글. 조인준 KBS 기술연구소 차장

C군의 네버엔딩 스토리, 디지털 영상처리의 이해 - 7

지난 연재에서 Fill과 Key 영상을 이용한 영상합성의 기본원리에 대한 설명을 마무리하며 당시 소개하던 내용보다 한 단계 더 진화된 Pre-multiplied Alpha라고 불리는 기술을 다음 연재에 이어서 소개하기로 했습니다. 그래서 이번 연재는 지난 연재에 이어 아주 자연스러운 연결로 Pre-multiplied Alpha라고 불리는 방식의 영상합성 방법을 설명하겠습니다.

그림 1. 배경영상과 그래픽의 합성 예

[그림 1]과 같이 배경영상 위에 별 모양의 그래픽 영상을 합성하는 경우, 지난 연재에서 설명한 일반적 방식은 [그림 2]와 같은 Fill과 Key 영상을 사용합니다. 지난 연재를 못 보신 분들을 위해 Fill과 Key 영상을 이용한 영상합성 방법을 간단히 상기하겠습니다. [그림 2] 우측의 Key 영상은 좌측의 Fill 영상이 다른 영상 위에 합성될 때 얼마의 불투명도를 갖고 합성되는지를 나타내는 영상입니다(백색 : 불투명, 검정색 : 투명, 회색 : 반투명). Key 영상이 최대로 백색인 화소에서 Fill 영상은 완벽한 불투명이 되어 배경영상을 완전히 덮고, Key 영상이 최대로 검정인 화소에서 Fill 영상은 완벽한 투명이 되어 배경영상만이 보이게 됩니다. 마지막으로 Key 영상이 회색인 화소에서 Fill 영상은 반투명으로 배경영상에 합성되며, 해당 화소의 Key 영상이 어두울수록 투명도가 높게 배경영상에 합성됩니다.

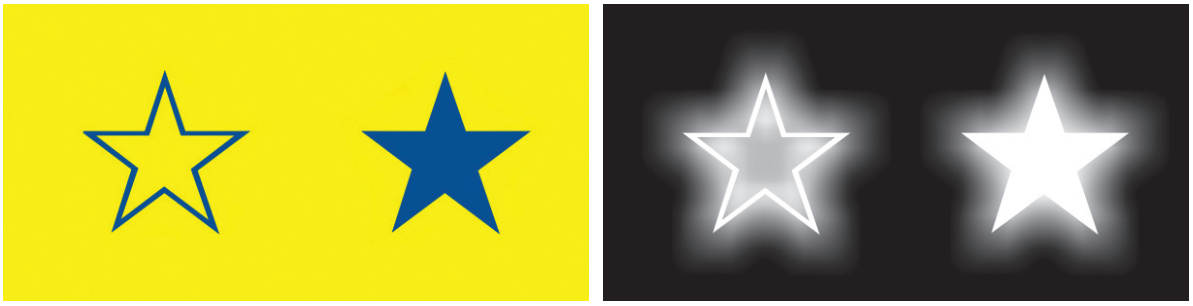

그림 2. 일반적인 Fill(좌)과 Key(우) 영상 (Unshaped Video)

하지만 [그림 2]와 같은 방식의 Fill과 Key 영상만 존재하는 것이 아닙니다. 약간 복잡해 보이지만 Pre-multiplied Alpha라고 불리는 또 다른 방식이 존재하며 [그림 3]과 같은 Fill과 Key 영상을 사용합니다. 참고로 [그림 2]의 일반적 Fill/Key 영상은 Unshaped Video라고 부르며, [그림 3]의 Pre-multiplied Alpha 방식의 Fill/Key 영상은 Shaped Video라고 부르기도 합니다. Shaped Video라는 명칭의 의미를 차차 알게 되겠지만, Fill 영상이 Key 영상에 따른 모양을 갖게 되므로 Shaped Video라고 합니다. 물론, 배경영상 위에 합성된 결과영상은 [그림 2]의 Unshaped Video와 [그림 3]의 Shaped Video 둘 다 똑같습니다.

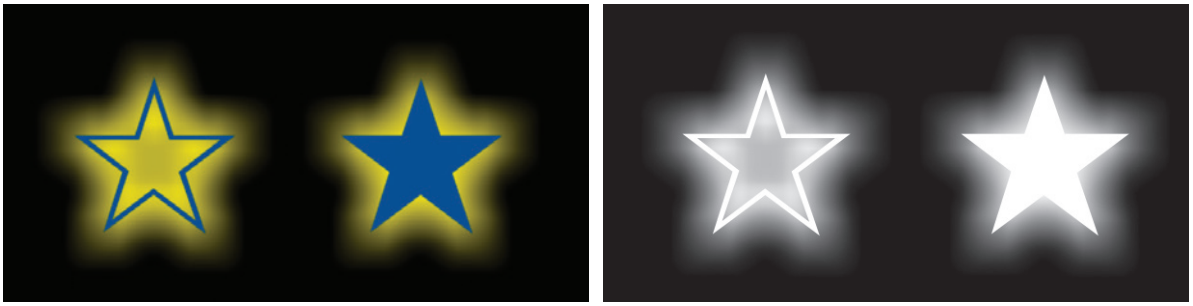

그림 3. Pre-multiplied Alpha 방식의 Fill(좌)과 Key(우) 영상 (Shaped Video)

[그림 2]의 일반적인 방식의 Fill/Key 영상과 [그림 3]의 Pre-multiplied Alpha 방식의 Fill/Key 영상을 비교해보면, 양쪽의 Key 영상은 동일하지만 Fill 영상이 다르다는 것을 누구나 쉽게 알 수 있습니다. 그렇다면 [그림 3]의 Fill 영상은 어떻게 만들어지며, 어떤 원리로 배경영상에 합성되기에 [그림 2]의 일반적 방식과 같은 결과를 내는 걸까요?

우선 Pre-multiplied Alpha 방식의 Fill 영상을 만드는 방법에 관해 [수식 1]을 들어 설명하겠습니다. [수식 1]은 [그림 2]와 [그림 3]의 Fill 영상의 화소별 R, G, B 값을 식으로 정리한 것입니다.

$$R_{Fill\ Shaped} = R_{Fill} \times \frac{Key}{255}$$

$$G_{Fill\ Shaped} = G_{Fill} \times \frac{Key}{255}$$

$$B_{Fill\ Shaped} = B_{Fill} \times \frac{Key}{255}$$

수식 1

[수식 1]의 $R_{Fill\ Shaped}$, $G_{Fill\ Shaped}$, $B_{Fill\ Shaped}$는 [그림 3]의 Pre-multiplied Alpha 방식의 Fill 영상에서 각 화소가 갖는 R, G, B 값, R_{Fill}, G_{Fill}, B_{Fill}는 [그림 2]의 일반적 Fill 영상에서 각 화소가 갖는 R, G, B 값, Key 는 [그림 2]와 [그림 3]의 동일한 Key 영상의 각 화소가 갖는 밝기값을 나타냅니다. [수식 1]에서 $\frac{Key}{255}$ 를 사용한 이유는 0~255 사이의 값을 갖는 8비트 디지털 영상을 사용한다는 가정하에 Key 값을 0~1 사이의 값으로 변환하기 위한 것입니다. [수식 1]을 가지고 간단한 사고실험 수준에서 [그림 2]의 Fill 영상이 [그림 3]의 Fill 영상으로 바뀌는 것을 쉽게 이해할 수 있습니다. [수식 1]을 보면 Key가 0인 곳에서 Pre-multiplied Alpha 방식의 Fill은 검은색이 되는 것을 알 수 있습니다. 또한, Key가 255인 곳에서는 $\frac{Key}{255}$가 1이 되므로 [수식 1]을 통해도 Fill 영상에는 변화가 없게 됩니다. 마지막으로 Key가 회색인 부분에서는 Pre-multiplied Alpha 방식의 Fill 영상이 어두워지며 검은색이 섞인 것처럼 보이게 됩니다. 이유는 [수식 1]의 $\frac{Key}{255}$ 이 1보다 작은 값을 가지게 되어 색을 원래보다 어둡게 만들기 때문입니다.

[수식 1]을 이용해서 Unshaped Video라고 불리는 일반적인 Fill/Key 영상을 Shaped Video라고 불리는 Pre-multiplied Alpha 방식의 Fill/Key 영상으로 바꾸었다면, 배경영상 위에 합성했을 때 [그림 2]와 [그림 3]의 두 종류의 Fill/Key를 이용한 방법들 모두 같은 결과를 얻으려면 어떻게 해야 할까요? 이에 대한 답은 [수식 2]와 [수식 3]을 통해 얻을 수 있습니다.

$$R_{Output} = R_{Fill} \times \frac{Key}{255} + R_{Background} \times (1 - \frac{Key}{255})$$
$$G_{Output} = G_{Fill} \times \frac{Key}{255} + G_{Background} \times (1 - \frac{Key}{255})$$
$$B_{Output} = B_{Fill} \times \frac{Key}{255} + B_{Background} \times (1 - \frac{Key}{255})$$

수식 2

$$R_{Output} = R_{Fill\ Shaped} + R_{Background} \times (1 - \frac{Key}{255})$$
$$G_{Output} = G_{Fill\ Shaped} + G_{Background} \times (1 - \frac{Key}{255})$$
$$B_{Output} = B_{Fill\ Shaped} + B_{Background} \times (1 - \frac{Key}{255})$$

수식 3

[수식 2]는 지난 연재에 소개되었던 식으로써 [그림 2]와 같은 일반적 Fill/Key 그래픽 영상을 이용하여 [그림 1]과 같은 배경영상과 그래픽 영상의 합성을 수행하는 식입니다. [수식 2]의 R_{Output}, G_{Output}, B_{Output}는 [그림 1]의 합성된 영상에서 각 화소가 갖는 R, G, B 값, R_{Fill}, G_{Fill}, B_{Fill}는 [그림 2]의 Fill 영상에서 각 화소가 갖는 R, G, B 값, $R_{Background}$, $G_{Background}$, $B_{Background}$는 [그림 1]의 실사 배경영상과 같이 그래픽과 합성되기 전의 영상에서 각 화소가 갖는 R, G, B 값, Key 는 [그림 2]의 Key 영상의 각 화소가 갖는 밝기값을 나타냅니다. 이 [수식 2]에 [수식 1]을 적용하여 $R_{Fill} \times \frac{Key}{255}$, $G_{Fill} \times \frac{Key}{255}$, $B_{Fill} \times \frac{Key}{255}$을 $R_{Fill\ Shaped}$, $G_{Fill\ Shaped}$, $B_{Fill\ Shaped}$로 바꾸면 [수식 3]과 같이 됩니다. [수식 3]의 $R_{Fill\ Shaped}$, $G_{Fill\ Shaped}$, $B_{Fill\ Shaped}$는 [그림 3]의 Pre-multiplied Alpha 방식의 Fill/Key 영상에서 각 화소가 갖는 R, G, B 값입니다. 이렇게 만들어진 [수식 3]을 이용하여 [그림 3]의 Pre-multiplied Alpha 방식의 영상을 다른 영상 위에 합성할 수 있습니다.

불가피하게 수식과 그림을 오가며 복잡하게 설명되어 핵심을 명쾌하게 짚어내기 힘든 앞의 설명을 간단히 정리하면, [수식 3]은 [수식 2]에 [수식 1]을 통한 치환을 적용하여 만든 식으로써 Pre-multiplied Alpha 방식의 영상을 합성할 때 사용되며, 일반적인 Fill/Key 영상의 합성에 쓰이는 [수식 2]와 등가의 식입니다. 그런데 [수식 3]과 [수식 2]가 등가라면 둘 중 하나의 방법만 쓰지 않고 왜 굳이 복잡하게 두 가지 방법을 만들어서 사용

하는 걸까요? 이 질문에 답하기 위해 [수식 3]의 유도과정과 사용법에 대한 이해를 바탕으로 한 단계 심화된 내용으로 들어가 보겠습니다.

[수식 3]과 [수식 2]를 번갈아 보다 보면, [수식 2]와 비교해 [수식 3]의 우항 앞부분에서 $\frac{Key}{255}$이 사라진 것이 가장 먼저 눈에 띕니다. "설마 $\frac{Key}{255}$하나 없어지는 것 때문에 Pre-multiplied Alpha 방식의 영상합성 방법을 만들었을까?"하고 의아해하실 수 있습니다만, 실제로 계산식에서 $\frac{Key}{255}$하나 없애려고 Pre-multiplied Alpha 방식의 Shaped Video를 사용합니다. 그렇다면 $\frac{Key}{255}$하나가 없는 경우 얻는 이득은 무엇일까요? 뻔하디뻔한 대답이지만 바로 '계산량 감소'라는 이득을 얻게 됩니다. HD를 예로 들어 계산량 감소를 설명해보겠습니다. HD 배경영상에 그래픽 영상을 합성할 경우 각 화소의 R, G, B 값을 계산할 때마다 $\frac{Key}{255}$이 사라지게 되므로 1920×1080×3개의 $\frac{Key}{255}$가 사라지게 됩니다. 그런데 [수식 2]에서 보이듯이 $\frac{Key}{255}$ 하나에 Key를 255로 나누는 나눗셈 한 번과 이 나눠진 값을 R_{Fill}, G_{Fill}, B_{Fill}에 곱해주는 곱셈 한 번이 필요하므로 1920×1080×3개의 $\frac{Key}{255}$가 사라지면 1920×1080×3×2 = 12,441,600번의 나눗셈/곱셈이 사라지게 됩니다. 현행 HD는 1초에 약 30프레임이므로 Pre-multiplied Alpha 방식의 Shaped Video를 사용하면 초당 12,441,600×30 = 373,248,000번의 나눗셈/곱셈이 사라집니다. 게다가 배경영상에 합성되는 그래픽 영상이 하나가 아닌 복수일 경우에는 더욱 엄청난 수의 나눗셈/곱셈이 사라집니다. 제한된 H/W 자원으로 고성능을 구현해야 하는 시스템에서 계산량 절감은 굉장한 이점입니다. 물론 이 사라진 계산들은 공짜가 아니라 Fill/Key 영상을 만들 때 Pre-multiplied Alpha 방식으로 만들어야 가능한 것입니다. 그렇다면 자연히 다음과 같은 질문이 따르게 됩니다.

"Pre-multiplied Alpha 방식으로 Fill 영상을 만들기 위해서는 일반적인 Fill 영상에 [수식 1]을 적용하는 계산이 필요하니 결과적으로 계산의 총량은 절감 안 되는 것 아닌가?"

이에 대한 대답은 "아닙니다. 모든 경우 완벽히는 아니지만 계산의 총량은 대부분 절감됩니다."입니다. 우선 하나의 예를 들어보겠습니다. 배경영상이 동영상이고, 그 배경영상 위에 합성되는 그래픽은 정지영상이라고 가정하겠습니다. 그래픽은 정지영상이므로 한 번만 Pre-multiplied Alpha 방식으로 Fill 영상을 만들면 됩니다. 하지만 배경영상은 동영상이므로 모든 프레임에서 [수식 3]을 적용하여 그래픽을 합성해야 합니다. 세세하게 따지지 않고 식으로만 봐도 [수식 1]은 한 번만 사용했으며, [수식 3]은 모든 프레임에서 매번 반복되므로 엄청난 계산량의 절감이 확실합니다.

그렇다면 위의 경우와 다르게 배경영상도 동영상, 그 위에 합성되는 그래픽도 동영상일 경우 Pre-multiplied Alpha 방식이 가지는 특징은 무엇일까요? 이 경우 다음의 두 가지로 나누어 생각할 수 있습니다.

① 동영상 그래픽이 사전제작 그래픽일 경우

② 동영상 그래픽이 실시간 그래픽일 경우

우선 ①과 같이 그래픽이 실시간 그래픽이 아닌 사전제작 그래픽이라면, 방송시스템 전체에서 제작 시 영상합성을 위해 수행하는 '실시간 계산량'은 [수식 3]에 해당하는 계산만 수행되므로 확실히 절감됩니다. 다음으로 ②와 같은 경우, 모든 영상은 동영상이므로 [수식 1]과 [수식 3] 모두 모든 프레임에서 반복되며, 시스템의 총량으로 볼 때 일견 계산량의 절감은 없어 보입니다. 하지만 그래픽을 생성할 때 미리 연산을 더 해서, 뒤에 이어지는 영상합성 과정에서 계산량을 줄여주는 효과가 있습니다. 이 효과로 인해 전체 제작 과정에서 그래픽의 생성보다 합성 과정이 더 많다면 계산량의 절감이 확실하게 이루어집니다.

지금까지 Shaped Video라고도 불리는 Pre-multiplied Alpha 방식의 Fill/Key 영상을 이용한 영상합성 방법에 대해 소개를 드렸습니다. 다음 연재에서는 영상의 확대 및 축소가 실제로 어떤 계산과정을 통해서 이루어지는지 상세하게 설명드리겠습니다.