Computer Graphics 과제 3 보고서

컴퓨터 공학부

2004-11881 고우종

# 개발 환경

1. Microsoft Windows 7 Professional K (64bit)
2. Microsoft Visual Studio 2010 Professional
3. Mac OS X 10.6.7

# 실행 방법

Hw.sln 파일을 열어서 hw3 프로젝트 선택 후 Release나 Debug 모드에서 Run (F5)을 하시면 컴파일 및 실행이 됩니다. 이번 과제는 이전 과제들과는 달리 속도가 조금 요구되므로 되도록이면 release 모드에서 실행하시는 것을 추천 드립니다. 그리고 파일 입력은 “data.txt” 파일로 받습니다.

혹시 **hw1** 프로젝트가 선택되어 컴파일 된다면, 왼쪽 사이드 바에서 hw3 프로젝트에서 오른쪽 버튼을 클릭하여 “Set as StartUp project”를 클릭 후 다시 컴파일 및 실행해주시기 바랍니다. 설정하여 제출하였는데 이상하게 가끔씩 풀리는 경우가 있는 것 같습니다.

# Swept Surface 구현

이번 세 번째 과제 Swep surface 만들기는 우선 cross section의 control point들을 입력 받아서 spline으로 닫힌 곡선을 그리는 것으로 시작합니다. 그러고 나서 위치, 방향, 크기의 변화 역시 spline으로 interpolation 한 후 해당 cross section들을 하나하나 위치로 이동시키고 방향을 맞추고 크기를 조절한 다음 외곽점들을 triangle strip을 이용하여 연결해주는 것으로 끝납니다.

참고로 triangle strip으로 된 곡면 말고 wireframe 형태로도 볼 수 있는데 그 키는 아래와 같습니다. 라이트가 잘 안 비춰지는 곳의 구조를 자세히 보고 싶다거나 아니면 내부 구조가 궁금할 때 매우 유용합니다.

* ‘w’ – wireframe 모드로 보기

대략 위와 같은 방식으로 Swept surface를 생성하는데, spline 생성을 위해 Catmull-Rom, B-spline, Natural cubic spline, cubic b-spline subdivision 그리고 interpolatory subdivision 기능을 구현하였습니다. 기본 요구사항 및 Extra로 제시된 모든 spline 생성 방법들을 모두 다 구현했습니다. 그리고 cross section 생성시에 쓰이는 spline과 위치, 크기, control point의 interpolation에 쓰이는 spline을 구분하여 컨트롤할 수 있도록 구현했습니다.

구현 방법이나 알고리즘들은 대부분 강의 PPT에 있는 내용들을 참고하여 구현하였습니다. 특히 Natural cubic spline은 연립방정식을 코드 내에서 푸는 것이 어려운 점이 많았습니다. 그래서 PPT에 있는, Natural cubic spline 형태를 B-spline 형태로 변환하는 행렬을 사용하여 Natural cubic spline의 control point들을 B-spline의 control point들로 변환 후 구현해둔 B\_Spline() 함수를 이용하여 생성했습니다.

기본적인 Bezier curve나 B-spline 생성은 매트릭스를 이용하지는 않았고 그냥 basis function들과 컨트롤 포인트의 곱으로 계산했습니다. 그리고 각 curve들의 parameter t에서의 tangent vector를 derivation을 통해 구해뒀는데, 이를 뒤에 렌더링 과정에서 변환 후 normal vector로 이용했습니다. 하지만 subdivision의 경우 normal vector를 구하는 법을 배우지 않아서, 오늘 교수님께서 질문 드렸더니 근사해서 계산하라는 답변을 받았습니다. 따라서 subdivision들은 주변의 다른 vertex들과의 차를 이용하여 normal vector를 approximation하였습니다. 그러한 연유로 subdivision 방법들로 cross section을 그릴 경우 몇몇 상황에서 normal vector가 정확하지 않아서 표면의 색이 좀 부드럽지 않게 표시되는 경우가 있습니다.

그리고 추가적으로 저는 크로스 섹션 수정 모드에서 컨트롤 포인트가 새로 추가되어도 다른 크로스 섹션들이 영향 받게 하고 싶지 않았기 때문에 컨트롤 포인트 보간에서는 조금 다른 방법을 택했습니다. 컨트롤 포인트끼리를 보간하는게 아니라 그 컨트롤 포인트를 이용해 이미 그려진 크로스 섹션 외곽선 위의 점 하나하나를 가지고 보간을 하였습니다. 즉 여러 크로스 섹션에서 같은 index를 가지는 점들을 모아서 그 점들을 가지고 새로 spline을 하여 중간 외곽선들을 구했습니다. 따라서 컨트롤 포인트 개수가 크로스 섹션마다 서로 상이하여도 중간 크로스 섹션들을 만들어 낼 수 있습니다.

마지막으로 한가지 구현사항에 있어서 주의가 필요한 부분은 Catmull-Rom이나 B-spline 등을 spline 생성 방법으로 사용할 경우 **첫 번째 크로스 섹션과 마지막 크로스 섹션은 화면에 표시되지 않으므로** 데이터에 입력해도 화면상에서 확인할 수 없다는 점입니다. 이는 강의시간에 교수님께서 Catmull-Rom의 양 끝점의 tangent 값이 정의되지 않기 때문에 양끝 섹션은 표시하지 않기로 한 것을 따랐습니다.

그럼 이제 크로스 섹션의 외곽선 생성과 위치, 크기, 컨트롤 포인트 보간을 하는 spline을 선택하는 방법을 설명 드리겠습니다. 우선 크로스 섹션의 외곽선 생성 방법은 키보드에서 1에서 5까지의 숫자를 누르는 것으로 바꿀 수 있습니다.

‘1’ – B-spline

‘2’ - Catmull-Rom

‘3’ – Natural Cubic Spline

‘4’ – B-spline Subdivision

‘5’ – Interpolating Subdivision

마찬가지로 키보드에서 6에서 0까지의 숫자를 누르면 위치, 크기, control point의 spline을 생성하는 spline 방법들을 바꿀 수 있습니다.

‘6’ – B-spline

‘7’ - Catmull-Rom

‘8’ – Natural Cubic Spline

‘9’ – B-spline Subdivision

‘0’ – Interpolating Subdivision

이는 나중에 나오는 cross section 수정 모드 (키보드 ‘e’)에서도 어떤 방법으로 그리고 있는지를 화면의 표시되는 텍스트를 통해 알 수 있습니다. 그리고 참고로 방향을 보간하는 spline의 경우 강의와 PPT를 통해 배운 방법이 unit quaternion을 De Casteljau Algorithm과 SLERP을 통해서 만드는 방법 밖에 없었으므로 방향을 보간하는 방법은 이 방법만을 사용하였고 따로 바꿀 수 있게 하지 않았습니다.

# Cross section 수정 모드 (키보드 ‘e’)

키보드 ‘e’ 키를 누르면 cross section을 수정할 수 있는 2D 모드로 진입합니다. 사용할 수 있는 키들은 아래와 같습니다.

1. 다음, 이전 크로스섹션 – ‘+’, ‘-‘
2. 컨트롤 포인트 이동 - 마우스 왼쪽 버튼 클릭 후 드래그
3. 컨트롤 포인트 삭제 - 컨트롤 포인트에 마우스 오른쪽 버튼 클릭
4. 컨트롤 포인트 생성 - 빈 공간에 마우스 오른쪽 버튼 클릭
5. Zoom in/out - ‘z’, ‘Z’
6. 3D 모드로 전환 – ‘e’

주의할 점은 Catmull-Rom이나 B-spline 등을 spline 생성 방법으로 사용할 경우 첫 번째 크로스 섹션과 마지막 크로스 섹션은 화면에 표시되지 않으므로 수정해도 변화를 확인할 수 없다는 것입니다. 컨트롤 포인트를 움직일 경우 수정내용은 바로 바로 실시간으로 렌더링에 반영됩니다.

좌측 하단에서는 지금 수정하고 있는 크로스 섹션이 몇 번째 크로스 섹션인지와 확대 비율을 확인할 수 있습니다. 그리고 왼쪽 상단의 메시지를 통해 지금 어떠한 spline 생성 방법으로 cross section의 외곽선 생성 그리고 위치, 크기, 컨트롤 포인트 보간이 이루어지는지도 확인할 수 있습니다.

# 테스트 셋

총 3개의 테스트 셋이 포함되어있습니다.

1. Screw.txt
2. Bowl.txt
3. Tree.txt