일반논문 (Regular Paper)
방송공학회논문지 제23권 제5호, 2018년 9월 (JBE Vol. 23, No. 5, September 2018)
https://doi.org/10.5909/JBE.2018.23.5.682
ISSN 2287-9137 (Online) ISSN 1226-7953 (Print)

MMT 프로토콜 기반의 다중쓰레드를 활용한 ARQ 패킷 오류 제어 기법

원 광 은[a)], 안 은 빈[a)], 김 아 영[a)], 이 홍 래[a)], 서 광 덕[a)‡]

ARQ Packet Error Control Scheme Using Multiple Threads Based on MMT Protocol

Kwang-eun Won[a)], Eun-bin Ahn[a)], Ayoung Kim[a)], Hong-rae Lee[a)], and Kwang-deok Seo[a)‡]

요 약

본 논문에서는 MMT(MPEG Media Transport) 프로토콜 기반의 대용량 멀티미디어 전송에서 다중쓰레드를 활용한 ARQ 패킷 오류 제어 기법을 제안한다. 송신 측에서는 영상을 구성하는 각 프레임을 MMT 프로토콜을 기반으로 패킷 단위로 잘라 패킷의 헤더(Header)에는 패킷이 포함된 프레임의 순서, 표현 시간 정보 등을 저장하고 페이로드(Payload)에는 프레임을 구성하는 직접적인 정보를 저장하여 IP(Internet Protocol) 망으로 전송한다. 수신 측에서는 수신한 패킷의 오류 발생 여부를 판단하여 오류가 발생한 경우 재전송을 통해 오류를 제어하고 수신한 패킷의 헤더에 저장된 정보에 따라 패킷을 프레임으로 재구성한다. 이때 다중쓰레드 기반의 전송 방식을 설계 및 적용하여 각 쓰레드가 하나의 프레임을 맡아 패킷화(packetization)하고 전송함으로써 대용량 멀티미디어의 전송 효율을 높인다. 또한 오류가 발생한 패킷을 재전송 할 경우 단일쓰레드를 사용할 때 나타날 수 있는 문제점을 해결함으로써 다중쓰레드 전송 방식의 효율성을 검증한다.

Abstract

In this paper, we propose an ARQ packet error control scheme using multiple threads in delivering massive capacity of multimedia based on MMT(MPEG Media Transport) protocol. On the sending side, each frame that constitutes an image is packetized into MMT packets based on MMT protocol. The header of the packet stores the sequence number of the frames contained in the packet and the time of presentation information. The payload of the packet stores the direct information that comprises the frame. The generated MMT packet is transmitted to the IP network. The receiving side checks if any error has occurred in the received packet. For any identified error, it controls the error through ARQ error control scheme and reconfigure the frame according to the information stored in the header of the received packet. At this point, a multi-threading based transport design is constructed so that each thread takes over a single frame, which increases the transmission efficiency of massive capacity multimedia. The efficiency of the multi-threading transport method is verified by solving the problems that might arise when using a single-thread approach if packets with errors are retransmitted.

Keyword : MMT protocol, Multi-thread processing, Multi-thread transport, ARQ error control

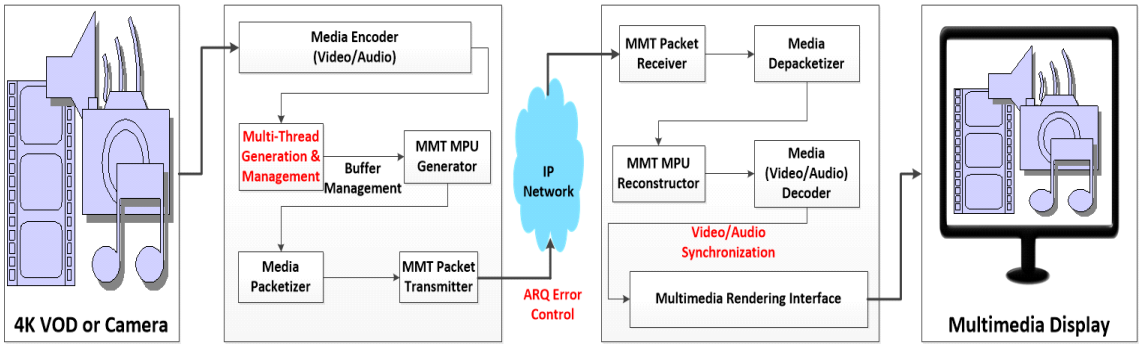


그림 1. 본 연구에서 구현한 MMT 프로토콜 기반의 대용량 멀티미디어 송수신 구조
Fig. 1. Architecture of the huge-volume multimedia transceiver system implemented in this research

I. 서 론

UHD 시대에 사용될 대용량 멀티미디어 부호화 표현 및 다중화 전달을 위해 MPEG에서는 MPEG-H를 표준화하였다. 그 중 MMT(MPEG Media Transport) 기술은 MPEG에서 표준화가 진행된 차세대 미디어 전송 규격으로서 방송망(Broadcast Network)과 인터넷망을 동시에 서비스 할 수 있는 대용량 초고화질 하이브리드 미디어 전송에 적합한 기능을 제공한다[1][2][3]. MMT 기술은 기존의 방송 및 인터넷 망을 통한 미디어의 포장과 저장 및 전송을 위해 개발된 기술들을 향후 스마트 시대에 맞게 전송 환경에 상관없이 혼합하여 사용할 수 있도록 하는 목표를 가지고 있다[4].

대용량 초고화질의 멀티미디어를 실시간으로 전송할 경우 대량의 데이터 패킷을 네트워크로 보내기 때문에 네트워크 대역폭의 과부하를 야기함과 동시에 데이터 안정성과 멀티미디어 사이의 동기화 처리에 문제를 발생시킨다[5][6]. 이를 위해 본 논문에서는 대용량 초고화질 실시간 멀티미디어 전송 시 MMT 프로토콜 기반의 다중쓰레드 전송 기법을 설계하고 이 제안 기법을 실제 오류가 발생하는 인터넷을 통한 대용량 멀티미디어 전송에 적용하여 효율성을 검증하고자 한다. 이를 위해서, 송신 측에서는 MMT 기술을 기반으로 대용량 VOD 또는 초고화질 실시간 스트리밍 영상을 전송한다. 이때 대용량 미디어 데이터의 전송 효율을 높이기 위해 다중쓰레드 기법을 적용한다. 수신 측에서는 제안하는 기법을 통해 수신한 멀티미디어 데이터의 오류 유무를 확인해 오류가 발생한 데이터 패킷은 재전송을 통해 복구한다. 그리고 다중쓰레드를 사용함으로써 나머지 데이터 패킷은 정상적으로 전송이 가능하므로 단일쓰레드 사용 시 오류 복구를 위한 재전송 과정에서 영상이 정지하는 현상을 방지한다.

이러한 목표를 달성하기 위해 본 연구에서 구현한 MMT 프로토콜 기반의 대용량 멀티미디어 송수신 구조는 그림 1과 같다. 대용량 멀티미디어 데이터는 송신 측의 인코더를 통해 필요에 따라 압축 또는 무압축 된다. 인코딩된 대용량 멀티미디어를 효율적으로 MMT 패킷화하고 전송하기 위해 다중쓰레드 기법이 사용되며 나뉜 패킷은 네트워크를 통해 전송된다. 수신 측에서는 수신한 MMT 패킷을 디코딩해 오류가 발생한 패킷에 대해서는 오류를 복구한 뒤 미디어 간 동기화를 맞추고 렌더링한다.

II. 관련 기술 개요

1. MMT 표준 기술 개요

MMT는 그림 2와 같이 크게 압축된 미디어에 대한 포장

a) 연세대학교 컴퓨터정보통신공학부(Division of Computer and Telecommunications Engineering, Yonsei University)
‡ Corresponding Author : 서광덕(Kwang-deok Seo)
E-mail: kdseo@yonsei.ac.kr
Tel: +82-33-760-2788
ORCID: http://orcid.org/0000-0001-5823-2857
※ This research was supported by the Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(NRF-2018R1D1A1B07047065).
· Manuscript received July 19, 2018; Revised August 24, 2018; Accepted, August 24, 2018.

기능을 담당하는 Encapsulation Function (E-layer), 전송 기능을 담당하는 Delivery Function (D-layer), 그리고 시그널링 기능을 담당하는 Signaling Function (S-layer)으로 구성되어 있다[1].

포장 기능 영역(E-layer)은 부호화된 미디어 데이터를 네트워크의 페이로드에 실어 보내기 위해 캡슐화(encapsulation)하는 포맷을 정의한다. MPU(Media Processing Unit)로 미디어 컨텐츠의 논리적인 구조를 정의하고 이는 MMT 기술을 이용할 때 독립적으로 완벽하게 소비가 가능한 미디어의 단위이며 ISO BMFF(ISO Based Media File Format)의 형태로 생성된다. 이 포맷에는 컨테이너 포맷, 미디어 패킷화 포맷 등이 포함된다.

전송 기능 영역(D-layer)은 포장된 미디어 데이터를 다른 네트워크로 전송하기 위한 응용계층 전송 프로토콜과 다양한 타입의 미디어가 저장될 페이로드의 포맷을 정의하는 영역이다. 이 영역에서는 네트워크로 전송하기 위한 MPU들의 분할(fragmentation)과 결합(aggregation)을 수행한다. 또한 네트워크 흐름 다중화, 네트워크 패킷화, QoS(Quality of Service) 보장을 위한 AL-FEC와 ARQ(Automatic Repeat reQuest) 등이 이 영역에 포함된다.

시그널링 기능 영역(S-layer)은 MMT 미디어 패키지의 전송과 소비에 필요한 시그널링 메시지의 포맷을 정의한다. 서버와 클라이언트 간 컨텐츠 전송을 위해 패키지 내에 포함된 페이로드의 포맷과 패키지를 구성하고 있는 프로토콜

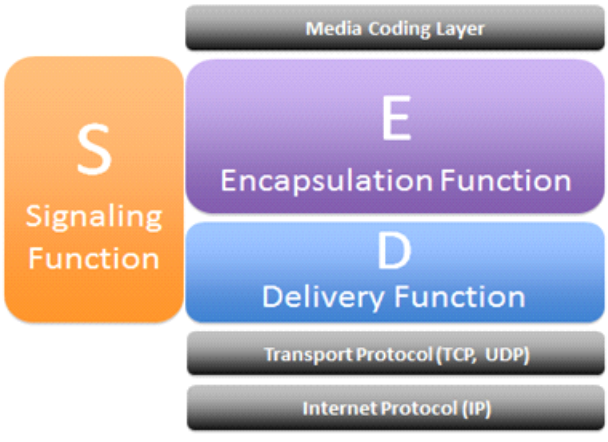


그림 2. MMT 기능의 계층 구조
Fig. 2. Layer structure of the MMT functional areas

구성에 대한 정보를 제공하며 5개의 메시지가 정의되어 있다.

2. 쓰레드(Thread) 기반 설계 및 구현 기술

쓰레드라는 용어는 프로세스에서 실행되는 흐름(Flow)의 단위로 정의할 수 있다. 그림 3에 보이듯이, 하나의 프로세스는 여러 개의 쓰레드를 가질 수 있고 1개일 경우 단일쓰레드(Single-Thread), 그 이상은 다중쓰레드(Multi-Thread)라고 한다[7]. 1개 이상의 쓰레드가 존재할 경우 기본이 되는 쓰레드는 메인쓰레드(Main-Thread)라고 하며 단일쓰레드일 경우 그 자체로 메인쓰레드가 된다. 쓰레드는 프로세스 내에서 스택(Stack) 영역만 따로 할당을 받고 Code, Data, Heap 영역을 공유한다. 스택 영역은 프로그램이 동작하기 위한 인자와 프로세스의 상태를 저장하는데 사용된다. Code 영역에는 특정 프로세스의 동작 상태를 저장하는 곳으로 프로세스가 독립적인 포인터를 가지고 작동하기 위한 영역이며 Data 영역은 하나의 프로세스가 작동 중에 필요로 하는 데이터를 저장하기 위한 영역이고 Heap 영역은 프로그램이 동작할 때 필요한 데이터 정보를 임시적으로 저장하는 영역이다.

다중쓰레드는 여러 개의 쓰레드가 하나의 작업을 매우 짧은 단위로 분할하여 다수의 쓰레드에 할당함으로써 동시에 병렬로 전송을 처리하는 것처럼 작업한다[8]. 그러나 쓰레드를 사용할 때는 쓰레드 간 실행 순서를 예측할 수 없고 공유 데이터를 갱신하는 타이밍에 다른 쓰레드에서 자원을 사용한다면 갱신되지 못한 데이터가 사용될 수 있다. 이처럼 쓰레드를 사용할 경우에 공유하는 데이터의 안전성에 문제가 생길 수 있다. 때문에 쓰레드를 구현할 경우 임계영역(Critical Section)이라는 제약사항을 두어 한 쓰레드가 일정 상태가 될 때까지 자원을 독점으로 사용할 수 있는 권리를 준다. 쓰레드가 짧은 시간동안 자원을 독점할 경우 다른 쓰레드는 대기 상태에 들어가고 자원의 사용이 끝난 쓰레드는 대기하는 쓰레드에게 신호를 보내 쓰레드가 다음 동작을 할 수 있도록 해야 한다[9]. 만약 이 과정이 일어나지 않는다면 각 쓰레드는 다른 쓰레드가 끝나기만을 기다리는 대기 상태가 계속 종료되지 않는 이른바 교착 상태(Dead Lock)에 빠지게 된다.

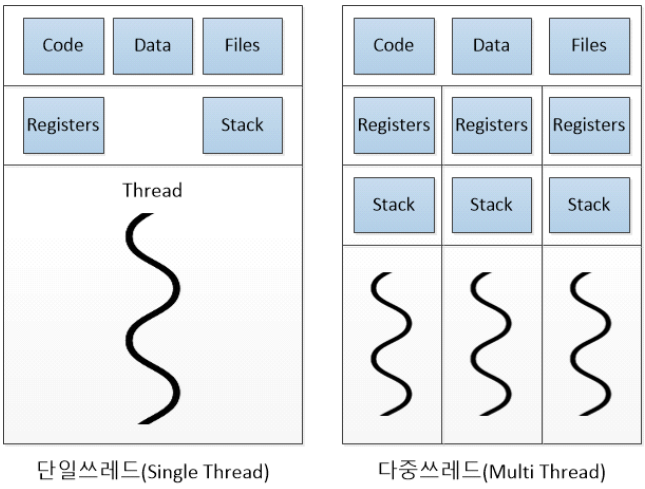


그림 3. 단일쓰레드와 다중쓰레드의 개념도
Fig. 3. Concepts of single-thread and multi-thread

III. 제안하는 MMT 기반의 다중쓰레드 패킷 오류 제어 기법

MMT 프로토콜 기반의 전송에서, 송신 측에서는 대용량 미디어 데이터를 나누어 보낼 때 독립적으로 완벽하게 소비가 가능한 단위인 MPU(Media Processing Unit)로 데이터를 분할하게 되는데, 송신 측과 수신 측에서 한 번에 하나의 데이터를 캡슐화하고 하나의 채널을 사용해 전송하게 되면 MMT 시스템에서 지원하는 시그널링을 사용하더라도 데이터 자체를 보내는 데에 시간적, 물리적 자원 낭비를 초래하게 된다. 본 논문에서 제안하는 방법은 다중쓰레드 기반으로 공유 데이터를 분할 및 전송함으로써 대용량 멀티미디어를 효율적으로 전송하고 이때 MMT 패킷에 발생하는 오류를 제어하는 방법이다.

그림 4는 단일쓰레드 기반으로 입력된 프레임 데이터를 MMT 기술을 사용해 패킷화하고 전송하는 송수신 구조이다. 송신 측의 Sending Unit과 수신 측의 Receiving Unit은 서로 쌍(Pair)을 이루는 단일쓰레드이며 서로 데이터를 주고 받는다. FFMPEG 라이브러리 함수를 사용해 영상을 한 프레임 단위로 큐(Queue)에 저장하고 저장된 프레임 데이터를 전송에 필요한 MMT 패킷 단위로 분할한다. 이때 MMT 패킷은 해당 패킷과 패킷이 속하는 프레임에 대한 여러 정보뿐만 아니라 미디어 종류에 따라 인코딩 및 디코딩에 필요한 코덱(Codec) 정보를 가지고 있으며 코덱의 종류에 따라 가지는 정보가 조금씩 달라질 수 있다. 또한 분할 시 MMT 패킷의 헤더 중 timestamp 필드에 해당 비디오 데이터와 오디오 데이터의 DTS(Decoding Time Stamp) 및 PTS(Presentation Time Stamp) 정보를 담아 전송하기 때문에 각 패킷이 수신 측에 도착하였을 때 디코딩 시간과 표현 시간을 알 수 있다.

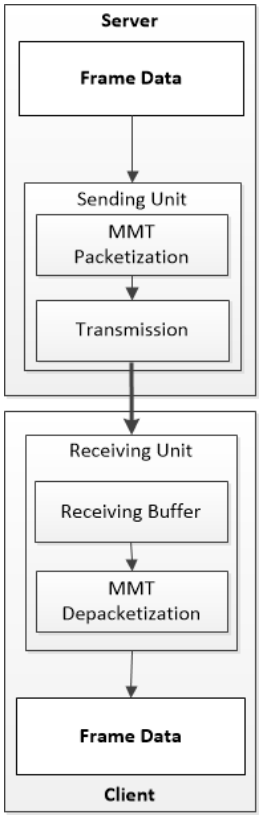


그림 4. 단일쓰레드 기반의 MMT 패킷의 송수신 구조
Fig. 4. Structure of the MMT packet transceiver based on single-thread

프레임 데이터의 크기가 크지 않을 때에는 위와 같은 방식으로 송신 측에서 프레임을 패킷화해 전송하고 수신 측에서는 수신된 패킷을 저장한 뒤 역패킷화(depacketization) 과정을 거쳐 영상을 재생한다. 그러나 초고화질의 대용량 멀티미디어 데이터를 전송할 경우 기존의 네트워크를 통해 패킷이 전송되는 시간은 변함이 없다하더라도 프레임 데이

터의 크기가 매우 크다. 따라서, 송수신 측에서 프레임 데이터를 패킷화 및 역패킷화하는데 더 많은 시간이 소요되고 이는 영상의 정상적인 재생에 문제를 발생시킬 수 있다. 이처럼 대용량 멀티미디어 데이터를 전송할 때 다중쓰레드를 사용하면 위와 같은 문제를 해결할 수 있다.

그림 5는 제안하는 다중쓰레드 기반의 대용량 초고화질 멀티미디어 송수신 구조를 나타낸다. 송신 측에서는 FFMPEG 라이브러리 함수를 통해 영상을 프레임 단위로 나누어 저장하고 각각의 쓰레드는 한 프레임을 할당받아 분할해 MMT 패킷화하고 네트워크를 통해 수신 측으로 전송한다. 수신 측에서는 송신 측과 쌍(Pair)을 이루는 쓰레드를 통해 MMT 패킷을 전송받아 역패킷화 과정을 거쳐 프레임을 재구성한다. 이렇게 다중쓰레드를 통해 대용량 멀티미디어를 전송할 경우 네트워크 포트를 여러 개 사용하는 다중채널(Multi-channel)과 같은 효과를 얻을 수 있다.

다중쓰레드를 사용할 경우 각각의 쓰레드에 할당되는 프레임 데이터 크기가 다르고 쓰레드 간 스케줄링으로 인해 수신 측에서 재구성되는 프레임의 순서가 송신 측에서 전송한 순서와 다를 수 있다. 이는 패킷 데이터를 정상적으로 수신하더라도 영상 자체에 오류를 유발시키기 때문에 수신 측의 프레임이 저장되는 큐(queue)에서는 재순서화(reordering) 과정을 거쳐 프레임을 정렬시켜야 한다.

앞서 수신 측에서 다중쓰레드를 통해 프레임을 수신할 경우 프레임의 재순서화의 중요성에 대해 기술하였다. 이는 각 프레임을 구성하는 패킷에도 해당한다. 캡슐화된 패킷은 IP 망을 통해 전송되기 때문에 프레임을 구성할 때 재순서화가 반드시 필요하다. 대용량 초고화질 멀티미디어의 경우 한 프레임을 구성하는 MMT 패킷의 개수가 많기 때문에 전송 과정에서 데이터 패킷의 손실(missing), 오류(error) 그리고 전송 지연(delay)에 의해 해당 패킷이 포함된 프레임의 재생 시간까지 전송되지 못하는 경우가 다른 일반적인 화질의 영상을 전송할 때보다 자주 발생할 수 있다. 위와 같은 문제가 발생할 경우 해당 패킷을 재전송(retransmit)하는 과정을 통해 오류를 제어해야 한다.

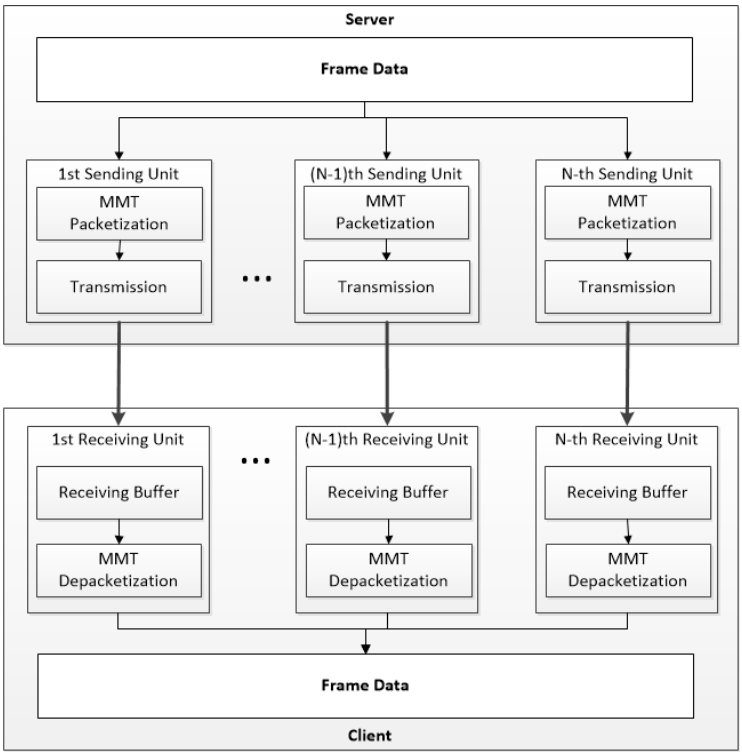


그림 5. MMT 기반의 다중쓰레드 송수신 모델
Fig. 5. Structure of the MMT packet transceiver based on multi-thread

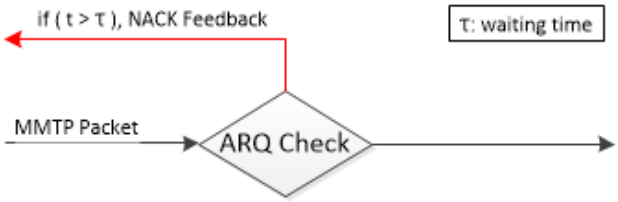


그림 6. 오류 제어를 위한 재전송 요청 판단 절차
Fig. 6. Decision process of retransmission request for error control

본 논문에서 구현한 재전송 요청 판단 절차의 방식은 그림 6과 같다. 수신 측에서는 모든 패킷의 전송 신뢰성을 확보하기 위해 수신된 각각의 패킷이 정상적으로 수신되었는지 확인한다. 만약 정해진 시간(waiting time)만큼 기다린 이후에도 MMT 패킷이 수신되지 않으면 패킷이 손실되었다고 판단하고 재전송 요청을 위한 NACK(Negative ACKnowledge) 피드백(feedback) 메시지를 MMT 시그널링 규격에 따라 송신 측으로 전송한다[1]. 이 때 오류가 발생한 패킷을 정상적으로 수신할 때까지 이 과정을 반복해서 수행한다.

패킷에 오류가 발생한 경우 해당 패킷을 전송하려는 Sending Unit과 Receiving Unit 쌍은 다른 패킷 전송을 중지하고 오류 제어를 위한 재전송을 한다. 따라서, 정상적인 패킷으로 프레임을 구성하는 시간이 프레임의 재생 시간보다 뒤로 밀리는 경우가 발생하고 이전에 큐에 저장되어 있던 프레임 데이터를 전부 사용해 큐가 고갈되면 결과적으로 영상이 정지하게 된다.

이처럼 단일쓰레드 환경에서 재전송 요청이 일어날 경우 영상 재생에 영향을 줄 수 있다. 따라서 재전송 요청이 일어나는 도중에도 정상적인 패킷을 사용해 프레임을 구성하고 재전송 받은 패킷은 추후에 프레임을 구성할 수 있도록 설계하는 것이 효율적인 방법이다. [10]에서 제안된 기술과 달리 본 논문에서는 각 쓰레드마다 대용량 멀티미디어를 전송하기 때문에 기존의 IP망으로 전송하는 것은 네트워크에 과부하를 유발시킬 수 있다. 피드백 채널에서의 메시지 전송도 이미 네트워크 사용량이 많은 환경에서는 부하가 될 수 있기 때문에 이러한 오류를 감안하여 별도의 피드백 채널을 구현하지 않았다. 이러한 환경을 고려하여 피드백 메시지 채널을 만드는 대신 수신 측에서는 일정 시간마다 수신 패킷을 확인하여 데이터 패킷을 수신하지 못할 경우에만 송신 측에 재전송 요구를 보냄으로써 네트워크에 과부하를 줄이는 방향으로 설계하였다. 이를 위해 본 논문에서 제안하는 바는 MMT 기반에서

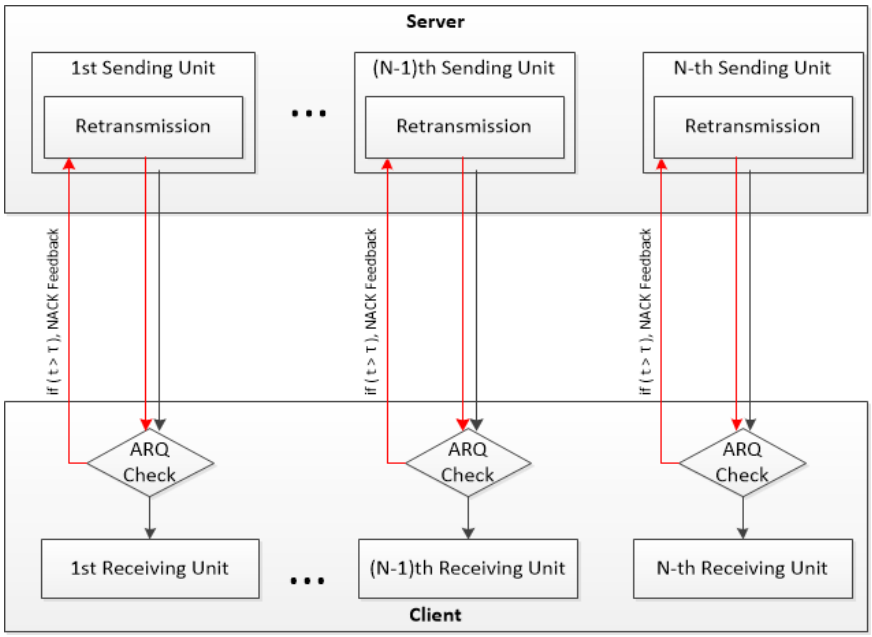


그림 7. 다중쓰레드 기반 ARQ 패킷 오류 제어 메커니즘
Fig. 7. ARQ packet error control mechanism based on multi-thread

Sending Unit과 Receiving Unit 쌍에 ARQ 모듈을 삽입해 다중채널(Multi-Channel)을 다중쓰레드와 동일하게 구성하여 각각 독립적으로 비디오 및 오디오 데이터를 주고받는 방식이다. 그림 7은 제안하는 다중쓰레드를 기반으로 ARQ 패킷 오류를 제어할 수 있는 오류 제어 메커니즘이다. 다중쓰레드를 통해 대용량 미디어의 전송 속도를 높임과 동시에 하나의 쓰레드에서 오류 제어를 위해 패킷을 재전송 받는 시간 동안 다른 쓰레드에서 정상적으로 패킷을 전송받게 함으로써 오류 제어 시 다중쓰레드 환경의 효율성을 높일 수 있다.

그림 8은 본 논문에서 제안하는 MMT 프로토콜 기반의 다중쓰레드를 활용한 ARQ 패킷 오류 제어 송수신 모델이다. MMT 프로토콜을 기반으로 송신 측인 서버와 수신 측인 클라이언트 사이에 다중쓰레드를 사용해 대용량 멀티미디어를 전송한다. 각각의 쓰레드마다 오류 제어를 위한 재전송 요청을 판단함으로써 일정 쓰레드가 오류 발생 패킷에 의한 재전송 기능을 수행할 때 나머지 쓰레드는 정상적인 패킷을 전송받아 프레임을 구성하기 때문에 수신 측에서는 끊김 현상 없이 영상을 재생할 수 있다.

이때 프레임을 구성하는 패킷에 오류가 발생해 이를 복구하기 위해 패킷을 재전송하는데 필요한 패킷 재전송 가능 시간 구간은 식 (1)과 같다.

현재 재생되는 프레임의 재생시간이 S이고 손실된 패킷이 포함된 프레임의 재생시간이 S'라고 하였을 때, 오류가 발생한 패킷이 포함된 프레임에서의 패킷 재전송 여유 시간 T는 다음의 구간 값을 만족해야 한다.

$$S < T < S' \tag{1}$$

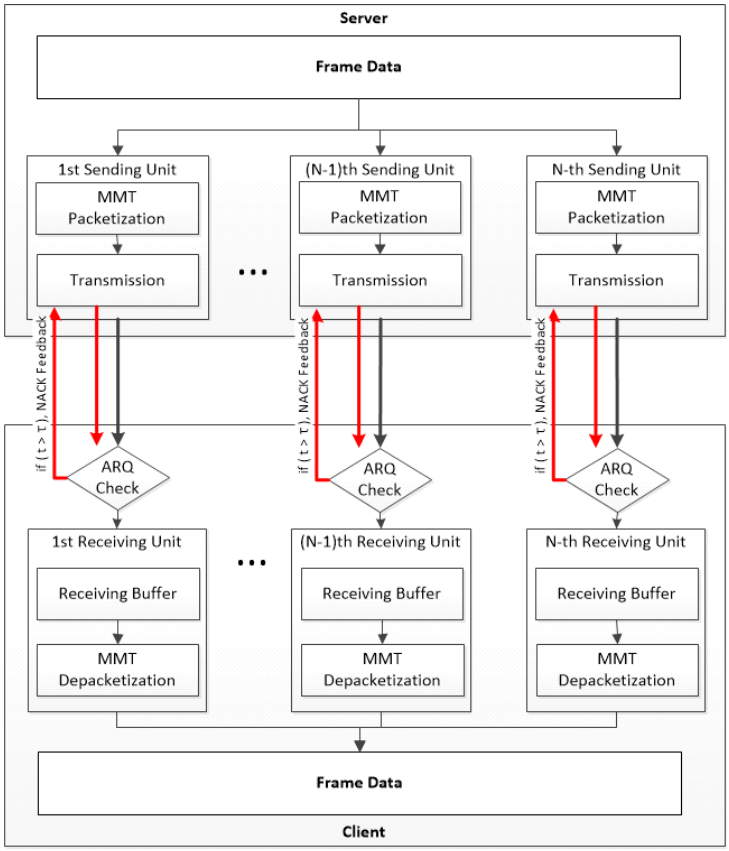


그림 8. 제안하는 MMT 프로토콜 기반의 다중쓰레드를 활용한 ARQ 패킷 오류 제어 송수신 모델
Fig. 8. Model of the ARQ packet error control transceiver based on MMT protocol

만약 T가 위 구간을 벗어난 값을 가질 경우 오류가 발생한 패킷을 재전송 받아 프레임을 구성하더라도 이미 프레임 재생 시간을 벗어났기 때문에 쓸모없는 프레임이 된다. 위의 경우 오류가 발생한 패킷이 포함된 프레임 대신에 정상적으로 구성된 직전 프레임 데이터를 그대로 사용함으로써 수신 측에서는 시각적인 화질 저하를 줄일 수 있다.

IV. 실험결과

본 논문에서 제안한 MMT 프로토콜 기반으로 프레임 데이터를 분할해 전송할 때 단일쓰레드와 다중쓰레드의 ARQ 오류 제어 기법의 효율성을 확인하기 위해 Microsoft에서 제공하는 'Network Emulator for Windows Toolkit'를 활용하였다[11][12]. 실험에서 임의의 패킷 손실률은 20%이고 MMT ARQ 모듈의 waiting time은 100ms로 설정하였으며 4K 해상도 영상 파일의 5초부터 20초 사이 구간에서 패킷을 손실시켜 다중쓰레드 사용에 따른 MMT ARQ 모듈의 복구 성능과 전송 속도에 미치는 영향을 실험을 통해 확인하였다. 본 실험에 앞서 앞 장에서 기술한 Sending Unit과 Receiving Unit 쓰레드 쌍을 Transport Unit이라 칭하였다.

그림 9는 Transport Unit이 하나인 단일쓰레드 상황에서 패킷이 손실되는 현상을 버퍼의 상태를 통해 확인한 그래프이다. 단일쓰레드가 정상적인 패킷을 수신하지 못하고 오류가 발생한 패킷에 대한 재전송을 계속해서 수행하기 때문에 패킷 손실이 발생하는 5초 부근부터 이전에 정상적인 패킷 수신과 프레임 구성을 통해 버퍼에 쌓여있던 프레임 데이터가 점차 줄어들어 7초 부근에서 버퍼에 쌓인 프레임의 개수가 0으로 수렴하는 것을 확인할 수 있다. 프레임 데이터가 0이 되는 것은 오류 발생 이전에 미리 저장한 프레임

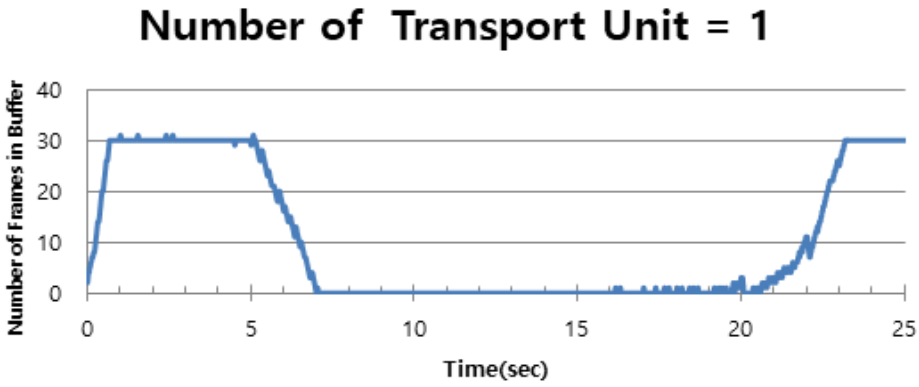


그림 9. 쓰레드가 1개인 경우, 패킷 손실에 따른 버퍼 상태 그래프
Fig. 9. Buffer status according to the packet losses (number_of_thread==1)

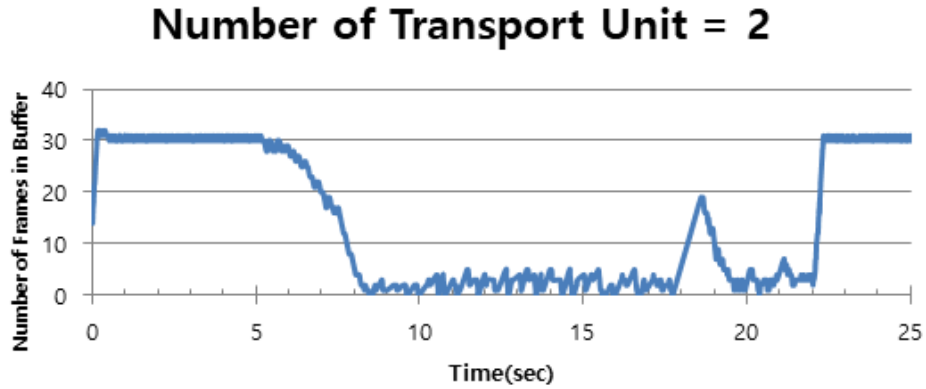


그림 10. 쓰레드가 2개인 경우, 패킷 손실에 따른 버퍼 상태 그래프.
Fig. 10. Buffer status according to the packet losses (number_of_thread==2)

데이터가 모두 소진되어 영상이 출력되지 않음을 뜻한다. 그림 9에서는 7초부터 버퍼에 데이터가 없기 때문에 영상 출력이 끊겨 보이게 된다. 이후 20초 부근에서 손실이 일어나지 않으면서 쓰레드가 패킷 재전송 대신 정상적으로 패킷을 수신해 프레임을 구성해 다시 버퍼에 프레임 데이터가 쌓이게 되고 영상이 정상 출력되는 모습을 확인할 수 있다.

그림 10은 쓰레드가 2개인 경우 패킷 손실에 따른 버퍼 상태를 나타낸다. 쓰레드가 1개일 때와 비슷하지만, 하나의 쓰레드를 추가적으로 사용함으로써 하나의 쓰레드가 그림 9와 같이 재전송에 사용되어 정상적인 패킷을 수신하지 못하더라도 나머지 쓰레드는 정상적으로 패킷을 수신하기 때문에 비디오 재생에 필요한 프레임을 구성할 수 있고 버퍼가 완전히 고갈되지 않는다. 그러나 각 쓰레드는 대용량의 멀티미디어 데이터 패킷을 전송 또는 필요에 따라 재전송하고 수신한 프레임을 구성하기 때문에 정상적인 프레임 재생 속도보다 데이터 전송과 프레임 구성에 시간이 더 소모될 경우 버퍼에 저장된 영상 프레임이 계속해서 소진되어 그림 9의 실험 결과와 같이 버퍼에 저장된 프레임 데이터의 량이 0이 될 수 있고, 이 경우 영상이 끊기는 상황이 발생한다.

그림 11과 12는 각각 쓰레드가 4개와 8개인 경우 패킷 손실에 따른 버퍼 상태를 나타낸다. 그림 11은 오류 발생 구간에서 잠깐씩 버퍼에 저장된 프레임 데이터의 개수가 줄어들지만 그림 9나 그림 10과 같은 구간에서 패킷 재전송

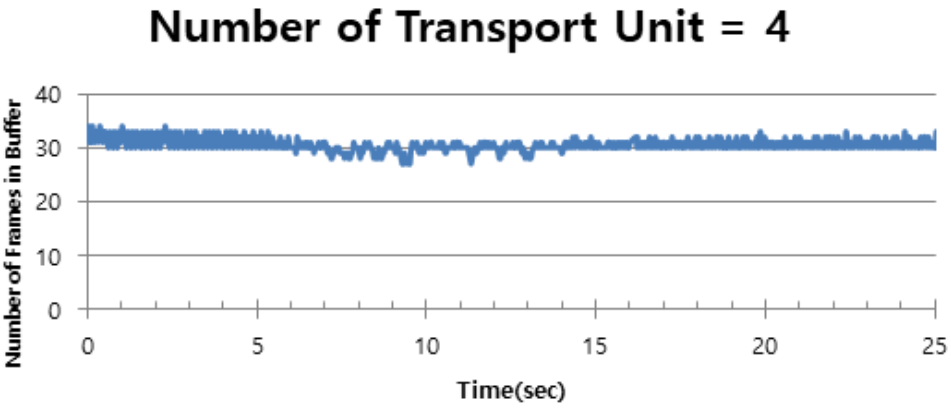


그림 11. 쓰레드가 4개인 경우, 패킷 손실에 따른 버퍼 상태 그래프
Fig. 11. Buffer status according to the packet losses (number_of_thread==4)

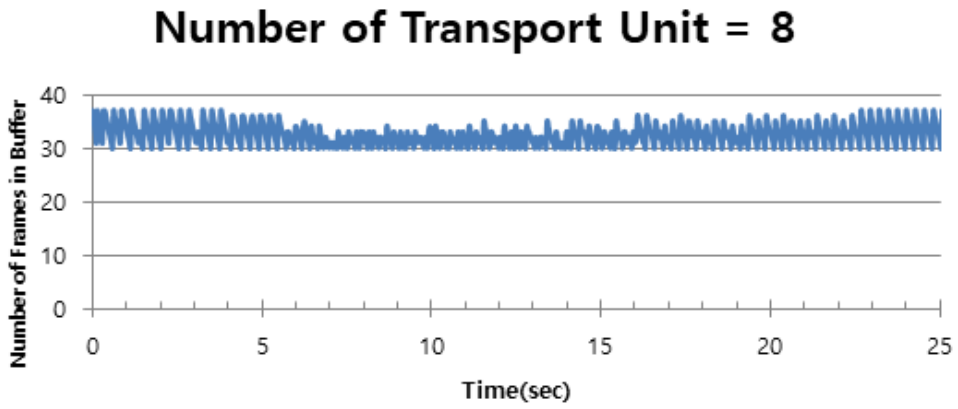


그림 12. 쓰레드가 8개인 경우, 패킷 손실에 따른 버퍼 상태 그래프
Fig. 12. Buffer status according to the packet losses (number_of_thread==8)

이 일어났을 때, 재전송 기능을 하는 쓰레드보다 정상 패킷을 수신하여 프레임을 구성하는 쓰레드가 많기 때문에 프레임 데이터를 저장하는 버퍼가 줄어드는 현상이 현저하게 감소하는 것을 확인할 수 있다. 만약 네트워크의 상태가 좋지 않아 여러 쓰레드에서 동시에 패킷 오류가 발생하는 경우가 아니라면 프레임 데이터가 저장된 버퍼가 고갈되는 상황이 발생하지 않는다.

위의 실험과 같이 쓰레드 개수가 증가하면 오류 발생 구간에서 버퍼에 저장된 프레임 데이터가 감소하는 기울기가 점점 줄어들고 오류 발생 구간이 지났을 때 다시 정상적으로 버퍼에 프레임 데이터가 저장되는 기울기는 점점 가파르게 변하는 것을 확인할 수 있다.

이상의 실험 결과를 통해 MMT 기반으로 대용량 멀티미디어 전송 시 패킷 오류가 발생했을 때, 한 번에 한 가지 기능을 수행하는 단일쓰레드의 단점을 보완하여 오류 발생 구간에서는 영상 재생에 필요한 프레임 데이터가 저장된 버퍼가 고갈되는 속도를 늦추고, 오류 발생 이후 구간에서는 버퍼에 저장되는 정상적인 프레임 데이터의 개수를 빠르게 복구하는 다중쓰레드를 활용한 ARQ 패킷 오류 제어의 효율성을 확인할 수 있다.

V. 결 론

본 논문에서는 MMT 프로토콜 기반의 다중쓰레드를 활용한 ARQ 패킷 오류 제어 기법에 대해 제안하였다. 제안된 다중쓰레드를 활용한 패킷 오류 제어는 네트워크 상황이 좋지 않아 단일쓰레드 환경에서 패킷 전송 시 오류가 발생할 때 쓰레드가 재전송에 사용됨으로써 정상적인 패킷을 수신하지 못해 영상의 재생 시 발생할 수 있는 문제를 해결하였다. 즉, 다중쓰레드를 사용하여 재전송에 사용되는 쓰레드를 분리 구현함으로써 제안하는 방식의 효율성을 확인하였다. 또한, 실험 결과를 통해 단일쓰레드에서는 패킷 오류 발생 시 재전송으로 인해 영상의 재생에 필요한 프레임 데이터가 저장된 버퍼가 고갈되어 영상의 재생이 끊기는 문제가 있음을 확인하였고, 제안하는 다중쓰레드를 사용함으로써 재전송시 버퍼가 고갈되는 속도를 늦추고 오류가 발생하지 않으면서 정상적인 패킷의 수신을 통해 버퍼가 복구되는 시간을 줄일 수 있음을 확인하였다.

참 고 문 헌 (References)

[1] ISO/IEC 23008-1:2014 (First edition), Information technology High efficiency coding and media delivery in heterogeneous environments Part 1: MPEG media transport (MMT), Jun. 2014.

[2] Y. Sohn, M. Cho, and J. Paik, “Design of MMT-based broadcasting system for UHD video streaming over heterogenous networks,” Journal of Broadcast Engineering, vol. 20, no 1, pp. 16-25, Jan. 2015.

[3] S. Aoki, K. Otsuki, and H. Hamada, “Effective usage of MMT in broadcasting systems,” in Proc. of IEEE Int. Symp. Broadband Multimedia Systems and Broadcasting (BMSB), pp. 1-6, 2013.

[4] T. Jung, H. Lee, and K. Seo, “Overview on MPEG MMT technology and its application to hybrid media delivery over heterogeneous networks,” In Proc. of Pacific Rim Conference on Multimedia, pp. 660-669, 2015.

[5] S. Cho, J. Lee, and K. Park, “Low delayed mobile live streaming method and its implementation,” In Proc. of IEEE Int. Conf. Multimedia and Expo Workshops (ICMEW), pp. 1-3, 2015.

[6] Y. Hu, S. Xie, and Y. Xu, “Dynamic VR live streaming over MMT,” in Proc. of IEEE Int. Symp. Broadband Multimedia Systems and Broadcasting (BMSB), pp. 11-16, 2017.

[7] A. Silberschatz, “Operating System Concepts 8th Edition,” Hongnung Publishing Company, pp.169-172, Feb. 2013.

[8] D. Kang and H. Park, “A design and implementation of transmit/receive model to speed up the transmission of large string-data sets in TCP/IP socket communication,” Journal of Korea Institute of Information and Communication Engineering, vol. 17, no. 4, pp. 885-892, Apr. 2013.

[9] J. Jeong, et al. “Performance comparison method for multi-core based application software architecture alternatives,” Journal of KIISE: Software and Applications, vol. 39, no 1, pp. 1-11, Jan. 2012.

[10] C. Kim, et al. “An efficient delay-constrained ARQ scheme for MMT packet-based real-time video streaming over IP networks,” Journal of Real-Time Image Processing, vol. 12, no. 2, pp. 257-271, 2016.

[11] ISO/IEC PDTR 23008-13, Information technology-high efficiency coding and media delivery in heterogeneous environments Part 13: MPEG media transport implementation guidelines.

[12] Available at http://network-emulator-for-windows-toolkit.software.informer.com/.

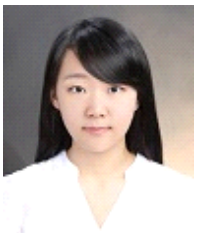
저 자 소 개

원 광 은

- 2016년 2월 : 연세대학교 컴퓨터정보통신공학부 학사
- 2018년 8월 : 연세대학교 전산학과 석사
- ORCID : https://orcid.org/0000-0002-6370-0608
- 주관심분야 : 영상통신, 멀티미디어 통신시스템

안 은 빈

- 2016년 8월 : 연세대학교 컴퓨터정보통신공학부 학사
- 2017년 3월 ~ 현재 : 연세대학교 전산학과 석박통합과정
- ORCID : https://orcid.org/0000-0001-7681-4682
- 주관심분야 : 영상처리, 영상통신, 360/VR 영상



김 아 영

- 2016년 2월 : 연세대학교 컴퓨터정보통신공학부 학사
- 2016년 3월 ~ 현재 : 연세대학교 전산학과 석박통합과정
- ORCID : https://orcid.org/0000-0002-3793-1365
- 주관심분야 : 영상통신, 실시간 스트리밍, 360/VR 영상

이 홍 래

- 2010년 2월 : 연세대학교 컴퓨터정보통신공학부 학사
- 2012년 8월 : 연세대학교 전산학과 석사
- 2012년 9월 ~ 현재 : 연세대학교 전산학과 박사과정
- ORCID : https://orcid.org/0000-0001-6985-6016
- 주관심분야 : 영상부호화, 영상통신, 멀티미디어 통신 프로토콜

서 광 덕

- 1996년 2월 : KAIST 전기 및 전자공학과 학사
- 1998년 2월 : KAIST 전기 및 전자공학과 석사
- 2002년 8월 : KAIST 전기 및 전자공학과 박사
- 2002년 8월 ~ 2005년 2월 : LG전자 단말연구소 선임연구원
- 2012년 9월 ~ 2013년 8월 : Courtesy Professor, Univ. of Florida, USA
- 2005년 3월 ~ 현재 : 연세대학교 컴퓨터정보통신공학부 교수
- ORCID : http://orcid.org/0000-0001-5823-2857
- 주관심분야 : 영상부호화, 영상통신, 디지털 방송, 멀티미디어 통신시스템