A Real Time Draggable Frame Capture System with Mobile Device

by

Ching-Chun Lu

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2017

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

In this report, a real time draggable frame grabber media system is proposed. The user can simultaneously acquire the snapshot from a live video playing on a display by simply using an intuitive drag hand gesture with their mobile device. We achieve the operating time shorter than 1 second, which let user be able to get the frame of video showing on the display instantly. Moreover, the system supports many-to-many service with iBeacon and multiple displays. The system includes the server, client and video source. We will introduce about structure, implementation and performance analysis.

Acknowledgements

I would like to thank my supervisor Prof. Pin-Han Ho for all the support and encouragement to my study in MASc program and this work. I would also like to thank all people who made this thesis possible.

Dedication

This is dedicated to the one I love.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

With the rapid development of mobile industry and internet technology, our life styles have been changed considerably, such as the way we communicate with each other and the tool that we utilize to record our life. Moreover, the technology improvement of mobile multimedia and cloud computing provides more entertaining services with better quality of service (QoS) [1]

1.1 Motivation

Web services with cloud server is wild use in online advertising system, however, most of them are putting contents in the server previously. For more dynamic and interactive use, we try to set the event which is holding as data source. For example, in a concert or a live ball game, users can get the snapshots, live video stream or information just updated in real time. This could be achieved and implemented with the developed mobile cloud computing technology.

1.2 Thesis Contributes

A real time draggagle frame grabber system is purposed in this report. The system presents a real time frames capture for mobile users from multiple video displays as one of the content provider. In other words, customers with mobile devices can get the snapshot of the video

which is showing on the display with a simple drag movement. In present time there are many way of mobile advertisement, now we are doing it LIVE.

Figure 1.1. shows the architecture of part of the system. It consists display (video source); frame grabber device; cloud sever and application for mobile users.

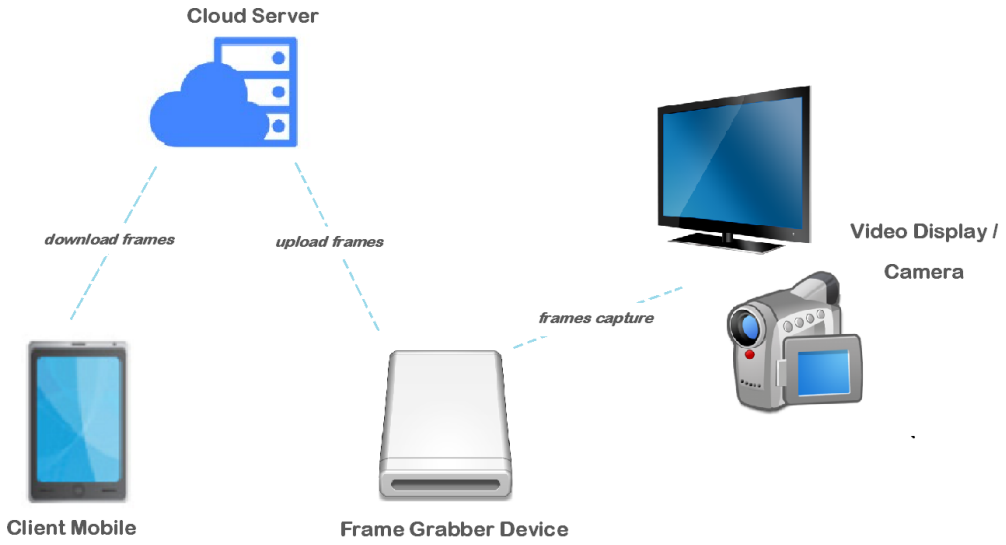


Figure 1.1: Illustration of the draggable frame grabber system

The frames source could be from camera recording or a video playing display with hdmi output. The display, which could be LED, LCD, digital billboard, or any other format such as projector screens, is used to provide visualization function to the viewers. The frame grabber (epiphan DVI2USB 3.0) device is designed to capture video from any VGA, DVI, or HDMI source with resolutions up to 19201200 at 60 frames per second. The frame grabber captures frames from the HDMI output video source, and connected to a computer via USB 3.0. Frames are uploaded to google cloud storage and saved in the server. The other content provider is the boradcasting slideshow. The mobile application detects hand movement of user, and a request will be sent to the cloud sever once a drag detected to download the relative image. The following section will expound the system in detail.

Chapter 2

Background

In this chapter, we will intriduce the cloud computing and the Google Cloud Platform. Moreover, there is a study of a smart signage system with smartphone, wireless communication and digital display.

2.1 Cloud computing

Cloud computing is an Internet-based technology that provides various computing and storage services. Users can store and share their data through the cloud instead of their own local device. Moreover, they are able to develop and deploy applications with developer tools on the cloud. Astri [2] reviews and identifies the critical success factors as the impact and adventage of cloud computing, which are cost reducing, flexible, redundancy and reliability, scalability, collaboration, efficiency,virtually and availability.

2.1.1 NIST Definition of Cloud Computing

The U.S. National Institute of Standards and Technology (NIST) Definition of Cloud Computing lists five essential characteristics, three service models, and four deployment models of Cloud Computing [3].

On-demand self-service. A user can get the cloud service with his needs such as storage or launching applications without going through to the service provider. **Broad network access**. A user could access advanced multimedia services anytime, anywhere,

and from any device without any limitations. **Resource pooling**. Multi-tenancy enables the system to serve different consumers (tenants) whereby each is isolated from the other. Provider pooled the computing resources to serve multiple cloud service consumers by using multi-tenancy models. It allows several users to use the same resources without aware each other and the resource location. **Rapid elasticity**. The automated capability of the cloud to scale the resources. The service can be adjusted very flexibly depends on the consumer demand, which means additional storage space or server bandwidth can be done on-demand. Users can purchase any quantity at any time, and data can be replicated to several data center around the world. **Measured service**. Any services in the cloud system are controlled and monitored by the provider, such as purchasing, storage, processing, bandwidth, and active user accounts. This characteristic makes sure the resource management and the transparent services between consumer and provider.

Software as a service (SaaS) allows users to use the providers applications on the cloud, which is what most people think when they say cloud service. Most APIs are also associated with SaaS. Platform as a service (PaaS) is similar to SaaS, but instead of only use the applications or services. It provides a developing environment such as develop tools, IDE and the management of the environment processing (design, development, test and deploy). Consumer can use the application developing environment which includes programming-language execution environment, database, and web serve without managing the cloud structure (networking, operating system or storage). Infrastructure as a service (IaaS) provides virtualised hardware or computing infrastructure, which offers a self-service environment for users with compute, storage and networking service. This let consumer feel like using the real device and no need to think about management or maintenance of the device. Table 2.1 shows the variations of the service models.

Service \ Model	SaaS	PaaS	IaaS
Hosted applications suites of services	✓		
Development tools and database management	✓	✓	
Operation system	✓	✓	✓
Virtualization	✓	✓	✓
Servers and storage	✓	✓	✓
Networking firewalls/security	✓	✓	✓
Data center physical plant/building	✓	✓	✓

Table 2.1: Service models variation

Private cloud is created by a single organization. The data can only be used in the

specific private network and cannot be shared by other organization. If the data center is shared, then we call it is a virtual private cloud. Community cloud is built by many organizations that have shared interests. The goal is to achieve the optimize benefits within these organizations. Public cloud is provisioned for open use by the general public, it could also be charged and permission needed depends on service provider. Consumers use the service without considering the management of the system such as set up, host or back up the data center. Hybrid clouds can be any combination of the above three. For example, the developer may launch the less sensitive data in the public cloud but put more sensitive data in private cloud.

2.1.2 Cloud Aware Multimedia Applications

Zhu [4] presented a media life circle of cloud-aware multimedia shown as Figure 2.1, which is composed of acquisition, storage, processing, dissemination, and presentation.

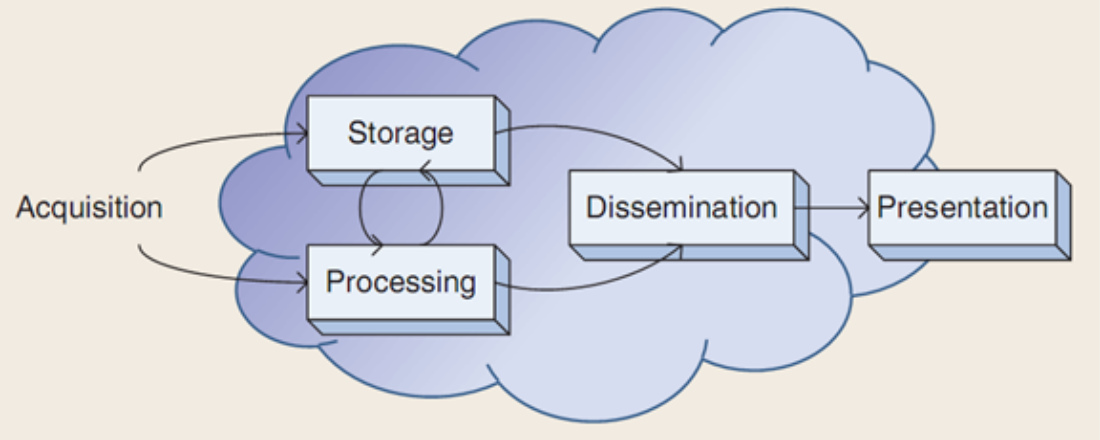


Figure 2.1: Media life circle of cloud-aware multimedia

- **Storage and sharing (storage and dissemination):** always-on is the advantage of cloud storage, which let users can access and share contents from any device at any time.
- **Authoring and Mashup (storage and processing):** Combines data, presentation or functionality from different multimedia resources for creating new services. The cloud-based multimedia authoring and mashup can offer many editing function and users do not need to consider the software maintenance or pre-installation.

- **Adaptation and Delivery (processing and dissemination):** In order to increased mobility, flexibility and scalability of cloud technology, adaptation and delivery takes a very important role. Hummaida [5] mentions adaptation is to increase or reduce resource allocation to a workload, also defined the process Cloud Systems Adaptation as a change to provider revenue, data centre power consumption, capacity or end-user experience where decision making resulted in a reconfiguration of compute, network or storage resources.

- **Media Rendering:** The challenge of rendering on mobile devices is the limited wireless bandwidth and the mobile phone has limited computing capability, memory size, and battery life. Since the cloud has the GPU comes with the capability to collaborative and interactive 3D experience, the rendering task would be shifted form the client to cloud server. The task now applies the Internet service that accepts 3D scene descriptions for video games, animated movies, simulators or visualized design.

2.1.3 Google Cloud

SADA system surveyed 200+ IT managers about their uses of public cloud services in 2015 [6]. The result shows that more than 84% are using public cloud infrastructure and google cloud platform is a very popular one among the cloud platforms. Figure 2.2 shows the interest of google cloud platform has grown rapidly in the last 5 fives form the Google trends explore [7], numbers represent the high peak of popularity. More and more manager would like to migrate since the public cloud comes with low cost, more secure, flexible and cost-effective.

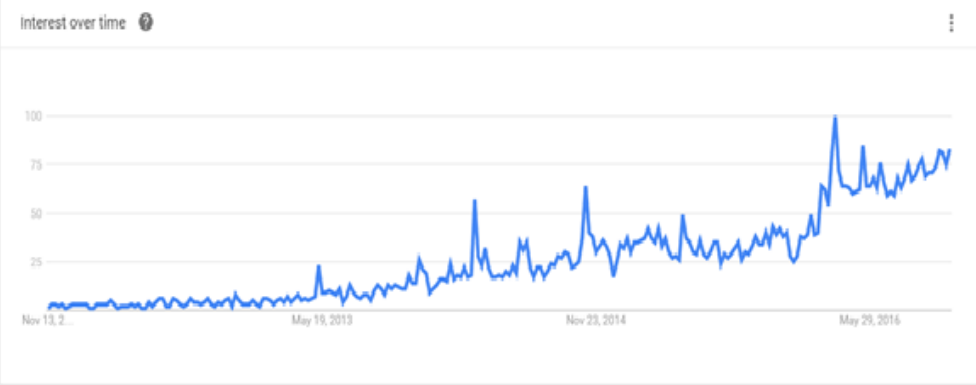


Figure 2.2: Google cloud platform interest over last five years

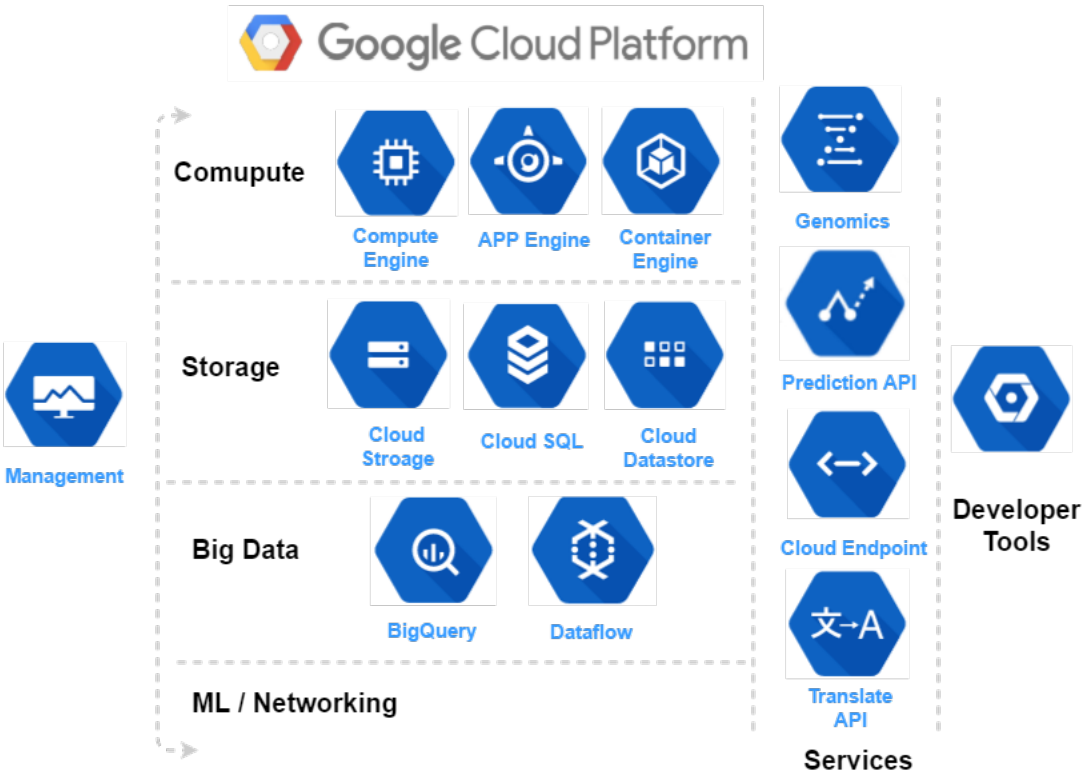


Figure 2.3: Google cloud platform diagram

Figure 2.3 shows the products and services of Google cloud platform, which offers options for storing and analyzing Data in Googles Cloud [8]. Flexible computing options come with virtual machines. App Engine is a platform-as-a-service provides SDK and tools for developing application in Java and Python. Google compute engine is an Infrastructure-as-a-Service provides virtual machine and the infrastructure for user to run applications. Here is an example shows the difference between these two computing options. For App Engine, some simple codes or clicks in Java or Python is enough to deploy a basic application. For compute engine, user needs to access to the virtual machine instances and web server such as Apache to get into the infrastructure to deploy and display the application. There are scalable and resilient storage and database service, moreover, Big Data offers the service of interactively analyzing massive datasets. BigQuery is a massively parallel query data processes that analyzes large data sets using SQL queries. Cloud dataflow offers execution for large-scale data processing for ETL, batch computation, and stream data processing. The difference between storage and BigData is huge. BigData lets users process a large

amount of data and analyze them quickly. The storage service lets user stores contents and hosts the database.

2.2 Smart Signage

A WiFi based interactive signage system leads to applications as for the scenarios of entertainment events, exhibitions, and assemblies with signages for real-time demonstration of media content. A recent success story is reported by [9], which is a draggable cyber-physical broadcast/multicast (B/M) media delivery system (or called CY system in the rest of the paper) for real-time media content retrieval by a large number of viewers. The CY system is based on a cyber-physical B/M protocol that synchronizes the content on the digital displays with the viewers smart phones. It supports interaction via one-to-many B/M such that multiple users can obtain content from a display with a dragging hand gesture. Users quality of experience (QoE) is characterized by the response time that is desired to be within 1 second.

The CY system relies on a quasi real-time approach where the WiFi router constantly broadcasts the media content in separate files, one after the other, in a round-robin fashion. In the meantime, it broadcasts a small tag corresponding to the file being displayed. Note that the WiFi transmits each content file not necessarily related to that being displayed in the signage. Once a smart phone actively running the corresponding application comes into the transmission range of the WiFi router, it obtains all the files and temporarily stores the files in its local hard driver. After a content capturing request, the smart phone checks the latest received tag and plays back the corresponding file. The tags are broadcast as a mean for synchronization between the smart phone and the content being displayed on the signage, which leads to an impression to the viewer that the media content being displayed in the signage is real-time obtained due to the dragging hand gesture. The response time performs well if the target content is ready in the buffer while acquiring; however, the proposed problem is that since it is a round-robin fashion, user has to wait one whole round if the package lost or just missing the content when get into the broadcast group. We improve the scheduling method to minimize the average delay time with considering the relationship between showing content and transmitting packet and focusing on the target to find the weight.

Chapter 3

System

In this chapter, the system will be detailed introduced with two different types of content provider. One is a local server with frame grabber to capture live video. Then upload contents to cloud server stores and manages as database. The other one is displayed as slideshow with perserved contents and broadcasting. A schdeling method is proposed to improve the performance. The mobile application let user to download contents.

3.1 Frame Grabber

Frame grabber (also called video grabbers or capture cards) is an electronic device that can capture video form the analog signal or digital video stream [10]. Frame grabber can grab video via the interfaces such as parallel digital, Camera Link, IEEE 1394 (FireWire) and GigE Vision. We can control the color space conversion, image scaling and re-sampling for application requirement. The technology can be wild used in live event production, healthcare, manufacturing and any other imaginable situation with capturing, streaming and recording.

3.1.1 Epiphan Frame Grabber Device

Epiphan DVI2USB 3.0 captures images or video from HDMI, SDI, DVI or VGA sources and connects to the computer with USB 3.0 port [11]. There is no external power supply since it powered through the USB port. It has the capture rate at 60 frames per second with resolutions up to 19201200 in YUV 4:2:0 color format. There is a LED indicator

Connecters	DVI-I (integrated, digital & analog) USB 3.0 B-Type connector
Input	VGA and DVI (HDMI video compatible)
Video Sampling	24 bits per pixel, 8:8:8 format 16 bits per pixel, 5:6:5 format 8 bits per pixel, 3:3:2, 3:2:3, 2:3:3 or 256-grey scale format 4 bits per pixel, 16-grey scale format
Supported video modes	Up to 19201200
Update rate	Up to 60 fps
HDMI Audio	16-bit PCM encoded audio at 32 kHz, 44.1 kHz, and 48 kHz sampling rates (Windows and Linux only)
OS Support	Windows 7, 8, 10 (i386, x64) Mac OS X 10.10 and up (i386, x86_64) Linux (x86, x86_64) DirectShow (Windows), Video4Linux (Linux), and Quicktime (Mac OS X) supported.

Table 3.1: Technical specifications of the Epiphan frame grabber device

shows the current status of the device. The device can also be connected by USB 2.0, but the frame grab rate will be reduced. Table 3.1 shows the device technical specification.

The Epiphan device provides unified API to access USB or network frame grabbers. and also supports different operating systems, (Windows, Mac OS X and Linux) however, frmgrab.dll is required at runtime while using Windows system. The local server (or local computer) can connect with frame grabber device by USB or network with the location parameter (local, net, serial number, id). Serial number is given to specify a local or a network frame grabber device. The system checks the local frame grabber first then goes to network. If it success detects the video, V2U_TURE will be returned. There are three main video mode (vm) parameters defined as screen width (pixels), screen height (pixels) and vertical refresh rate (mHz). If no signal is detected, all fields are set to be zero.

Figure 3.1 shows software design of the local server. Once the local server starts running, the program splits into two subroutines. First subroutine connects to the frame grabber to receive data. It first detects whether there is an Epiphan device connected, and will reply frame grabber not found if cannot find one. We can also check whether the frame grabber is ready or not by the LED indicator which shows the status on the device. If the program detects the frame grabber but does not find a connected video source, it returns V2U_FALSE and the message no signal detected. After pass the video mode (vm) detection, frames will be captured and stored in the local server with the vm parameter (width, height and vfreq).

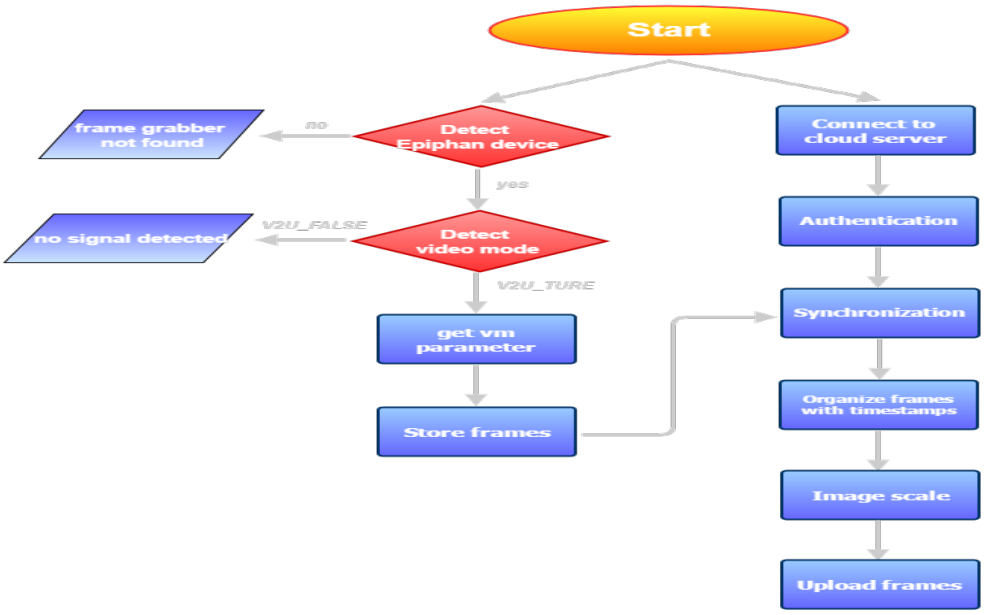


Figure 3.1: Software design of the local server

The other one subroutine connects to the cloud server. It is required to be authenticated with the cloud server to get the permission to transfer or read/write contents to the bucket. The program keeps receiving timestamps from the cloud server uniform clock for synchronization. The system synchronization will be detailed elaborated in the next section. Here is one thing needed to be mentioned, the stored frames are also used in this process as showed the store frames and synchronization blocks are connected. Which means the system does not synchronize with the cloud server only one time at the beginning, it keeps receiving timestamps. Once there is a new frame, it will be organized with the updating time. The images are scaled for better performance and then upload to the cloud server.

3.2 Scheduling for broadcast

With a cyber-physical broadcast/multicast (B/M) media system as the other content provider, we can now provide two different kinds of content source service - snapshots from live video or preserved slideshow images. In this section, we study the CY system and propose an improved scheduling method base on the system.

The CY system relies on a quasi real-time approach where the WiFi router constantly broadcasts the media content in separate files, one after the other, in a round-robin fashion. In the meantime, it broadcasts a small tag corresponding to the file being displayed. Note that the WiFi transmits each content file not necessarily related to that being displayed in the signage. Once a smart phone actively running the corresponding application comes into the transmission range of the WiFi router, it obtains all the files and temporarily stores the files in its local hard driver. After a content capturing request, the smart phone checks the latest received tag and plays back the corresponding file. The tags are broadcast as a mean for synchronization between the smart phone and the content being displayed on the signage, which leads to an impression to the viewer that the media content being displayed in the signage is real-time obtained due to the dragging hand gesture.

The CY system can successfully yield a response time shorter than 1 second at the assumption that the dragging event happens after the required content file is ready for being display in his/her smart phone; otherwise the viewer has to wait for receiving the necessary file and subject to longer response time. The key problem leading to the above mentioned shortfall is on the lack of coordination between the scheduling of transmitted content files by the WiFi router and the displaying contents. We investigate the coordination based on a novel scheduling scheme for the file transmission such that the response time at the viewer can be minimized.

The average response time has been studied as the function of T_g (gesture detection time), S_f (the size of the draggle file), B (bit rate of the channel) and T_d (decision time). However, the conventional transmitting method is in round-robin fashion, which the router launching the data packet without considering what is shown on the display, may not be effective. In a round-robin fashion, a user smartphone may have massive delay time while receiving contents which are not available to drag. Therefore, enhance the performance of the signage system by adjusting schedule of content is studied in this section.

Without loss of generality, we consider a single device with a set of K files to display, denoted as C_k for $k = 1K$, each with a display time as T_k for $k = 1K$. Let D be the time for completing the delivery of a single content file to the mobile device and is assumed equal in the following analysis. Table 3.2 shows the architecture of contents in m devices.

Parameter r_j is the dynamic weight of transmission for each content. For example, if $r_1 = 1$ and $r_2 = 1$ while T_1, that means the router transmits C_1once first then C_2 once and repeats the action until T_2. To achieve the smallest delay time, focus on the target to find the weight. It is obvious that $r_j = 1$ and others weight 0 can make it. Which means the router only transmits C_j during T_j. However, r_j and r_{j+1} should both be concerned not only because after transmit C_j a few times the difference of delay time will be very

r_k	C_{1k}	C_{2k}	C_{3k}		C_{mk}
$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$
r_3	C_{13}	C_{23}	C_{33}	$\dots$	C_{m3}
r_2	C_2	C_{22}	C_{32}		C_{m2}
r_1	C_1	C_{21}	C_{31}		C_{m1}
	Device 1	Device 2	Device 3		Device m

Table 3.2: Devices and contents architecture

small which can be ignorant but also while sending C_{j+1} that user can save the waiting time while T_{j+1}. Moreover, the router sends C_j at the begin of T_{j+1} for the situation that the user joins the network for dragging C_j at the end of T_j. Figure. 3.2 shows an example of the scheduling method.

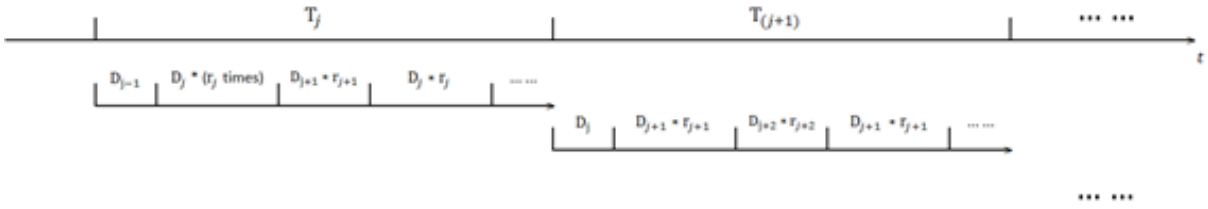


Figure 3.2: An example of the scheduling method with considering r_j and r_{j+1} and send the previous C_{j-1} once at the beginning of T_j.

Since the parameter r is 0 except r_j and r_{j+1}, when the user gets into the B/M group and is interested in C_j, the time interval $r_{j-1}$and T_j lead to further discussions. First, if the user joins the system in T_j, C_j is showing on the monitor for user to drag. To calculate n_{r_j} which is the number of times that C_j can be transmitted in T_j, $n_a = r_j * floor(\frac{n-1}{r_j+r_{j+1}})$, $n_b = r_{j+1} * floor(\frac{n-1}{r_j+r_{j+1}})$ and $z = mod(\frac{n-1}{r_1+r_2})$.

1. Case 1: if $z <= r_j, (floor(\frac{z}{r_j}))$

$$n_{r_j} = n_a + z \text{and} n_{r_{j+1}} = n_b \tag{3.1a}$$

2. Case 2: if $z > r_j$

$$n_{r_j} = n_a + r_j \text{and} n_{r_{j+1}} = n_b + (z - r_j) \tag{3.1b}$$

The number of successful trial is incremented when target is transmitted, with the parameter x. The delay time of the target each single round can be expressed as:

$$s = floor(\frac{x-1}{r_j + r_{j+1}}) * (r_j + r_{j+1}) * T + mod((x-1), r_j) * D \tag{3.2}$$

C_j, j=	1	1	1	2	2	1	1	1	2	2
x	1	2	3			4	5	6		
$floor(\frac{x-1}{r_j})$	0	0	0			1	1	1		
mod(x-1,r_j)	0	1	2			0	1	2		
Dealy time	0	D	D			5D	6D	7D		

(a) When $r_1 = 3$ and $r_2 = 2$; T = 10D;

$Q_{d(j)}$, j=	1	1	1	1	2	2	1	1	1	1	2	2	1	1	1	1	2	2	1	1
x	1	2	3	4			5	6	7	8			9	10	11	12			13	14
$floor(\frac{x-1}{r_j})$	0	0	0	0			1	1	1	1			2	2	2	2			3	3
mod(x-1,r_j)	0	1	2	3			0	1	2	3			0	1	2	3			0	1
Delay time	0	D	2D	3D			6D	7D	8D	9D			12D	13D	14D	15D			18D	19D

(b) When $r_1 = 4$ and $r_2 = 2$; T = 20D;

Table 3.3: An example for calculating delay time each single round

Table 3.3 shows two examples with different weight for calculating dealy time in one single round. There is a parameter y counting the number of big cycles when the monitors display every item. For every two cycles router is expected to transmit one item for n times, assume 1 x n. Therefore, the different rounds of delay time and probability can be expressed as:

1. Case 1: $x <= n_{r_j}$

$$floor(\frac{x-1}{r_j}) * (r_j + r_{j+1}) * D + mod((x-1), r_j) * D + D$$

for $y = 0$ and $x > 1$

$$floor(\frac{x-1}{r_j}) * (r_j + r_{j+1}) * D + mod((x-1), r_j) * D$$

for $y > 0$ and $x <= 1$ (3.3a)

2. Case 2: $x = n_{r_j} + 1$

$$n * D \tag{3.3b}$$

3. Case 3: $x > n_{r_j} + 1$

   let $x_2 = x - n_{r_j}$

$$\begin{aligned} &floorfloor(\frac{x_2 - 1}{r_{j+1}}) * (r_j + r_{j+1}) * D \\ &\quad + mod((x_2 - 1), r_{j+1}) * D + (k + 1) * n * D + D + r_j * D \end{aligned} \tag{3.3c}$$

PROBABILITY $= p * (1 - p)^{x-1}$

Hence, the average delay time E is:

$$E = \sum(DELAYTIME * PROBABILITY) \tag{3.4}$$

Eq.(3.4) shows that E is the function of p, n, k. Consider the different joined times of user in T_j. t_{join} is the instance time when the user joins the broadcast group and T_{join} is the time duration of t_{join} minus the beginning of T_j. Since T can be separated into n parts, T_{join} can be presented as n_{drag} which means the number of times of content witch router has transmitted and $n_{drag} < n$. n_g is the number of times that content can be transmitted in T_j but the user would miss since the joined time maybe not at the beginning of T_j.

$g = floor(\frac{n_{drag}-1}{r_j+r_{j+1}}) * r_j; h = floor(\frac{n_{drag}-1}{r_j+r_{j+1}}) * r_{j+1}; z_g = mod(\frac{n_{drag}-1}{r_j+r_{j+1}})$

1. Case 1:

$$(z_g r_j), n_g = g + z_g and n_h = h \tag{3.5a}$$

2. Case 2:

$$(z_g > r_j), n_g = g + r_j \text{ and } n_h = h + (z - r_j) \tag{3.5b}$$

The number of successful trial can be shown as $x^{'} = xg$. Therefore, the probability is $p * (1 - p)^{x^{'}-1}$ and the delay time is Eq.(3.3c) (the previous one) minus $(n_{drag} - 1) * D$.

$$DELAYTIME = \begin{cases} floor(\dfrac{x-1}{r_j}) * (r_j + r_{j+1}) * D + mod((x-1), r_j) * D \\ \qquad\qquad -(n_{drag} - 1) * D \qquad \text{for } x \leq n_{r_j} \\ n * D - (n_{drag} - 1) * D \qquad \text{for } x = n_{r_j} + 1 \\ floor(\dfrac{x_2 - 1}{r_{j+1}}) * (r_j + r_{j+1}) * D + mod((x_2 - 1), r_{j+1}) * D \\ \qquad +(k+1) * n * D + D + r_j * D - (n_{drag} - 1) * D \\ \qquad\qquad \text{for } x > n_{r_j} + 1 \end{cases} \tag{3.6}$$

Moreover, if the user joins the system in T_{j-1} which means C_j is not showing on the monitor for user to drag. The user can receive the content but not available to drag until T_j. Assume delay time increments after dragging, there will be no delay time if the user obtains C_j in T_{j-1}. Therefore, the parameter w, which is the number of times of C_j has been sent in T_{j-1} for estimating probability. Using Eq.(3.1) to find n_{r_j} in T_{j-1}. ($r_{j-1} =$ and $r_j =$) and Eq.(3.5) to find n_h. Hence, $w = n_{r_j} - n_h$ and the probability is $p * (1-p)^{(w+x-1)}$. Delay time is the same as Eq.(3.3c) when n_{drag} is 0.

3.3 Server

This section elaborates the cloud server and iBeacon. Contents stored in the Google Cloud Storage. The iBeacon provides the locational signal to let mobile device can designate specific content within multiple displays.

3.3.1 Google Cloud Storage

Google cloud storage (GCS) is the product of cloud platform in google infrastructure with high security and fast retrieval from data center around the global. It is highly structured with project based. Major divisions or containers in the project are called buckets, and objects are in the buckets. A bucket is similar to a directory and an object is similar to a

file. Buckets provide access level control (ACLs) and project specific, in other words, the buckets cant be shared within different projects. It is similar that objects cant be shared between buckets. There are many ways to interact with GCS. The first one is the web-based interface GCS manager or a simple web UI with basic communication and management of the buckets and objects called Google cloud developer console. HTTP standard methods support transfer commands such as get, put and post. gsutil is the tool comes with google cloud SDK to access GCS form the command line, which gives users a more advanced and powerful way of managing tasks. Programmatically there are REST (representational state transfer) APIs, users can easily use them with different programming language.

The upload is strongly consistent, which means once the file is uploaded into a bucket in cloud storage there will be a success response, and the file is immediately ready for download. GCS supports browser authenticated (OAuth 2.0) file transfer. The permission is required for users to be able to download the objects, or users can get the objects through the provided anonymous access. The transfer comes with the ability to pause and resume in case of communication failure or other problem, the uploading or downloading will resume automatically. It can resume exactly where it left off rather than start over.

Before uploading data to cloud server, we have to supply credentials and a project ID to get authorized and start using Google's APIs. For authenticating API requests, it can be Google Cloud SDK (locally), OAuth2 access token (not refreshable) or generate a JSON service account key. While using the Storage blob with ACLs, we can start the service to upload the file to the bucket. The cloud Datastore is a storage system generated by App Engine. It hosts dynamic application data while GCS hosts exist file uploading. The GCS can also be used with App Engine while using the REST API or App Engine API.

3.3.2 Synchronization

In order to let different devices have the same value of time, we predefine a reference clock based on Greenwich Time in the server for time uniform to achieve the synchronization. Figure 3.3 shows the illustration of the system synchronization rely on the reference clock. The local server updates the time and gets timestamps once connecting to the server. All the uploading contents are arranged with timestamps, which ensures the time accuracy and data organization. When a user wants to download the content with a Drag order, the system will send a request and calculate the current time to acquire the target content.

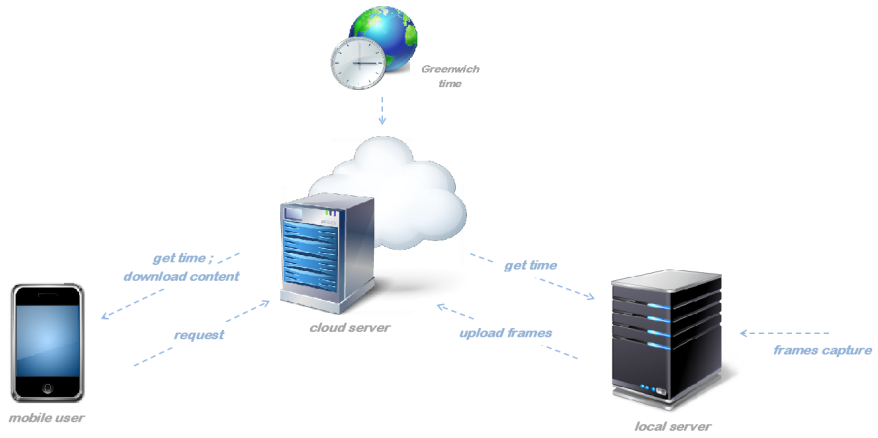


Figure 3.3: Illustration of synchronization

3.3.3 iBeacon

Now the system has the capability to offer one-tomany service with the structure mentioned in the previous sections. In advance, iBeacon technology helps us achieve the many-tomany service. Ning [12] introduced an iBeacon-based location-aware advertising system. The technology is based on the standard Bluetooth Low Energy (BLE) protocol, which supports smart phone to identify their position and location in indoor environment. The distance level for the signal range of the iBeacon are Immediate (0 - 0.5 meters), Near (0.5 3 meters) and Far (3 30 meters). iBeacon is a transmit-only device which transmits data package to clients. An iBeacon data package (31 bytes) consists Prefix (9 bytes), UUID (16 bytes), Major Number (2 bytes), Minor Number (2 bytes) and TX power (1 bytes). Universally unique identifier (UUID) identifies the beacon. Major Number identifies a subset of beacons within a large group and Minor Number identifies a specific beacon. TX power measures the signal strength at one meter from the device. The distance between beacon and mobile device can be calculated with the signal strength. An Estimote iBeacon device can be detected with UUID and radar range information shown as Figure 3.4.

The iBeacon technology comes with the advantage of low power requirement, small size and low cost for in-door positioning. Moreover, it has compatibility with a large installed base of mobile phones, tablets and computers. Mobile application measures the distance with the iBeacon devices, it other word, the user in the iBeacon signal region could be

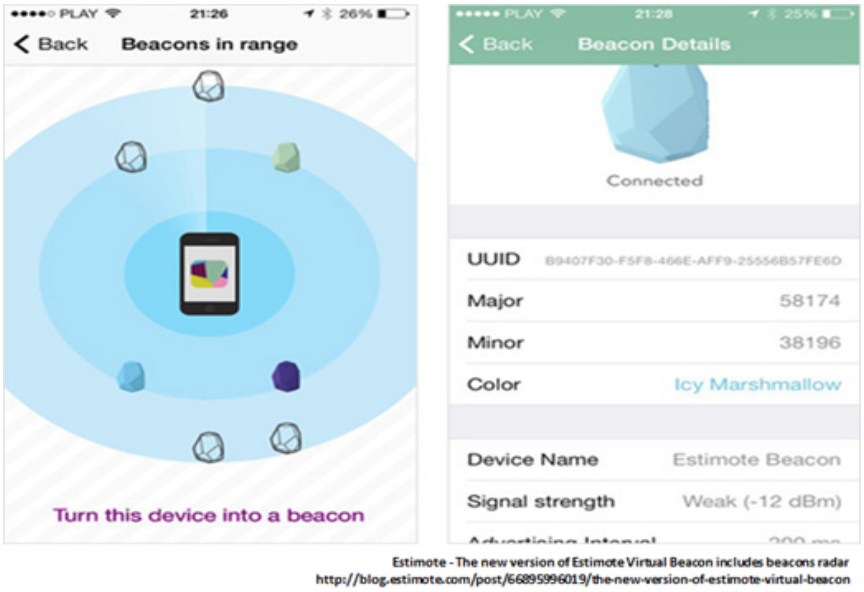


Figure 3.4: Illustration of Estimate iBeacon radar range and user interface details

precisely positioned. We can set more than one display with video source, and users can specify the target display with the nearest one. The next section we are going to introduce a broadcast schedule method as one additional content provider for multiple displays.

3.4 Mobile Application

The application is developed in Android for mobile device. Figure 3.5 shows the software design of mobile application. User need to login to the server to start running the application. The application connets to the cloud server for live video frame and also gets into the brodcast group for the slide show content. The program splits into two subroutines, one for iBeacon and the other one for drag detection. The first subroutine searches iBeacon and positional signal. The scanning application reads the iBeacon data package for UUID, major number and minor number to get information about the beacon. The other one subroutine detects the dragging event. Since the application gets the iBeacon information, it can tell the cloud server which display is the target. It also updates time with the uniform clock once connecting to the server. Therefore, with the display information and the updating timestamps, the request will be send to server to search and download the target content.

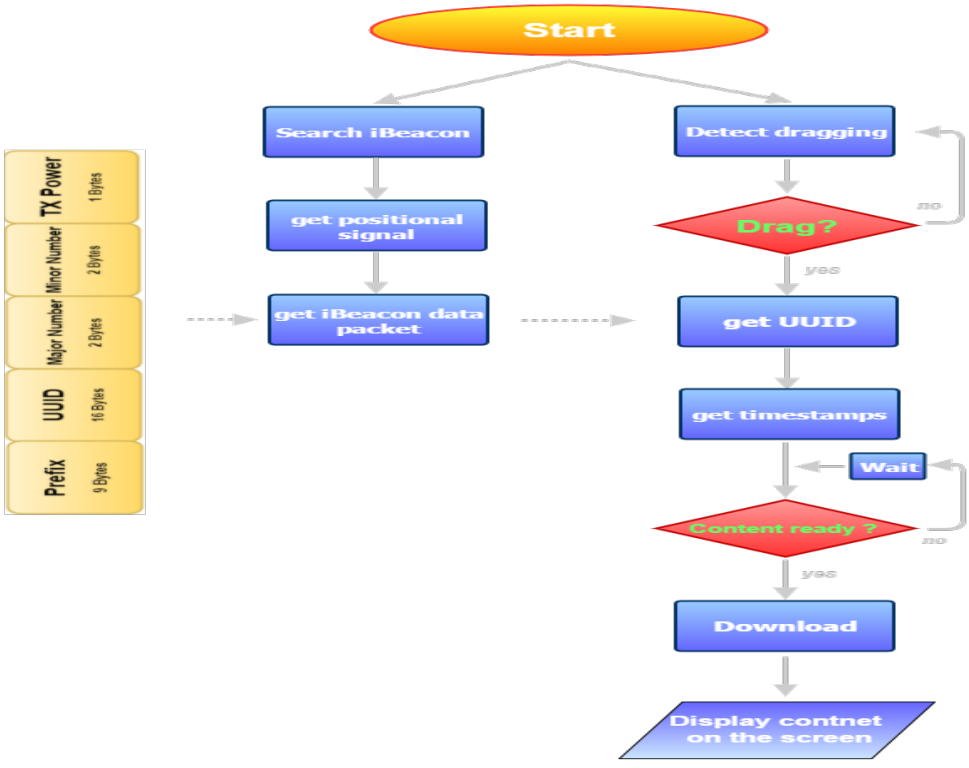


Figure 3.5: Software design of the mobile application

3.4.1 Detection of Dragging Hand Gesture

Instead of setting a traditional button for users to do the download order, we decide to use one simple dragging gesture as shown in Figure 3.6. The mobile device uses sensors such as gyroscope (angular speed around each axis) and accelerometer (acceleration in three axes) embedded in the smartphone to track gesture movement. Although it is essential to have gesture spotting and gesture segmentation in traditional hand gesture recognition [13], only one simple hand movement is required in the system which makes it easier [9]. The dragging event can be detected by checking the range of the values of tri-axis accelerometer when total acceleration, $A = \sqrt{(A_X)^2 + (A_Y)^2 + (A_Z)^2}$ (where A_X, A_Y and A_Z are the acceleration values of X, Y and Z-axis of the accelerometer respectively). Since this hand gesture is so simple, the advantage of the approach is the detection accuracy and a very short response time, which also enhances user experience.

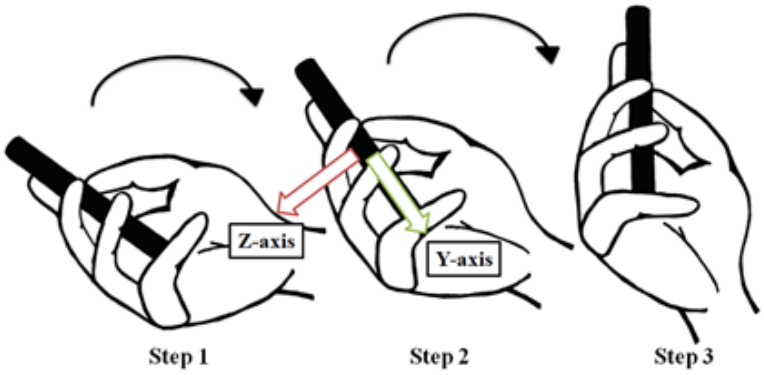


Figure 3.6: Illustration of the dragging hand gesture

Chapter 4

Implementation and Experiment

In this chapter, we set up and demonstrate the system.

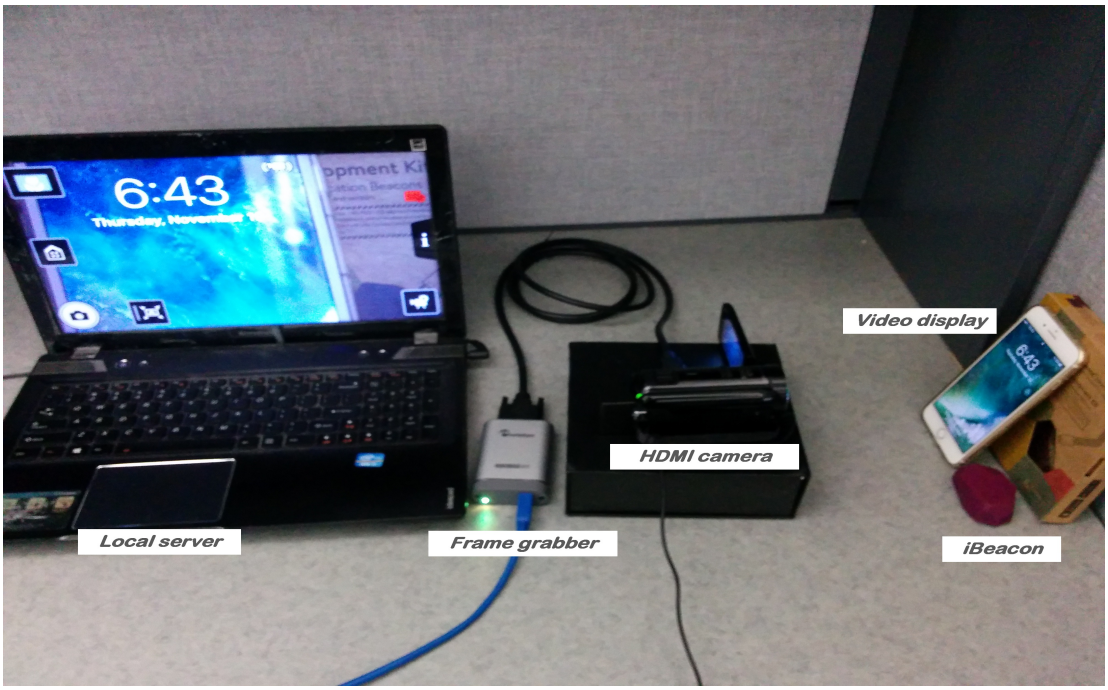


Figure 4.1: Implementation of the frame grabber and local server

Figure 4.1 shows the impalement of the frame grabber and local server. The camera takes video form the smartphone screen as the video source. We put a clock as video source

to show it is dynamic and no delay. There is an iBeacon set beside the video display for location aware of multiple displays to achieve many-to-many service. The Epiphan device captures frames from the HDMI output camera, the flashing LED indicates the grabber is working properly. A computer is set as the local server and connects to the frame grabber via USB 3.0. The local server works on receiving frames and uploading contents to the cloud server.

4.1 The Demonstration of System

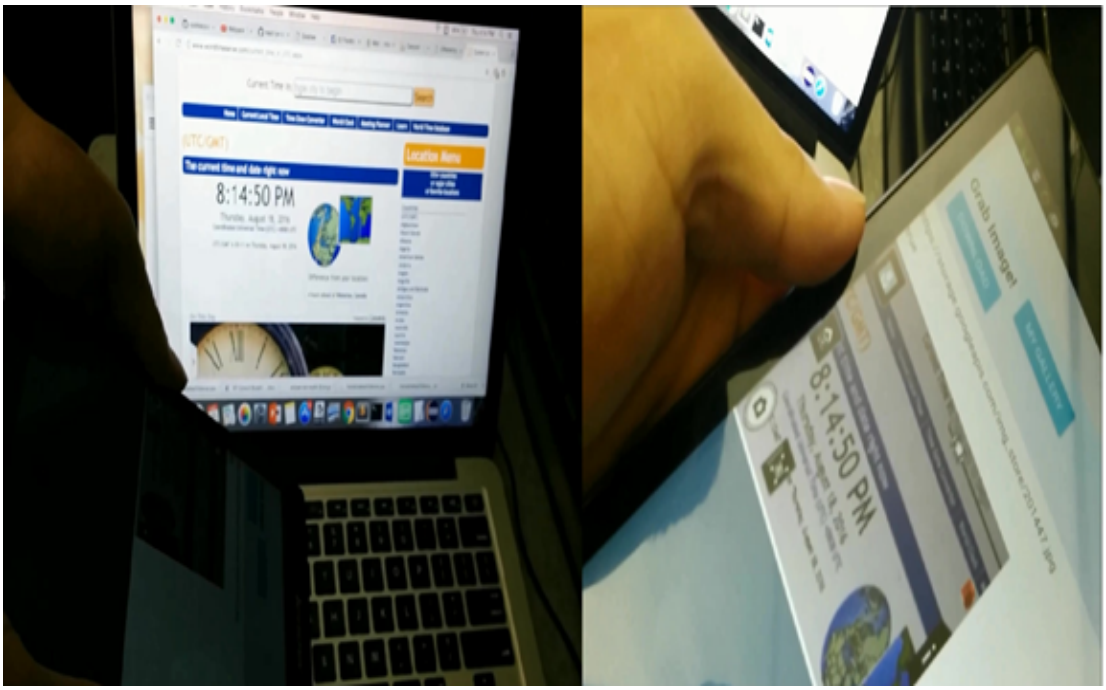


Figure 4.2: Demonstration and the result

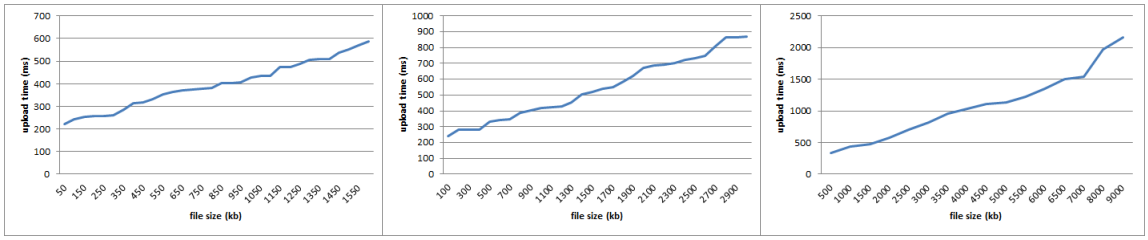
The demonstration result is showed as Figure 4.2. A camera shooting on a clock is set as the video source for the synchronization check, if we can get the same "second" displayed on the monitor once we do the "drag" and it is showed on the smartphone screen in a very short time, then we can say the image is captured in real time. The result shows that we dragged at the second of '50' and the image matched.

4.2 System Performance

The performance of the system is characterized by the following factor: T_c (frame capture time), T_u (frame upload time), T_r (sever response time) and T_{dl} (frame download time). T_r is the time for the server to response the request after the dragging event from user, which means how long the customer has to wait for the download to start. T_r is corresponding with $T_c + T_u$.

4.2.1 Uploading time cost

In case to let user gets the idea image in a very short time, we have to make sure the server response time is shorter than 1 second at the assumption that the dragging event happens and the system can provide the target file immediately, otherwise the user has to wait for a long time to receive the image or gets the image which is not the target. Since the capture rate for the frame grabber device is up to 60 fps, the upload time takes a key part of the system performance. The resolution of the original frame is 1920*1080 with size 6000 kb, which costs too much time to upload. Therefore, we need to resize the frame before the upload step. For better performance, we examine different file size against upload time, Figure 4.3 (a)(b)(c) show the relationship between the image size and the time cost.



(a) (b) (c)

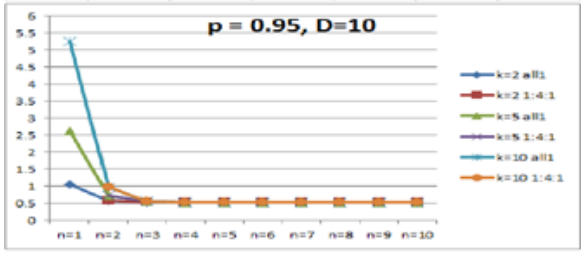
Figure 4.3: File size (kb) against upload cost time (ms). (a) with interval 50 kb (50 1550) (b) with interval 100 kb (100 2990) (c) with interval 500 kb (500 9000)

As the result of the Figure 4.3, we can find that it will cost more than 1 second if the file is larger than 3500 kb. For more ideal case, it will be better to let the cost time less than 0.5 second, which means the file size is less than 1300 kb.

4.2.2 Broadcast delay

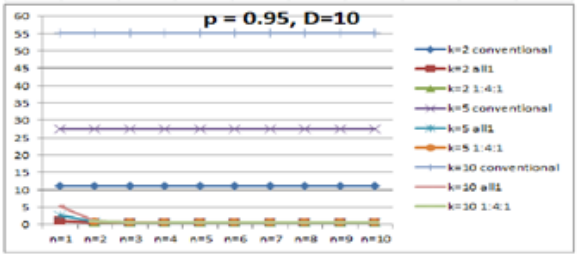
According to Eq.(3.6), Figure 4.4 shows the plot of n against average delay time when k = 2,5 and 10 and under the condition of p = 0.9 or 0.95, D=1 or 10. Comparing when the scheduling method is the (1) conventional one, (2) $r_j = 1$ and others weight 0 and (3) $r = 0$ except r_j and r_{j+1} but sent r_{j-1} once first $r_j : r_j = 4 : 1$ from the simulation result). If there are more contents in one device which means k is large, it costs more time to run a big cycle. The plot shows there is no relation between n and the conventional scheduling method because router only sends data regardless of what is being shown on the monitor. But for the methods considering parameter r, the way router sends data depends on the display time, if the display time is larger, the number of times of D is increases hence n will be larger. The plot shows that if n increases, the result of average delay time will be closer to the ideal case (2).

	k=2		k=5		k=10	
	all1	1:4:1	all1	1:4:1	all1	1:4:1
n=1	1.0526		2.6316		5.2632	
n=2	0.5764	0.5764	0.7268	0.7268	0.9774	0.9774
n=3	0.5301	0.5301	0.5413	0.5413	0.5601	0.5601
n=4	0.5266	0.5266	0.5273	0.5273	0.5286	0.5286
n=5	0.5263	0.5263	0.5264	0.5264	0.5265	0.5265
n=6	0.5263	0.5264	0.5263	0.5265	0.5263	0.5266
n=7	0.5263	0.5264	0.5263	0.5264	0.5263	0.5264
n=8	0.5263	0.5264	0.5263	0.5264	0.5263	0.5264
n=9	0.5263	0.5264	0.5263	0.5264	0.5263	0.5264
n=10	0.5263	0.5264	0.5263	0.5264	0.5263	0.5264

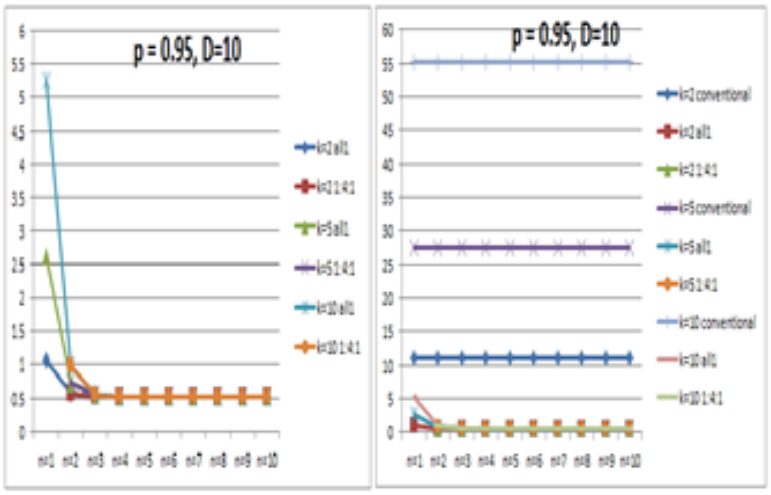


(a)

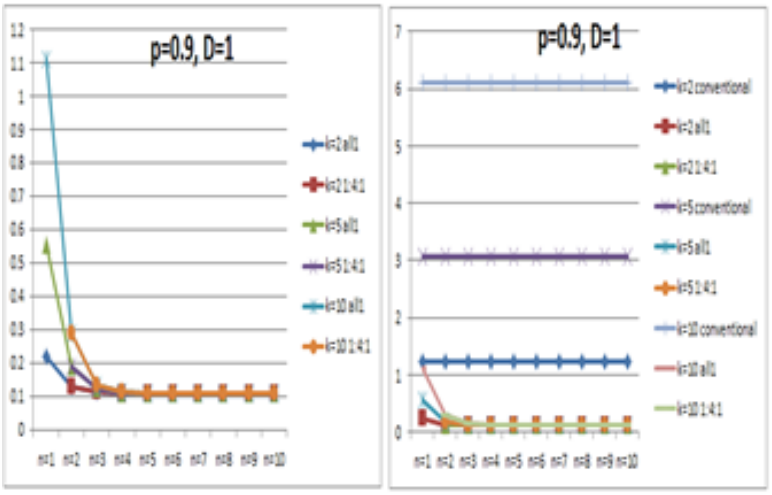
	k=2			k=5			k=10		
	conventional	all1	1:4:1	conventional	all1	1:4:1	conventional	all1	1:4:1
n=1	11.0526	1.0526		27.6316	2.6316		55.2632	5.2632	
n=2	11.0526	0.5764	0.5764	27.6316	0.7268	0.7268	55.2632	0.9774	0.9774
n=3	11.0526	0.5301	0.5301	27.6316	0.5413	0.5413	55.2632	0.5601	0.5601
n=4	11.0526	0.5266	0.5266	27.6316	0.5273	0.5273	55.2632	0.5286	0.5286
n=5	11.0526	0.5263	0.5263	27.6316	0.5264	0.5264	55.2632	0.5265	0.5265
n=6	11.0526	0.5263	0.5264	27.6316	0.5263	0.5265	55.2632	0.5263	0.5266
n=7	11.0526	0.5263	0.5264	27.6316	0.5263	0.5264	55.2632	0.5263	0.5264
n=8	11.0526	0.5263	0.5264	27.6316	0.5263	0.5264	55.2632	0.5263	0.5264
n=9	11.0526	0.5263	0.5264	27.6316	0.5263	0.5264	55.2632	0.5263	0.5264
n=10	11.0526	0.5263	0.5264	27.6316	0.5263	0.5264	55.2632	0.5263	0.5264



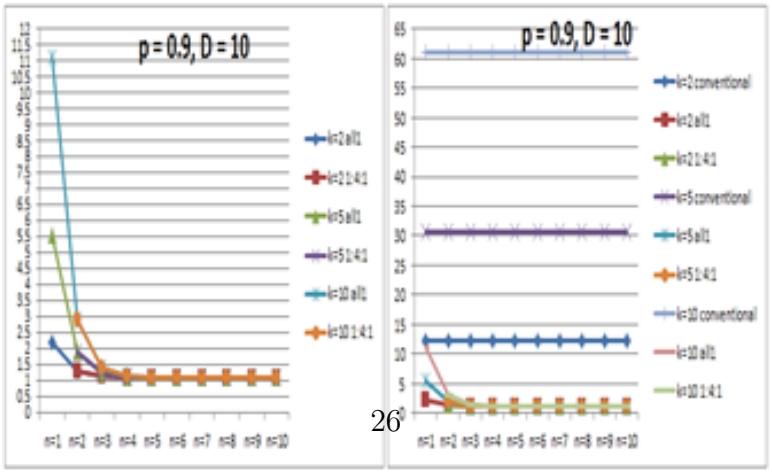
(b)



(c)



(d)



(e)

Figure 4.4: Numerical analysis of the average delay time against n (Comparing scheduling methods (1)(2)(3))

4.2.3 Delay

The best scenario for the system is that the user can get the content immediately after the drag, however, delay cannot be ignored in real life. T_r is the interaction delay perceive by the user which has a significant impact of the system performance. If we have to let user feel almost no delay, the system response time should be minimized. The factors for the delay are network delay, application process, cloud server process and waiting time for the target content ready.

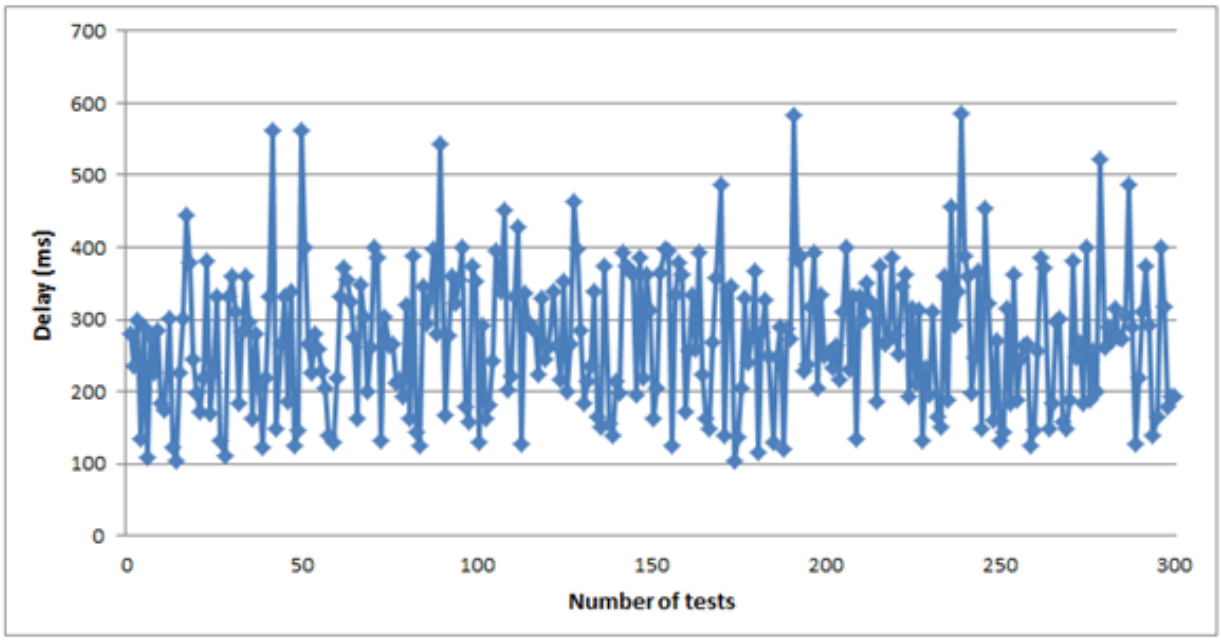


Figure 4.5: System response time test

Figure 4.5 shows the delay test for the system response. There are 300 record data for the time cost to start download after a drag detected. The result shows it normally takes 100 ms $\sim$ 400 ms for the system response. And we can see all data points are above 100ms which means it takes at least 100 ms delay for the network delay or the synchronized process. The waiting time for the target content ready depends on the upload rate and the timing user makes the order. It would be almost no waiting if the user drags when the content almost complete upload, otherwise, the system response cost more time to finish the upload.

4.3 Summary

In this chapter, we show the system deployment and demonstration. The system performance is evaluated by the delay and frames upload time. The time cost to start download after the drag event detected is measured to determine the delay, and frames are scaled to adjust upload time. The system response time cost around $200 \sim 400$ ms. Although the upload and download time influenced by the network speed condition, the upload time is also adjusted to be around $250 \sim 400$ ms. With the above time cost controlled within 1 sec, the system can offer the proper contents to user.

Chapter 5

Summary

Cloud technology is very popular nowadays and is wild use in the advertisement system. Instead store data in the cloud server in previous, contents are capture form the live event and upload to the server in real time. The thesis presents a system with the ability to interact with multiple sources or displays with a simple drag gesture in LIVE.

5.1 Thesis Summary

A real time draggable frame grabber media system is proposed in this thesis. The system consist frame grabber, cloud server, iBeacon and mobile application. there are two types of content provider, snapshots from live video or preserved slideshow images. The frame grabber captures frames as one contents provider, frames are rescaled and then upload to the cloud server. We choose google cloud storage as cloud server, and contents are saved in the cloud server as database. There is a reference clock in the server for synchronization, and contents are organized by the timestamps. The iBeacon device provides position signal for mobile device receiving the location information. In this way the mobile application can tell the server which specific display is the target. A broadcast media system has been studied as the ohter content provider and a scheduling method has been proposed to enhance the performance. The mobile application does the iBeacon and the drag event detection. A request will be sent to server for the specific content (with timestamps) in the specific display (iBeacon information) after a drag detected.

5.2 Future Work

Since the uploaded frames are saved in the server as database, it provides the ability for further management. Contents can be advanced used in Google Cloud Platform with developed functions. A simple idea is that data still saved in the Cloud Storage but can be transfer through Dataflow and further use as in BigQuery for analyze or Bigtable for applications and reports. With APP Engine and its API, it is friendly to build mobile application on GCP. The future work is first to achieve higher transfer rate. The other is to manage the frames to the server such as make it possible to let user download gif picture or video stream (moreover, specific interval) in real time or more live interaction. Although there are many applications can receive live video stream, but it does cost data usage and smartphone battery consumption. A well-developed cloud platform provides a possible way to manage and organize data in the cloud server to enhance the user experience.

5.3 Conclusion

In this thesis, a real time draggable frame grabber media system is proposed. The user can simultaneously acquire the snapshot from a live video playing on a display by simply using an intuitive drag hand gesture with their mobile device. With the up to 60 fps capture rate and the image size adjustment for better uploading cost time, the response time is less than 1 second after users drag hand movement. As what shown in the experiment, we can get the same second content after the drag immediately. We also improve the scheduling method to minimize the average delay time of broadcasting slideshow images with considering the relationship between showing content and transmitting packet and focusing on the target to find the weight.

References

[1] H. Moustafa, N. Marchal, and S. Zeadally. Mobile Multimedia Applications: Delivery Technologies. *IT Professional*, 14(5):12–21, Sept 2012.

[2] LY. Astri. A Study Literature of Critical Success Factors of Cloud Computing in Organizations. *Procedia Computer Science*, 59:188–194, 2015.

[3] P. Mell and T. Grance. The NIST definition of cloud computing. *Communications of the ACM*, 53(6):50, 2010.

[4] W. Zhu, C. Luo, J. Wang, and S. Li. Multimedia cloud computing. *IEEE Signal Processing Magazine*, 28(3):59–69, 2011.

[5] AR. Hummaida, NW. Paton, and R. Sakellariou. Adaptation in cloud resource configuration: a survey. *Journal of Cloud Computing*, 5(1):1–16, 2016.

[6] SADA systems. Public Cloud More Secure Than Corporate Data Centers. https://sadasystems.com/2016-public-cloud-survey-Infographic.pdf, 2016.

[7] GoogleTrends. https://www.google.com/trends/explore?q=Google%20Cloud%20Platform. [Norvember-2016].

[8] Google. Google Cloud Platform. https://cloud.google.com/. [Norvember-2016].

[9] J. She, J. Crowcroft, H. Fu, and PH. Ho. Smart signage: An interactive signage system with multiple displays. In *Green Computing and Communications (GreenCom), 2013 IEEE and Internet of Things (iThings/CPSCom), IEEE International Conference on and IEEE Cyber, Physical and Social Computing*, pages 737–742. IEEE, 2013.

[10] Wikipedia. Frame grabber — wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/Frame_grabber. [Online; accessed 13-Norvember-2016].

[11] Epiphan. SDK release notes. https://www.epiphan.com/products/dvi2usb-3-0/.

[12] J. Ning. An iBeacon-Based Location-Aware Advertising System. University of Waterloo, 2016.

[13] S. Zhou R. Xu and W. Li. MEMS accelerometer based nonspecific-user hand gesture recognition. *IEEE sensors journal*, 12(5):1166–1173, 2012.