เฟรมเรทที่ปรับเปลี่ยนได้และการเลือกเฟรมอัตโนมัติภายในบัฟเฟอร์บนเครื่องแม่ข่ายสำหรับผสานวีดีโอ

Adaptive Frame Rate and Automatic Frame Selection in Buffer Based On Video Mixer Server

กฤษฎิ์ธนิก ศรีธนสาร สุวิพล สิทธิชีวภาค
สาขาวิชาวิศวกรรมโทรคมนาคม คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

บทคัดย่อ

วีดีโอคอนเฟอร์เรนซ์ที่ใช้งานอยู่ในปัจจุบันนี้ผ่านระบบเครือข่ายที่หลากหลายและซับซ้อน ย่อมต้องมีการปรับปรุงให้เหมาะสมกันกับสภาพแวดล้อมของระบบเครือข่าย รวมทั้งคุณภาพของการบริการ (QoS) บทความนี้นำเสนอวิธีการในการส่งสัญญาณภาพวีดีโอของวีดีโอเซอร์เวอร์ ที่ได้รับสตรีมมิ่งวีดีโอจากเครื่องลูกข่ายแหล่งต่าง ๆ ด้วยวิธีการรวบรวมภาพและประเมินความสามารถของระบบเครือข่ายจากแหล่งที่มา เพื่อตอบสนองสัญญาณภาพเฟรมต่อวินาทีอย่างเหมาะสมตามเวลาจริงเมื่อมีดีเลย์เกิดขึ้นในระบบเครือข่ายที่แตกต่างและความสามารถในการส่งผ่านข้อมูลที่ผันแปรได้ตลอดเวลาตามสภาพการใช้งานของเครือข่ายนั้น ๆ ในตอนท้ายบทความนี้ ได้กล่าวถึงวิธีการเลือกเฟรมในบัฟเฟอร์ และผลการทดลองที่ได้จากเลือกเฟรมเรทที่เหมาะสม

คำสำคัญ : วีดีโอคอนเฟอร์เรนซ์, การปรับแต่งเฟรมเรท, ผสานเฟรมภาพ, ไคลเอนท์-เซอร์เวอร์, JMF

Abstract

For the fact that many modern video conference systems could be used on various complex network systems, it is necessary to make improvement of the proper network adapters to be suitable for the environment of the networks and the quality of service (QoS). This paper would present the practical algorithm to send the video signal of the video server receiving streaming video signals from any client. The method of frame mixing and network ability approximation would be applied on the study. It is to suitably continuously respond the video frame per second in real time whenever having any delay in any network and variable data transfer in any network condition. Finally, the suitable frame selection algorithm in buffer and its result would be mentioned.

Keywords : video conference, adaptive frame rate, frame mixing, client-server, JMF

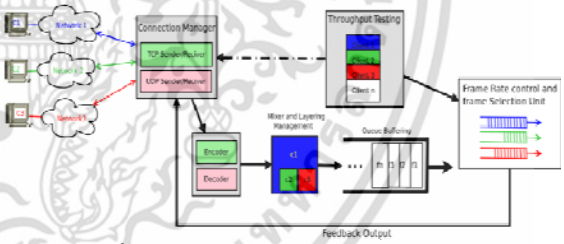
1. บทนำ

ปัจจุบันมีการส่งวีดีโอผ่านเครือข่ายเพิ่มมากขึ้น ทั้งในแบบเครือข่ายปกติและไร้สาย[1] ที่มีคุณภาพของช่องสัญญาณเปลี่ยนแปลงตลอดเวลา โดยเฉพาะการส่งวีดีโอสตรีมที่รันบนแอพพลิเคชันต่าง ๆ อย่างไรก็ตาม ยังคงเป็นงานที่ท้าทาย เนื่องจากระบบเครือข่ายอาจมีการเปลี่ยนแปลงแบบพลวัต มีจำนวนช่องสัญญาณที่จำกัด และการสูญเสียของเพ็กเก็จข้อมูลวีดีโอสตรีมที่มีเนื้อหาองค์ประกอบแบบบิตเรทต่ำ[2] รวมทั้งความสามารถของการประมวลผลของอุปกรณ์ลูกข่ายที่มีอยู่อย่างจำกัด จะมีประโยชน์มากหากสามารถปรับแต่งคุณสมบัติบางประการของวีดีโอที่ได้รับจากเครื่องลูกข่ายจากแหล่งต่างๆ มาประมวลผลบนเครื่องวีดีโอแม่ข่าย และตอบกลับไปยังแหล่งที่มานั้นหลังจากการการประเมินความสามารถของข้อมูลที่เดินทางกลับจากเครื่องวีดีโอแม่ข่ายถึงปลายทาง ดังนั้นต้องอาศัยกลไกที่ทำงานโดยหลักการของสถาปัตยกรรมแบบ ไคลเอนต์-เซิร์ฟเวอร์ เพื่อการวิเคราะห์ความสามารถในการส่งข้อมูลระหว่างเครื่องแม่ข่ายและเครื่องลูกข่ายบนเครือข่ายที่แพ็กเกจข้อมูลถูกส่งไปแล้วนำค่าความสามารถดังกล่าวมาคำนวณหาจำนวนเฟรมต่อวินาทีที่เหมาะสมและใกล้เคียงเวลาจริงมากที่สุดเมื่อมีดีเลย์เกิดขึ้น

จากที่ได้กล่าวมาแล้วนั้นสิ่งสำคัญคือ วิธีการออกแบบการทำงานของระบบ วีดีโอมิกเซอร์เซิร์ฟเวอร์ที่ใช้ในการผสมวิดีโอสตรีมในคอนเฟอร์เรนซ์ (chat room) กลุ่มเดียวกันในลักษณะของการแชร์บัฟเฟอร์ แล้วเลือกเฟรมภาพที่อยู่ในบัฟเฟอร์ดังกล่าว จำนวนเฟรมที่ถูกเลือกนั้นมาจากการคำนวณ ที่ได้ประเมินความสามารถของระบบเครือข่ายจากแหล่งที่มาของสมาชิกในกลุ่มของคอนเฟอร์เรนซ์นั่นเอง ประเด็นคือ เมื่อความสามารถของระบบเครือข่ายในกลุ่มคอนเฟอร์เรนซ์ต่างกัน การคำนวณเลือกเฟรมจากบัฟเฟอร์ก็ต่างกัน อาจทำให้บัฟเฟอร์นั้นไม่เพียงพอ (insufficient buffer) ต่อจำนวนเฟรมขาเข้า และถูกดันออกในรูปแบบของ FIFO เพื่อลดปัญหาดังกล่าวจึงต้องนำหลักการแบบ store and forward เพื่อให้เฟรมในบัฟเฟอร์ ล้นออกมา (overflow) แล้วนำเฟรมข้อมูลดังกล่าวมาประมวลผลต่อไป เฟรมที่ถูกเลือกมานั้นเป็นลักษณะแบบสุ่มจึงมีความจำเป็นที่จะต้องทำสำเนาและลำดับชุดของเฟรมต่อวินาทีในบัฟเฟอร์ใหม่ เพื่อชดเชยและลดค่าของเวลาที่สูญเสียไป จึงจะทำให้ค่าเฟรมเรทที่ได้นั้นเหมาะสมมากที่สุด

2. โครงสร้างและการออกแบบของระบบ Video Mixer Server

โครงสร้างและการออกแบบภายในของระบบวีดีโอมิกเซอร์ประกอบไปด้วย 5 ส่วนประกอบหลัก คือ ส่วนของการรอรับการติดต่อจากเครื่องลูกข่าย (Connection Listeners) ส่วนของการทดสอบประสิทธิภาพในการส่งข้อมูลระบบเครือข่าย (Throughput Testing), ส่วนของการเข้าและถอดรหัส (Encoder and Decoder), ส่วนผสานและจัดตำแหน่งเฟรมภาพ (Mixer and Layering Management) และหน่วยของลำดับในการทำบัฟเฟอร์ (Queue Buffering) องค์ประกอบของแต่ละส่วน แสดงดังรูปที่ 1



รูปที่ 1. โครงสร้างของระบบวีดีโอมิกเซอร์

2.1 การรับการติดต่อจากเครื่องลูกข่าย (Connection Listeners)

ด้วยหลักการทำงานของเครื่องแม่ข่าย-ลูกข่าย ประกอบด้วยโปรโตคอล TCP [3] สำหรับการรับส่งข้อความในการควบคุมการทำงานของระบบ (massage control) ระหว่างเครื่องไคลเอนต์-เซิร์ฟเวอร์ เช่น ข้อความที่ใช้สำหรับการล็อกอิน เข้า-ออก ระบบ ข้อความสำหรับบอกจำนวนและรายละเอียดของผู้เข้าร่วมในกลุ่มเดียวกัน สำหรับข้อมูลวีดีโอนั้นมีความจำเป็นในการรับและส่งข้อมูลที่ต้องอาศัยความเร็วและต้องการความถูกต้องของ

ข้อมูลที่น้อยกว่า ดังนั้น RTP [4] ซึ่งจัดอยู่ในโปรโตคอลชนิด UDP จึงมีความเหมาะสมมากกว่าในการส่งข้อมูลในลักษณะของวีดีโอสตรีมมิ่ง

2.2 ทดสอบข้อมูลที่ส่งผ่านระบบเครือข่าย (Throughput Testing)

ความสามารถในการส่งข้อมูลผ่านระบบเครือข่ายนั้นย่อมต่างกันทั้งในรูปแบบของสถาปัตยกรรมโครงสร้างพื้นฐาน กระทั่งช่วงเวลาของการใช้งาน ดังนั้นจำนวนเฟรมต่อวินาทีใด ๆ ขึ้นอยู่กับปริมาณข้อมูลที่ส่งได้ในเครือข่าย คำนวณได้จาก

$$T = \frac{g(x)}{\Delta t} \quad (1)$$

เมื่อ T แทนค่าของ Throughput ที่ได้ในหน่วย ไบต์ต่อวินาที $g(x)$ คือจำนวนของ ไบต์ที่ถูกส่งจนถึงปลายทาง โดยที่ x คือ ค่าที่ได้จากการสุ่มของจำนวณเต็ม ระหว่าง 1 ถึง 9 Δt คือ ช่วงเวลาใด ๆ

$$g(x) = random(x) \times 1024 \qquad 1 \le x < 10 \quad (2)$$

นำมาคำนวณหาเฟรมต่อวินาที λ ได้โดย

$$\lambda = \frac{8 \cdot T}{n(w \cdot h)} \quad (3)$$

n คือค่าความลึกของสีในหน่วยบิต w และ h คือขนาดความกว้างและความสูงของเฟรม โดยที่การทดสอบความสามารถในการส่งข้อมูลจะมีการทดสอบเป็นระยะ ๆ ทุก x วินาที

2.3 การเข้าและถอดรหัส (Encoder and Decoder)

ข้อมูลวีดีโอที่รับมามีการเข้ารหัสแบบ low bit rate ในรูปแบบของ H263 [5] หรือรูปแบบอื่น ผ่าน RTP โปรโตคอล ซึ่งต้องทำการถอดรหัสที่เหมาะสม แล้วทำการดักจับสตรีมของวีดีโอแบบเฟรมต่อเฟรมออกมา ในลักษณะภาพข้อมูลดิบ จากนั้นนำข้อมูลที่ได้ ไปเข้ารหัสเป็น JPEG ก่อนที่จะส่งต่อไปยัง ส่วนของการผสานและจัดตำแหน่งเฟรมภาพ กระบวนการเข้าและถอดรหัสนี้จะเกิดขึ้นกับทุก ๆ วีดีโอสตรีมที่ถูกส่งมาเป็นเซสชัน โดยแต่ละเซสชันนั้นก็จะแยกออกจากกันโดยอิสระ

2.4 ผสานและจัดตำแหน่งเฟรมภาพ (Frame Mixer and Layering Management)

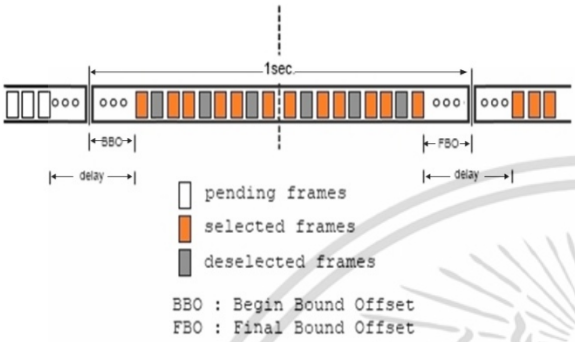
สตรีมของภาพนิ่งของแต่ละเซสชันที่วิ่งเข้ามาในส่วน ของ Frame Mixer [6] จะถูกลำดับสัดส่วนลงใน layout ภายใน layout จะมีตำแหน่งและขนาดที่ถูกระบุเอาไว้สำหรับแต่ละเซสชัน เฟรมจากเซสชันถูกวาดลงใน layout แบบ FIFO เฟรมจากเซสชันใดมาถึงก่อนจะเป็นเลเยอร์พื้นหลัง (Background Layer) ส่วนที่เหลือจะเป็นเลเยอร์ส่วนหน้า (Foreground Layer) ทั้ง Background Layer และ Foreground Layer ถูกแนบเข้าด้วยกันเป็นเฟรมใหม่ ซึ่งจะถูกส่งไปยัง บัฟเฟอร์ต่อไป

2.5 ลำดับในการทำบัฟเฟอร์ (Queue Buffering)

สิ่งสำคัญที่ทำให้สตรีมมิ่งมีความต่อเนื่องนั่นคือ บัฟเฟอร์ ซึ่งมีองค์ประกอบหลักสองส่วนคือจำนวนและขนาดของบัฟเฟอร์ ขนาดของบัฟเฟอร์ยิ่งมากก็ยิ่งทำให้สตรีมราบรื่นและต่อเนื่องแต่ก็ต้องแลกมาด้วยค่าของ ดีเลย์และเวลาในการประมวลผล (Processing Time) เพิ่มมากขึ้น ถ้าขนาดบัฟเฟอร์น้อยเกินไปก็ทำให้สตรีมไม่ราบรื่นเท่าที่ควร ดังนั้นขนาดของบัฟเฟอร์ต้องมีความเหมาะสม บัฟเฟอร์ส่วนใหญ่จะถูกออกแบบมาให้ทำงานแบบเก็บและคาย (Store and Forward) ในบทความนี้จะมีการกำหนดขนาดบัฟเฟอร์ที่ไม่สูงมากเพื่อลดค่าดีเลย์ แก้ไขความต่อเนื่องของสตรีมโดยอาศัยการทำ Overflow Buffer กล่าวคือ ไม่ว่าอัตราเฟรมต่อวินาทีขาเข้ามีความเปลี่ยนแปลงอย่างต่อเนื่อง เฟรมที่จะนำไปใช้งานจริงคือ เฟรมที่ล้นออกมาจาก Buffer จากนั้นกลุ่มเฟรมดังกล่าวจะถูกนำไปคัดเลือกและนำไปใช้งานต่อไป

3. วิธีปรับเฟรมเรทและเลือกเฟรมในบัฟเฟอร์

สตรีมของกลุ่มภาพนิ่งถูกดึงออกจากบัฟเฟอร์อย่างต่อเนื่องจากนั้น จะถูกทำสำเนาลงสู่บัฟเฟอร์ย่อย ๆ จำนวนของบัฟเฟอร์ย่อย ๆ นี้จะมีจำนวนเท่ากับสตรีมเซสชัน รูปแบบของกลุ่มภาพนิ่งในบัฟเฟอร์ย่อยนี้เรียกว่า Image Unit รูปที่ 2 แสดงโมเดลของ Image Unit



รูปที่ 2 รูปแบบกลุ่มภาพนิ่งในบัฟเฟอร์ Image Unit

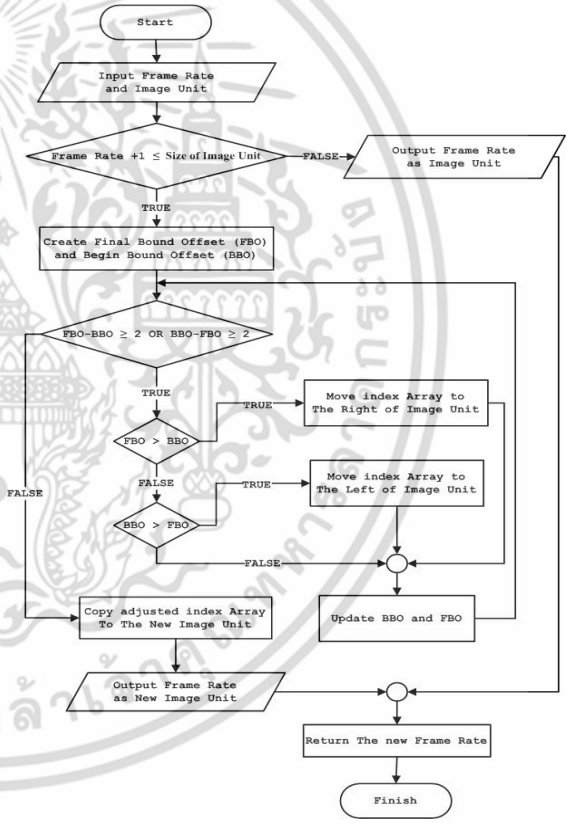
การเลือกเฟรมภาพจะเลือกกลุ่มของภาพต่อหน่วยเวลา (วินาที) ภายใน Image Unit ของบัฟเฟอร์ย่อยนี้ จำนวนของเฟรมที่จะเลือกมาจากการคำนวณเฟรมเรทที่ได้จาก ข้อ 2.2 ถ้า λ คือเฟรมเรทที่เหมาะสมของแต่ละ stream i ภายในบัฟเฟอร์ ณ เวลาต่าง ๆ t λ_{FBO_t} และ λ_{BBO_t} แทนอินเทอร์นอลดีเลย์ของเฟรมที่ไม่ถูกเลือกช่วงระยะต้นและอินเทอร์นอลดีเลย์ของเฟรมที่ไม่ถูกเลือกช่วงระยะปลายภายใน Image Unit ตามลำดับ แล้วค่าคิวอิ้งดีเลย์ d_{que_t} ของบัฟเฟอร์ คือ

$$d_{que_t} = \sum_{i=1}^{n} (\lambda_{FBO_t} + \lambda_{BBO_t})_i \qquad (4)$$

เฟรมใดภายในบัฟเฟอร์ที่จะถูกเลือกนั้นอาศัยอัลกอริทึมโดยมีจำนวนเฟรมเรทที่ได้จากข้อ 2.2 เป็นอินพุท และใช้ Index Array เป็นตัวระบุตำแหน่งเฟรมภายใน Image Unit สามารถแสดงได้ด้วย pseudo code และ flow chart ดังต่อไปนี้

```
Object[] selectFrameInTimelineBuffer(int frameRate, Object[] inImageUnit) {
        int unitSize = imageUnit.size;
        Object[] tempImgUnit = new Object[frameRate];
        Object[] outImageUnit = new Object[frameRate];
        if( frameRate +1 ≤ unitSize ){
                int innerOffset = unitSize / (frameRate+1);
                for(int i=0; i < frameRate; i++) {
                        tempImgUnit[i] = inImageUnit[i+innerOffset]
                }
                int finalBoundOffset = last inImageUnit index - last tempImgUnit index
                int beginBoundOffset = first tempImgUnit index - first inImageUnit index
                while((finalBoundOffset - beginBoundOffset ≥ 2) or
                                (beginBoundOffset - finalBoundOffset ≥ 2) ) {
                        if (finalBoundOffset > beginBoundOffset ) {
                                move tempImgUnit index array to the right inImageUnit;
                        }
                        if (beginBoundOffset > finalBoundOffset ) {
                                move tempImgUnit index array to the left inImageUnit;
                        }
                        update finalBoundOffset and beginBoundOffset
                }
                for(int j = 0; j < frameRate; j++) {
                        outImageUnit[j] = tempImgUnit[j];
                }
                return outImageUnit;
        }
        else {
                return inImageUnit;
        }
}
```

รูปที่ 3 pseudo code อัลกอริทึม ในการเลือกเฟรม

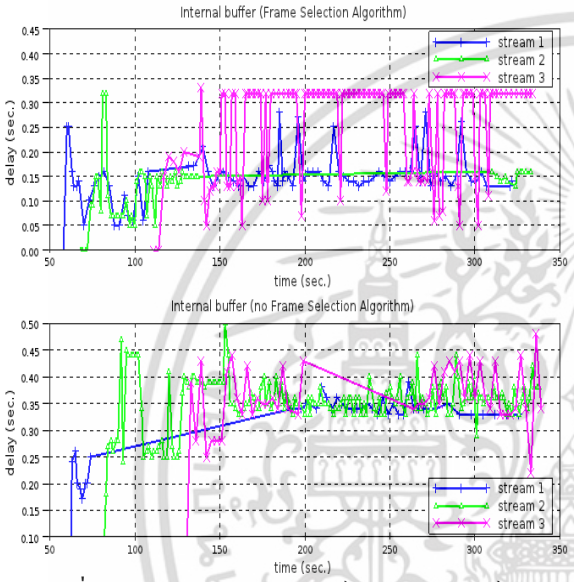


รูปที่ 4 Flow Chart แสดงอัลกอริทึม ในการเลือกเฟรม

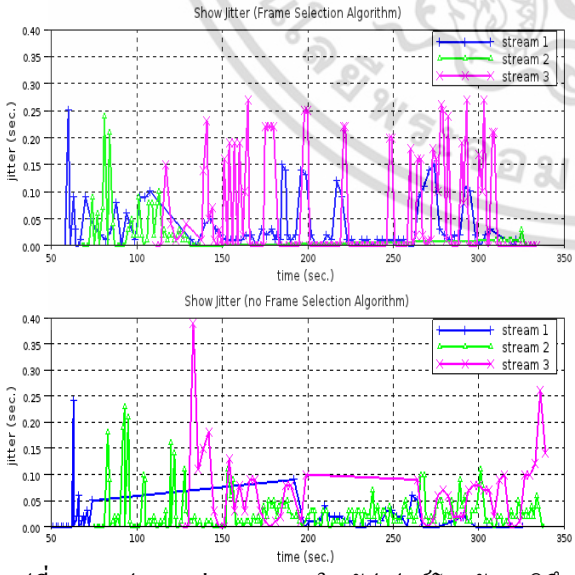
4. ผลการทดลอง

บทความนี้ได้แสดงผลการทดลอง เปรียบเทียบค่าของดีเลย์ที่เกิดขึ้นในบัฟเฟอร์ ค่าความคลาดเคลื่อนของเวลา หรือ jitter และค่าของเฟรมเรท ที่เกิดขึ้นจากการปรับเฟรมเรทที่ใช้อัลกอริทึมในการเลือกเฟรมและการปรับ

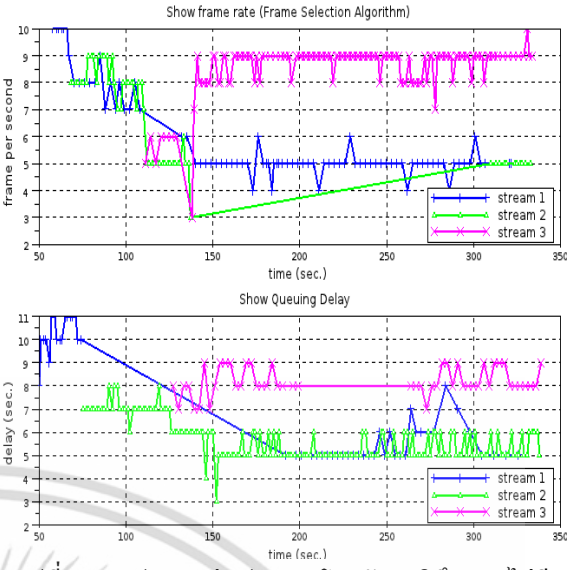
เฟรมเรทที่ไม่ได้ใช้อัลกอริทึม กำหนดให้ stream 1, stream 2 และ stream 3 คือวีดีโอที่ถูกส่งผ่านโครงข่ายที่แตกต่างกันคือ โครงข่าย WAN โครงข่าย LAN และโครงข่าย WLAN ตามลำดับ โครงสร้างต่าง ๆ ในการออกแบบระบบนี้ นำมาสร้างและพัฒนาเป็น application โดย application ดังกล่าวใช้ภาษา Java และ JMF : Java Media Framework [7,8] ซึ่งเป็น Application Programming Interface ในการพัฒนา



รูปที่ 5 กราฟแสดงค่าของดีเลย์ภายในบัฟเฟอร์โดยอัลกอริทึมและไม่มีอัลกอริทึมเพื่อเลือกเฟรม



รูปที่ 6 กราฟแสดงค่า jitter ภายในบัฟเฟอร์โดยอัลกอริทึมและไม่มีอัลกอริทึมเพื่อเลือกเฟรม



รูปที่ 7 กราฟแสดงค่าเฟรมเรทโดยอัลกอริทึมและไม่มีอัลกอริทึมเพื่อเลือกเฟรม

ผลการทดลองที่นำเสนอด้วยกราฟจากรูปที่ 5 ถึงรูปที่ 7 สรุป เป็นตารางค่าเฉลี่ย Mean และค่าเบี่ยงแบนมาตรฐาน S.D. จากพารามิเตอร์ดังต่อไปนี้คือ เฟรมเรท (FPS) ดีเลย์ภายในบัฟเฟอร์ และค่าความคลาดเคลื่อนของเวลา jitter ระหว่าง การใช้ อัลกอริทึมและไม่ใช้อัลกอริทึมเพื่อเลือกเฟรมในบัฟเฟอร์ ในช่วงวินาทีที่ 130 – 340 ซึ่งเป็นช่วงเวลาที่ stream อยู่ในบัฟเฟอร์ ดังตารางที่ 1

	Frame Selection Algorithm						No Frame Selection Algorithm					
	Delay		Jitter		FPS		Delay		Jitter		FPS	
	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.
stream1 (WAN)	0.150	0.037	0.034	0.045	5.9	0.4	0.343	0.014	0.015	0.020	5.4	0.7
stream2 (LAN)	0.151	0.009	0.006	0.008	6.6	0.6	0.363	0.028	0.024	0.024	5.3	0.5
stream3 (WLAN)	0.255	0.094	0.057	0.092	8.6	0.7	0.357	0.076	0.068	0.072	8.2	0.6

ตารางที่ 1 แสดงค่า FPS ค่าดีเลย์ และค่า jitter ของแต่ละ stream โดยอัลกอริทึมและไม่มีอัลกอริทึมเพื่อเลือกเฟรม

5. สรุป

จากการทดลองเห็นได้ชัดว่า การส่งวีดีโอสตรีมภายหลังการปรับเฟรมเรทเพื่อให้เหมาะกับ Throughput นั้น ยังไม่เพียงพอที่จะทำให้การส่งวีดีโอสตรีมดีขึ้นได้ เนื่องจากมีดีเลย์เกิดขึ้นในระบบ ดังนั้นต้องอาศัยวิธีการที่จะช่วยลดค่าดีเลย์ดังกล่าว นั่นคือ อัลกอริทึมในการเลือกเฟรมภายในบัฟเฟอร์ ค่าเฉลี่ยของดีเลย์ที่เกิดขึ้นในบัฟเฟอร์ โดยอัลกอริทึมมีค่าที่น้อยกว่าประมาณ 2.3 เท่า

เมื่อเทียบกับไม่มีอัลกอริทึมเพื่อเลือกเฟรมในบัฟเฟอร์สำหรับเครือข่ายแบบ WAN และ LAN และน้อยกว่าประมาณ 1.4 เท่า สำหรับเครือข่ายแบบไร้สายหรือ WLAN ในขณะเดียวกันค่าความคลาดเคลื่อนของเวลา หรือ jitter ก็มีค่าน้อยกว่าแบบไม่ใช้อัลกอริทึมเช่นกันในกรณีเครือข่ายที่เป็น LAN และ WLAN ในทางกลับกันการใช้อัลกอริทึมในการเลือกเฟรม ยังให้ค่าเฟรมเรท หรือ FPS ที่สูงกว่าเมื่อเทียบกับแบบไม่มีอัลกอริทึม ดังนั้นการใช้อัลกอริทึมเพื่อช่วยเพิ่มประสิทธิภาพของการส่งวีดีโอ นั้นสามารถทำได้ดี แต่ยังมีข้อจำกัดบางประการคือ การใช้งานจะรู้สึกถึงอาการกระตุกของภาพ เป็นระยะๆ ถึงแม้ค่าเฉลี่ย jitter มีค่าน้อย แต่ก็ให้ค่าส่วนเบี่ยงเบนมาตรฐานที่มากกว่าค่าเฉลี่ย jitter การกระจายตัวของ jitter ที่เกิดขึ้น สาเหตุเกิดจาก ตัวอัลกอริทึมมีการปรับและขยับช่วงของจำนวนเฟรมของ Image Unit ภายในบัฟเฟอร์อยู่ตลอดเวลาและไม่สนใจว่าเฟรมที่อยู่ภายในบัฟเฟอร์นี้ล่วงเวลาในการประมวลผลและแท้จริงแล้วมีช่วงระยะห่างไปจากเฟรมที่ไม่ถูกเลือกแล้วกี่วินาที แต่สนใจเพียงว่าเฟรมปัจจุบันและเฟรมที่กำลังจะวิ่งเข้ามาในบัฟเฟอร์จะต้องส่งอย่างไรให้ทันและสัมพันธ์กับค่า Throughput นั่นเอง

6. กิตติกรรมประกาศ

ผู้วิจัยขอขอบคุณ รศ.ดร. สุวิพล สิทธิชีวภาค สำหรับเครื่องแม่ข่ายที่ใช้สำหรับการทดสอบ และสำนักบริการคอมพิวเตอร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ที่เอื้อเฟื้อระบบเครือข่ายในการทดสอบและสนับสนุนงานวิจัย

7. เอกสารอ้างอิง

[1] A. Majumda, D.G. Sachs and I.V. Kozintsev, K. Ramchandran and M.M. Yeung, “Multicast and unicast real-time video streaming over wireless lans,” IEEE Trans. Circuits and Systems for Video Technology, Vol.12, No.6., pp.524–534, Jun., 2002.

[2] K. Tan, R. Ribier and S. Liou, “Content-sensitive video streaming over low bitrate and lossy wireless network,” ACM 2011. 9th international conference on Multimedia, pp. 512–515, Sept., 2001.

[3] J. Postel, “Transmission Control Protocol,” RFC-793, USC/Information Sciences. Sept., 1981.

[4] H. Schulzrine, S. Casner, R. Frederick, V. Jacobson, “RTP: A Transport Protocol for Real-Time Applications,” RFC1889., Jan., 1996.

[5] R.E. Parker, Jr. and M. Tummala, “Modeling of H.263 encoded low bitrate video traffic for tactical video conferencing applications,” Conference Record of the Thirty-Second Asilomar Conference, Signals, Systems & Computers, Vol.2, pp.1626-1630, 1998.

[6] Xin-Gang Liu, Kook-Yeol Yoo and Kwang Deok Seo, “A Fast Video Mixing Method For Multiparty Video Conference,” ICIAR, Vol.3656, pp.320-327, 2005.

[7] L. Mengual, J. Bobadilla, R. Caballero, G. Hernández, “Design and Testing of two secure Video Conferencing applications based on JMF (Java Media Framework) and VIC (Video Conferencing Tool),” ICDT 2006. International Conference on, 29-31 Aug 2006.

[8] R. Gordon and S. Tally, JMF Java Media Framework, Prinice-Hall, 1998.