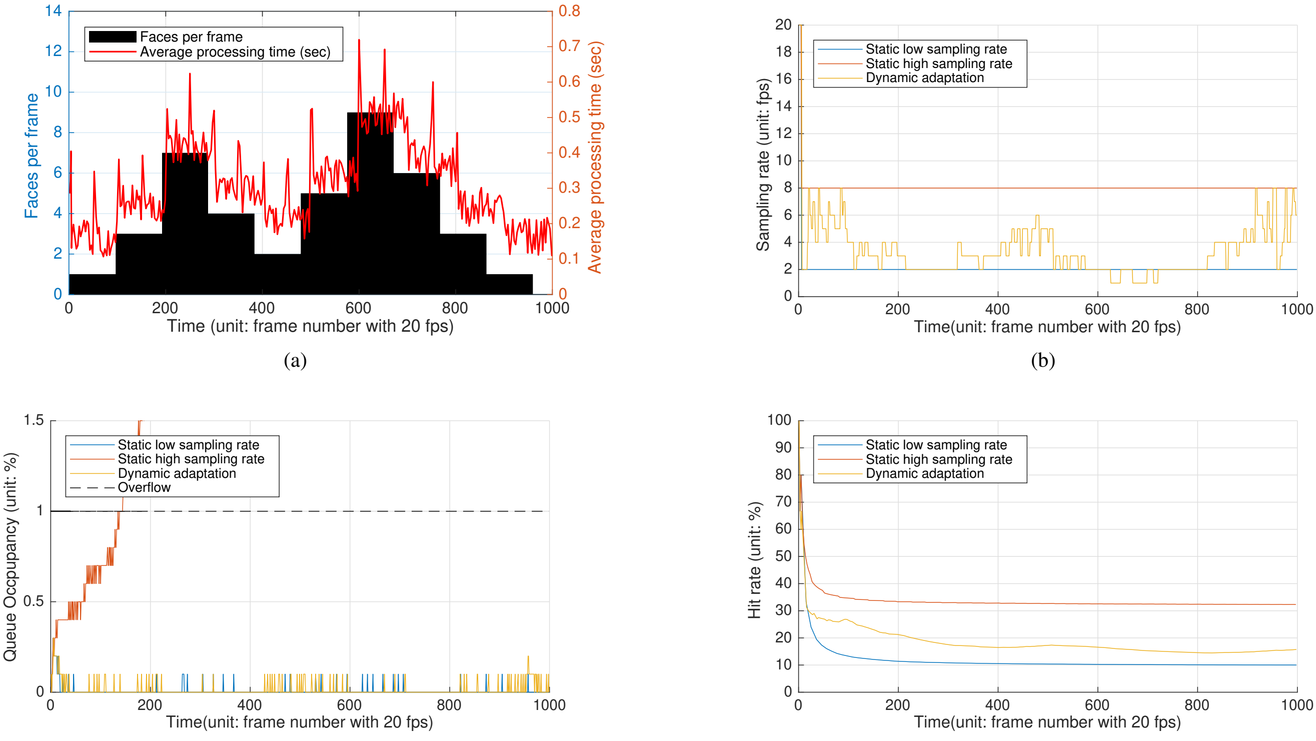
In the section that follows, we will compare the performance on the preset scenario with and without the dynamic sampling rate adaptation algorithm proposed in this paper. The considerations of the performance are queue stability which means the extent to which the stable point is formed within $Q_{\max}$ and hit-rate which is represented by the percentage of cumulative sampled images over total frames.

Fig. 3(b) represents the sampling rate with fps as a unit when the corresponding frame is input from the image acquire. The red, blue, and yellow lines represent the case of using static high, static low, and dynamically adapted sampling rate, respectively. Fig. 3(c) represents the ratio of the current queue length to the user defined queue limitation $Q_{\max}$ when the image is input from the image acquire. For example, in our test scenario, if the length of the queue was 3 when a new image was entered, the queue occupancy would be 30% because the limitation of the queue in this scenario is defined as 10. When the queue occupancy is over than 100%, we can consider it as overflow of the queue and it means that the system losses its reliability. Fig. 3(d) represents the hit-rate, which is the ratio of the number of processed frames to the total image frames that flowed. In our test scenario, assuming that reality is flowing at 20 fps, if the system processes 60 frames while a minute, the hit-rate will be 50% (60/120).

B. Performance Comparison

In order to evaluate the performance of the proposed dynamic sampling rate adaptation with Lyapunov optimization framework, we have compared it with static high sampling rate and static low sampling rate. The static low sampling rate stands for the case where the sampling rate is focused on the stability of the queue by setting it low. Therefore, with this scenario, frame per seconds is selected as 2 for static low sampling rate to handle the worst case. On the other hand, the static high sampling rate stands for the case where the sampling rate is relatively focused on the performance, and thus frame per second is selected as 8 for static high sampling rate. The static high sampling rate algorithm can handle (i) the case for momentary performance degradation and (ii) the case where it has longer processing time for taking items that will not happen often. In the performance evaluation, we choose the static values, i.e., 2 and 8, based on trial-and-error based simulations which can clearly show the novelty. As expected, if the static low sampling rate is lower (e.g., 1 instead of 2), more stability can be achieved. Similarly, if the static high sampling rate is higher (e.g., 10 instead of 8), it can achieve higher performance. Even though various static setting can be available, the given two example values (i.e., 2 and 8) for this performance evaluation are good enough to show the limitations on static sampling rate based algorithms. In addition, it is possible to consider dynamic sampling rate based algorithms in order to compare to the proposed algorithm. However, most of them are not associated with queue-based system modeling and its corresponding stochastic optimization. However, the proposed algorithm is designed for time-average performance maximization subject to queue stability, thus queue-related evaluations are essential, as shown in Figs. 2(c) and 3(c), whereas the conventional dynamic sampling rate based algorithms are not associated with queue-based modeling in general. Thus, the other dynamic sampling rate algorithms are not considered in this evaluation because we cannot do apple-to-apple comparisons with the proposed algorithm.

As shown in Figs. 3(c) and 3(d), the hit-rate of the system is reasonably good, however it becomes unstable even to induce queue overflow when the performance of the system is momentarily decreased in the case of static high sampling rate. On the other hands, in the case of static low sampling rate, it is stable during all operations, however it shows poor performance in general. Comparing to these results, in the case of the proposed dynamic sampling rate adaption algorithm, it shows high hit-rate while maintaining its stability. Therefore we can con-



(a) (b) (c) (d)

Fig. 3. Data-intensive performance evaluation results of sampling rate adaptation with (8). Fig. 3(a)–(d) show the performance evaluation results depending on various criteria such as sampling rate dynamics, queue stability, and hit ratio which is represented by the percentage of cumulative sampled image over total frames respectively as time elapsed: (a) Scenario, (b) changes in sampling rate, (c) stability of the queue, and (d) hit rate.

firm the effectiveness of the proposed dynamic sampling rate adaptation.

IV. DISCUSSION

In this section, several discussions conduct in order to provide useful insights for system design and implementation.

A. *Use of Accurate Processing Rates*

As mentioned above, the proposed algorithm only uses processing rate p as a side information (i.e., variables) for sampling rate optimization. Therefore, it does not matter whether there exists some prediction error or not. Thus, the proposed algorithm uses simply calculated value using the processing time measured in the previous timeslot as the processing rate p. However, it is still possible to improve the performance of the system by conducting accurate prediction of p. Furthermore, considering additional methods for more accurate prediction of p, it can be directly applied to the proposed algorithm without additional computational complexity.

B. *Hardness on $Q_{\max}$ Determination*

In the proposed algorithm, the observed time constraint is based on the user-defined limit length of queue $Q_{\max}$. This $Q_{\max}$ also refers to the system memory limit itself, or the limit beyond which the queuing delay is estimated to be too long to satisfy the time constraints. In latter situation, it is practically impossible to determine the upper bound of the queue length because it should be exactly predictable how many times are spent per image. In most cases, it will be able to restore queue stability before exceeding the upper bound. However, it is obviously risky which may introduce queue overflow. As an alternative to this problem, we can consider to determine $Q_{\max}$ dynamically as well as the sampling rate.

C. *Frequent Sampling Rate Updates*

Even though the processing speed of the system is set to change frequently than usual, the sampling rate appeared in Fig. 3(b) shows that the sampling rate is frequently fluctuating. It is the result of the determining new optimal sampling rates for frame-by-frame processing. Therefore we can make the sampling rate less fluctuate by reducing the period to determine the new sampling rate. However, if the decision period is too short, it may not reflect the instantaneous performance degradation, thus the stability of the system can not be guaranteed. Therefore, there exists a new trade-off about this, however we do not discuss it in this paper, and it can be additional new sampling rate decision criteria for frame-by-frame processing.

D. *Complicated Computational Procedures*

The proposed closed-form equation for time-average dynamic optimization is quite simple and thus the computational complexity of the algorithm is polynomial-time. However additional simplification can be achieved using $x(t) = X_{\max} \cdot (|Q(t)|/Q_{\max})$ instead of the proposed equation where $X_{\max}$ stands for maximum hit-rate. This simplified equation is also able to prevent queue overflow and maximizes hit-rate. However, the user has to always suffer certain amounts of queuing delays due to the fact that the queue backlog is always exhausted. This can be an inconvenient factor depending on the situation even though the time constraint is satisfied. Therefore, it can be recommended to use for the systems which are not sensitive to response time.

V. CONCLUDING REMARKS

Today, we can observe various surveillance-based security applications with real-time computer vision system spreading into our daily lives even in form of mobile and IoT applications that often have limitations in the amount of computation resources available. In this system, utilization of buffer/queue for allowing certain amounts of delays is useful for making these offloading-based distributed systems efficient. However, to execute the computer vision based intensive computation on the devices which have insufficient computational resource such as mobile and embedded IoT devices, the sampling rate should be adapted dynamically during operation and offloading. This situation is due to the fact that the limitations let the performance of the IoT devices be fluctuated severely and eventually it makes hard to obtain reasonable and reliable static sampling rate which is able to guarantee queue stability and performance maximization. In this paper, we propose a dynamic sampling rate adaptation algorithm based on Lyapunov optimization framework which optimizes time-average utility subject to queue stability. To the best of our knowledge, the proposed algorithm in this paper is the first trial which focuses on system-wide optimal control in automated vision-based surveillance systems under the consideration of queue stability. Furthermore, we discuss several view points those can be useful for system design and implementation. The performance of the proposed dynamic control algorithm are verified through experiment-based intensive performance evaluation.

As future work, further investigations should be conducted for resolving the discussed issues in terms of the use of accurate processing rate (in Section IV.A) and max-queue determination (in Section IV.B).

REFERENCES

[1] A. Booranawong, N. Jindapetch, and H. Saito, “Adaptive filtering methods for RSSI signals in a device-free human detection and tracking system," *IEEE Syst. J.*, vol. 13, no. 3, pp. 2998–3009, Sept. 2019.

[2] A. V. Savkin and H. Huang, “A method for optimized deployment of a network of surveillance aerial drones," *IEEE Syst. J.*, vol. 13, no. 4, pp. 4474–4477, Dec. 2019.

[3] C. Huang, H. Wang, H. Zhou, S. Xu, and D. Ren, “EVAC-AV: The live road surveillance control scheme using an effective-vision area-based clustering algorithm with the adaptive video-streaming technique," *IEEE Syst. J.*, vol. 11, no. 3, pp. 1228–1238, Sept. 2017.

[4] X. Zhang, H. Wu, M. Wu, and C. Wu, “Extended motion diffusion based change detection for airport ground surveillance," *IEEE Trans. Image Process.*, vol. 29, pp. 5677–5686, 2020.

[5] H. Du, L. Chen, J. Qian, J. Hou, T. Jung, and X. Li, “PatronuS: A system for privacy-preserving cloud video surveillance," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1252–1261, June 2020.

[6] J. Kim and W. Lee, “Feasibility study of 60 GHz millimeter-wave technologies for hyperconnected fog computing applications," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1165–1173, Oct. 2017.

[7] OpenFace, https://cmusatyalab.github.io/openface/, 2017.

[8] B. Amos, B. Ludwiczuk, and M. Satyanarayanan, “OpenFace: A general-purpose face recognition library with mobile applications," *Carnegie Mellon Univ., School of Computer Science, TR CMU-CS-16-118*, June 2016.

[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE CVPR*, June 2014, pp. 580—587.

[10] A. Goyal *et al.*, “Automatic border surveillance using machine learning in remote video surveillance systems," *Emerging Trends in Elec., Communi., and Inf. Technol.*, vol. 569, pp 751–760, Sept. 2019.

[11] G. Sreenu and M. A. Saleem Durai, “Intelligent video surveillance: A review through deep learning techniques for crowd analysis," *J. Big Data*, vol. 6, no. 48, June 2019.

[12] H. Chang, D. Zhao, C. H. Wu, L. Li, N. Si, and R. He, “Visualization of spatial matching features during deep person re-identification," *J. Ambient Intell. Humaniz. Comput.*, Feb. 2020.

[13] Q. Guo, Q. Liu, W. Wang, Y. Zhang, and Q. Kang, “A fast occluded passenger detector based on MetroNet and Tiny MetroNet," *Information Sciences*, vol. 534, pp. 16–26, Sept. 2020.

[14] C. Sipetas, A. Keklikoglou, and E. J. Gonzales, “Estimation of left behind subway passengers through archived data and video image processing," *Trans. Res. Part C: Emer. Technol.*, vol. 118, p. 102727, Sept. 2020.

[15] G. T. S. Ho,Y. P. Tsang, C. H. Wu, W. H. Wong, and K. L. Choy, “A computer vision-based roadside occupation surveillance system for intelligent transport in smart cities," *Sensors*, vol. 19, no. 8, p. 1796, Apr. 2019.

[16] D. Kim, J. Kwon, and J. Kim, “Low-complexity online model selection with Lyapunov control for reward maximization in stabilized real-time deep learning platforms," in *Proc. IEEE SMC*, Oct. 2018.

[17] D. Kim, J. Kim, and J. Y. Bang, “A reliable, self-adaptive face identification framework via Lyapunov optimization," in *Proc. ACM SOSP AISys*, Oct. 2017.

[18] K. S. Kim, D. Kwon, Y. Kim, J. Kim, and J. Kim, “Self-adaptive machine learning operating systems for security applications," in *Proc. IEEE ICASSP*, Apr. 2018.

[19] M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*, Morgan & Claypool, 2010.

[20] S. Jung, J. Kim, and J.-H. Kim, “Intelligent active queue management for stabilized QoS guarantees in 5G mobile networks," *IEEE Syst. J.*, 2021.

[21] J. Kim, G. Caire, and A. F. Molisch, “Quality-aware streaming and scheduling for device-to-device video delivery," *IEEE/ACM Trans. Netw.*, vol. 24, no. 4, pp. 2319–2331, Aug. 2016.

[22] J. Koo, J. Yi, J. Kim, M. A. Hoque, and S. Choi, “REQUEST: Seamless dynamic adaptive streaming over HTTP for multi-homed smartphone under resource constraints," in *Proc. ACM Multimedia*, Oct. 2017.

[23] M. Choi, A.F. Molisch, and J. Kim, “Joint distributed link scheduling and power allocation for content delivery in wireless caching networks," *IEEE Trans. Wireless Commun.*, vol. 19, no. 12, pp. 7810–7824, Dec. 2020.

[24] J. Koo, J. Yi, J. Kim, M. A. Hoque, and S. Choi, “Seamless dynamic adaptive streaming in LTE/Wi-Fi integrated network under smartphone resource constraints," *IEEE Trans. Mob. Comput.*, vol. 18, no. 7, pp. 1647–1660, July 2019.

[25] M. Choi, J. Kim, and J. Moon, “Wireless video caching and dynamic streaming under differentiated quality requirements," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 6, pp. 1245–1257, June 2018.

[26] M. Choi, A. No, M. Ji, and J. Kim, “Markov decision policies for dynamic video delivery in wireless caching networks," *IEEE Trans. Wireless Commun.*, vol. 18, no. 12, pp. 5705–5718, Dec. 2019.

[27] M. Choi, J. Kim, and J. Moon, “Dynamic power allocation and user scheduling for power-efficient and delay-constrained multiple access networks," *IEEE Trans. Wireless Commun.*, vol. 18, no. 10, pp. 4846–4858, Oct. 2019.

[28] N.-N. Dao, D.-N. Vu, W. Na, J. Kim, and S. Cho, “SGCO: Stabilized green crosshaul orchestration for dense IoT offloading services," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 11, pp. 2538–2548, Nov. 2018.

APPENDIX A
THE EFFECT OF $\mathsf{Out}(x(t), t)$

The $\mathsf{Out}(x(t), t)$ and $Q(t)$ in our Lyapunov optimization framework are constants which are independent to sampling rate $x(t)$. According to the fact that (7) equals to

$$x^* \leftarrow \arg \max_{x(t)\in\mathcal{X}} \left\{V \cdot x(t) - Q(t) \cdot x(t) + Q(t) \cdot p\right\}, \quad (9)$$

where the change in p only makes a vertical parallel shift to the function. This vertical shift does not affect on the result of the (7), thus the $\mathsf{Out}(x(t), t)$ can be ignored.

APPENDIX B
V DERIVATION FOR (7)

Due to the fact that $V \cdot x(t)$ and $Q(t) \cdot x(t)$ are linear, the V which is the gradient of first term has to be lower than $Q(t)$ in order to to select sampling rate $x(t)$ less than 0 which is optimum when the queue length is $Q_{\max}$. Then the gradient of second term when $Q(t) = Q_{\max}$. Finally, $V = Q_{\max}$.

APPENDIX C
V DERIVATION FOR (8)

According to the Lyapunov drift model, i.e.,

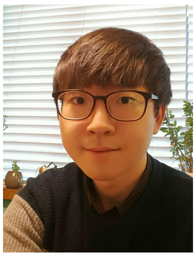
$$V \cdot x(t) \cdot \frac{(2p - x(t))^{\frac{Q(t)}{Q_{\max}}}}{p} - Q_{\max} \cdot x(t), \tag{10}$$

which is a convex quadratic function, the optimal sampling rate $x(t)$ is chosen by the partial derivative of (10), i.e.,

$$\frac{\partial}{\partial x(t)} \left(V \cdot x(t) \cdot \frac{(2p - x(t))^{\frac{Q(t)}{Q_{\max}}}}{p} - Q_{\max} \cdot x(t) \right), \tag{11}$$

and letting this (11) be zero. Therefore, in order to select sampling rate x less than 0 which is optimum when the queue length is maximum, the (11) should be less than equal to zero; and this holds when $x = 0$ and $Q(t) = Q_{\max}$.

Based on this observation, it can be justified that $V = Q_{\max}/2$.



Dohyun Kim is currently a Researcher at Naver Webtoon Corporation, Seongnam, Korea. He received his B.S. and M.S. degrees in Computer Science and Engineering from Chung-Ang University, Seoul, Korea, in 2018 and 2020, respectively. His research focus includes computer vision and deep learning theories and their applications.

Soohyun Park is currently a Ph.D. candidate in Electrical Engineering at Korea University, Seoul, Republic of Korea. She received her B.S. degree in Computer Science and Engineering from Chung-Ang University, Seoul, Republic of Korea, in 2019. Her research focus includes deep learning and machine learning algorithms and their applications. She was a recipient of IEEE Vehicular Technology Society (VTS) Seoul Chapter Award (2019).

Joongheon Kim (M'06–SM'18) has been with the School of Electrical Engineering, Korea University, Seoul, Korea, since 2019, where he is currently an Assistant Professor. He received the B.S. and M.S. degrees in computer science and engineering from Korea University, Seoul, Korea, in 2004 and 2006, respectively; and the Ph.D. degree in computer science from the University of Southern California (USC), Los Angeles, CA, USA, in 2014. Before joining Korea University, he was with LG Electronics (Seoul, Korea, 2006–2009), InterDigital (San Diego, CA, USA, 2012), Intel Corporation (Santa Clara in Silicon Valley, CA, USA, 2013–2016), and Chung-Ang University (Seoul, Korea, 2016–2019).

He serves as an Associate Editor for *IEEE Transactions on Vehicular Technology*. He internationally published more than 80 journals, 110 conference papers, and 6 book chapters. He also hols more than 50 granted patents. He was a Recipient of Annenberg Graduate Fellowship with his Ph.D. admission from USC (2009), Intel Corporation Next Generation and Standards (NGS) Division Recognition Award (2015), IEEE Vehicular Technology Society (VTS) Seoul Chapter Award (2019), *IEEE Systems Journal* Best Paper Award (2020), and IEEE ICOIN Best Paper Award (2021).

Jae young Bang received the M.S. and Ph.D. degrees in Computer Science from the University of Southern California. He was a USC Annenberg graduate fellow. He is currently a Researcher at Kakao Corporation. His research interest spans from collaborative software development to distributed software system architectures. More information is available on his homepage: http://ronia.net/.

Soyi Jung has been a Research Professor at Korea University, Seoul, Republic of Korea, since 2021. She also holds a postdoctoral visiting scholar position at Donald Bren School of Information and Computer Sciences, University of California, Irvine, CA, USA. She received her B.S., M.S., and Ph.D. degrees in electrical and computer engineering from Ajou University, Suwon, Republic of Korea, in 2013, 2015, and 2021, respectively. From March 2015 to February 2016, she worked as a researcher at Korea Testing and Research (KTR) Institute.

Her current research interests include network optimization for autonomous vehicles communications, distributed system analysis, big-data processing platforms, and probabilistic access analysis. She was a Recipient of Best Paper Award by KICS (2015), Young Women Researcher Award by WISET and KICS (2015), Bronze Paper Award from IEEE Seoul Section Student Paper Contest (2018), ICT Paper Contest Award by Electronic Times (2019), and IEEE ICOIN Best Paper Award (2021).