

# 국어 영역

## [1~4] 2021수능

최근의 3D 애니메이션은 섬세한 입체 영상을 구현하여 실물을 촬영한 것 같은 느낌을 준다. 실물을 촬영하여 얻은 자연 영상을 그대로 화면에 표시할 때와 달리 3D 합성 영상을 생성, 출력하기 위해서는 모델링과 렌더링을 거쳐야 한다.

최근의 3D 애니메이션은 **섬세한 입체 영상**을 구현하여 실물을 촬영한 것 같은 느낌을 준다. 실물을 촬영하여 얻은 자연 영상을 그대로 화면에 표시할 때와 달리 **3D 합성 영상**을 생성, 출력하기 위해서는 모델링과 렌더링을 거쳐야 한다.

실물을 촬영하여 그대로 화면에 표시하면 되는 자연 영상과는 달리 3D애니메이션에서 실물을 촬영한 것 같은 느낌을 주기 위해서는 모델링과 렌더링을 거쳐 생성, 출력되는 **3D 합성 영상**이라는 섬세한 입체 영상을 구현해야 한다. ‘모델링과 렌더링을 거쳐 3D 합성 영상이 어떻게 구현되는지’ 서두에서의 큰 논의범주 방향 설정 완료.

->

**\*\* 글의 서두에서는 최초의 논의범주를 단순 화제가 아니라 방향으로 설정해야 하기 때문에 특히 신경 써서 의식적으로 문장과 문장을 의미 단위로 연결해주어야 한다.**

-여기서는 3D 애니메이션이 실물을 촬영한 것 같은 느낌을 주기 위해 구현하는 것이 3D 합성 영상임을 연결해주는 것. 그럼으로써 서두에서의 **최초 논의범주**를 ‘어떻게 모델링과 렌더링을 거쳐 **3D 합성 영상을 생성, 출력하는지**’로 잡아주는 것.

Tip. 물론 지시어, 같은 활자 등 형식적 요소의 도움으로 문장들을 연결할 수도 있지만 그래서는 표기에 그칠 뿐 아무 의미도 도출되지 않는다. 되도록 의미 단위로 연결하여 조금 더 글에 대한 이해도를 높이는 방향으로 습관을 들여놓도록 하자.

**모델링**은 3차원 가상 공간에서 물체의 모양과 크기, 공간적인 위치, 표면 특성 등과 관련된 고유의 값을 설정하거나 수정하는 단계이다. 모양과 크기를 설정할 때 주로 3개의 정점으로 형성되는 삼각형을 활용한다. 작은 삼각형의 조합으로 이루어진 그물과 같은 형태로 물체 표면을 표현하는 방식이다. 이 방법으로 복잡한 굴곡이 있는 표면도 정밀하게 표현할 수 있다. 이때 삼각형의 꼭짓점들은 물체의 모양과 크기를 결정하는 정점이 되는데, 이 정점들의 개수는 물체가 변형되어도 변하지 않으며, 정점들의 상대적 위치는 물체 고유의 모양이 변하지 않는 한 달라지지 않는다. 물체가 커지거나 작아지는 경우에는 정점 사이의 간격이 넓어지거나 좁아지고, 물체가 회전하거나 이동하는 경우에는 정점들이 간격을 유지하면서 회전축을 중심으로 회전하거나 동일 방향으로 동일 거리만큼 이동한다. 물체

표면을 구성하는 각 삼각형 면에는 고유의 색과 질감 등을 나타내는 표면 특성이 하나씩 지정된다.

**모델링**은 3차원 가상 공간에서 물체의 모양과 크기, / 공간적인 위치, / 표면 특성 등과 관련된 고유의 값을 설정하거나 수정하는 단계이다.

앞에서 봤던 모델링을 구체화하고 있네. 끊어준 순서대로 각각 그것과 관련한 고유의 값을 어떻게 설정하고 수정하는지 설명해줄 것 같고.

**모양과 크기**를 설정할 때 주로 3개의 정점으로 형성되는 삼각형을 활용한다. 작은 삼각형의 조합으로 이루어진 그물과 같은 형태로 물체 표면을 표현하는 방식이다. 이 방법으로 복잡한 굴곡이 있는 표면도 정밀하게 표현할 수 있다.

모양과 크기 설정할 때 구체적으로 어떻게 하는지부터 설명하네. 3개의 정점으로 형성되는 삼각형들의 조합으로 이루어진 그물과 같은 형태로 물체 표면을 표현해서 복잡한 굴곡이 있는 표면도 정밀하게 표현해낸다고.

이때 삼각형의 꼭짓점들은 물체의 모양과 크기를 결정하는 정점이 되는데, 이 정점들의 개수는 (물체가 변형되어도) -> C 변하지 않으며, / 정점들의 상대적 위치는 (물체 고유의 모양이 변하지 않는 한) -> C 달라지지 않는다. (물체가 커지거나 좁아지는 경우)에는 -> 정점 사이의 간격이 넓어지거나 좁아지고, (물체가 회전하거나 이동하는 경우)에는 -> 정점들이 간격을 유지하면서 회전축을 중심으로 회전하거나 동일 방향으로 동일 거리만큼 이동한다. / 물체 표면을 구성하는 각 삼각형 면에는 고유의 색과 질감 등을 나타내는 표면 특성이 하나씩 지정된다.

이때 삼각형의 꼭짓점들은 물체의 모양과 크기를 결정하는 정점이 되는데 라고 했으니 아직까지 논의범주가 모양과 크기인 것 같네. 삼각형의 꼭짓점들은 물체의 모양과 크기를 결정하는 정점들이고, 이 정점들의 개수는 물체가 변형되어도 변하지 않으며 상대적 위치는 고유의 모양이 변하지 않는 한 달라지지 않는다고. 물체가 커지거나 좁아지는 경우랑 물체가 회전하거나 이동하는 경우는 의미상 고유의 모양이 변하지 않는 한을 구체화한 말들이고 정점 사이의 간격이 넓어지거나 좁아지고, 정점들이 간격을 유지하면서 회전축을 중심으로 회전하거나 동일 방향으로 동일 거리만큼 이동한다는 건 상대적 위치가 달라지지 않는다는 말이구나. 어 그런데 다음 문장에서 표면의 한 면당 하나의 표면 특성이 지정된다고. **그럼 앞 문장에서 공간적인 위치에 대한 말을 했다는 거네.** 어디부터인지 끊어보면 정점들의 개수는 물체가 변형되어도 변하지 않고 뒤의 상대적 위치 이야기부터구나. **그러니까 상대적 위치가 공간적인 위치 얘기였네.** 여기까지 모델링의 각 논의범주별로 어떻게 고유의 값을 설정하고 수정하는지를 읽었다.

->

\* 두 번째 이상의 문단들에서는 앞 문단(들)과 어떠한 관계를 갖는지를 연결해 줌으로써 해당 문단 논의의 진행 방향을 확보할 수 있다.

- 서두에서의 최초 논의범주인 모델링과 렌더링을 거쳐 -> 3D 합성 영상을 생성 중 모델링부터 설명하고 있다고 연결해둠으로써 모델링에서 구체적으로 어떻게 모양과 크기/공간적인 위치/표면 특성과 관련된 고유의 값을 설정하거나 수정하는지를 해당 문단의 방향으로 확보하는 것.

\* 한 문장 내에서 병렬적인 논의가 이어지거나, 동일한 층위/위계의 대상 내지는 주장들이 병렬되면 /로 끊어두면서 항목화 시켜두어야 한다. (한 문장의 분절적인 의미 파악이 곧 이해이다) 이후 진행되는 논의가 어느 항목에 속하는지를 계속 연결해두면서 읽어 내려간다. 즉, 논의범주 단위별로 항목화 해준다.

- 여기서는 단순히 모델링이 -를 설정하거나 수정하는 단계이군이라고 아니라 모델링이 / /를 각각 설정하거나 수정하는 단계이군이라고 분절적으로 파악해두는 것. 실제 이후에 이 항목들의 순서대로 논의가 진행되고 있다. 모양과 크기에 대한 논의가 어디까지 이어지는지를 끊어두고, 공간적인 위치까지 끊어두고, 표면특성까지 끊고.

Tip. 이는 문장과 문장을 연결할 때도 필요한 능력치이다. 지금 읽고 있는 문장과 이전 문장이 첫째로, 둘째로 따위의 명시적 표지 없이 병렬적인 논의를 전개하고 있을 때 그것을 /로 끊어 분절적으로 파악해두지 않으면 후에 표지 없이 이어지는 서술들이 그냥 병렬적인 논의들의 상위범주로만 읽히고 문제에서 서술들이 어느 항목에 붙는지를 물어봤을 때 대처할 수가 없게 된다.

\* 표지 없이 나열되는 문장들 간의 관계는 스스로 설정해두면서 자신이 지금 무엇을 보고 있는지를 생각해 두어야 한다.

- 여기서는 (물체 고유의 모양이 변하지 않는 한) -> 상대적 위치는 변하지 않는다는 문장을 구체화하는 문장으로 뒤의 두 문장을 연결해 두는 것. 즉 아직까지 다음 논의로 넘어가지 않았다는 것.

+ 想以愆 ; 그런데 이렇게 표지 없이 이어지는 서술들을 논의범주 단위별로 끊고 앞의 항목에 붙여두는 데에 있어 선제적으로 반응하기는 여간 어려운 일이 아닐 수 없다. 예컨대 이 지문에서 공간적인 위치라는 항목이 어디서부터 어디까지인지를 읽으면서 바로 알 수는 없는 노릇이다. 그럴 때는 표면특성이라는 다음 항목이 나오기 전까지를 그 전 논의범주로 규정하고, 어디서부터였는지를 찾으려 거슬러 올라가는, 상당히 능동적인 행동이 필요하다. 이것 역시 여러분들이 도달하셔야 하는 상황식 독해의 일종인 것이다.

공간에서의 입체에 대한 정보인 이 데이터를 활용하여, 물체를 어디에서 바라보는가를 나타내는 관찰 시점을 기준으로 2차원의 화면을 생성하는 것이 렌더링이다. 전체 화면을 잘게 나눈 점이 화소인데, 정해진 개수의 화소로 화면을 표시하고 각 화소별로 밝기나 색상 등을 나타내는 화숫값이 부여된다. 렌더링 단계에서는 화면 안에서 동일 물체라도 멀리 있는 경우는 작게, 가까이 있는 경우는 크게 보이는 원리를 활용하여 화숫값을 지정함으로써 물체의 원근감을 구현한다. 표면 특성을 나타내는 값을 바탕으로, 다른 물체에 가려짐이나 조명에 의해 물체 표면에 생기는 명암, 그림자 등을 고려하여 화숫값을 정해 줌으로써 물체의 입체감을 구현한다. 화면을 구성하는 모든 화소의 화숫값이 결정되면 하나의 프레임이 생성된다. 이를 화면출력장치를 통해 모니터에 표시하면 정지 영상이 완성된다.

공간에서의 입체에 대한 정보인 이 데이터를 활용하여, -> 물체를 어디에서 바라보는가를 나타내는 관찰 시점을 기준으로 2차원의 화면을 생성하는 것이 렌더링이다.

이 데이터를 활용하여 2차원의 화면을 생성하는 것이 렌더링이다. 아 쪽 위에 정의를 보니까 모델링은 3차원 가상 공간에서 고유의 값을 설정하고 수정하는 단계라고 했으니까 공간에서의 입체에 대한 정보였나 보고 렌더링은 그것을 활용하여 2차원의 화면을 생성하는 것이네. 모델링에서 렌더링으로 이야기가 넘어가 구체적으로 어떻게 2차원의 화면을 생성하는지 설명할 것 같고.

전체 화면을 잘게 나눈 점이 화소인데, 정해진 개수의 화소로 화면을 표시하고 각 화소별로 <- 밝기나 색상 등을 나타내는 화숫값이 부여된다. 렌더링 단계에서는 화면 안에서 동일 물체라도 멀리 있는 경우는 작게, 가까이 있는 경우는 크게 보이는 원리를 활용하여 -> 화숫값을 지정함으로써 => 물체의 원근감을 구현한다. 표면 특성을 나타내는 값을 바탕으로, 다른 물체에 가려짐이나 조명에 의해 물체 표면에 생기는 명암, 그림자 등을 고려하여 -> 화숫값을 정해줌으로써 => 물체의 입체감을 구현한다. / (화면을 구성하는 모든 화소의 화숫값이 결정되면) -> 하나의 프레임이 형성된다. 이를 (화면출력장치를 통해 모니터에 표시하면) 정지영상이 완성된다.

화면을 나눈 점인 화소에 화숫값을 부여한다고. 아 렌더링에서는 구체적으로는 동일 물체라도 멀리 있으면 작게, 가까이 있으면 크게 보이는 원리를 활용해서 화숫값을 부여해 원근감 구현. 표면 특성값 바탕으로 명암, 그림자 등 고려해서 화숫값을 부여해 입체감 구현한다고. 이렇게 모든 화소의 화숫값을 결정해서 형성된 하나의 프레임을 화면출력장치로 모니터에 표시하면 정지영상이 완성된다고 하네. 여기까지 화소에 화숫값을 부여한 뒤 하나의 프레임을 형성하고 2차원 화면에 표시하는 데까지, 렌더링을 읽은 것 같다.

->

\* 두 번째 이상의 문단들에서는 앞 문단(들)과 어떠한 관계를 갖는지를 연결해 줌으로써 해당 문단 논의의 진행 방향을 확보할 수 있다.

-서두에서의 최초 논의범주인 모델링과 렌더링을 거쳐 -> 3D 합성 영상을 생성 중 모델링까지 읽었고 렌더링에서 어떻게 2차원의 화면을 생성하는지를 방향으로 설정하는 것.

\*\*\* 앞에서 정리한 개념/서술 등이 재진술,구체화/대비되고 있을 때 그것을 끌고 내려와서 넣어 읽는다. => 상향식 독해

-여기에서는 공간에서의 입체에 대한 정보인 이 데이터를 보고 모델링은 3차원 가상 공간에서 고유의 값을 설정하고 수정하는 단계였다는 것을 끌고 내려와서 3차원 입체에 대한 정보인 그 데이터를 활용하여 2차원의 화면을 생성하는 것이 렌더링이네 하고 넣어 읽는 것.

Tip. 상향식 독해의 단초를 여기서의 '이'와 같은 지시어나 보조사(형식적 요소들=활자)로만 정리해둔다면 시험장에서 큰일이 생긴다. 결과적으로 그냥 펜으로 두 문장을 겹으로만 이어둘 수가 있기 때문이다. 반드시 3차원 가상 공간에서의 입체에 대한 정보를 받아 2차원 화면을 생성한다는 '의미'를 살려서 '이해'해주셔야 한다.

\* 한 문장의 의미를 분절적으로 파악하는 것이 문장의 '이해'이다.

-여기에서는 단순히 렌더링의 정의이다라고 구조적으로 파악하지 말고, '~이 데이터를 활용하여 -> 관찰시점을 기준으로 2차원의 평면을 만드는 것'이 렌더링이라고 이해해두어야 한다.

Tip. 한 문장의 의미를 분절적으로 파악하는 방식을 모두 유형화할 수는 없지만, 크게 선후, 인과, 조건, 항목화 등이 있다. 이는 나열되는 문장들의 의미를 분절적으로 파악해 연결해 두는 방식이 길어진 문장 안으로 삽입된 것으로 보면 된다. 선후는 일련의 순서를 갖는, 화살표로 표시할 수 있는 정보들을 -> 분절적으로 파악해주는 것. 인과는 원인과 <-( 결과 )로 파악할 수 있는 정보들을 분절적으로 파악해주는 것. 조건은 ( 경우 ) -> 어떠한지를 분절적으로 파악해주는 것. 항목화는 병렬적으로 나열된 대상, / 주장, / 논의들을 끊어서 분절적으로 파악해주는 것이다. 물론 계속 강조하지만 활자 단위가 아니라 의미 단위로 분절화 시켜서 이해하려고 노력 해주셔야 한다. 자꾸 활자나 형식적 요소 등을 통해 구조적으로만 파악하려고 하시면 내용을 이해하는 것이 목적인 '올바른 독해'에서 점점 멀어지게 된다.

\* 문장과 문장을 붙여 읽어서 의미를 도출하는 것은 '활자'가 아니라 '의미'단위로 하셔야 한다.

-여기에서는 '화면'이라는 같은 글자 단위가 아니라, 2차원의 화면을 생성할 때 구체적으로는 전체 화면을 잘게 나눈 화소에 화소값을 부여하는 방식으로 진행하네 라고 붙여 읽어주셔야 한다.

Tip. 물론 단초가 어 화면? 방금 문장에서의 2차원의 화면? 처럼 형식적 요소=활자가 될 수 있겠으나 붙여 읽으신 결과가 그냥 펜으로 문장과 문장 겹으로 이어두는 것이 되어서는 안된다는 이야기이다.

\* 표지 없이 나열되는 문장들 간의 관계는 스스로 설정해두면서 자신이 지금 무엇을 보고 있는지를 생각해 두어야 한다.

-여기에서는 화면을 구성하는 화소들에 화소값을 부여한다는 구도를 잡아주는 두 번째 문장과 뒤의 두 문장들의 관계를 '구체적으로 어떻게 화소값을 부여하여 결과로 무엇을 구현하는지'로 설정해 두는 것.

+ 想以懸 ; 그런데 이렇게 표지 없이 이어지는 서술들을 논의범주 단위별로 끌고 앞의 항목에 붙여두는 데에 있어 선제적으로 반응하기는 여간 어려운 일이 아닐 수 없다. 예컨대 여기에서는 화소에 화소값을 부여한다는 구도를 잡은 두 번째 문장 뒤에 이어지는 두 문장을 구체적으로 어떻게 화소값을 부여하여 결과로 무엇을 구현하는지로 붙여주기가 쉽지만은 않다. 그럴 때는 다음 문장에서 모든 화소의 화소값이 결정되고 와 같이 논의를 정리해줄 때 어 지금까지 화소의 화소값을 결정하는 과정이었구나 하고 잠시 눈을 위로 돌려서 정리해주시고 넘어가면 되겠다. (상향식 독해)

모델링과 렌더링을 반복하여 생성된 프레임들을 순서대로 표시하면 동영상이다. 프레임을 생성할 때, 모델링과 관련된 계산을 완료한 후 그 결과를 이용하여 렌더링을 위한 계산을 한다. 이때 정점의 개수가 많을수록, 해상도가 높아 출력 화소의 수가 많을수록 연산 양이 많아져 연산 시간이 길어진다. 컴퓨터의 중앙처리장치(CPU)는 데이터 연산을 하나씩 순서대로 수행하기 때문에 과도한 양의 데이터가 집중되면 미처 연산되지 못한 데이터가 차를 기다리는 병목 현상이 생겨 프레임이 완성되는 데 오랜 시간이 걸린다. CPU의 그래픽 처리 능력을 보완하기 위해 개발된 ㉠그래픽처리장치(GPU)는 연산을 비롯한 데이터 처리를 독립적으로 수행할 수 있는 장치인 코어를 수백에서 수천 개씩 탑재하고 있다. GPU의 각 코어는 그래픽 연산에 특화된 연산만을 할 수 있고 CPU의 코어에 비해서 저속으로 연산한다. 하지만 GPU는 동일한 연산을 여러 번 수행해야 하는 경우, 고속으로 출력 영상을 생성할 수 있다. 왜냐하면 GPU는 한 번의 연산에 쓰이는 데이터들을 순차적으로 각 코어에 전송한 후, 전체 코어에 하나의 연산 명령어를 전달하면, 각 코어는 모든 데이터를 동시에 연산하여 연산 시간이 짧아지기 때문이다.

(모델링과 렌더링을 반복하여 생성된 프레임들을 순서대로 표시하면) -> 동영상이다. // 프레임을 생성할 때, 모델링과 관련된 계산을 완료한 후 그 결과를 이용하여 -> 렌더링을 위한 계산을 한다. <- 이때 정점의 개수↑가 많을수록, 해상도↑가 높아 -> 출력 화소의 수↑가 많을수록 -> 연산 양↑이 많아져 => 연산 시간이 길어진다. 컴퓨터의 중앙처리장치(CPU)는 (데이터 연산을 하나씩 순서대로 수행하기 때문에) -> (과도한 양의 데이터가 집중되면) -> 미처 연산되지 못한 데이터가 차를 기다리는 병목 현상이 생겨 => 프레임이 완성되는 데 오랜 시간이 걸린다.

앞에서 렌더링까지 거쳐 형성된 프레임들을 순서대로 표시하면 동영상이 된다고. 아 정지영상들을 순서대로 합쳐놓은 이게 3D 합성 영상이네. 서두에서의 첫 논의범주 닫고. 근데 프레임을 생성할 때라면 아까 렌더링 얘기인데. 모델링과 관련된 계산을 완료한 후에 그 결과를 이용해서 렌더링을 위한 계산을 한다고. 이때면 렌더링 계산할 때고 정점의 수와 출력 화소의 수가 많으면 연산 양 많아져서 연산 시간 늘어난다고. CPU가 데이터 처리를 순서대로 해서 처리할 데이터가 많으면 병목현상 생겨 오랜 시간 걸린다는 것은 이걸 구체화해 설명해준 거네. 정점의 수와 출력 화소의 수가 데이터라고. 연산 시간이 많이 걸리는 게 문제면 그걸 어떻게 해결하는지도 나오겠네.

CPU의 그래픽 처리 능력을 보완하기 위해 <- 개발된 그래픽처리 장치(GPU)는 연산을 비롯한 데이터 처리를 독립적으로 수행할 수 있는 장치인 코어를 수백에서 수천 개씩 탑재하고 있다. GPU의 각 코어는 그래픽 연산에 특화된 연산만을 할 수 있고 / CPU의 코어에 비해서 저속으로 연산한다. 하지만 GPU는 동일한 연산을 여러 번 수행해야 하는 경우, 고속으로 출력 영상을 생성할 수 있다. <- (왜냐하면 GPU는 (한 번의 연산에 쓰이는 데이터들을 순차적으로 각 코어에 전송)한 후, -> (전체 코어에 하나의 연산 명령어를 전달하면), 각 코어는 모든 데이터를 동시에 연산하여 -> 연산 시간↓이 짧아지기 때문이다.

아 GPU가 CPU에서 생긴 연산시간 오래 걸리는 문제를 해결하겠다고 그걸 그래픽 처리 능력 문제라고 한다고. 코어가 되게 많네. 이게 그래픽 연산에 특화된 연산만을 하면 도움이 되겠는데 저속으로 연산한다고. 근데도 아까 CPU가 동일 연산을 여러 번 할 때 병목 현상으로 연산시간이 오래 걸린 문제를 해결할 수 있다고. 아 데이터를 처리할 코어에 다 보내놓고 한 번에 연산하네. CPU처럼 순서대로 데이터 하나씩 연산하는 게 아니니까 문제의 이유를 잘 해소했네. 연산시간 짧아지겠고.

->

\* 서두에서의 첫 논의범주가 닫히는 시점을 파악하는 것은 바로 뒤에서 새로운 논의범주를 잡으려는 의식적 노력을 기울이기 위함이다.

-여기에서는 동영상이 된다는 보고 프레임을 화면에 순서대로 표시해 정지영상들을 결합해서 3D 합성 영상을 만든다는 것을 파악한 뒤, 프레임을 생성할 때를 렌더링의 구체화로 잡아주는 것.

\* 표지 없이 논의범주가 전환될 때는 기민하게 반응하여 앞 내용과 연결해 두어야 한다. 만약 연결되지 않는다면 새로운 논의범주이므로 잠시 기존 이야기로 돌아올 때까지 버텨준다.

-여기에서는 프레임을 생성할 때 라는 말을 보고 위의 렌더링에서 화솟값 부여한 뒤 하나의 프레임 형성하는 부분과 연결해주는 것.

\* 표지 없이 나열되는 문장들 간의 관계는 스스로 설정해두면서 자신이 지금 무엇을 보고 있는지를 생각해 두어야 한다.

- 여기서는 정점의 수와 출력 화소의 수가 많아 연산 양 많아지고 연산시간 늘어난다는 문장의 구체화로 뒤 문장을 규정해두는 것. 그럼으로써 정점의 수와 출력 화소의 수가 데이터라는 것과 연산시간이 길어지면 프레임 생성에 시간이 오래 걸려 문제가 발생한다는 의미를 도출해내는 것.

\* 한 문장의 의미를 분절적으로 파악하는 것이 문장의 '이해'이다.

- 여기에서는 GPU의 각 코어는 그래픽 연산에 특화된 연산만을 할 수 있고 / CPU의 코어에 비해서 저속으로 연산한다는 분절적으로 파악해 그래픽 연산에 특화된 연산만을 한다면 도움이 되겠는데 저속으로 연산하는 게 도움이 되는지에 대한 생각을 한 뒤, 자연스럽게 뒤에서 그럼에도 고속으로 연산할 수 있다는 말에 '어떻게'라는 궁금증을 갖는 것.

\* 문제와 해결이라는 흐름에서 중요한 것은 '문제가 왜 문제인지', '해결책이 그 문제의 이유를 어떻게 해소하는지'를 이해하는 작업이다. 단순히 문제와 해결이라는 활자만 읽어내면 아무런 의미가 생기지 않는다.

- 여기에서는 단순히 연산시간이 짧아져서 문제가 해결되었다고 읽어내지 말고, CPU에서 순서대로 데이터를 연산해서 병목현상으로 연산시간이 길어졌던 문제를, GPU에선 일단 코어에 데이터 다 보내놓고 한 번에 연산함으로써 문제의 이유를 해소했음을 이해하는 것.

+ 想以慰 ; 기술지문의 일반적인 흐름. 여러분들이 보시는 17학년도 이후의 기술지문들은 16학년도까지 이과 학생들이 특정 과학 지문들에서 너무 유리하다는 여론이 형성된 이후 새롭게 등장한 것들로, 나름의 일반적인 흐름을 갖고 있다. 물론 이 사실을 알고 있다는 것 자체가 여러분들에게 큰 도움이 되지는 않을 것이고, 지문 읽을 때 참고 정도의 도움을 받을 수 있으면 좋겠다. 기술지문은 보통 특정 기술의 목적을 구현하기 위한 장치가 소개되고, 서두에서 장치가 -> 목적을 구현하는 것이 방향으로 설정되곤 한다. 이후에는 장치를 구성하는 요소들 각각의 역할이 이어지며 기술의 작동원리를 형성하고 목적을 구현하는 데까지가 첫 번째 논의범주로 잡힌다. 하지만 이것은 언제나 완벽하지 않아서, 일정한 한계나 결함을 갖기 마련이고, 그것을 어떻게 수정, 보완하는지 까지가 두 번째 논의범주로 잡힌다. 여러분들이 이 사실을 알고 있음으로써 얻을 수 있는 효용은 첫째 기술 지문에서 서두에서 잡아내야 할 첫 논의범주(방향/흐름)이 장치의 -> 목적구현이라는 것. 둘째는 장치를 구성하는 요소가 어떠한 역할을 하는지를 분절적으로 읽어두어야 한다는 것. (문제에서 항목을 뒤섞어서 물어본다) 셋째 목적구현이 어느 단계에서 이루어지는지 혹은 특정 요소가 결정적 역할을 하는지에 주목해두는 것. (역시 작동원리 마지막 즈음에 구현되지 않는다면 문제에서 어느 단계냐를 물어보며 어떤 요소가 결정적인 역할을 했는지를 정답선지로 쫓는다) 넷째 수정 보완하는 흐름에서 어떻게 문제의 이유를 해소하는지를 이해해두어야 한다는 것. 다만, 기술지문도 평가원 독서지문이다. 얼마든지 범주가 가 능하니 내신식으로 외워서 어떻게 해보겠다는 생각은 버릴 것.

1. ②

정답 해설 : ② 표면특성이 어떻게 결정되는지에 대한 논의범주는 읽었던 기억이 없다. 표면특성과 화솨값이 어떤 관계가 있는지 찾으러 돌아가 보니 오히려 화솨값을 결정하는 방식 두 번째에서 표면특성을 나타내는 값을 바탕으로 명암과 그림자 등을 고려하여 화솨값을 정해준다고 했었다. 표면특성이 화솨값에 의해 결정되는 것이 아니라 화솨값이 표면특성에 의해 결정되는 것이다. 인과아웃.

- ① 자연 영상은 실물을 촬영하여 얻은 것. 그것과 같은 느낌을 주기 위해 3D애니메이션에서는 3D 합성 영상을 구현한다는 맥락으로 읽었다. 그 3D 합성 영상을 생성, 출력하기 위한 것이 모델링과 렌더링이므로 자연 영상에서는 모델링과 렌더링 단계가 필요 없다.
- ③ 물체의 원근감과 입체감은 렌더링 단계에서 화솨별로 화솨값을 부여하는 구체적 방식들에서 봤었다. 렌더링이 관찰 시점을 기준으로 2차원 화면을 생성하는 단계이니 물체의 원근감과 입체감이 구현되는 기준이 관찰 시점이라고 하면 맞겠다.
- ④ 해상도가 높으면 연산 양이 많아져 연산 시간이 길어지는 문제가 생기는 흐름이었다.
- ⑤ 병목 현상은 CPU가 데이터를 순차적으로 하나씩 연산하다 보니 처리해야 할 데이터가 쌓였을 때 그것을 연산하는 데 오랜 시간이 걸리는 문제를 말하는 것이었다. 따라서 연산할 데이터의 양이 처리 능력을 초과할 때 생기는 것이라고 할 수 있겠다.

2. ②

정답 해설 : ② 물체의 복잡한 곡면을 표현하는 것에 대한 이야기로 돌아가 보면 작은 삼각형의 조합으로 이루어진 그물과 같은 형태로 물체 표면을 표현하여 복잡한 굴곡이 있는 표면도 정밀하게 표현할 수 있다고 나와 있다. 사실관계 적절.

- ① 가려져 보이지 않는 부분들에 대한 이야기는 돌아가 보면 렌더링에서 화솨값 부여하는 방식 두 번째에 다른 물체에 가려짐이나 조명에 의해 물체 표면에 생기는 명암, 그림자 등을 고려하여 화솨값을 정해 준다고 되어 있다. 모델링이 아니라 렌더링에 대한 설명이다. 항목 아웃.
- ③ 표면 특성 부여하는 것에 대한 이야기는 모델링 쪽에 항목으로 잡아 두었다. 돌아가 보면 삼각형 하나에 하나의 색상의 표면특성을 부여하는 것이므로 사실관계 아웃.
- ④ 정점들로 표현된 물체를 2차원 평면에 존재하도록 배치하는 것은 렌더링 단계에서의 이야기이다. 항목 아웃.
- ⑤ 관찰시점들을 순차적으로 저장하는 것은 동영상 만들 방식이었다. 모델링이 아니라 3D 합성 영상에 대한 설명이다. 항목 아웃.

+ 想以慰 ; 제발 이렇게 큰 논의범주 하나에 대해 물어봤을 때 그 논의범주로 자꾸 돌아가서 그 문장들하고 선지하고 비벼대지(?) 마라. 서술을 바탕으로 항목(논의범주)으로 돌아가라. 대부분 항목 아웃으로 처리할 수 있다. 예컨대 모델링에 대해 물어보는 것처럼 보였지만 실상은 렌더링에 대한 선지거나 3D 합성 영상에 대한 선지거나 할 수가 있다는 말이다.

3. ④


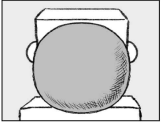
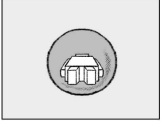
정답 해설 : ④ (데이터 연산을 하나씩 순서대로 처리해야 한다면) -> 다수의 코어가 작동하는 경우의 연산시간 = 1개의 코어를 가진 경우의 연산시간인가. GPU는 CPU에서 데이터 연산을 순차적으로 수행해서 처리가 밀리는 문제의 이유를 코어에 데이터 다 보내 놓고 한 번에 연산함으로써 해소했다. 그런데 CPU랑 동일하게 순차적으로 처리하게 할 경우 1개의 코어를 가지나 다수의 코어를 가지나 한 번에 하나씩만 처리할 수 있어서 딱히 연산시간이 빨라지지 않을 것이다. 따라서 순차적으로 연산해야 하면 다수의 코어나 1개의 코어나 연산 시간이 동일하다. 사실 관계 적절.

- ① 동일 개수의 정점 위치 연산 시, 코어 개수가 많아지면 -> 총 연산시간이 길어지는가. 코어 개수가 많아지면 데이터를 보다 많이 한 번에 처리할 수 있겠다. 연산시간은 짧아지겠다. 사실 관계 아웃.
- ② 10개의 연산을 10개의 코어에서 동시에 진행하려면 <- 10개의 연산 명령어가 필요한가. 연산명령어는 잘 기억 안 나는데 돌아가 보면 데이터 코어에 다 보내 놓고 하나의 연산 명령어를 전달하여 동시에 연산한다고 했으므로 10개의 코어에서 동시에 진행해도 1개의 연산 명령어만 있으면 되겠다. 사실 관계 아웃.
- ③ GPU에서 1개의 코어만 작동할 때의 연산시간 = 1개의 코어를 가진 CPU의 연산시간인가. 아니다. 코어 하나의 연산 시간은 CPU보다 길어서 잠시 의아해했던 기억이 있다. 사실 관계 아웃.
- ⑤ 10개의 데이터를 10개의 코어에서 처리할 경우 전송시간 = 1개의 데이터를 1개의 코어에 전송하는 시간인가. 모른다. 전송 시간에 대한 이야기는 언급되지 않았다. 논의범주 없음.

+ 異想以慰 ; 새롭게 추가된 정답 근거? 논의 범주 없음. 최근 들어 정답 선지 아닌 선지들에서 조금 많이 눈에 들어와서 이쯤에서 언급한다. 여러분들이 지문을 전부 '이해'하지 못해서서가 아니라 지문에 애초에 그런 이야기가 없어서 틀린 선지일 수도 있다. 특히 2022 예비평가 기술지문의 31번의 2번 선지를 보시면, 충전 허용 전류보다 전류의 세기가 크면 문제가 생긴다는 언급이 있는데 지문에는 전류의 세기가 일정해야 한다는 말만 있지 왜 일정해야 하는지, 또 전류의 세기가 충전허용전류보다 어때야 하는지에 대한 언급은 전혀 없다. 혹자는 충전허용전류보다 전류의 세기가 크면

안 되기 때문에 일정하게 해야 한다는 말이 있는 것이라고 할 수도 있겠지만 정말 시험 시간 안에 그런 이해를 갖추는 것이 가능하다고 보는가? 마찬가지로 이 문제의 5번 선지에서 전송시간이 어떠한지에 대해 시험 시간 안에 이 지문만 읽고 이해를 가지는 것이 가능하다고 보는가? 여러분들은 지문을 출제하는 교수도, 지문과 문제를 분석해서 여러분들을 가르치는 저도 아닙니다. 시험 시간 안에 어떤 판단을 해야 할지를 생각해 보세요. 지문에 그런 이야기 없었다고 생각하고 '논의 범주 없음'으로 처리하는 것이 가장 '호트러짐이 없이 올바른' 사고와 행동입니다. 예비평가에 정답선지로 꽂혀 있길래 슬금슬금 기어 나올 것 같아서 언급했습니다.

4.

	[장면 구상]	[장면 스케치]
장면 1	주인공 '네모'가 얼굴을 정면으로 향한 채 입에 아직 불지 않은 풍선을 물고 있다.	
장면 2	'네모'가 바람을 불어 넣어 풍선이 점점 커진다.	
장면 3	풍선이 더 이상 커지지 않고 모양을 유지한 채, '네모'는 풍선과 함께 하늘로 날아올라 점점 멀어지는 모습이 보인다.	

# <보기> 분석 - 장면 1을 보고는 아무 생각도 들지 않는다. 장면 2를 보고 지문과 연결 지을 수 있었다면 풍선이 점점 커진다고 했으니 모델링 쪽에 물체가 커지거나 작아지는 경우 -> 정점 사이의 간격이 넓어지거나 좁아진다고 잡아줄 수 있겠다. 장면 3에선 지문과 연결 지을 수 있었다면 더 이상 커지지 않고 모양을 유지한 채 점점 멀어진다고 했으니 동일 물체라도 멀리 있는 경우는 작게 보이는 원리를 활용하여 -> 화숫값을 지정함으로써 => 원근감을 구현한다로 잡아줄 수 있겠다.

정답 해설 : ④ 장면 3의 모델링 단계에서 / 풍선에 있는 정점들이 이루는 삼각형들이 작아지는가. 정점들이 이루는 삼각형들이 무엇에 따라 작아지는지 지문으로 돌아가 보면 모델링 쪽에는 삼각형이 작아진다는 서술이 없고 렌더링 쪽에서 동일 물체라도 멀리 있는 경우는 작게, 가까이 있는 경우는 크게 보이는 원리를 활용하여 -> 화숫값을 지정함으로써 => 원근감을 구현한다는 서술밖에 없다. 네모가 점점 멀어진다고 했으니 작게 보이는 원리를 활용하여 화숫값을 지정하는 것은 맞지만 삼각형이 작아지는지는 확실치 않고 맞다고 하더라도 모델링이 아니라 렌더링쪽 이야기이다. 항목 아웃.

① 장면 1의 렌더링 단계에서 / 가려 보이지 않는 입 부분의 삼각형들의 표면 특성이 -> 화숫값을 구하는 데 사용되지 않는다. 화숫값을 부여하는 렌더링 단계의 서술로 돌아가 보면 표

면 특성을 나타내는 값을 바탕으로 다른 물체에 가려짐이나 조명에 의해 생기는 명암, 그림자 등을 고려하여 화숫값을 정해줌으로써 물체의 입체감을 구현해준다고 나와 있다. 표면 특성을 바탕으로 화숫값을 정해주는 것은 가려진 부분이 아니라 2차원 평면에서 그 위에 있는 보이는 부분이며, 고려하는 것도 가려진 부분의 표면 특성이 아니라 가려짐에 의해 생기는 명암과 그림자 등이므로 가려 보이지 않는 입 부분의 삼각형들의 표면 특성은 화숫값을 구하는 데 사용되지 않는다. 사실 관계 적절.

② 장면 2의 모델링 단계에서 / 풍선에 있는 정점의 개수는 유지되는가. 정점의 개수가 무엇에 따라 유지되고 변하는지 지문으로 돌아가 보면 모델링 쪽에 정점들의 개수는 <- 물체가 변형되어도 변하지 않는다는 서술이 있다. 풍선이 점점 커져도 정점들의 개수는 변하지 않겠다. 사실 관계 적절.

③ 장면 2의 모델링 단계에서 / 풍선에 있는 정점 사이의 거리가 멀어지는가. 정점 사이의 거리가 무엇에 따라 멀어지는지 지문으로 돌아가 보면 물체가 커지는 경우임을 알 수 있다. 풍선이 점점 커진다고 했으므로 정점 사이의 거리도 이에 따라 멀어지겠다. 사실 관계 적절.

⑤ 장면 3의 렌더링 단계에서 / 전체 화면에서 화숫값이 부여되는 화소의 개수는 변하지 않는가. 화숫값이 부여되는 화소의 개수가 변하는가 변하지 않는가 돌아가 보자니 전체 화면을 잘게 나눈 점인 화소마다 화숫값을 부여한다는 서술과 그것의 구체화된 문장 2개밖에 없다. 화소의 개수를 변하게 만드는 요인은 없으므로 화소의 개수는 변하지 않는다. 사실 관계 적절.