

실시간 문자효과기(프리즘젬) 개발

박성준*, 이경국, 박근수, 이범구
한국방송공사 기술연구소

The Development of Real-time Character Effector : PRISM GEM

Sung-Choon Park*, Kyung-Kook Lee, Keun-Soo Park, Beom-Goo Lee
Technical Research Institute, Korean Broadcasting System(KBS)

* Email : scpark@tri.kbs.co.kr Tel. : 82-2-781-5964 Fax : 82-2-781-5999

요 약

본 논문에서는 문자나 그래픽 요소에 대한 실시간 다중 디지털 효과를 구현한 프리즘젬 장비를 소개한다. 프리즘 젬은 문자를 포함한 여러 그래픽 요소에 대하여 각기 다른 디지털 효과를 편집하여 실시간으로 동시에 효과를 수행할 수 있어서 방송 화면에 다양한 표시 효과를 제공한다. 프리즘젬은 그래픽 보드와 특별히 제작된 효과용 보드를 장착한 PC형 장비로 소형이며 경제적이다. 프리즘젬은 다양한 문자와 그래픽을 편집할 수 있고, 이들에 대한 위치 변환, 크기 변환, 회전, 기울임, 구부림, 퍼뜨림, 클리핑, 스크롤, 블링킹 등의 효과와 함께 곡선이나 나선형과 같은 다양한 경로 이동 효과를 줄 수 있다. 본 논문에서는 프리즘젬 장비의 하드웨어 구성, 실시간 다중 디지털 효과를 위한 효과 보드의 동작 원리, 실시간 처리 성능, 복합적인 모양 변형 효과를 구현하기 위한 효과 변환 함수의 모델링 방법, 경로 이동 효과를 포함한 효과 애니메이션 방법, 운용 소프트웨어의 기능 등에 대한 내용을 기술한다.

I. 서 론

스캐너(FSS : Flying Spot Scanner)와 같은 자막 처리 장비의 대체용으로 사용되기 시작한 문자발생기(Character Generator)는 컴퓨터 하드웨어 및 그래픽 기술의 놀라운 발전 속도에 힘입어 다양한 자막과 그래픽을 처리할 수 있게 되었고, 지금은 방송 프로그램 제작의 기본 장비로 인식되고 있다.

그런데, 최근에는 뉴스, 스포츠 및 오락 프로그램 등의 제작에 있어서, 자막과 그래픽의 다이내믹한 효과에 대한 요구가 점차 증가하고 있다. 그러나, 문자발생기로만 처리하기에는 아직 한계가 있다. 특히, 예고 프로그램처럼 여러 가지 내용에 각기 다른 효과를 동시에 주기 위해서는 별도의 디지털 영상 효과기(Digital Video Effect)를 사용한다 하더라도 여

러 번의 편집을 거치거나 여러 대의 장비를 동원해야 가능하다. 즉, 비용 및 시간상으로 비효율적이다.

이러한 문제점을 극복하기 위해서는 다음과 같은 조건을 만족하는 장비가 필요하다. 첫째, 문자를 포함한 그래픽 요소에 대해 실시간 디지털 효과가 가능해야 한다. 둘째, 경제성을 고려한 저가의 장비로 문자발생기와의 연결이 간편해야 한다. 셋째, 가장 중요한 조건 중의 하나로 여러 그래픽 요소들에 대해 독립적인 효과를 동시에 수행할 수 있어야 한다.

KBS 기술연구소에서는 이와 같은 조건을 모두 만족하는 보드를 개발하여 젬(GEM : Graphic Effect Machine)이라 이름지었다.

젬은 기능적으로는 문자를 포함한 여러 그래픽 요소의 독립적인 실시간 효과를 수행할 수 있으면서도, 장비의 소형화와 경제성을 고려하여 PC에 장착 가능하게 한 장의 보드로 구성하였으며, 이것은 문자발생기와의 연결을 자연스럽게 한다. 젬은 그림 1.과 같은 복잡한 효과를 실시간으로 동시에 수행할 수 있게 한다.



그림 1. 다중 그래픽 요소에 대한 독립적인 실시간 디지털 효과

KBS 기술연구소에서는 프리즘, 프리즘플러스, 프리즘프로로 이어지는 문자발생기를 자체 기술로 개

발하여 왔다. 이 중 가장 최근에 개발된 프리즘프로와 켄을 결합하여 프리즘켄을 탄생시켰다[1]. 프리즘켄은 프리즘프로의 성능을 향상시킴과 동시에 켄 보드를 장착하여 문자와 그래픽 요소에 다양한 효과를 실시간으로 사용할 수 있다.

프리즘켄은 펜티엄 CPU를 사용하는 PC 시스템을 기본으로 하고 있으며, 32MB의 메인 메모리와 1GB의 하드 디스크를 사용한다. 프리즘켄에 장착된 그래픽 보드의 출력은 디지털 영상 데이터로 켄 보드와 내부적으로 연결된다. 켄 보드의 출력은 RGB 신호로 나오며, 이것을 컬러 인코더로 인코딩하여 컴포지트 신호로 변환하여 사용한다.

프리즘켄은 '95년에 개발하여 제1회 전국동시지방선거 개표방송에서 처음으로 사용되었으며, ABU 총회에 논문을 발표하여 호평을 받은바 있다[3]. '96년에는 켄 보드에 화소보간회로를 추가하여 화질을 개선시켜 15대 국회의원선거 개표방송에 사용하였으며, NAB Award에서 본선 진출 작품으로 선정되기도 하였다. 또한, 프리즘켄은 『다중 화면 처리 방법 및 그 장치』로 현재 국내 특허 출원 중에 있다.

프리즘켄의 소프트웨어로 '95년에 기본적인 효과 제작 도구인 켄툴(GEMTOOL)을 개발하여 선거방송등에 사용하였고, '96년에는 사용자의 편리성을 대폭 강화한 운용 소프트웨어를 새롭게 개발하여 현업에 활용하고 있다[2].

II. 프리즘켄의 하드웨어 구성 및 동작 원리

1. 시스템의 전체적인 구성

프리즘켄은 펜티엄 CPU, 32MB의 메인 메모리, 1GB의 하드 디스크가 내장된 PC 시스템에 타가 (TARGA+ 64) 그래픽 보드와 켄 보드를 장착한 본체와, 편집을 위한 모니터, 키보드와 마우스, 그리고 컬러 인코더 등으로 구성된다. 이들의 구성 및 연결은 그림 2와 같다.

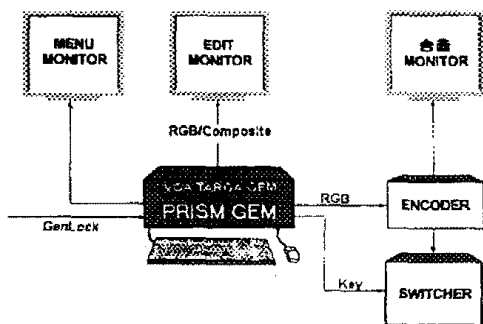


그림 2. 프리즘켄의 구성 및 연결

프리즘켄에서 문자와 그래픽 요소의 내용은 타가

그래픽 보드에 그려지며, 이 출력은 디지털 영상 데이터의 형태로 켄 보드에 입력되어 효과가 수행된다. 켄 보드는 타가 보드로부터 타이밍 데이터도 받아서 동작하게 되는데, 원래의 타가 보드에는 디지털 영상 데이터나 타이밍 데이터의 출력 단자가 없으므로 타가 보드를 개조하여 데이터 출력부를 추가하였다. 데이터 출력부는 작은 보드로 만들어 타가 보드에 부착하였다.

2. 효과 보드의 구성 및 동작 원리

켄 보드의 구성도는 그림 3과 같으며 다음과 같이 동작한다.

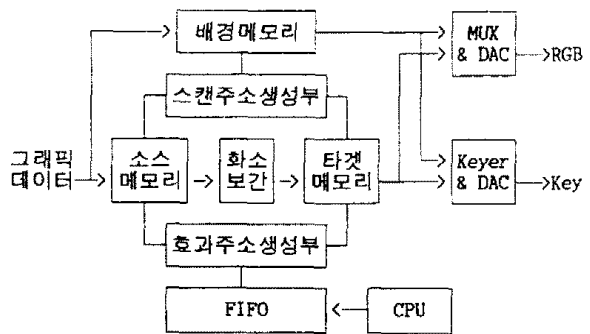


그림 3. 켄 보드 구성도

그래픽 보드에서 입력되는 영상 데이터는 소스 메모리(Source Memory)로 입력된다. 소스 메모리의 입력 데이터는 스캔 주소 생성부(SAG : Scan Address Generator)에서 만들어지는 보통의 순차 주소로 쓰여지고 효과 주소 생성부(EAG : Effect Address Generator)의 소스 메모리 주소로 읽혀진다. EAG의 소스 메모리 주소로 읽혀진 영상 데이터는 타겟 메모리(Target Memory)에 EAG의 타겟 메모리 주소로 저장되며, 이 과정에서 효과 화면이 만들어진다.

EAG에서는 Raytheon Semiconductor사의 이미지 리샘플러(Image Resampler)인 TMC2301을 사용하는 데, 소스 메모리 주소 X, Y와 타겟 메모리 주소 U, V를 식(1)에 의해 발생시킨다[4].

$$\begin{aligned} X(U, V) &= a_1U^2 + a_2U + a_3UV + a_4V^2 + a_5V + a_6 \\ Y(U, V) &= \beta_1U^2 + \beta_2U + \beta_3UV + \beta_4V^2 + \beta_5V + \beta_6 \end{aligned} \quad (1)$$

$$a_7 \leq X \leq a_8, \quad a_9 \leq U \leq a_{10}, \quad \beta_7 \leq Y \leq \beta_8, \quad \beta_9 \leq V \leq \beta_{10}$$

여기에서 $a_1, \dots, a_{10}, \beta_1, \dots, \beta_{10}$ 의 값은 EAG 매개변수로서 이 매개변수를 변화시켜 2차원 이동, 확대, 축소, 회전, 비선형 효과 등을 만든다[5]. 따라서, 한 화면을 여러 부분으로 나눈 뒤, 각 부분에 서로 다른 매개변수를 사용하면 화면의 각 부분에 서로 다른 효과 화면을 생성시킬 수 있다. 켄 보드는 한 화면에 대하여 최대 32개까지 영역별로 서로 다른 효과를 줄 수 있다.

효과 화면의 생성에 쓰이는 EAG의 소스 메모리 주소는 일반적으로 소수점 부분의 값을 갖게 되므로 이 지점의 정확한 영상 데이터의 값은 주위 네 지점 픽셀들을 보간하여 구하는 것이 좋다. 따라서 주위 네 지점의 픽셀을 쉽게 구하기 위하여 소스 메모리는 프레임 구조의 동일한 네개의 스트라이프 메모리(Stripe Memory)로 구성된다.

매개변수는 PC로부터 매 필드별로 EAG에 공급되며 그 중간에 쓰인 FIFO(First In First Out)는 PC가 보다 효율적으로 매개변수를 EAG에 공급할 수 있도록 한다. 즉, FIFO 내에 한 필드 분의 매개변수를 저장하고, EAG에서 주어진 영역의 주소 생성이 끝나면, 즉시 다음 영역의 매개변수를 FIFO에서 EAG로 입력하는 방식으로, PC의 계산 처리와는 독립적으로 동작한다. 이와 같은 방법으로 실시간 다중 효과를 수행할 수 있다.

소스 메모리와 타겟 메모리는 각각 프리즈(Freeze) 기능이 있어서 프리즈 시에 그래픽 보드에 배경 화면을 준비하여 배경 화면용 프레임 메모리로 보낼 수 있다. 효과가 생성되어 저장된 타겟 메모리는 배경 화면과 혼합된 뒤 DAC를 거쳐 아날로그 컴포넌트 상태로 출력된다. 내부에서 처리된 키신호는 타겟 메모리와 배경 화면의 믹스 회로에 작용하며, 일정 시간 지연된 뒤 DAC를 거쳐 외부로 출력된다.

3. 프리즘점의 실시간 처리 성능

캠 보드는 타가 그래픽 보드로부터 타이밍 신호를 받아서 동작한다. 타가 그래픽 보드와 캠 보드는 512픽셀x486라인의 해상도로 사용하는데, 이 모드에서의 비디오 클럭 주파수는 9.702MHz로 초당 처리할 수 있는 픽셀수는 616픽셀x525라인x30프레임/초=9,702,000픽셀이다. 프리즘점에서 효과 보드의 FIFO로 매개변수를 전달하는 시간은 프레임당 약 4스캔라인이 걸리므로 초당 처리 픽셀수는 대략 9,702,000-616픽셀x4라인x30프레임/초=9,628,000픽셀이다. 이것은 초당 처리해야 할 화면의 픽셀수인 512픽셀x486라인x30프레임/초=7,464,960픽셀에 비하여 약 130% 정도이다. 따라서, 프리즘점은 최대 32개의 독립적인 영역에 대하여 이들 면적의 합이 출력 화면의 130%이내일 경우에는 실시간으로 처리할 수 있다.

III. 효과 변환 함수 모델링

1. 모양 변형 효과 함수 모델의 설정

복합적인 모양 변형 효과는 EAG에서 식(1)에 따른 매개변수의 변화에 의해 수행된다. 그런데, 식(1)은 일반적인 2원 2차방정식이므로 각항의 계수 지정 방식으로는 원하는 모양 변형 효과를 얻기가 쉽지

않다. 따라서, 일반적으로 디지털 효과에 사용되는 사용자 매개변수, 예를 들면, 이동, 확대, 축소, 회전, 기울임, 구부림, 퍼뜨림 등의 정도값에 의해 모양을 변형시킬 수 있는 효과 변환 함수의 모델링이 필요하다.

효과 변환 함수는 기본적으로 식(2)와 같은 행렬 변환을 사용한다.

$$\begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} = M_{Effect} \begin{pmatrix} U \\ V \\ 1 \end{pmatrix} \quad (2)$$

효과 변환 행렬은 모양 변형 효과에 사용되는 사용자 변수의 종류에 따라 달라지는데, 프리즘점에서는 다양하고도 복합적인 모양 변형 효과를 얻기 위하여 식(3)과 같은 변환 행렬로 모델링하였다.

$$\begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} = M_o \cdot f_i \cdot M_n \cdot M_c \cdot M_s \cdot M_k \cdot M_r \cdot M_t \begin{pmatrix} U \\ V \\ 1 \end{pmatrix} \quad (3)$$

M_o : 변환 대상 문자/그래픽 객체의 위치 및 영역 결정 행렬

f_i : 비선형 변환값을 산출하는 함수

M_n : 구부림, 퍼뜨림 등 비선형 효과 영역을 얻기 위한 행렬

M_c : 모양 변형의 중심을 주기 위한 행렬

M_s : 크기 변화를 주기 위한 행렬

M_k : 기울임을 주기 위한 행렬

M_r : 회전을 주기 위한 행렬

M_t : 위치 이동을 주기 위한 행렬

\cdot : 단순 곱이 아니고 행렬과 함수의 적용을 의미함

이는 XY평면에서 UV평면으로의 변환으로 볼 때 적용 순서를 역순으로 고려해야 한다. 즉, 먼저 변환할 영역을 지정하고, 비선형 변환을 적용한 후, 비선형 변환에 의해 변화된 영역을 얻고, 변형 중심을 얻어서, 이에 대해 크기를 변환하고, 기울임을 준 다음, 회전을 처리하고, 최종적으로 위치를 이동시킨다. 이것이 일반 사용자의 모양 변형에 대한 잠재적 관념에 가장 잘 부합하는 순서라고 생각된다.

2. 모양 변형 효과 함수 모델의 전개

앞에서 설정한 효과 변환 모델을 전개하여 보자. 프리즘점은 타겟 메모리의 위치로부터 소스 메모리의 위치를 결정하는 역매핑(Inverse Mapping)을 사용하므로 변환 행렬은 역변환(Inverse Transform) 형태이다.

효과를 주기 위한 문자 및 그래픽 요소의 소스 메모리 위치값(sO_x, sO_y)과 영역 크기(sO_w, sO_h)가 결정되면, 위치값만큼 위치 변환을 하여 원점으로 이동시킨다. 변환 행렬 M_o 는 식(4)와 같다.

$$M_o = \begin{pmatrix} 1 & 0 & sO_x \\ 0 & 1 & sO_y \\ 0 & 0 & 1 \end{pmatrix} \quad (4)$$

여기에 먼저 비선형 변환을 가한다. 식(1)에서 이

차항과 교차항의 계수를 변화시키면 비선형적인 모양 변화가 일어나는데, 비선형의 모양 변화를 정규화 시키기 위해서는 식(5)와 같은 비선형 모양 변환 함수와 식(6)과 같은 모양 변형에 따라 변화된 영역을 산출하는 함수가 필요하다.

$$f_i : \begin{pmatrix} X \\ Y \end{pmatrix} = f_i(U, V, \hat{a}) \quad (5)$$

$$g_i : \begin{pmatrix} U_0 \\ V_0 \\ W_0 \\ H_0 \end{pmatrix} = g_i(sO_w, sO_h, \hat{a}) \quad (6)$$

여기에서 $\hat{a} = (a_0, a_1, a_2, a_3, \dots)$ 는 효과 결정 변수의 집합으로 구부림, 퍼뜨림, 찌그러뜨림, 밀소정도 등과 같은 비선형 변형 효과의 방향이나 정도 등이 포함된다. 식(5)는 설정된 효과 파라미터 \hat{a} 따라 모양 변화를 결정하는 함수로 U, V에 대한 2차함수이고, 변형 효과에 따라 적합한 함수를 사용한다. 식(6)은 \hat{a} 에 따라 변형된 모양을 포함하는 영역의 시작 위치 (U_0, V_0)와 크기 (W_0, H_0)를 산출하는 함수로, 원래 영역의 넓이 sO_w 와 높이 sO_h 에 대한 함수이다. 비선형 효과 영역 행렬 M_n 은 식(7)과 같이 된다.

$$M_n = \begin{pmatrix} 1 & 0 & W_0 \\ 0 & 1 & H_0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & g_{i_u_0}(sO_w, sO_h, \hat{a}) \\ 0 & 1 & g_{i_v_0}(sO_w, sO_h, \hat{a}) \\ 0 & 0 & 1 \end{pmatrix} \quad (7)$$

다음에는 확대, 축소, 기울임, 회전 등 선형 변형을 위해 문자 및 그래픽 요소의 모양 변형 효과 중심점을 결정한다. 중심점은 μW_0 와 νH_0 로 결정되며, μ 와 ν 의 변역은 각각 $0 \leq \mu \leq 1$, $0 \leq \nu \leq 1$ 이다. W_0 와 H_0 는 비선형 변환에 의해 변화된 영역 크기이다. 선형 변환을 위하여 변형 중심점을 원점으로 위치 변환하여야 하므로 모양 변형 중심점 변환 행렬 M_c 는 식(8)과 같다.

$$M_c = \begin{pmatrix} 1 & 0 & \mu W_0 \\ 0 & 1 & \nu H_0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & \mu g_{i_w_0}(sO_w, sO_h, \hat{a}) \\ 0 & 1 & \nu g_{i_h_0}(sO_w, sO_h, \hat{a}) \\ 0 & 0 & 1 \end{pmatrix} \quad (8)$$

선형 변형인 크기 변환, 기울임, 회전, 이동에 대한 변환 행렬은 각각 식(9), 식(10), 식(11), 식(12)와 같다.

$$M_s = \begin{pmatrix} \frac{S_y}{S_x \cdot S_y} & 0 & 0 \\ 0 & \frac{S_x}{S_x \cdot S_y} & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (9)$$

S_x : X축 크기 변환, S_y : Y축 크기 변환

$$M_k = \begin{pmatrix} \frac{\cos \theta_2}{\sqrt{\cos(\theta_1 - \theta_2)}} & -\frac{\sin \theta_2}{\sqrt{\cos(\theta_1 - \theta_2)}} & 0 \\ \frac{\sin \theta_1}{\sqrt{\cos(\theta_1 - \theta_2)}} & \frac{\cos \theta_1}{\sqrt{\cos(\theta_1 - \theta_2)}} & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (10)$$

θ_1 : X축 기울임, θ_2 : Y축 기울임

$$M_r = \begin{pmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (11)$$

θ : 회전 각도

$$M_t = \begin{pmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{pmatrix} \quad (12)$$

t_x : X축 이동, t_y : Y축 이동

따라서, 효과 변환 함수는 식(13), 식(14)와 같이 전개되며, 최종적인 X, Y 값은 각각 식(15), 식(16)이 된다. 이것은 결국 U, V에 대한 이차식이 되고, 식(1)과 같은 형태로 정리하여 EAG의 매개변수를 결정한다.

$$\begin{pmatrix} U \\ V \\ 1 \end{pmatrix} = M_c \cdot M_s \cdot M_k \cdot M_r \cdot M_t \begin{pmatrix} U \\ V \\ 1 \end{pmatrix} \quad (13)$$

$$\begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} = M_n \cdot f_i \cdot M_c \begin{pmatrix} U \\ V \\ 1 \end{pmatrix} \quad (14)$$

$$X = f_i(U + g_{i_u_0}(sO_w, sO_h, \hat{a}), V + g_{i_v_0}(sO_w, sO_h, \hat{a}) + sO_x) \quad (15)$$

$$Y = f_i(U + g_{i_u_0}(sO_w, sO_h, \hat{a}), V + g_{i_v_0}(sO_w, sO_h, \hat{a}) + sO_y) \quad (16)$$

3. 효과 애니메이션

여기서는 정지된 모양 변환이 아닌 움직임 효과에 대해 기술한다. 각종 변환에 대한 EAG 매개변수들을 썸 보드 내의 FIFO로 매 프레임마다 보냄으로써 실시간의 동적인 효과를 구현할 수 있다. 동적인 효과를 주기 위하여 키프레임(Keyframe)과 키프레임 사이의 인터프레임(Interframe)의 EAG 매개변수는 키프레임의 효과 변수를 보간(Interpolation)한 후 효과 변환 함수에 의해 얻는다.

키프레임간의 이동에는 점프형, 직선형, 곡선형, 공진형 경로 이동 방식을 지원한다. 점프형은 보간 없이 순간적으로 이동 변환을 줌으로써 구현한다.

직선형은 키프레임의 위치를 연결하는 직선 경로로의 이동 방식으로 인터프레임의 위치 P_i 는 식(17)에 의해 얻어지며, 시간 함수 $F(t)$ 를 변경하여 이동하는 속도에 변화를 가할 수 있다. 시간 함수 $F(t)$ 는 시간에 대한 일반 n차식이 가능하며, 가속 및 감속의 자유로운 조절이 가능하다.

$$P_i = P_{k1} + (P_{k2} - P_{k1})u \quad (17)$$

$u = F(t), T_{k1} \leq t \leq T_{k2}, 0 \leq u \leq 1$
 P_{k1} : 키프레임1 위치, P_{k2} : 키프레임2 위치
 T_{k1} : 키프레임1 시간, T_{k2} : 키프레임2 시간

곡선형은 베지어 곡선(Bezier Curves) 알고리즘을 이용한다[6]. 즉, 양쪽 키프레임의 접선 벡터(Tangent Vector)를 정의하는 두 개의 제어점(Control Point)을 지정하여 베지어 곡선을 생성하며,

인터프레임의 위치는 이 곡선을 따라 움직인다. 베지어 곡선의 행렬 표현은 식(18)이므로, 인터프레임의 위치 P_i 는 식(19)와 같다. 시간 함수 $F(t)$ 는 시간에 대한 1차식을 사용한다.

$$Q_i(u) = (u^3 \ u^2 \ u \ 1) \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} P_i \\ P_{i+1} \\ P_{i+2} \\ P_{i+3} \end{pmatrix} \quad (18)$$

$$P_i = ((P_{k2}-P_{k1})-3(P_{c2}-P_{c1}))u^3 + 3(P_{c2}-2P_{c1}+P_{k1})u^2 + 3(P_{c1}-P_{k1})u + P_{k1} \quad (19)$$

$u = F(t)$, $T_{k1} \leq t \leq T_{k2}$, $0 \leq u \leq 1$
 P_{k1} : 키프레임1 위치, P_{k2} : 키프레임2 위치
 P_{c1} : 제어점1 위치, P_{c2} : 제어점2 위치
 T_{k1} : 키프레임1 시간, T_{k2} : 키프레임2 시간

공전형은 나선형 움직임이 가능하도록 두 개의 공전 중심과 각각의 공전 반경을 설정한다. 두 키프레임의 위치를 각각 P_{k1} , P_{k2} 라하고, 공전 중심 위치를 각각 P_{o1} , P_{o2} 라 하면, 각 키프레임과 공전 중심 사이의 거리 r_1 , r_2 와 방향각 α_1 , α_2 는 식(20)과 같다.

$$r_1 = |\overrightarrow{P_{o1}} - \overrightarrow{P_{k1}}| \quad r_2 = |\overrightarrow{P_{o2}} - \overrightarrow{P_{k2}}| \quad (20)$$

$$\alpha_1 = \tan^{-1}(\overrightarrow{P_{o1}} - \overrightarrow{P_{k1}}) \quad \alpha_2 = \tan^{-1}(\overrightarrow{P_{o2}} - \overrightarrow{P_{k2}})$$

따라서, 인터프레임 시간에서의 공전 중심 P_o 와 공전 반경 r , 방향각 α 는 식(21)과 같으며, 인터프레임의 위치 P_i 는 식(22)와 같다. 식(21)에서 시간 함수 $F(t)$ 는 시간에 대한 1차식을 사용한다.

$$P_o = P_{o1} + (P_{o2} - P_{o1})u$$

$$r = r_1 + (r_2 - r_1)u \quad (21)$$

$$\alpha = \alpha_1 + (\alpha_2 - \alpha_1 \pm 2\pi\phi)u$$

$u = F(t)$, $T_{k1} \leq t \leq T_{k2}$, $0 \leq u \leq 1$
 ϕ : 회전수, T_{k1} : 키프레임1 시간, T_{k2} : 키프레임2 시간

$$P_i = P_o + \begin{pmatrix} r \cos \alpha \\ -r \sin \alpha \end{pmatrix} \quad (22)$$

IV. 프리즘젯 운용 소프트웨어의 기능

프리즘젯 운용 소프트웨어는 문자, 도형, 심벌들에 대한 효과를 편집하고 이를 실시간으로 송출하는 기능을 갖는다.

편집 기능은 크게 문자 및 그래픽 편집과 효과 편집으로 나눌 수 있다. 문자 및 그래픽 편집에서는 다양한 형태의 문자와 도형을 제작하고, 팁스(Tips) 등의 그래픽 프로그램에서 제작한 심벌이나 배경 그림을 불러다 사용할 수 있다. 그림 4.는 문자 편집 시의 메뉴 화면의 모습이다.

문자 편집에서는 다양한 서체의 한글, 영문, 한자 및 특수 문자를 임의의 크기와 색상으로 제작할 수 있다. 또한, 여러 가지의 그림자 및 테두리 형태, 방향, 두께 조정 기능, 횡서와 종서 기능, 문단 배열 기

능 등을 제공하며, 이미 제작된 문자에 대한 자유로운 편집과 속성 변경이 가능하다. 특히, 자주 사용되는 문자 형태를 저장하여 필요할 때 그림 형태의 아이콘을 선택하여 불러 쓰는 라이브러리 기능도 있다. 그래픽 편집은 선, 사각형, 원 등과 같은 도형에 대하여 문자와 비슷한 속성으로 자유로운 편집이 가능하다.

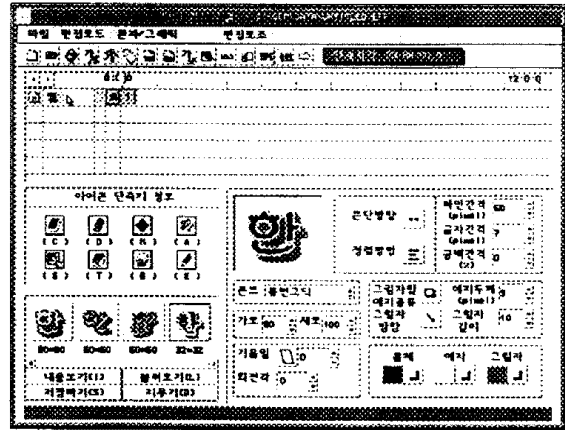


그림 4. 문자 편집 시의 메뉴 화면

효과 편집에서는 문장, 도형 및 심벌 등 생성된 객체에 대하여 효과를 생성하는데, 효과를 생성하는 방법은 객체에 미리 제작된 전체적인 효과를 불러와서 적용하는 방법과 각각의 키프레임 모양과 이동 경로를 제작하는 방법이 있다. 그림 5.는 효과 편집 시의 메뉴 화면의 모습이다. 메뉴 화면은 효과에 대한 키프레임의 위치, 이동 경로 및 시간을 편집할 수 있도록 구성된다.

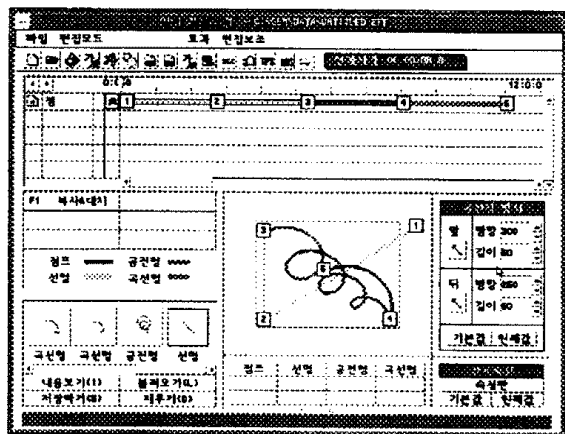


그림 5. 효과 편집 시의 메뉴 화면

키프레임의 모양 편집에서는 확대, 축소, 회전, 기울임, 클리핑, 스크롤, 밀소, 구부림, 퍼뜨림, 찌그러뜨림 등의 효과를 복합적으로 만들 수 있다. 키프레임과 키프레임 사이의 모양은 매 필드별로 인터폴레

이선 된다. 키프레임의 모양은 라이브러리로 저장 가능하며, 변형의 모습이 표시된 라이브러리 아이콘을 선택하여 불러다 다른 객체에 적용시킬 수 있다. 키프레임 모양 변형 효과의 예를 그림 6에 보인다.

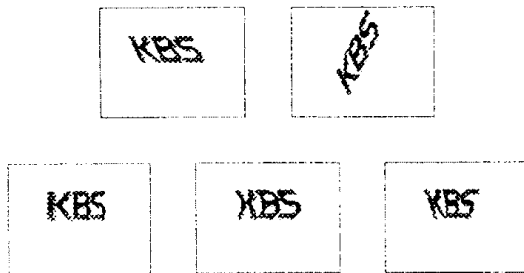


그림 6. 키프레임 모양 변형 효과

키프레임의 모양과 위치 결정 후에 키프레임간의 이동 경로를 편집하게 되는데, 이동 경로에는 점프 방식, 직선형 방식, 곡선형 방식, 공전형 방식이 있다. 직선형 방식에서는 속도 완급의 조절이 가능하며, 곡선형 방식은 베지어 곡선을 이용하여 다양한 곡선형 경로를 생성할 수 있다. 또한, 공전형 방식은 공전에 대한 시작 중심과 끝 중심, 회전 방향 및 회전수를 조정하여 나선형 경로 등을 생성할 수 있다. 이동 경로도 라이브러리로 저장 가능하며, 저장된 것은 불러다 재 사용할 수 있다. 그림 7에 이동 경로에 따른 효과의 예를 보인다.

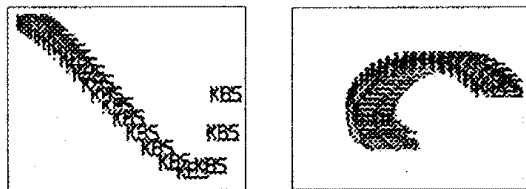


그림 7. 이동 경로 효과

여러 개의 키프레임에 대하여 모양과 이동 경로를 생성하면 객체에 대한 하나의 효과가 되며, 이것을 효과 윈도우라 하는데, 하나의 객체에 대하여 여러 개의 효과 윈도우를 생성할 수 있다. 즉, 하나의 내용을 가지고 각기 다른 여러 효과를 동시에 낼 수 있다. 한 객체에 대한 효과 윈도우들의 집합을 효과 그룹이라 한다. 효과 윈도우 및 효과 그룹 역시 라이브러리로 저장하였다가 다른 객체에 불러오기로 적용시킬 수 있다. 이렇게 하면 효과 제작에 걸리는 시간을 상당히 단축시킬 수 있다.

위와 같이 편집된 효과는 하나의 효과 화일로 저장할 수 있으며, 송출 프로그램에서 저장된 효과 화일들을 이용하여 송출 순서 및 시간 등을 정하여 송출할 수 있다. 송출에서 객체에 대한 효과는 실시간으로 이루어진다.

V. 결 론

프리즘셈은 문자발생기 기술과 디지털 영상 효과 기술에 기반을 둔 그래픽 문자 효과기로서, 문자를 포함한 여러 그래픽 요소에 대하여 각기 다른 효과를 동시에 실시간으로 수행한다. 이러한 기능은 그래픽 보드와의 연계성을 고려한 셈 보드의 설계, EAG와 FIFO를 이용한 다중 효과 처리 등에 기초한다. 또한, 프리즘셈의 운용 소프트웨어는 키프레임 편집 기술에 기초하여 효과의 다양성을 증폭시키며, 효과 라이브러리의 사용으로 효과 편집을 신속히 할 수 있다.

프리즘셈은 여러 개의 객체에 대하여 확대, 축소, 회전, 기울임, 콜리핑, 스크롤, 밀소, 구부림, 퍼뜨림, 찌그러뜨림 등의 모양 변환 효과와 점프, 직선 움직임, 곡선 움직임, 나선형 움직임 등의 이동 경로를 독립적으로 생성하거나 적용할 수 있다.

프리즘셈은 타이틀 편집과 같이 문자 및 그래픽 요소에 디지털 효과를 적용하는 분야, 스포츠나 오락 프로그램 등과 같이 다양하고도 다이내믹한 효과를 필요로 하는 분야, 그리고, 프로그램 예고 편집과 같이 여러 가지 내용에 각기 다른 효과를 동시에 적용하는 분야 등에 특히 유용하게 활용될 수 있다.

프리즘셈은 문자와 그래픽 요소에 대한 실시간 다중 디지털 효과라는 새로운 방송 기술 분야를 개척하였다는 데에 그 의미가 크며, 나아가 방송의 품질과 생산성 향상에도 크게 기여할 것으로 기대된다.

참 고 문 헌

- [1] 한국방송공사 기술연구소, 영상정보처리 시스템 연구보고서, pp.3-97, 1995 12월
- [2] 한국방송공사 기술연구소, 프리즘셈 운용 소프트웨어 개발보고서, 1996 10월.
- [3] K. S. Park, K. K. Lee, Y. C. Yoon and S. C. Park, "The Development of Graphic Effect Machine(GEM)", ABU Technical Review, No.163, pp.16-21, March-April, 1996.
- [4] Raytheon Semiconductor, "TMC2301 CMOS Image Resampling Sequencer", 1994 Data Book, pp.2_187-2_211, 1993.
- [5] John Eldon and John Watson, "Non-Linear Operations with the TMC2301 Image Resampling Sequencer", Raytheon Semiconductor, 1994 Data Book, pp.7_95-7_103, 1993.
- [6] Richard H. Bartels, John C. Beatty and Brain A. Barsky, An Introduction to Splines for use in Computer Graphics and Geometric Modeling, pp.211-245, 1987.