화면 전환 효과를 위한 비디오 편집 시스템

김 정 훈†

요 약

본 논문에서는 LDP, VCR, Camcoder 등으로부터 아날로그 신호를 입력받아 디지탈 데이타로 변환한 후 이를 원하는 형태로 편집하여 다시 아날로그 형태로 변환시키는 시스템을 설계 및 구현하고자 한다. 특히 본 논문에서는 비디오 데이타를 편집할 때 유용하게 쓰일 수 있는 다양한 화면 전환 효과 알고리즘을 제시하며 일부는 이미 개발되어 있는 단일 이미지 처리 알고리즘을 응용한다. 마지막으로 기존의 전문 비디오 편집 시스템들과 본 편집 시스템과의 성능을 비교 분석해 본다.

A video editing system for transition effects

Jeong Hoon Kim†

ABSTRACT

In this paper, I tried to take analog signal from LDP, VCR, Camcoder and to convert to digital data. Then I designed and implemented a system to edit the data and to reconvert the edited data to analog signal. For this study, I partially used single image processing algorithms which were developped before my work and suggested algorithms of various transition effects which could be useful for video data editing. In conclusion, I compared the conventional professional video systems with mine and analyzed them.

1. 서 론

최근들어 멀티미디어 기술과 통신 기술의 발전으로 키오스크(Kiosk), 화상회의, 멀티미디어 데이타를 이용한 인터넷 서비스 등의 분야가 실생활에 응용되고 있으며 현재는 일반인들까지도 이러한 기술의 혜택을 받고 있다[7, 9]. 키오스크, 화상회의, 인터넷 서비스 등의 응용에 사용되는 멀티미디어 데이타는 그 전달 효과가 커서 기존의 텍스트를 기본으로 하는 응용보다 많은 이점이 있으나 큰 저장 공간과 처리 시간이 길다는 단점 때문에 일반화되기에는 많은 어려움이 있었다. 그러나 CD-ROM과 같은 저장 매체의 발달과 컴퓨터의 처리 능력 향상으로 이러한 문제점이 상당 부분 해결되었으며 이제는 그 매체를 처리할 수 있는 도구 개발이 필요하게 되었다. 현재 멀티미디어 데이타를 처리할 수 있는 도구들은 많이 개발되어 있으나 비디오 매체를 다룰 수 있는 도구들은 흔하지 않다. Adobe사의 Premiere와 Ulead사의 Video studio가 비디오를 처리할 수 있는 도구로 알려져 있으나 이는 모두 외국 제품이며 국내에서 개발된 것은 거의 전무한 상태이다[2, 12, 18]. 이외의 대부분의 도구들은 단순히 비디오를 재생하는데 중점을 두었기 때문에 편집에는 한계가 있으며 화면전환 효과와 같이 시각적 효과를 부각시킬 수 있는 비디오 고유의 효과는 지원하지 못하는 단점이 있다. 물론 비디오 화면 전환 효과들은 이미 방송에서 보편화되어 있지만 이는 아날로그 데이타를 편집하는 것으로서 디지탈 데이타를 편집하는 것은 아니다. 그러나 아날로그 데이타가 아닌 디지탈 데이타로서 비디오 편집이 가능해진다면 다음과 같은 이점을 얻을 수 있다[16].

(1) 임의 접근 방식으로 빠른 탐색이 이루어질수 있다.

† 정 회 원 : 현대전자(주) 소프트웨어연구소 주임연구원
논문접수 : 1995년 8월 16일, 심사완료 : 1995년 12월 12일

(2) 데이타의 손상없이 무한 복제가 가능하다.
(3) 여러가지 다양한 방법으로 편집이 가능하다.
(4) 적은 노력으로 대화형식의 비디오 프리젠테이션 시스템을 만들 수 있다.
(5) 각 영상을 용이한 방법으로 관리가 가능해진다.
(6) 컴퓨터 환경에서 조작할 수 있다.

본 논문에서는 LDP, VCR, Camcoder 등으로부터 아날로그 신호를 입력받아 디지탈 데이타로 변환한 후 이를 원하는 형태로 편집하여 다시 아날로그 신호로 변환시키는 시스템을 설계 및 구현하고자 한다. 물론 이미 존재하는 디지탈 데이타를 입력받을 수 있으며 편집된 디지탈 데이타를 출력결과로 얻을 수 있다. 특히 본 논문에서는 비디오 데이타를 편집할 때 유용하게 쓰일 수 있는 다양한 화면 전환 효과 알고리즘을 제시하며 일부는 이미 개발되어 있는 단일 이미지 처리 알고리즘을 응용한다. 마지막으로 본 논문에서 제시한 알고리즘을 사용하여 편집 시스템을 직접 구현해 보고 타 시스템과 비교하여 그 유용성을 살펴본다.

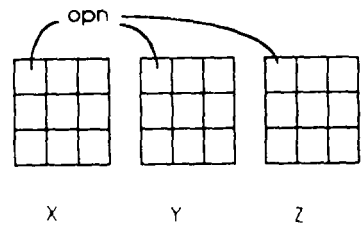
2. 이미지 처리 효과 알고리즘

2.1 이미지 대수(Image Algebra)

한 이미지를 컴퓨터에서 표현할 때 그 표현 방식은 숫자이며 수학적인 연산을 행할 수 있다. 이미지에 대한 대수학적인 연산(algebraic operation)은 점대응(pointwise) 연산과 행렬(matrix) 연산 두 가지 형태가 있으며 다음과 같이 표현될 수 있다[8, 10].

$$X \text{ opn } Y = Z \qquad (1)$$

(1) 식에서 X와 Y는 이미지 또는 스칼라이며 Z는 이미지가 된다. X와 Y가 이미지이면 같은 차수 또는 같은 공간 해상도를 가져야 한다. 점대응 연산일 경우, opn은 X와 Y에 각각 대응되는 요소들의 연산을 의미한다. 이를 이해하기 쉽게 그림으로 표현하면 다음과 같다.



(그림 1) 점대응 연산
(Fig. 1) Pointwise operation

이러한 이미지 대수는 다음과 같이 3가지로 나누어 생각해 볼 수 있다.

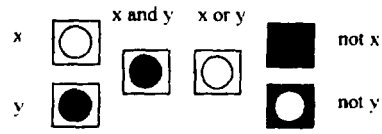
(1) 산술 연산 (Arithmetic Operations)
(2) 불리안 연산 (Boolean Operations)
(3) 행렬 연산 (Matrix Operations)

2.2 산술 연산

이미지간의 산술 연산은 더하기(addition), 빼기(subtraction), 곱하기(multiplication), 나누기(division) 연산을 의미한다. (1)식에서 opn은 원하는 산술 연산으로 대치될 수 있다. 각 픽셀 값은 0에서 255까지의 값을 갖는다고 할 때 이 범위를 벗어나는 픽셀 값은 적절한 값으로 대치되어야 한다. 더하기 연산은 X와 Y의 각 픽셀 값의 합을 의미하며 밝아진 혼합된 이미지가 생성된다. 만약 Y가 이미지가 아닌 스칼라이면 Z는 X 보다 더 밝아진다. 반대로 빼기 연산은 X와 Y의 각 픽셀 값의 차를 의미하며 Y가 스칼라이면 Z는 X 보다 더 어두워진다. 곱하기, 나누기 연산은 대응되는 각 픽셀 값의 곱하기와 나누기 연산을 의미한다.

2.3 불리안 연산

수행될 수 있는 불리안 연산자는 AND, OR, NOT 등이다. 이 세가지를 조합해서 얻어낼 수 있는 불리안 함수로는 NAND, NOR, XOR 등이 있다. 불리안 연산은 이미지 데이타가 이진(bi-



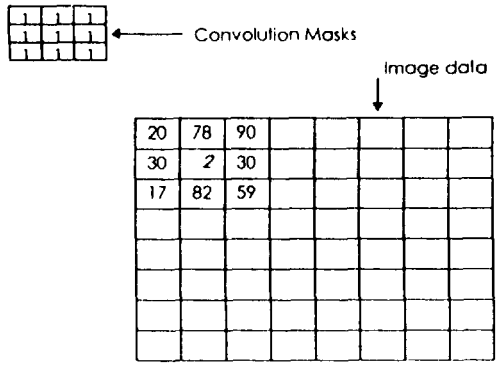
(그림 2) 이진 형태의 불리언 연산
(Fig. 2) Binary boolean operation

nary) 형태일 때 이해하기 쉬운데 간단한 예는 (그림 2)와 같다.

AND 연산을 사용하여 이미지를 어둡게 만들 수 있으며 OR 연산으로 이미지를 밝게 만들 수 있다.

2.4 행렬 연산

더하기 연산이나 빼기 연산일 경우 행렬 연산은 위의 산술 연산과 같다. 그러나 곱하기 연산일 경우에는 산술 연산과 달리 행렬 연산은 벡터곱(vector product)이 된다. 한 이미지에 효과를 부여하기 위해 가해지는 행렬을 회선 마스크(convolution masks) 또는 회선 행렬이라 하는데 이 행렬은 다양한 공간 처리 변환을 위해 사용된다. 회선 행렬이 실제로 이미지에 적용되는 예를 다음 (그림 3)에서 보였다. 사용되는 행렬은 이미지의 내용을 흐릿하게 만들기 위한 행렬이며 3x3으로 가정하였다. 현재 처리되고 있는 픽셀 값이 이미지 행렬의 (1,1) 요소일 때 2라는 값은 (2)식에 의해 주변의 값들과 비슷한 값으로 변화된다. 이러한 회선 행렬에 의해 전체적인 이미지는 흐릿해질 수 있으며 또 다른 회선 행렬을 사용하여 이미지의 선명도를 높힐 수도 있다. 이러한 회선의 이용은 변화시키고자 하는 이미지의 경계에 도달했을때 문제점이 발생될 수 있는데 이러한 문제점은 영상의 모서리에 있는 데이타에 대해서는 행렬 연산을 하지 않음으로써 해결될 수 있다. 행렬 연산을 통하여 이미지를 흐릿하게 또는 선명하게 하는 것 이외에도 모자이크, 물결 무늬, 엠보싱 등의 효과를 가 할 수 있다.



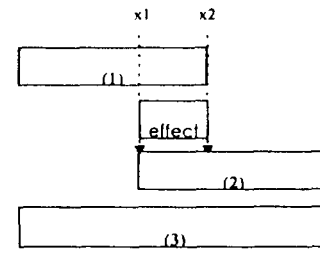
(그림 3) 회선 행렬의 적용
(Fig. 3) Application of convolution matrix

$$(1x20+1x78+1x90+1x30+1x2+1x30+1x17+1x82+1x59)/9=408/9=45 \quad (2)$$

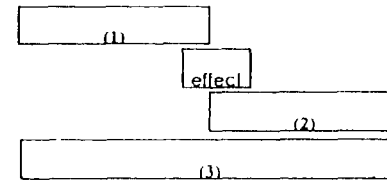
3. 화면 전환 효과 알고리즘

비디오 데이타는 여러개의 정지 영상들이 모여져서 만들어지게 되므로 정지 영상 처리를 위한 알고리즘을 응용하여 화면 전환 효과에 적용할 수 있다. 일반적으로 많이 사용되는 화면 전환 효과에는 페이드(fade), 디졸브(dissolve), 와이프(wipe), 체커박스(checker box), 키잉(keying) 등의 효과가 있다. 이들 효과들은 두 비디오 스트림이 오버랩되는 경우와 그렇지 않은 경우로 나누어 볼 수 있다. 다음 (그림 4)는 비디오 스트림 (1)의 뒷 부분과 (2) 스트림의 앞 부분이 오버랩되면서 효과가 삽입되는 경우이고 (그림 5)는 스트림 (1)의 뒷 부분에 이어서 (2) 스트림이 진행되는 경우이다. (3) 스트림은 효과가 삽입되어 새로 생성되는 스트림을 의미한다. (그림 4)의 경우에 해당하는 화면 전환 효과는 와이프, 디졸브 등이며 (그림 5)의 경우는 페이드 효과를 예로 들 수 있겠다.

본 논문에서 제시하는 화면 전환 효과 알고리즘에 사용되는 상수값들은 다음과 같다.



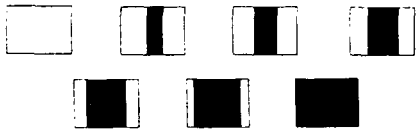
(그림 4) 오버랩되는 화면 전환 효과
(Fig. 4) Overlapped transition effect



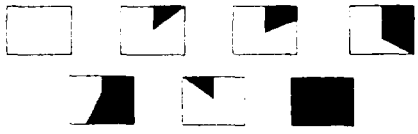
(그림 5) 오버랩되지 않는 화면 전환 효과
(Fig. 5) Not Overlapped transition effect

X, Y : 효과 처리를 위해 입력되는 영상
Z : 효과 처리의 결과 영상
IMAGE COUNT : 화면전환 효과를 위해 새로 생성되는 이미지 수
IMAGE WIDTH : 이미지의 폭
IMAGE HEIGHT : 이미지의 높이

다음에 기술하는 화면 전환 효과 알고리즘에서 계산된 픽셀 값이 0 보다 작거나 255 보다 클 경우는 적당한 값으로 대치되어야 한다. (그림 6)과 (그림 7)은 문개방 와이프(door open wipe)와 시계방향으로의 와이프(clockwise wipe)를 간략히 그림으로 나타낸 것이다. 3.1에서의 알고리즘은 2.2에서 살펴본 더하기 산술 연산을 반복적으로 적용하여 기술하였고 3.2, 3.3, 3.4절에서 제시한 알고리즘은 본 비디오 편집 시스템을 위해 새롭게 제안된 것들이다.



(그림 6) 문 개방 와이프
(Fig. 6) Door open wipe



(그림 7) 시계방향으로의 와이프
(Fig. 7) Clockwise wipe

3.1 페이드

페이드 효과는 영상이 점점 어두워지거나 밝아지면서 한 화면에서 다음 화면으로 전환되는 효과를 말한다. 이 효과는 전환 효과가 시작되는 프레임의 픽셀에 일정한 값을 더해 밝기를 조절한다[8, 10]. 영상이 시작되거나 끝날 때도 적용될 수 있다. 이를 구현한 알고리즘은 다음과 같다(점점 밝아지면서 다음 화면으로 전환되는 경우).

```
increase=255/IMAGE COUNT * 2;
step=0;
for(num=0;  num<IMAGE COUNT;num++){ //이
미지 수 만큼 반복
 for(i=0; i<IMAGE HEIGHT; i++){ //i,j는 이미
 지의 세로와 가로 픽셀의 위치를 나타냄
  for(j=0; j<IMAGE WIDTH; j++){
   if (num<IMAGE COUNT/2) { //생성되는 이미
   지의 절반은 (1)스트림에 대해 처리
    Z[i][j] = X[i][j] + step; //점점 밝아진다.
    Z 이미지의 대부분의 픽셀값이 255에 가까와짐
    step=step+increase;
   }
   else { //생성되는 이미지의 절반은 (2)스트림에
   대해 처리
    Z[i][j]=Y[i][j] + step; //점점 원래의 이미
   지로 돌아옴
    step = step - increase;
   }
  }
 }
}
step : 각 픽셀값에 더해지는 일정한 값 (  255 <=
     step <= 255 )
```

3.2 디졸브

디졸브 효과는 영상이 다른 영상으로 서서히 변하는 과정을 나타내는 효과로서 디졸브의 중간 단계는 사진을 이중인화한 것과 같은 효과를 나타낸다.

```
increase = 1 / IMAGE COUNT;
weight=0;
for (num=0; num<IMAGE COUNT; num++){ //
이미지 수 만큼 반복
 for (i=0; i<IMAGE HEIGHT; i++){ //i,j는 이미
 지의 세로와 가로 픽셀의 위치를 나타냄
  for(j=0; j<IMAGE WIDTH; j++){
  Z[i][j]=X[i][j]*(1 - weight) + Y[i][j]*weight;
  } //(1)스트림의 이미지와 (2)스트림의 이미지의 가
  중치를 서로 다르게 둠
 }  //효과 초기에는 (1)스트림의 이미지의 내용이 많
  이 포함되고 상대적으로 (2)스트림의 내용이 적게 포
  함됨
weight=weight+increase;
}
weight : X, Y 두 이미지의 각 픽셀 값을 혼합시킬 때 각
       이미지에 두게 되는 가중치
(0 <= weight <= 1)
increase : 0 <= increase <= 1
```

3.3 와이프

한 영상에서 다음 영상이 나타날 때 전환 형태와 방향을 조절할 수 있는 효과를 말한다. 예를 들면 다음 장면으로 문이 열리듯 혹은 닫히듯, 또는 시계 방향이나 반 시계 방향으로 돌아가면서 전환되는 효과를 줄 수 있다. 이외에도 왼쪽에서 오른쪽으로, 오른쪽에서 왼쪽으로, 위에서 아래로, 아래에서 위로의 장면 전환의 효과도 포함된다. 여기서는 대표적인 몇가지를 기술한다.

(1) 왼쪽으로의 와이프(Left Push)

다음 화면이 오른쪽에서 왼쪽으로 나타나면서 화면이 전환되는 효과

```
increase=IMAGE WIDTH / IMAGE COUNT;
step=IMAGE WIDTH;
for (num=0; num<IMAGE COUNT; num++){ //
이미지 수 만큼 반복
  for (i=0; i<IMAGE HEIGHT; i++){ //i,j는 이미
  지의 세로와 가로 픽셀의 위치를 나타냄
    for(j=0; j<IMAGE WIDTH; j++){
    if (j < step) //기준되는 step보다 작으면 (1)스트
    림 이미지의 내용을
      Z[i][j] = X[i][j];
    else //그렇지 않으면 (2)스트림 이미지의 내용을 결
    과 영상(Z)에 반영
      Z[i][j] = Y[i][j];
    }
  }
step=step-increase;
}
```

step : X 이미지와 Y 이미지를 구분하는 경계를 나타낸다
(0<=step<=IMAGE WIDTH)

(2) 문 개방 와이프(Door Open)

문이 열리듯 다음 화면이 나타나는 효과

```
increase=IMAGE WIDTH/(2*IMAGE COUNT);
step=IMAGE WIDTH/2;
for(num=0; num<IMAGE COUNT; num++){ //이
미지 수 만큼 반복
  for (i=0; i<IMAGE HEIGHT; i++){ //i,j는 이미
  지의 세로와 가로 픽셀의 위치를 나타냄
    for(j=0; j<IMAGE WIDTH; j++){
      if((j<step) || (j>=IMAGE WIDTH-step))
      //이미지의 가로에 대해서 j가 왼쪽 기준치(step)
        Z[i][j] = X[i][j]; //보다 작거나 오른쪽 기준
        치(IMAGE_WIDTH-step)보다 큰 경우
      else //그렇지 않을 경우
        Z[i][j] = Y[i][j];
    }
  }
  step=step-increase;
}
```

step : X이미지와 Y이미지를 구분하는 경계를 나타낸다
(0<=step<=IMAGE WIDTH/2)

(3) 중앙에서부터의 개방 와이프(Center Open)

화면 중앙에서 다음 화면이 서서히 나타나는 효과

```
wIncrease=IMAGE WIDTH / 2*IMAGE COUNT;
hIncrease=IMAGE HEIGHT / 2*IMAGE COUNT;
wStep=IMAGE WIDTH/2;
hStep=IMAGE HEIGHT/2;
for (num=0; num<IMAGE COUNT; num++){ //
이미지 수 만큼 반복
  for (i=0; i<IMAGE HEIGHT; i++){ //i,j는 이미
  지의 세로와 가로 픽셀의 위치를 나타냄
    for(j=0; j<IMAGE WIDTH; j++){
      if (((j < wStep) || (j>IMAGE WIDTH -
      wStep))
        && ((i< hStep) || (i>=IMAGE HEIGHT
        -hStep)))
        Z[i][j] = X[i][j];
    else
        Z[i][j]=Y[i][j];
    }
  }
  wStep=wStep-hIncrease;
  hStep=hStep-wIncrease;
}
```

wStep, hStep : X이미지와 Y이미지를 구분하는 경계를 나타낸다
(0 <= wStep <= IMAGE WIDTH/2)
(0 <= hStep <= IMAGE HEIGHT/2)

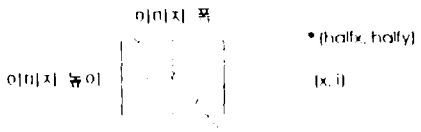
(4) 시계 방향의 와이프(Clockwise)

다음 화면이 시계 방향으로 돌면서 나타나는 효과

```
theta=360 / IMAGE COUNT;
halfx=IMAGE WIDTH / 2;
```

```
halfy=IMAGE HEIGHT / 2;
for (num=0; num<IMAGE COUNT; num++){ //
이미지 수 만큼 반복
  for (i=0; i<IMAGE HEIGHT; i++){ //i,j는 이미
  지의 세로와 가로 픽셀의 위치를 나타냄
    x=halfx + (i-halfy) / m; //기울기가 m이고
    (halfx, halfy)와 (x,i)두점을 지나는 직선의 방정식
      for(j=0; j<IMAGE WIDTH; j++){
        if ((0<theta<90) && (halfx<j<x))
          Z[i][j] = Y[i][j];
        else if ((90<theta<180) && (halfx<j) &
        & (x<j))
          Z[i][j] = Y[i][j];
        else if ((180<theta<270) && ((halfx<j)
        || (x<j)))
          Z[i][j] = Y[i][j];
        else if ((270<theta<360) && ((halfx<j)
        && (x>j)))
          Z[i][j] = Y[i][j];
        else
          Z[i][j] = X[i][j];
      }
  }
}
```

m은 (halfx, halfy)를 지나는 직선의 기울기를 나타내고 halfx와 halfy는 이미지의 가로와 세로의 중간 위치를 나타낸다. 또 x는 세로 픽셀이 i일때 (halfx, halfy)의 점을 지나는 가로 픽셀을 나타내는 점이며 이미지의 가로 크기보다 작다.



(그림 8) 한 이미지 내에서의 halfx와 halfy
(Fig. 8) Halfx and halfy in a image

3.4 키 잉

지정된 칼라에 대해서만 작업을 수행하는 효과를 말한다. 예를 들면 칼라 키잉(chroma keying)을 사용하여 지정된 칼라를 다른 칼라로 바꿀 수 있다.

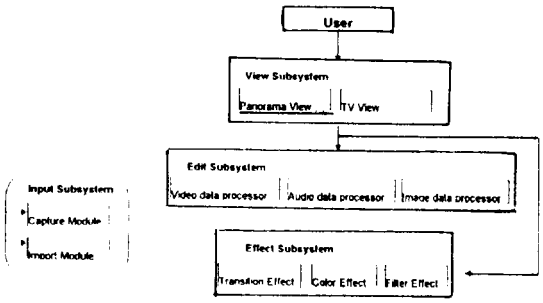
```
for (num=0; num<IMAGE COUNT; num++){ //
이미지 수 만큼 반복
  for (i=0; i<IMAGE HEIGHT; i++){ //i,j는 이미
  지의 세로와 가로 픽셀의 위치를 나타냄
    for(j=0; j<IMAGE WIDTH; j++){
      if (IsInRange(X[i][j]) //X[i][j]는 (i,j)번째 있
      는 픽셀의 RGB 값
        Z[i][j] = Y[i][j];
      else
        Z[i][j] = X[i][j];
      }
    }
}

IsInRange(R, G, B)
{
  if |R' - R| < 허용오차
    if |G' - G| < 허용오차
      if |B' - B| < 허용오차
        return TRUE;
  return FALSE;
}
```

X[i][j]는 (i,j)번째 픽셀의 R, G, B값을 의미하고 R', G', B'은 이미 정의한 RGB 값을 의미한다. 예를 들면 파란 하늘을 다른 그림으로 대치시키기 위해서는 위 R' G' B' 값은 (0,0, 255)가 되어야 한다.

4. 비디오 편집 시스템

이 장에서는 앞에서 언급한 이미지 대수 알고리즘과 화면 전환 알고리즘을 이용하여 비디오 편집 시스템을 설계하고 구현한다. 본 비디오 편집 시스템 설계 방식은 Rebecca[14] 방식을 따랐으며 Microsoft-Windows 3.1 환경에서 Visual C++1.5, MFC2.0, Video for Windows Development Kit를 사용하여 구현하였다[3, 11, 17]. 본 고에서 제시하는 비디오 편집 시스템의 전체적인 구조는 다음 4개의 부 시스템(subsystem)으로 구성되어 있다.

(1) 입력 부 시스템 (Input subsystem)
(2) 뷰 부 시스템 (View subsystem)
(3) 편집 부 시스템 (Edit subsystem)
(4) 효과 부 시스템 (Effect subsystem)



(그림 9) 비디오 편집 시스템의 메세지 전달 그래프
(Fig. 9) Message collaboration graph of video editing system

4.1 입력 부 시스템

입력 부시스템은 아날로그 신호를 캡쳐하는 부분과 본 비디오 편집기에서 사용하지 않는 형식을 적절한 형식으로 변환시키는 임포트(import) 부분이 있다. 캡쳐 부분에서는 프레임 단위의 디지탈 데이타의 캡쳐 뿐만 아니라 캡쳐 보드의 성능에 따라 초당 30프레임의 비디오 캡쳐를 지원한다. 정지 영상의 캡쳐 화면 크기는 640 x 480 크기를 지원하며 동영상의 경우는 320x240까지 지원한다. 임포트 부분에서는 기존의 AVI 비디오 화일, FLI/FLC의 애니메이션 화일, 그리고 다양한 형식의 이미지 화일을 편집 가능한 디지탈 데이타로 변환시키는 작업을 행한다. 오디오의 임포트는 Microsoft 사의 WAV에 한정한다.

4.2 뷰 부 시스템

편집을 위해 사용자는 필름이 나열되어 있는 것과 같은 파노라마 뷰를 사용할 수도 있고 또는 TV와 같이 한 화면만 나타나는 TV 뷰를 사용할 수도 있다. 기본적인 기능은 양쪽 뷰를 통해서 처리할 수 있으며 특정 뷰에 적합한 편집 기능이 존재한다. 예를 들어 화면 전환 효과와 오디오의 편집 기능에 대해서는 파노라마 뷰에서의 작업이 용이하며 영상 위에 다른 영상 혹은 이미지의 중첩(overlay)은 TV 뷰에서의 작업이 더 용이하다. 뷰를 통한 사용자 인터페이스는 다음 사항들을 고려하여 설계되고 구현되었다.

(1) 파노라마 뷰와 TV 뷰의 상호 보완성, 데이타의 일치성

(2) 직접적인 조작(Direct Manipulation)

(3) 편집 결과의 즉각적인 반응

4.2.1 파노라마 뷰

영상이 실제로 필름이 나열된 것과 같이 배열되어 있어 편집 영상을 일목요연하게 볼 수 있게 하였다. 영상을 표현하는 형식은 프레임 단위 또는 초 단위(1, 8 frame or 1, 2, 5, 10 seconds)의 설정이 가능한데 긴 단위의 설정은 전체 영상을 쉽게 볼 수 있게 하고 짧은 단위의 설정은 각각의 프레임에 대한 세밀한 편집을 가능하게 한다. 파노라마 뷰에서는 선택된 프레임의 자르기(cut), 복사(copy), 붙히기(paste), 삽입(insert), 드래그앤드롭(drag & drop) 등의 작업을 직접적인 조작으로 처리할 수 있게 하여 사용자의 직감적인 이해를 도와주며 2, 3장에서 설명한 여러 가지 특수 효과 작업을 할 수 있다. 입력 영상을 위한 트랙은 두개 존재하며 두개 이상의 영상을 동시에 편집할 수 있게 한다((그림 10)에서 상단에 위치한 A, B 트랙). 결과 영상을 위한 트랙이 존재하여 화면 전환 효과의 결과를 보여준다(A+B 트랙). 또한 오디오 트랙이 따로 존재하여 오디오 편집을 영상과 같은 방법으로 처리할 수 있게 한다(그림에서 하단에 위치한 A, B, C 트랙). A와 B트랙 사이에는 전환효과 윈도우에서 드래그앤드롭을 통해 가져온 아이콘을 놓는 위치이다. (그림 10)은 (그림 4)에서 설명한 오버랩되는 화면 전환 효과로서 A와 B트랙 사이의 중첩된 부분에 효과가 가해지는 경우이다.

4.2.2 TV 뷰

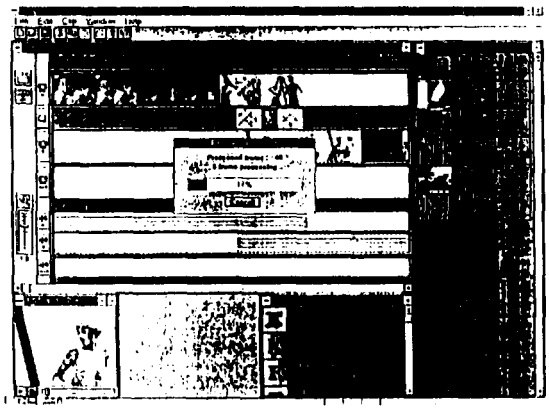
재생될 영상을 TV 모양의 뷰를 통해 나타냄으로써 간단하면서도 인상적인 느낌을 줄 수 있게 하였다. 이미지나 동영상의 오버레이와 같이 원래의 영상 크기를 유지하면서 작업을 하는 경우는 TV 뷰를 사용하게 된다. TV 뷰에서 작업되는 모든 것들은 객체로서 다루어 지게 되며 이동과 크기 조절이 가능하고 비디오 스트림 내의 삽입되는 시작 위치와 끝 위치를 지정할 수 있다. 작업한 결과는 바로 재생해 볼 수 있다(그림 11 참조). 다음은 위에서 열거한 두 뷰의 특징을 정리한 것이다.

4.3 편집 부 시스템과 효과 부 시스템

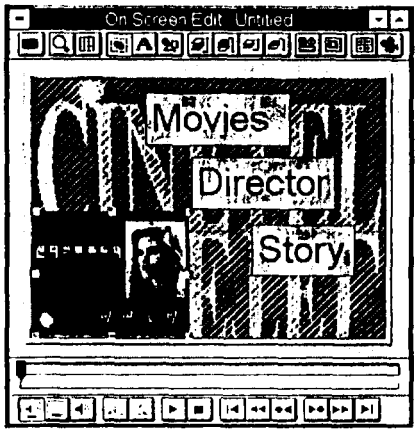
편집 부분에서는 복사, 자르기, 붙히기, 삽입 등의 일반적인 편집 작업을 담당한다. 이 작업은 영상 스트림과 오디오 스트림을 구별해서 편집될 수도 있고 같이 편집될 수도 있다. 효과 부 시스템에서는 다음의 3가지 효과를 제공한다.

(1) 칼라 효과
(2) 필터 효과
(3) 화면 전환 효과

단일 영상을 요구하는 효과에 대해서는, 즉 칼라 효과, 필터 효과들에 대해서는 다이얼로그 박스를 이용하여 효과를 부여한다. 칼라효과에서는 비디오 스트림의 흑백화, 회색화, 밝기 조절, 명암 조절 등의 작업을 행하며 필터효과는 화면의 좌우, 상하 대칭 이동, 화면의 선명화와 뭉개짐을 담당한다(그림 12 참조). 영상 클립이 두개 필요한 효과, 다시 말해서 화면 전환 효과에 대해서는 두 클립 트랙 사이에 효과 아이콘을 위치시킴(드래그앤드롭을 통하여)으로써 효과를 부여한다(그림 10참조). 이를 위해 화면전환 효과 트랙을 두 영상 트랙 사이에 둔다. 화면전환 효과 아이콘은 크기 조절이 가능하며 이를 통해 화



(그림 10) 파노라마 뷰에서의 작업 환경
(Fig. 10) Workspace of panorama view

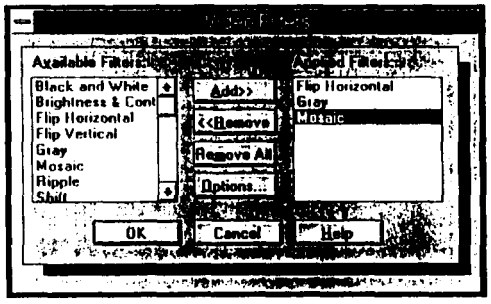


(그림 11) TV 뷰에서의 작업 환경
(Fig. 11) Workspace of TV view

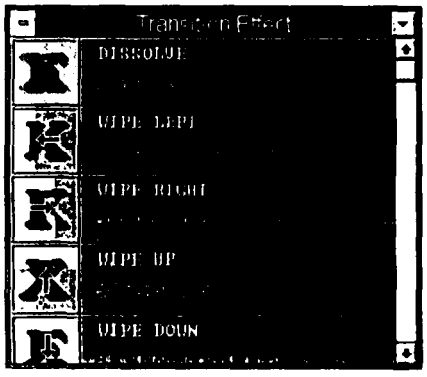
(표 1) TV 뷰와 파노라마 뷰의 기능 비교
(Table 1) Function comparison of TV and panorama view

기 능	TV	Panorama
자르기, 복사, 붙히기, 삽입	v	v
드래그앤드롭	v	v
칼라효과, 필터효과	v	v
키잉 효과	v	v
화면전환 효과	x	v
오버레이 작업	v	x
오디오 편집	x	v
재생	v	v

v : 가능 x : 불가



(그림 12) 필터 다이얼로그 박스
(Fig. 12) Filter dialog box

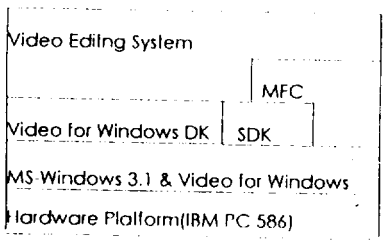


(그림 13) 화면 전환 윈도우
(Fig. 13) Transition effect window

면 전환 효과의 시간을 조절할 수 있게 한다. 화면전환 효과에서는 앞 절에서 설명한 페이드, 디졸브, 와이프, 체커박스, 키잉의 효과를 행한다 (그림 13 참조).

5. 실험 및 고찰

본 논문에서는 MS-Windows 3.1 환경에서 필터 효과, 화면 전환 효과, 그래픽 객체의 오버레이 작업 등을 행하면서 타 비디오 편집기와 장단점을 분석하였다. 본 비디오 편집기의 시스템 주변 환경은 (그림 14)와 같다.



(그림 14) 비디오 편집기 시스템 주변환경
(Fig. 14) Environment of video editing system

본 장에서는 위에서 제시한 비디오 편집 시스템을 이용하여 여러가지 화면 전환 효과들을 실험해보았다. 실험 대상은 320x240크기의 24 비트 실제색(true color)으로 이루어진 동영상이며 주어진 효과는 디졸브, 페이드, 다양한 형태의 와이프, 체커박스, 키잉 등이다. 한 프레임을 처리하는데 약 0.5초 정도 걸렸으며 이는 기존의 전문 비디오 편집 시스템인 Adobe사의 Premiere나 Ulead사의 Video Studio와 비슷하거나 더 빠른 처리 속도였다. 다음 〈표 2〉는 각 화면 전환 효과를 수행하는데 소요되는 시간을 비교,

〈표 2〉 화면 전환 효과 처리 시간 (30 프레임 처리)
〈Table 2〉 Processing time of transition effects (30 frame)

(단위) : 초

효과 \ 제품	본 편집 시스템	Premiere 1.0	VideoStudio2.0
페이드	110	없음	없음
디졸브	114	180	125
왼쪽으로의 와이프	105	143	110
문 개방 와이프	100	145	107
중앙 개방 와이프	106	147	95
시계방향의 와이프	115	없음	없음

정리한 것이다. 각 데이타는 30 프레임을 처리한 후 완성된 비디오 스트림을 제작하는데 걸린 시간을 나타낸다.

UI(User interface) 측면에서 본 Premiere나 Video Studio 경우, 새로 생성되는 클립은 재생할 수 있는 형태로만 보여지고 파노라마 뷰에 펼쳐진 형태로는 보여지지 않기 때문에 이를 편집하기 위해서는 다시 개방(open)해야하는 번거로움이 있었다. 본 비디오 편집 시스템에서는 A, B 트랙 이외에도 C 트랙을 하나 더 두어서 화면 전환 효과의 결과를 바로 C 트랙에 보여줌으로써 사용자가 편집 결과를 한 눈에 확인할 수 있게 하였다(그림 10 참조). 물론 편집 결과는 미리보기(preview)나 TV 뷰를 통해 바로 플레이백 해 볼 수 있게 하여 실제적인 효과의 결과도 확인해 볼 수 있게 하였다.

이외에도 본 비디오 편집 시스템은 비디오 CD를 제작하기 위한 편집 도구로서도 활용할 수 있게 메뉴나 그래픽 제작을 쉽게 할 수 있게 하였다. Video CD 2.0의 규격에는 리모콘으로 사용자가 메뉴를 선택할 수 있게 하여 사용자와의 상호작용(interaction) 기능이 추가되었는데 이를 위한 제작은 타 비디오 편집 시스템으로는 제작이 불편하거나 불가능하다. 본 비디오 편집 시스템에서는 TV 뷰를 통해 원하는 프레임에서 원하는 프레임까지 여러가지 그래픽 객체를 생성할 수 있게 하였다(위 그림 11참조). 그러나 최근에 개발된 Premiere의 화면 전환 효과의 개수가 75개 정도이고 Video Studio의 화면 전환 효과의 개수도 25개 정도인 것에 비추어 본다면 본 비디오 편집 시스템의 화면 전환 효과의 종류는 상당히 미흡하다고 할 수 있으며 이에 대한 계속적인 연구가 필요하리라 본다.

6. 결 론

본 고에서는 그 기술 개발이 거의 전무한 비디오 데이타의 편집과 효과를 다루었다. 특히 화면 전환 효과에 초점을 두어 기술했으며 이를 통해 비디오 데이타의 가치를 한층 더 높힐 수 있게 하였다. 화면 전환 효과의 일부는 정지 영상을 처리할 때의 알고리즘을 응용하여 적용하였으며

실험 결과 기존의 상용 도구들과 비슷한 성능 내지는 우수한 성능을 나타내었다. 본 논문에서 제시한 비디오 편집 시스템의 특징을 요약하면 다음과 같다.

(1) 다양한 화면 전환 효과 알고리즘 제공
(2) 정지 영상 처리 알고리즘을 비디오 효과에 적용
(3) 직관적인 사용자 인터페이스
(4) TV 뷰와 파노라마 뷰를 통한 편집의 편이성 도모
(5) 비디오 화면 상에 그래픽 객체 생성의 용이성

아직은 디지탈 데이타의 효과 편집이 값 비싼 워크스테이션 급에서나 가능하기 때문에 이를 대중화된 PC 상에서 구현한다는 것은 때이른 감이 없지 않다. 그러나 PC의 성능이 하루가 다르게 향상되고 있으며 동시에 영상 처리를 위한 보드가 계속 개발되고 있어 수년 내에 PC를 통한 영상 편집이 일반화되리라고 보여진다. 본 논문을 계기로 더 다양한 화면 효과들을 구비한다면 일반 가정이나 더 나아가서는 방송에도 사용될 수 있는 유용한 도구가 되리라고 생각된다.

참 고 문 헌

[1] A. Rosenfeld and A.C.Kak, "Digital Picture Processing," Academic Press, 1982

[2] Barbara Tansy, "Adobe Premiere User Guide," Adobe System Incorporated, 1993

[3] Ben Shneiderman, "Designing the User Interface," Addison-Wesley Publishing Company, 1992

[4] Bohdan O. Szuprowicz, "Implementing Multimedia for business," Computer Technology Research Corp., 1995

[5] Craig A. Cindley, "Practical Image Processing in C," Jhon Wiley and sons, 1991

[6] Dawson and Benjamin M., "Introduction to Image-Processing Algorithms," BYTE magazine, March 1987

[7] Deb Cameron, "The Internet A Global Business Oppotunity," Computer Technology Research Corp. 1994

[8] Garard J. Holzmann, "PICO-A Picture Editor," AT&T Technical Journal, March/April Vol.66, pp.2~13, 1987

[9] Harald Frater and Dirk Paulissen, "Multimedia Mania," Abacus, 1993

[10] Harley R. Myler and Arthur R. Weeks, "Computer Imaging Recipes in C," PTR Prentice, 1993

[11] James Rumbaugh, Michael Blaha, William Lorensen, "Object-oriented modeling and Design," Prentice-Hall International Inc., 1991

[12] Michael Feerer, "Premiere with a Passion," Peachpit Press, 1994

[13] Milan Sonka, Vaclav Hlavac, and Roger Boyle, "Image Processing, Analysis and Machine Vision," Chapman & Hall Computing, 1993

[14] Rebecca Wirfs-Brian Wilkerson, Laurea Wiener, "Designing object oriented software," Prentice Hall, 1990

[15] Rimmer, "Supercharged Bitmapped Graphics," McGraw-Hill, 1992

[16] Ron Wodaski, "PC Video Madness," SAMS Publishing, 1993

[17] Susan L. Flwler, Victor R. Stanwick, "The GUI style guide," AP Professional, 1995

[18] "Video Studio 2 User Guide," Ulead Systems Incorporated, 1994

김 정 훈

1991년 서울시립대학교 전산통계학과 졸업(이학사)
1994년 연세대학교 대학원 전산과학과 졸업(이학석사)
1994년~현재 현대전자(주) 소프트웨어연구소 근무
관심분야 : 하이퍼텍스트, 하이퍼미디어, 멀티미디어 데이타 처리