Testbed Design of Near Real-time Wireless Image Streaming with Apache Kafka for Road Traffic Monitoring

Aung Myo Htut[1], Soe Ye Htet[2], Khamxay Leevangtou[3], Kiattikun Kawila[4], Chaodit Aswakul[5]

Wireless Network and Future Internet Research Unit,
[1,2,3,5]Department of Electrical Engineering,
[4]Department of Computer Engineering,
Faculty of Engineering, Chulalongkorn University, Bangkok, Thailand
E-mail: [1]aung.my@student.chula.ac.th, [2]soe.y@student.chula.ac.th, [3]khamxay.l@student.chula.ac.th, [4]kiattikun.kaw@student.chula.ac.th, [5]chaodit.a@chula.ac.th

Abstract

In this paper, the testbed design of near real-time wireless image streaming is proposed for road traffic monitoring. Images of road segments are captured by Raspberry Pi cameras and sent wirelessly to a Kafka broker at the relevant traffic police controller box. Traffic police can pull these image sequences from the Kafka broker to watch and make a judgement on road traffic congestion levels. To avoid costly communication cabling, our system design uses wireless ad-hoc networking. As a proof of concept, a small testbed has been built with two Raspberry Pi's serving as Kafka producers and an Intel® NUC, serving as Kafka broker as well as Kafka consumer. Finally, based on the testbed, preliminary testing has been conducted with the focus on the retrievable image qualities and the end-to-end latency.

Keywords: real-time image streaming, wireless, Apache Kafka, Raspberry Pi

1. Introduction

Nowadays, big cities face inevitable congested road traffic problems. While the limitation of the road network capacity is a main cause of that problem [1], lacking of real-time or near real-time information about the road congestion status for the traffic signal light control operation is unarguably another main factor. If the traffic signal light can be controlled effectively, the road traffic will flow smoothly [2].

To control the traffic signal light, near real-time road traffic information (with up to 5 seconds latency) is needed. Such information is derived from the traditional loop detectors buried on the road surface, or as recently-made-popular from Closed-Circuit Television (CCTV). However, relying on the conventional CCTV with needed communication cabling infrastructure is costly. In this paper, we have adopted the concept of having instead this information sent to the traffic signal light control box in the form of image sequences over the wireless mesh networking links. These images must be updated frequently, in our work every 0.2 seconds, so that the images can be played back smoothly to traffic police. The challenge is in terms of the system scalability to cope with potentially large amount of data. Therefore, a big data framework is necessary.

With Apache Kafka [3], images can be captured by the Raspberry Pi camera and relayed to neighbouring Raspberry Pi's. In this case, a Raspberry Pi can serve as a Kafka producer and all the producers publish the image sequences as Kafka messages to the corresponding Kafka broker efficiently. Kafka consumers or traffic police users can then pull the messages from the broker to watch out for current incidences on their responsible road network.

The objective of this research is to design a small testbed of near real-time image streaming over the wireless ad-hoc links with the use case for road traffic monitoring with the enabler technology within the Apache Kafka. This paper reports the system design as well as, based on the constructed testbed, preliminary testing results with the focus on the retrievable image qualities and the corresponding end-to-end latency.

2. Kafka-Based System Design

Streaming is the way of transmitting and receiving the data in the continuous flow [4]. In a real-time or near real-time streaming, the smooth data flow is needed. To have a smooth data flow, the system needs a low latency and high enough bandwidth.

Apache Kafka (hereinafter called Kafka) [3] is one of the big data streaming platforms. Kafka is the publish-subscribe messaging system where data is sent as messages to a chosen topic managed by a Kafka broker. Any data can be sent by Kafka as the message formatted with the specific allowable data size. Large chunk of data is divided into small messages.

Despite the intent of Kafka for streaming large amount of data, Kafka has been used mostly for data that can be dealt with in the form of messages. Few works in the past have tried to adopt Kafka for real-time video streaming. As Kafka is not built for video-type data, there are limitations in video processing capability that must be efficiently modified to instantiate such a streaming session [5], [6].

To make a direct use of inherent Kafka features, in this paper, we have instead chosen to send the image sequence as a continuous flow of messages. In our proposed design, the network uses the wireless ad-hoc mode with the medium-range reachability distance according to the legally allowed maximum wireless transmission power over an Industrial, Scientific and Medical (ISM) 5-GHz band. Therefore, over a multiple-hop route, the end-to-end throughput and latency are expectedly a concern. To cope with those foreseeable problems, the image quality and updating frequency must be designed properly. Also, target traffic police users must be able to play back those image sequences smoothly enough to allow them to monitor the near real-time condition of the road traffics.

In this paper, Kafka producer is installed in the Raspberry Pi. This is a cost-effective hardware choice thanks to the computing power that should be enough to run all the necessary software modules for image capturing and streaming. Kafka broker is installed in the Intel® NUC to opt for a high computing power and storage space that would be needed as the Kafka broker and consumer functionalities.

To demonstrate our proof of concept, a small testbed has been constructed with 2 Raspberry Pi's and an Intel® NUC as shown in Table I. Two Raspberry Pi's are placed along the road side. Each Raspberry Pi camera captures images every 0.2 seconds. The images are published by relaying to the Intel® NUC serving as the Kafka broker over the wireless ad-hoc network links.

In the constructed testbed, there are two wireless ad-hoc hops and each hop spans from 200 to 250 meters. This testbed is expectedly installed at the crossover bridges along the Phaya Thai road segment in between Rama 1 road and Rama 4 road, Bangkok, Thailand. On this selected road segment, there are crossover bridges whose locations match with our wireless ad-hoc link span. That is, the bridges are separated with the distance around 250 meters as allowed by the line-of-sight 5-GHz Wi-Fi link being used in the testbed design.

Table I: Testbed Device Specification.

Device	Aim	Operation system
Raspberry Pi 3 Model B	Kafka producer	Ubuntu mate 16.04
Intel® NUC7i7BNH	Kafka Broker and Kafka consumer	Ubuntu mate 16.04

The images from the Raspberry Pi camera are published and stored with the specific topic inside the broker which can be placed inside the traffic police controller box, and hence a typical usage scenario in Figs. 1 and 2 The users inside the traffic police controller box can then pull the data from the broker by running a Kafka consumer software.

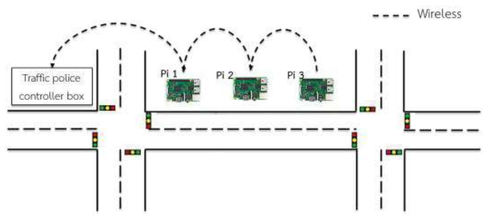

Fig. 1. Location of each Raspberry Pi and traffic police controller box in typical usage scenario.

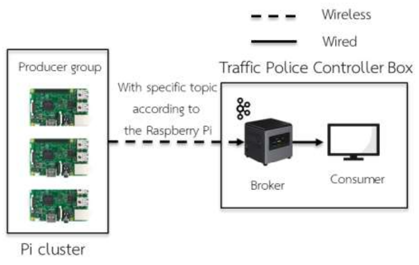
Wireless
Wired
Producer group
Traffic Police Controller Box
With specific topic according to the Raspberry Pi
Broker
Consumer
Pi cluster

Fig. 2. Proposed design of image streaming.

Each Raspberry Pi must relay the image data to the neighbouring Raspberry Pi in trying to reach finally at the Kafka broker in the traffic police controller box. For every 1 kBytes of captured image, at least 8 kBits / 0.2 sec = 40 kBits/s of link throughput is needed per hop. The Raspberry Pi nearer to the final node along the route needs higher throughput than that of the previous Raspberry Pi at the route upstream. The image size and end-to-end latency, both affecting the needed throughput as well as resultant image streaming quality, must therefore be investigated carefully in the testbed. Therefore, choosing the right quality of the image which is watchable for the user is the testbed testing target.

3. Preliminary Test

In the preliminary test, one Intel® NUC and two Raspberry Pi's have been used. All the devices have been installed at the crossover bridges on the Phaya Thai road as shown in Figs. 3-5. The road traffic condition has been captured in each Raspberry Pi camera and sent as an image sequence, with image snapshot examples shown in Fig. 6. In this test, distance between Intel® NUC and Raspberry Pi 1 has been 272 meters and distance between Intel® NUC and Raspberry Pi 2 has been 240 meters respectively.

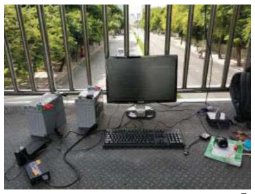
Fig. 3. Equipment settings of Intel® NUC-based Kafka broker and consumer.

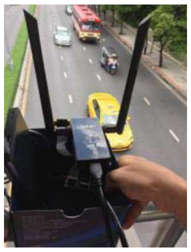
Fig. 4. Equipment settings of Raspberry Pi-based Kafka producer.

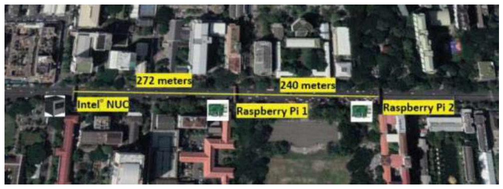

Fig. 5. Locations of devices during preliminary test at the crossover bridges on Phaya Thai road shown on Google® Earth.

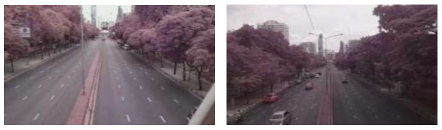
(a) (b)

Fig. 6. 320 × 180 resolution images from Raspberry Pi 1 and 2 in Figs. 6(a) and 6(b).

To carry out the preliminary test at this outdoor environment, Intel® NUC has been powered by the 12-Volt car battery. For each Raspberry Pi, a 20000-mA-hour power bank has been used. In order to have medium distance, a dual-band EDUP EP-AC1605 Wi-Fi USB adapter with two omnidirectional antennas have been installed to all devices (Raspberry Pi's and Intel® NUC). These antennas have been connected in the line-of-sight with 5-GHz band channel. The average TCP throughput has been found to be 10.2 MBits/s from Raspberry Pi 1 to Intel® NUC and 3.9 MBits/s from Raspberry Pi 2 to Intel® NUC.

All the Raspberry Pi's have produced the images to the Kafka broker at every 0.2 seconds. In this preliminary test, 320×180, 480×360, 720×480 and 1080×720 resolution JPEG images have been tested and the size of a captured image is averaged to 7.99 KBytes, 20.7 KBytes, 33.55 KBytes and 65.35 KBytes, respectively.

To measure the end-to-end latency, 15000 packets have been captured by TCPDUMP [7] at all the devices and the timestamped packets related only with images from Raspberry Pi 1 and Raspberry Pi 2 are filtered for latency computation. The clock of all the devices has been synchronised by Network Time Protocol (NTP) before starting the experiment. The measured average latency is shown in Fig. 7.

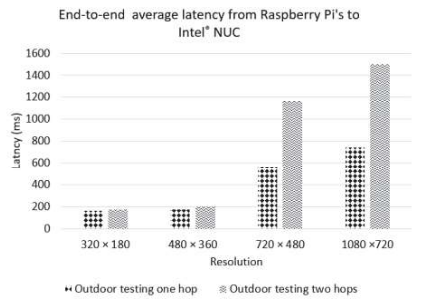

Fig. 7. End-to-end average latency from Raspberry Pi's to Intel® NUC.

Images from each Kafka producer are captured and streamed to the Kafka broker. To reduce the extra dalay in storing the images from the camera to Raspberry Pi and then reading them back, we have configured to stream directly those images to the Kafka broker just after capturing the images.

In the outdoor test, the distance is far and there are some noise and interference between each device. Therefore, some packets are lost, retransmitted and duplicated. Each producer needs to wait until the pervious image has arrived at the Kafka broker before sending the next image. This can give a good reliability, but the average latency is higher if the pervious image cannot at all arrive at the Kafka broker when for instance there is a wireless link failure.

By the tested results, users can watch the near real-time image streaming, as expected by the initial design target. The 320×180 and 480×360 resolution image sequences can give significantly lower latency than any higher-resolution image sequences. These images should be good enough in quality for traffic police users to watch over road traffic conditions.

4. Conclusion

In this paper, a small-testbed design of near real-time image streaming over wireless ad-hoc links with the use case for road traffic monitoring with the Apache Kafka has been given. User can monitor the near real-time streaming of the road traffic conditions in terms of image sequences with low latency. With the preliminary testbed, the image streaming quality and the end-to-end latency are useable. However, when the number of hops and distance is increased, the latency is increased.

In the future, after the proof of concept testbed has been verified of its functionalities, we would investigate the system scalability aspect by introducing more Kafka producers and brokers as well as consumers. It is hoped that this system can be cost-effectively useful for future realistic road traffic monitoring.

References

[1] M. Papageorgiou, C. Diakaki, V. Dinopoulou, A. Kotsialos, and Y. Wang, "Review of road traffic control strategies," *Proceedings of the IEEE,* vol. 91, no. 12, pp. 2043-2067, 2003.

[2] K. Nellore and G. P. Hancke, "A survey on urban traffic management system using wireless sensor networks," *Sensors,* vol. 16, no. 2, p. 157, 2016.

[3] G. Wang *et al.*, "Building a replicated logging system with Apache Kafka," *Proceedings of the VLDB Endowment,* vol. 8, no. 12, pp. 1654-1655, 2015.

[4] D. Wu, Y. T. Hou, W. Zhu, Y.-Q. Zhang, and J. M. Peha, "Streaming video over the Internet: approaches and directions," *IEEE Transactions on circuits and systems for video technology,* vol. 11, no. 3, pp. 282-300, 2001.

[5] X. Guo, Y. Cao, and J. Tao, "SVIS: large scale video data ingestion into big data platform," in *International Conference on Database Systems for Advanced Applications*, 2015, pp. 300-306: Springer.

[6] A. Ichinose, A. Takefusa, H. Nakada, and M. Oguchi, "A study of a video analysis framework using Kafka and spark streaming," in *Big Data (Big Data), 2017 IEEE International Conference on*, 2017, pp. 2396-2401: IEEE.

[7] "Tcpdump." [Online]. Available: https://www.tcpdump.org