

# MPEG-2 기반영상의 시스템 영역에서의 편집을 위한 PES 영역에서의 스트림 splicing에 관한 연구(비교, 분석)

김 동 준, 최 윤 식  
연세대학교 전기전자 공학과  
전화 : 02-2123-2774 / 핸드폰 : 019-658-1127

## A study of PES layer splicing for system layer editing on MPEG-2 based images(compare & analysis)

Dong-joon Kim, Yoon-sik Choe  
Dept. of Electrical and Electronic Engineering, Yonsei University  
E-mail : eastnent@hanmail.net

### Abstract

In this paper, We have studied for guaranteeing the clear display of MPEG-2 video sequence when conduct splicing of MPEG-2 system streams. we focus on the PES domain splicing considering video sequence. And we wish to make a base on the TS or PS domain splicing considering video sequence.

For that, first, we compared and analyzed problems that is raised when different two PES streams are spliced and effects that affect the video sequence. And based on this analysis, we have searched for methods that resolve the cause of problems that can be happened in the display of video sequence directly in PES domain.

### I. 서론

본 논문은 MPEG-2 시스템 스트림 영역에서의 스플라이싱이 MPEG-2 비디오 시퀀스의 예러 없는 디스플레이를 보장할 수 있는 방법을 연구한다.

기존의 스플라이싱은 MPEG-2 시스템의 TS 영역에서 이루어지도록 하고 있는데, 이렇게 함으로써 전송 중 실시간 스플라이싱이 가능하게 된다. 하지만, 이 스플라이싱은 비디오나 오디오 영역보다 상위인 시스템

영역에서 이루어지는 것이기 때문에, 비디오 문제로 한정시킨 영상편집의 관점에서 보면, 화면의 깨짐이 없는 임의의 픽처에서의 접합을 보장할 수 없었다.

본 논문에서는 MPEG-2 시스템의 PES 영역에서의 비디오 시퀀스를 고려한 스플라이싱을 다룸으로서 상위의 TS와 PS에서 동일한 목적의 스플라이싱을 수행할 수 있도록 하는 연구의 기반을 마련하고자 하였다.

그러기 위해서 먼저, 각각 다른 두 PES 스트림의 접합 시 발생하는 문제와 이로 인해 비디오 시퀀스에 미치는 영향을 비교, 분석하였고, 이러한 비교, 분석을 토대로, PES 영역에서 직접 비디오 시퀀스의 디스플레이에 일어날 수 있는 문제의 원인을 해결하는 방안을 모색하였다.

그리고, PES 패킷의 헤더 정보를 이용하면, PES 패킷의 상태와 관계없이 GOP 단위의 스플라이싱을 보장할 수 있음을 확인하였다.

### II. MPEG-2 시스템

#### 2.1 MPEG-2 시스템 스트림의 구조

MPEG-2 시스템 파트는 시스템 코딩에 관한 전반적인 분야의 권고안이다. 여기서 시스템 코딩이란 압축된 오디오나 비디오 스트림 뿐만 아니라 필요에 따라

사용자 데이터를 다중화하여 전송 혹은 저장에 적합하도록 만드는 포장(data formatting)에 관한 것이다.[1][4] 시스템 스트림이 형성되는 과정을 도시한 블록 다이어그램은 다음과 같다.[1]

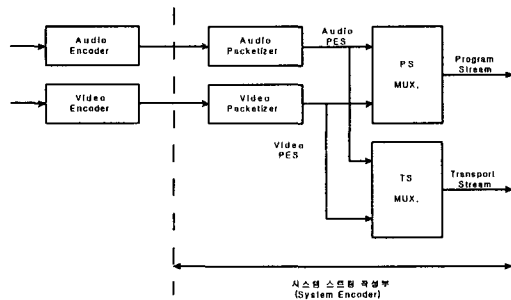


그림 1 MPEG-2 시스템 인코더의 개요

시스템 인코더의 입력 스트림들은 각각 packetizer에 의해 PES 패킷이라는 형태로 패킷화 된다. 그리고 이렇게 패킷화된 각각의 PES 패킷들은 다중화되어 사용 목적에 따라 TS 스트림과 PS 스트림이라는 두 가지 형태의 시스템 스트림으로 복호화 된다.

### 2.2 Packetized Elementary Stream(PES)

PES 패킷의 계층적 구조는 다음 그림과 같다.[1]

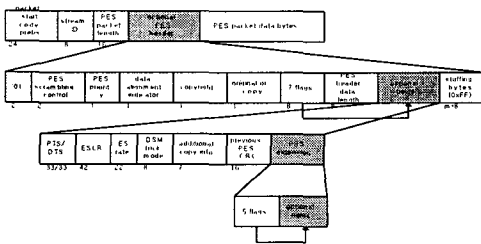


그림 4 PES packet syntax diagram

PES 패킷은 헤더와 유효부하로 구성되며 헤더에는 현재 패킷의 유효부하에 있는 데이터의 특성 및 전체 스트림을 구성하는 데이터와의 관계를 나타내는 정보 및 시간 정보 또 유효부하의 길이 정보 등이 있다.

그리고 유효부하는 elementary stream으로 채워지는데, 한 PES 패킷의 유효부하에는 여러 개의 access unit이 있을 수 있으며, 또 한 개의 access unit이 여러 개의 PES 패킷으로 나뉘어져 들어갈 수도 있다. 보통, 비디오의 access unit은 그 크기가 크기 때문

에 여러 개의 PES 패킷으로 나뉘어진다.[4]

### III. PES 스트림 스플라이싱을 위한 스트림의 형성

PES 스트림 영역에서의 비디오 스트림 스플라이싱은 서로 다른 비디오 시퀀스 데이터를 유효부하로 가진 두 개의 PES 스트림을 가정한다. 여기서 접합 부분의 앞부분에 오게 될 PES 스트림을 head 스트림, 뒷부분에 오게 될 스트림을 tail 스트림이라고 칭한다.

#### 3.1 Head 데이터 스트림의 형성

PES 영역에서의 비디오 스플라이싱을 위한 head 데이터 스트림을 형성할 때, PES 스트림이 잘려지는 형태는 완전한 PES 단위로 잘려지는 경우, PES의 유효부하의 중간지점에서 잘려지는 경우, PES 헤더부분만 남겨놓고 잘려지는 경우의 3가지로 분류될 수 있다. 하지만, 유효부하인 비디오 스트림의 관점에서 보면 PES 헤더만 남겨지는 경우를 제외하고는, 모두 픽처 단위에서 잘려지는 경우와 픽처 데이터 스트림의 중간부분에서 잘려지는 경우로 생각할 수 있다.

이 논문에서는 궁극적으로 비디오 시퀀스의 스플라이싱을 다루는 것이기 때문에, PES 영역에서는 완전한 PES 단위로 잘려지는 경우와 PES의 유효부하의 중간지점에서 잘려지는 경우만을 다루고, 또 비디오 스트림의 영역에서는 픽처 데이터 단위에서 잘려지는 경우만을 다루기로 한다.

PES의 유효부하에서 픽처 단위로 잘려지는 경우를 생각할 때, Head 스트림의 형성에서 PES 패킷이 완전한지 유효부하 중간부분에서 잘렸는지는 중요하지 않다. 왜냐하면, head 스트림을 만들기 위해 스트림을 절단할 때는 뒤에서부터 잘라내야 하므로 PES 패킷 헤더가 손상을 입지 않기 때문에 패킷의 형태가 유지되기 때문이다. 따라서, 여기서는 유효부하의 관점에서만 생각하기로 한다.

만약 B 픽처를 포함한 유효부하 뒤에서 잘려졌다면, 이 B 픽처의 참조픽처가 모두 앞부분의 PES 유효부하에 존재하므로, 문제가 발생하지 않는다. closed GOP로 설정된 GOP의 I 픽처 뒤에서 잘려진 경우도 역시 문제가 발생하지 않는다. 하지만, 개방된 GOP의 I 픽처 뒤에서 잘려진 경우나, 임의의 GOP의 P 픽처 뒤에서 잘려진 경우는 그 픽처들을 역방향 참조하게 되는 B 픽처들이 discard 되는 경우이므로, discard된 B 픽처의 개수만큼 프레임이 생략되는 결과가 나타난다.

### 3.2 Tail 데이터 스트림의 형성

PES 영역에서의 비디오 스플라이싱을 위한 tail 데이터 스트림을 형성할 때, PES 스트림이 잘려지는 형태는 완전한 PES 단위로 잘려지는 경우, PES 헤더가 손상되게 잘려지는 경우, 유효부하의 중간부분에서 잘려지는 경우의 3가지로 분류될 수 있다. 하지만, 유효부하인 비디오 스트림의 관점에서 보면 head 스트림의 경우와 마찬가지로, 픽처 단위에서 잘려지는 경우와 픽처 데이터 스트림의 중간부분에서 잘려지는 경우로 생각할 수 있다.

여기서는 PES 영역에서 완전한 PES 단위로 잘려지는 경우와 PES의 유효부하의 중간지점에서 잘려지는 경우, 그리고 헤더가 손상되어 유효부하만 있는 경우를 다루고, 또 비디오 스트림의 영역에서는 픽처 데이터 단위에서 잘려지는 경우만을 다루기로 한다.

이제 tail 스트림이 완전한 PES 패킷 단위로 잘려지고, PES의 유효부하에서는 픽처 단위로 잘려지는 경우를 생각해 보자.

만약 GOP 헤더의 closed\_gop가 1(closed GOP)일 때, GOP 헤더와 I 픽처를 포함한 유효부하 앞에서 잘려졌다면, 아무런 문제가 발생하지 않는다. 하지만, 개방된 GOP의 I 픽처 앞에서 잘려졌다면 그 I 픽처 뒤에 따라오는 B 픽처가 forward prediction을 할 수 있는 참조픽처가 없기 때문에 문제가 발생할 것이다. 그리고 임의의 GOP의 B 픽처나 P 픽처의 앞에서 잘려지는 경우에도, I, P 픽처같은 참조픽처가 없어지는 것이기 때문에 그 GOP 전체에 영향을 주게 된다.

Tail 스트림의 앞부분이 손상된 상태로 잘려져서 PES 패킷 헤더가 손상된 경우에는, 그 패킷의 유효부하가 비록 GOP 헤더를 포함하고 있어도 심각한 문제가 발생한다. PES 패킷 헤더가 없으면 PES 디코더에서 그 패킷을 인식하지 못하게 되기 때문에 그 유효부하도 자동적으로 discard 되기 때문이다. 이 때에도 결국, 한 GOP 전체에 대해서 에러가 발생하게 된다.

## IV. PES 시스템 영역에서의 비디오 스트림 스플라이싱

PES 시스템 영역에서 비디오 스트림을 고려하여 스플라이싱을 하려고 할 때 부딪히는 문제들은 보통 두가지의 관점에서 생각해 볼 수 있다. 그 중 하나는 PES 패킷 자체의 복호화에서 발생하는 문제로 볼 수 있고, 또 다른 하나는 PES 패킷들의 유효부하인 픽처들 사이의 관계에 의한 문제. 즉, elementary stream

상에서 발생하는 문제로 볼 수 있다. 하지만, 여기서는 PES 패킷 자체의 디코딩에서 발생하는 문제를 해결하는데 초점을 둔다. 그리고 접합부분의 유효부하는 GOP 단위가 보장되는 것을 가정한다.

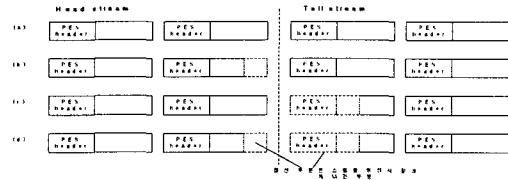


그림 4-1 PES 영역에서의 스플라이싱의 형태

- (a) 완전한 PES 패킷단위로 접합되는 경우
- (b) Head 스트림의 PES 패킷이 중간부분에서 잘린 경우
- (c) Tail 스트림의 처음 PES 패킷의 앞부분이 잘린 경우
- (d) (b),(c)가 복합된 경우

### 4.1 Tail 스트림이 PES 헤더로 시작되는 경우

이 경우에는 head 스트림의 마지막 부분이 완전한 PES 패킷의 형태로 끝났을 때, 스플라이싱에서 거의 문제가 발생하지 않는다. 하지만, head 스트림의 마지막 부분이 완전한 PES 패킷을 이루지 못하고, PES 패킷의 유효부하 중간에서 잘려졌다면 PES 디코더에서의 복호화시 심각한 문제가 발생할 수 있다. 그리고 이것은 비디오 시퀀스의 디스플레이에 직접적인 영향을 미치게 된다.

이러한 문제가 발생하는 이유는, PES 디코더에서 PES 패킷의 PES\_packet\_length의 값을 체크하고, 그 정보를 통해 패킷의 길이를 가정하기 때문이다. 이러한 PES 디코더의 동작원리 때문에 스플라이싱된 PES 스트림이 PES 디코더를 거치면서 비디오 시퀀스에 관련된 중요한 유효부하 정보를 잃어버리게 될 수 있다. 그리고 결국 비디오 시퀀스에 손상을 입히게 된다.

이러한 문제를 해결하기 위해서는 PES 디코더에서 정보의 손실 없이, 즉 discard가 이루어지는 부분이 없이 PES 패킷의 복호화가 이루어지도록 할 필요가 있다. 이를 위해서 PES 패킷 헤더의 PES\_packet\_length를 discard되는 부분이 없도록 조절해주는 방법을 제안한다.

스플라이싱이 일어나는 부분인 head 스트림의 마지막 PES 패킷과 tail 스트림의 첫 번째 PES 패킷이 디코더에서 각각 별개의 패킷으로 인식될 수 있도록 다음 식과 같이 head 스트림의 마지막 PES 패킷의 PES\_packet\_length값을 조절해준다.

$$PES\_packet\_length' = \text{기존의 } PES\_packet\_length \text{의 값} -$$

head 스트림의 마지막 패킷에서 잘려나간 부분의 바이트 수  
(식 1)

4.2 Tail 스트림이 유료부하로 시작되는 경우

이 경우는 head 스트림의 마지막 PES 패킷이 온전하더라도, tail 스트림의 첫 번째 PES 패킷의 헤더가 손상되어 그 패킷 전체가 인식되지 않기 때문에 문제가 발생한다.

이 문제를 해결하기 위해서는 4.1절에서와 마찬가지로 PES 디코더에서 정보의 손실 없이, PES 패킷의 복호화가 이루어지도록 할 필요가 있다.

여기서는 head 스트림의 마지막 PES 패킷 헤더의 PES\_packet\_length 값을 식 2나 식 3과 같이 증가시켜 줌으로서 PES 디코더가 tail 스트림의 첫 번째 PES 패킷의 유료부하를 head 스트림의 마지막 PES 패킷의 유료부하로써 인식하게 만들어주어야 한다. 이로써, tail 스트림의 유료부하는 보존된다.

$PES\_packet\_length' = \text{기존의 } PES\_packet\_length \text{의 값} + \text{tail 스트림의 첫 번째 패킷에서 남아있는 부분의 바이트 수}$   
(식 2)

위의 식 2는 head 스트림의 마지막 PES 패킷이 온전한 경우를 전제한 것이다. Head 스트림의 마지막 PES 패킷의 유료부하 중간부분이 잘려나간 경우에는 다음과 같이 계산된다.

$PES\_packet\_length' = \text{기존의 } PES\_packet\_length \text{의 값} - \text{head 스트림의 마지막 패킷에서 잘려져 나간 부분의 바이트 수} + \text{tail 스트림의 첫 번째 패킷에서 남아있는 부분의 바이트 수}$   
(식 3)

표 1 스플라이싱의 집합 형태에 따른 문제점

Head Str. / Tail Str.	완전한 PES 패킷	PES 패킷 유료부하 중간에서 절단
완전한 PES 패킷	문제없음	Tail 스트림 일부 discard -> GOP의 참조픽처에 문제 발생
PES 패킷 헤더 손상	Tail 스트림 일부 discard -> GOP의 참조픽처에 문제 발생	Tail 스트림 일부 discard -> GOP의 참조픽처에 문제 발생

표 2 집합 형태에 따른 문제점 해결방안

Head Str. / Tail Str.	완전한 PES 패킷	PES 패킷 유료부하 중간에서 절단
완전한 PES 패킷		Head 스트림의 PES_packet_length 감소
PES 패킷 헤더 손상	Head 스트림의 PES_packet_length 증가	Head 스트림의 PES_packet_length 감소 / 증가

V. 결론

본 논문에서는 MPEG-2 시스템 영역 중 PES 스트림 상에서 비디오의 seamless 스플라이싱을 구현하는 방법에 대해 고찰해 보았다.

본 연구에서는 TS나 PS보다는 하위의 시스템 영역인 PES 영역을 전제로 스플라이싱 작업을 수행하였으며, 그 결과 PES 패킷의 헤더 정보를 이용하면, PES 패킷이 어느 정도 손상된 상태에서도 tail 스트림 유료부하의 GOP 헤더부터 온전하다면 GOP 단위의 비디오 스트림 스플라이싱을 끊임없이 보장할 수 있음을 확인하였다.

이로써, PES 스트림 자체의 손상에 의한 비디오 스트림의 문제가 실시간으로 어느 정도 해결될 수 있음을 확인하였다.

참고문헌

- [1] ITU-T Recommendation H.222.0, "Information Technology - Generic Coding of Moving Pictures and Associated Audio Information : Systems", July, 1995
- [2] ITU-T Recommendation H.262, "Information Technology - Generic Coding of Moving Pictures and Associated Audio Information : Video", July, 1995
- [3] Janusz Konrad, "Visual Communication of Tomorrow : Natural, Efficient and Flexible", IEEE communications Magazine, VOL. 31, NO. 1, pp 126-133, Jan, 2001
- [4] 유시룡, 장규환, 이병욱, 김종일, 정해목, "MPEG 시스템", 대영사, 2000. 2. 10
- [5] Susie J. Wee, Bhaskaran Vasudev, "Splicing MPEG Video Streams in the Compressed Domain", IEEE, June 1997