공학석사 학위논문

Agnostic G^1 Gregory 곡면을 이용한 선체 곡면 모델링

Ship Hull Surface Modeling using Agnostic $\boldsymbol{G^1}$ Gregory Surfaces

2017 년 8 월

서울대학교 대학원

조선해양공학 전공

김 도 환

Agnostic G^1 Gregory 곡면을 이용한 선체 곡면 모델링

지도 교수 김 태 완

이 논문을 공학석사 학위논문으로 제출함

2017 년 6 월

서울대학교 대학원

조선해양공학 전공

김 도 환

김도환의 공학석사 학위논문을 인준함

2017 년 7 월

위 원 장 _____ (인)

부위원장 _____ (인)

위 원 _____ (인)

초 록

G^1 연속 조건을 만족하는 선체 곡면의 필요성은 선형 설계자와 선형을 활용하는 이후 설계 작업자들의 설계 결과물의 차이를 줄이기 위하여 꾸준히 대두되어왔다. 선형 설계 결과물과 모델링된 선체 곡면의 차이로 인하여 구조/저항 및 추진 성능에 대한 정확한 성능 평가가 이루어지지 않으며, 이는 곡판 전개 등의 생산 작업의 자동화율 감소에 영향을 준다.

G^1 연속 조건을 만족하는 선체 곡면 생성을 위한 다양한 연구가 진행되었지만, 선박 형상의 특성상 복잡한 설계선(lines)으로 생성되는 다양한 삼각/사각형 곡면 간 경계 조건에 모두 적용 가능한 알고리즘은 개발되지 않았다.

본 논문에서는 agnostic G^1 Gregory 곡면 생성 알고리즘을 선박에 적용하고 검증하였다. 해당 알고리즘은 기존 Gregory 곡면을 활용하여 삼각/사각형 곡면이 혼합되어 사용된 영역에서 곡면 간 G^1 연속 조건을 만족하도록 개선된 곡면 생성 알고리즘이다. Agnostic G^1 Gregory 곡면 생성 알고리즘을 선박의 lines 에 적용하여 선체 곡면을 생성하고, 곡면 간 법선 벡터 사이의 각도를 측정하여 G^1 연속 조건을 만족함을 검증하였다.

주요어 : G^1 연속 조건, Agnostic G^1 Gregory 곡면, 선체 곡면 모델링

학 번 : 2015-22861

목차

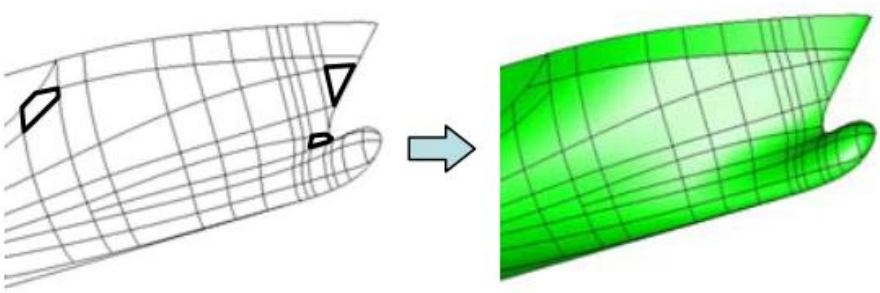
표 목차

그림 목차

1. 서론

1.1. 연구 배경

선형 설계자에 의한 선형이 완성되면 이것을 토대로 선박의 곡면을 생성하게 된다. 이때 생성되는 곡면은 각 패치의 경계 곡선 부근에서 부드럽게 이어지는 G^1 연속 조건을 만족하여야 한다. 연속 조건을 만족하지 못하는 곡면은 미관상 좋지 못하다. 또한 부드러운 선형이 아닐 경우 마찰에 의한 저항이 늘어나 선박의 저항 및 추진 성능에 좋지 않은 영향을 미치게 된다. 특히 선수 부분의 형상이나 선미 부분의 프로펠러 부근의 형상은 선박의 저항 및 추진 성능에 매우 큰 영향을 미치므로 이 부근의 곡면을 모델링 하는 작업은 매우 중요하다. 선형 중 이 부근은 다른 부분에 비해 곡률이 심하고 비정규(irregular) 형상을 포함한 복잡한 형태이므로 사각 패치(patch) 뿐만 아니라 삼각 패치가 혼재되어 설계된다. (**[그림 1-1]**에서도 확인할 수 있다.) 따라서 선체 곡면 모델링의 경우 삼/사각이 혼재되어 있는 경우에도 적용 가능한 곡면 모델링 알고리즘이 필요하다.



[그림 1-1] 곡선 그물망으로부터 선체 곡면을 모델링 하는 예[9].

1.2. 선행 연구

1 절에서 설명한 문제들을 해결하기 위해서 G^1 연속 조건을 만족하는 곡면 모델링에 관한 많은 연구가 진행되어 왔다. W. Du and J. Schmitt[3]이 제시한 방법은 local method 로서 장점을 지니고 있지만 사각 패치만을 이용하여 삼각, 사각 패치가 혼재된 선박의 곡선 그물망(curve network)에 적용하기에는 어려움이 있다. J. Peters[4]가 제시한 방법에선 사각 Bezier 곡면과 Bezier triangle 모두 이용하였으나 global method 이고 또한 꼭짓점에서의 curvature 에 관한 추가적인 제한 조건을 만족하여야만 했다. Q. Liu and T. C. Sun[5]은 두 가지 방법을 제시했는데, 하나는 6 x 6 차 Bezier 곡면을 이용한 방법으로 이는 local method 이고, 다른 하나는 4x4 차 Bezier 곡면을 이용하는 방법으로 global method 이다. 하지만 두 방법 모두 꼭짓점에서 조정점이 각각 pairwise-collinear [1] 해야만 하는 제한

[1] 꼭지점에서 곡선 쌍이 서로 접하게 만나야한다는 제한 조건을 의미한다.

조건을 가지고 있으며 또한 사각 패치 경우만을 다루었다는 한계점이 있다. X. Shi, T. Wang and P. Yu[6]는 G^1 연속 조건을 만족하는 5 차 B-spline 곡면을 이용해 모델링 하였다. 위 방법은 local method 이나 사각 곡면만을 이용하였다. W.-h. Tong,[7]의 방법은 삼각 spline 곡면을 이용한 local method 이나 삼각, 사각 패치가 혼재된 영역에 바로 적용할 수는 없다.

한편, 일반적인 곡면에 관한 연구뿐만 아니라 선체 곡면 모델링에 관한 연구도 수행되어 왔다. Joong-Hyun Rhim et. al.[9]은 B-spline 곡면을 이용하여 선체 곡면을 모델링 하였다. 비정규 곡선 그물망을 경계 곡선과 참조 곡선으로 재구성하여 정규 곡선 그물망으로 변환한 후, piecewise 곡면의 G^1 연속 조건을 맞추었다. 그러나 조선 산업의 특성상 선형은 다른 여타 제한 조건에 적합하게 선정된 최적화된 것이므로 수정이 힘들며, 경계 곡선을 지정하는 경우의 수가 무수히 많으므로 각 경우에 따라 생성되는 곡면과 품질이 달라진다는 한계점이 있다. 또한, Doo-Yeoun Cho et. al.[8]는 삼각, 사각 패치 모두를 이용한 local method 를 제시하였다. 이 방법은 꼭짓점에서의 valence 가 3, 4, 5 로 한정되어 있으며, 이 점을 기준으로 마주 보는 조정점은 서로 collinear 해야 한다는 제한 조건을 가지고 있다.

위와 달리 분할 곡면을 이용하여 곡면을 모델링 한 연구도 진행되어 왔다.[10],[11] Doo-Yeoun Cho et. al.[10]은 곡선 그물망의 각 영역에서 Catmull-Clark 분할 곡면을 생성한 후, 분할 곡면의 경계에서 G^1 연속 조건을 만족하게 하는 방법 및 곡면의 품질 향상을 위한 곡면 순정 방법을 제시하였다. 삼각형과 같은 비정규 위상은 정규

N-sided 조정 메시를 이용하여 다루었다. 최근에 Sebastian H. G et. al.[11]은 분할 곡면을 이용하여 텐서 곱 B-spline 곡면을 대체하는 방법을 제시했다. 그러나, 현재 분할 곡면을 지원하는 CAD/CAM 시스템이 거의 존재하지 않으므로 선박의 곡면 정보를 구조설계, 상세설계 등의 후행 공정에 원활하게 전달하기 어려운 문제점을 가지고 있다.

1.3. 연구 목적

선박 형상의 특성상 복잡한 설계선(lines)으로 생성되는 다양한 삼각/사각형 곡면 간 경계 조건에 적용 가능한 알고리즘이 필요하다. 앞서 2 절에서 살펴본 것처럼, 삼각/사각형 곡면이 모두 혼재 되어있는 곡선 그물망의 경우에, 두 곡면의 구분 없이 같은 G^1 연속 조건을 적용할 수 있는 알고리즘은 개발되지 않았다. 하지만 본 논문에서 적용한 G. Farin 의 방법은 이를 가능케하는 방법으로, Gregory 곡면을 이용하는 local method 이며 꼭짓점에서 조정점 각 쌍의 collinear 유무에 상관없이 coplanar 조건만을 만족하면 적용할 수 있는 알고리즘이다. 이러한 알고리즘을 실제 선박에 적용하기에 앞서, 이에 대한 적용가능성을 검토할 필요가 있었다. 이를 위해, 본 논문에서는 G. Farin 의 알고리즘 이론을 분석하고, 이를 선박의 몇몇 case 에 적용하였다.

1.4. 논문의 구성

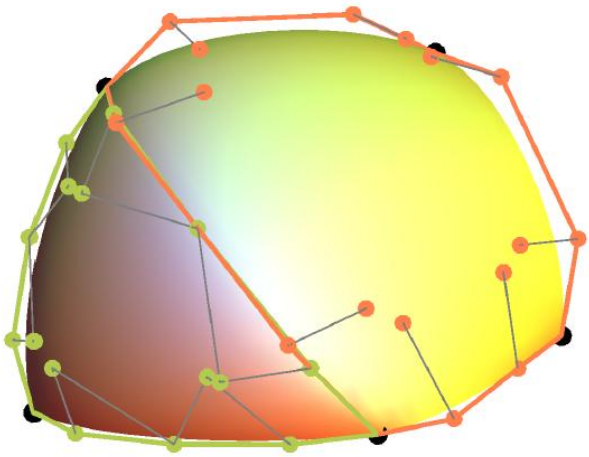
논문의 구성은 다음과 같다. 우선 본 알고리즘을 설명하기 위해 사각, 삼각 Gregory 곡면을 소개하고 이후 agnostic 의 의미를 두 곡면의 tangent ribbon 의 구조가 같음에 주목하여 설명한다. 이 곡면을 토대로 일반적인 G^1 연속 조건과 G. Farin 의 G^1 연속 조건을 제시하고 실제 적용될 조건식을 유도한다. 또한, 곡면을 fitting 하는 과정을 설명한다. 그리고 이를 종합하여 전처리 과정을 포함한, 실제 선박의 선형에 G^1 연속 조건을 만족하는 곡면을 모델링 및 생성하는 과정을 소개한다. 마지막으로 G^1 연속 조건을 만족하는 곡면을 가시화하고 곡면 간 각도를 측정하는 과정을 거쳐 본 알고리즘을 검증한다.

2. Agnostic G^1 Gregory 곡면

2.1. 명명법 및 용어

- $\mathbf{q}(t) = \sum_{i=0}^{n-1} \mathbf{q}_i B_i^{n-1}(t)$: 주어진 경계 곡선으로, n 차 Bezier 곡선.
- $\mathbf{q}^0(t) = \sum_{i=0}^{n-1} \mathbf{q}_i B_i^{n-1}(t)$: n−1 차 경계 곡선에서 접선 벡터가 있을 때, 그 접선 벡터의 시작하는 점(아래)에서의 곡선.
- $\mathbf{q}^n(t) = \sum_{i=0}^{n-1} \mathbf{q}_{i+1} B_i^{n-1}(t)$: n−1 차 경계 곡선에서 접선 벡터가 있을 때, 그 접선 벡터의 끝나는 점(위)에서의 곡선.
- $\widehat{\mathbf{q}}_i = \sum_{i=0}^{m} \widehat{\mathbf{q}}_i B_i^m(t)$: 주어진 경계곡선에서 degree elevation 시행 후의 곡선.
- $\mathbf{p}(t) = \sum_{i=0}^{n-1} \mathbf{p}_i B_i^{n-1}(t)$: 왼쪽 측면의 cross−derivative 곡선.
- $\mathbf{r}(t) = \sum_{i=0}^{n-1} \mathbf{r}_i B_i^{n-1}(t)$: 오른쪽 측면의 cross−derivative 곡선.
- $\mathbf{p}_i$: 왼쪽 측면의 cross−derivative 곡선에서 i번째 조정점
- $\mathbf{r}_i$: 오른쪽 측면의 cross−derivative 곡선에서 i번째 조정점
- $\mathbf{p}_1^e, \mathbf{r}_1^e, \mathbf{p}_2^e, \mathbf{r}_2^e, \mathbf{p}_0^e, \mathbf{p}_3^e, \mathbf{r}_0^e, \mathbf{r}_3^e$: 각 조정점을 추정한(estimation) 값.
- $\mathbf{p}'(t)$: $\frac{\mathrm{d}}{\mathrm{dt}}\mathbf{p}(t)$
- a × b : 두 벡터 a, b 의 외적

2.2. Gregory 곡면



[그림 2-1] Agnostic G^1 Gregory 곡면[1]

[그림 2-1]은 삼각, 사각 Gregory 곡면 사이에 G^1 연속 조건을 적용한 결과이다. 이를 적용하기 앞서 기본적으로 Gregory 곡면에 대해서 알아본다. Gregory 곡면은 사각, 삼각 두 가지 종류가 있다.

2.2.1. 사각 Gregory 곡면

$$\begin{matrix} \mathbf{b}_{00} & \mathbf{b}_{01} & \mathbf{b}_{02} & \mathbf{b}_{03} \\ \mathbf{b}_{10} & \mathbf{b}_{11} & \mathbf{b}_{12} & \mathbf{b}_{13} \\ \mathbf{b}_{20} & \mathbf{b}_{21} & \mathbf{b}_{22} & \mathbf{b}_{23} \\ \mathbf{b}_{30} & \mathbf{b}_{31} & \mathbf{b}_{32} & \mathbf{b}_{33} \end{matrix}$$

[그림 2-2] 3 차 사각 Bezier 곡면의 조정점

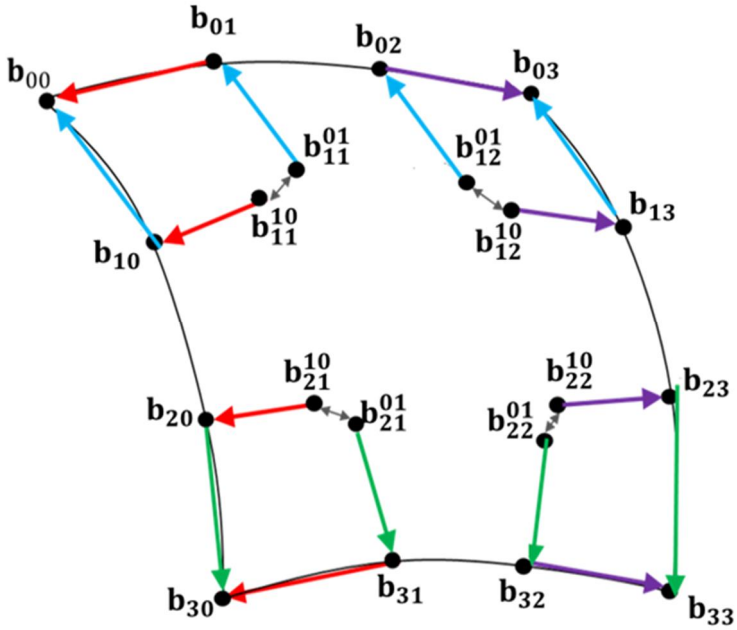
곡면 위의 점 $\mathbf{b}(u,v)$은 다음과 같이 구할 수 있다.

$$\mathbf{b}(u,v)=\sum_{i=0}^{3}\sum_{j=0}^{3}\mathbf{b}_{ij}B_i^3(u)B_j^3(v) \qquad \text{식 } 2-1$$

여기서, $0\le u,v\le 1$

$\mathrm{B}_i^3(u),\mathrm{B}_j^3(v)$는 식 2-2 와 같이 정의 된다.

$$\mathrm{B}_i^3(u)=\frac{3!}{i!\,(3-i)!}(1-u)^{3-i}u^i \qquad \text{식 } 2-2$$
$$\mathrm{B}_j^3(v)=\frac{3!}{j!\,(3-j)!}(1-v)^{3-j}v^j$$



[그림 2-3] 사각 Gregory 곡면

사각 Gregory 곡면의 조정점의 경우 일반적인 3 차 사각 Bezier 곡면과 같은 구조를 가졌지만, 내부의 조정점이 고정된 것이 아닌 변수로서 다음과 같은 식에 의해 정의된다.

$$\mathbf{b}_{11} = \frac{u}{u+v}\mathbf{b}_{11}^{10} + \frac{v}{u+v}\mathbf{b}_{11}^{01} \qquad \text{식 } 2-3$$

$$\mathbf{b}_{21} = \frac{1-u}{1-u+v}\mathbf{b}_{21}^{10} + \frac{v}{1-u+v}\mathbf{b}_{21}^{01} \qquad \text{식 } 2-4$$

$$\mathbf{b}_{12} = \frac{u}{1-u+v}\mathbf{b}_{12}^{10} + \frac{1-v}{1-v+u}\mathbf{b}_{12}^{01} \qquad \text{식 } 2-5$$

$$\mathbf{b}_{22} = \frac{1-u}{2-u-v}\mathbf{b}_{22}^{10} + \frac{1-v}{2-u-v}\mathbf{b}_{22}^{01} \qquad \text{식 } 2-6$$

이때 위 첨자 10 은 $\boldsymbol{u}$ 에 따라 변하는 경계에 더 큰 영향을 미친다는 사실을 나타내고, 마찬가지로 위 첨자 01 은 $\boldsymbol{v}$에 따라 변하는 경계에 더 큰 영향을 미침을 나타낸다. 식 2-3 ~ 식 2-6 에서 보듯이 내부의 8 개의 조정점이 블렌딩(blending) 되어 $\mathbf{b_{11}}, \mathbf{b_{21}}, \mathbf{b_{12}}, \mathbf{b_{22}}$ 가 결정된다.

2.2.1.1. 사각 곡면의 tangent ribbon

$$\mathbf{b}_{00}\ \mathbf{b}_{01}$$
$$\mathbf{b}_{10}\ \mathbf{b}_{11}^{10}$$
$$\mathbf{b}_{20}\ \mathbf{b}_{21}^{10}$$
$$\mathbf{b}_{30}\ \mathbf{b}_{31}$$

[그림 2-4] $v = \mathbf{0}$ 일 때 접평면을 이루는 조정점

$$\mathbf{b}_{02}\ \mathbf{b}_{03}$$
$$\mathbf{b}_{12}^{10}\ \mathbf{b}_{13}$$
$$\mathbf{b}_{22}^{10}\ \mathbf{b}_{23}$$
$$\mathbf{b}_{32}\ \mathbf{b}_{33}$$

[그림 2-5] $v = \mathbf{1}$ 일 때 접평면을 이루는 조정점

$$\mathbf{b}_{00}\ \mathbf{b}_{01}\ \mathbf{b}_{02}\ \mathbf{b}_{03}$$
$$\mathbf{b}_{10}\ \mathbf{b}_{11}^{01}\ \mathbf{b}_{12}^{01}\ \mathbf{b}_{13}$$

[그림 2-6] $u = \mathbf{0}$ 일 때 접평면을 이루는 조정점

$$\mathbf{b}_{20}\ \mathbf{b}_{21}^{01}\ \mathbf{b}_{22}^{01}\ \mathbf{b}_{23}$$
$$\mathbf{b}_{30}\ \mathbf{b}_{31}\ \mathbf{b}_{32}\ \mathbf{b}_{33}$$

[그림 2-7] $u = \mathbf{1}$ 일 때 접평면을 이루는 조정점

Gregory 곡면은 Bezier 곡면과 달리 각 모서리에서의 cross boundary derivative 를 각 쌍의 조정점으로 정의할 수 있다. **[그림 2-4]**에서 보듯이 $(\mathbf{b}_{00}, \mathbf{b}_{01})$, $(\mathbf{b}_{10}, \mathbf{b}_{11}^{10})$, $(\mathbf{b}_{20}, \mathbf{b}_{21}^{10})$, $(\mathbf{b}_{30}, \mathbf{b}_{31})$ 각 쌍에 의해 좌측의 cross boundary derivative 를 구할 수 있다. 마찬가지로 **[그림 2-5]**에서, $(\mathbf{b}_{02}, \mathbf{b}_{03})$, $(\mathbf{b}_{12}^{10}, \mathbf{b}_{13})$, $(\mathbf{b}_{22}^{10}, \mathbf{b}_{23})$, $(\mathbf{b}_{32}, \mathbf{b}_{33})$ 각 쌍에 의해 우측의 cross boundary derivative 를 구할 수 있다. **[그림 2-6]**, **[그림 2-7]**에서도 같은 방식으로 cross boundary derivative 를 구할 수 있음을 알 수 있다. 이러한 cross boundary derivatives 는 **[그림 2-3]**에서 각각의 색상을 띈 화살표로 확인할 수 있다. 각각의 경계를 따라 접평면을 정의할 수 있는 위와 같은 형태를 tangent ribbon 이라고 한다. 사각 Gregory 곡면의 경우 4 개의 tangent ribbon 이 있다.

2.2.2. 삼각 Gregory 곡면

$$\mathbf{b}_{040}$$
$$\mathbf{b}_{031}\mathbf{b}_{130}$$
$$\mathbf{b}_{022}\mathbf{b}_{121}\mathbf{b}_{220}$$
$$\mathbf{b}_{013}\mathbf{b}_{112}\mathbf{b}_{121}\mathbf{b}_{310}$$
$$\mathbf{b}_{004}\mathbf{b}_{103}\mathbf{b}_{202}\mathbf{b}_{301}\mathbf{b}_{400}$$

[그림 2-8] 4 차 삼각 Bezier 곡면의 조정점

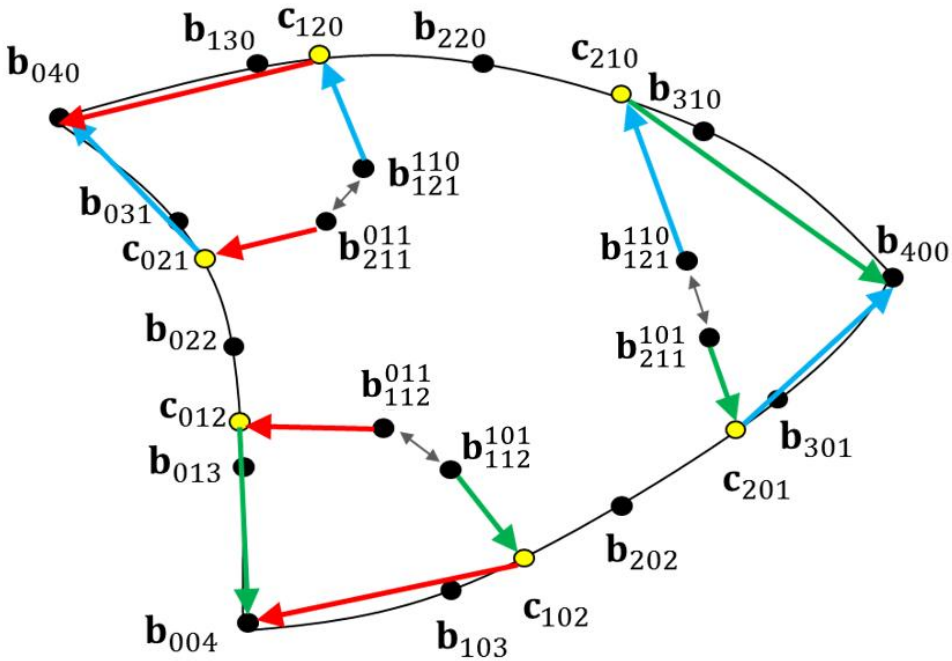
4 차 삼각 Bezier 곡면의 조정점의 경우 그림 2-9 와 같은 구조를 가진다. 곡면 위의 점 $\mathbf{b}(u,v,w)$은 다음과 같이 구할 수 있다.

$$\mathbf{b}(u,v,w) = \sum_{i+j+k=4} \mathbf{b}_{ijk} B^4_{ijk}(u,v,w) \qquad \text{식 } 2-7$$

여기서, u, v, w는 무계중심표계로, $u+v+w=1$이다.

$B^4_{ijk}(u,v,w)$는 식 2-8 과 같이 정의 된다.

$$B^4_{ijk}(u,v,w) = \frac{24}{i!\,j!\,k!} u^i v^j w^k \qquad \text{식 } 2-8$$



[그림 2-9] 삼각 Gregory 곡면

삼각 Gregory 곡면의 조정점의 경우도 일반적인 4 차 삼각 Bezier 곡면과 같은 구조를 가졌지만, 내부의 조정점이 고정된 것이 아닌 변수로서 다음과 같은 식에 의해 정의된다.

$$\mathbf{b}_{112} = \frac{u}{u+v}\mathbf{b}_{112}^{101} + \frac{v}{u+v}\mathbf{b}_{112}^{011} \quad \text{식 2-9}$$

$$\mathbf{b}_{211} = \frac{w}{w+v}\mathbf{b}_{211}^{101} + \frac{v}{w+v}\mathbf{b}_{211}^{110} \quad \text{식 2-10}$$

$$\mathbf{b}_{121} = \frac{u}{u+w}\mathbf{b}_{121}^{110} + \frac{w}{u+w}\mathbf{b}_{121}^{011} \quad \text{식 2-11}$$

이때 위 첨자 101 은 경계 곡선 $(\boldsymbol{u}, \mathbf{0}, \boldsymbol{w})$에 더 큰 영향을 받고, 마찬가지로 위 첨자 110 은 경계 곡선 $(\boldsymbol{u}, \boldsymbol{v}, \mathbf{0})$에 더 큰 영향을 받으며, 위 첨자 011 은 경계 곡선 $(\mathbf{0}, \boldsymbol{v}, \boldsymbol{w})$에 더 큰 영향을 받는다는 사실을 나타낸다. 식 2-9 ~ 식 2-11 에서 보듯이 내부의 6 개의 조정점이 블렌딩되어 $\mathbf{b}_{112}, \mathbf{b}_{211}, \mathbf{b}_{121}$가 결정된다.

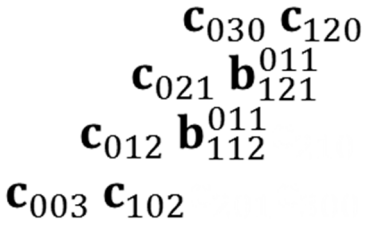
2.2.2.1. 삼각 곡면의 tangent ribbon

여기서 4 차 Bezier 곡선인 경계 곡선을 3 차 Bezier 곡선으로 표현하면 다음과 같다.

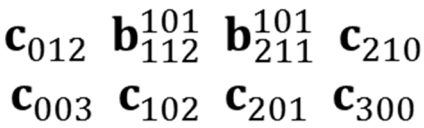
$$\mathbf{c}_{030}$$
$$\mathbf{c}_{021}\ \mathbf{c}_{120}$$
$$\mathbf{c}_{012}\qquad \mathbf{c}_{210}$$
$$\mathbf{c}_{003}\ \mathbf{c}_{102}\ \mathbf{c}_{201}\ \mathbf{c}_{300}$$

[그림 2-10] 3 차 삼각 Bezier 곡면의 경계 곡선의 조정점

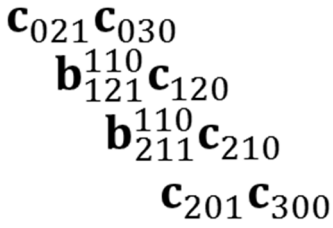
이제 사각 곡면에서처럼 다음과 같이 tangent ribbon 을 정의할 수 있다.



[그림 2-11] $\boldsymbol{u} = \mathbf{0}$에서의 tangent ribbon



[그림 2-12] $\boldsymbol{v} = \mathbf{0}$에서의 tangent ribbon



[그림 2-13] $\boldsymbol{w} = \mathbf{0}$에서의 tangent ribbon

삼각 곡면의 tangent ribbon 또한 사각 곡면의 것처럼 각 경계 곡선에서의 cross boundary derivative 를 구함으로써 접평면을 정의할 수 있다. [그림 2-11]에서 보듯이 $(\mathbf{c}_{030}, \mathbf{c}_{120})$, $(\mathbf{c}_{021}, \mathbf{b}_{121}^{011})$, $(\mathbf{c}_{012}, \mathbf{b}_{112}^{011})$, $(\mathbf{c}_{003}, \mathbf{c}_{102})$ 각 쌍에 의해 u=0 에서의 cross boundary derivative 를 구할 수 있고, 마찬가지로 [그림 2-12]에서 $(\mathbf{c}_{003}, \mathbf{c}_{012})$, $(\mathbf{c}_{102}, \mathbf{b}_{102}^{101})$, $(\mathbf{c}_{201}, \mathbf{b}_{211}^{101})$, $(\mathbf{c}_{003}, \mathbf{c}_{102})$에 의해 v=0 에서의 cross boundary derivative 를, [그림 2-13]에는 $(\mathbf{c}_{030}, \mathbf{c}_{021})$, $(\mathbf{c}_{120}, \mathbf{b}_{121}^{110})$, $(\mathbf{c}_{210}, \mathbf{b}_{211}^{110})$,

($\mathbf{c}_{300}, \mathbf{c}_{201}$) 각 쌍에 의해 w=0 에서의 cross boundary derivative 를 구할 수 있다. 이러한 cross boundary derivatives 는 [그림 2-9]에서 각각의 색상을 띤 화살표로 확인할 수 있다.

2.2.3. Agnostic 의 의미

$$\begin{matrix} \mathbf{b}_{00} & \mathbf{b}_{01} \\ \mathbf{b}_{10} & \mathbf{b}_{11}^{10} \\ \mathbf{b}_{20} & \mathbf{b}_{21}^{10} \\ \mathbf{b}_{30} & \mathbf{b}_{31} \end{matrix} \qquad \begin{matrix} \mathbf{b}_{02} & \mathbf{b}_{03} \\ \mathbf{b}_{12}^{10} & \mathbf{b}_{13} \\ \mathbf{b}_{22}^{10} & \mathbf{b}_{23} \\ \mathbf{b}_{32} & \mathbf{b}_{33} \end{matrix}$$

$$\begin{matrix} \mathbf{b}_{00} & \mathbf{b}_{01} & \mathbf{b}_{02} & \mathbf{b}_{03} \\ \mathbf{b}_{10} & \mathbf{b}_{11}^{01} & \mathbf{b}_{12}^{01} & \mathbf{b}_{13} \end{matrix} \qquad \begin{matrix} \mathbf{b}_{20} & \mathbf{b}_{21}^{01} & \mathbf{b}_{22}^{01} & \mathbf{b}_{23} \\ \mathbf{b}_{30} & \mathbf{b}_{31} & \mathbf{b}_{32} & \mathbf{b}_{33} \end{matrix}$$

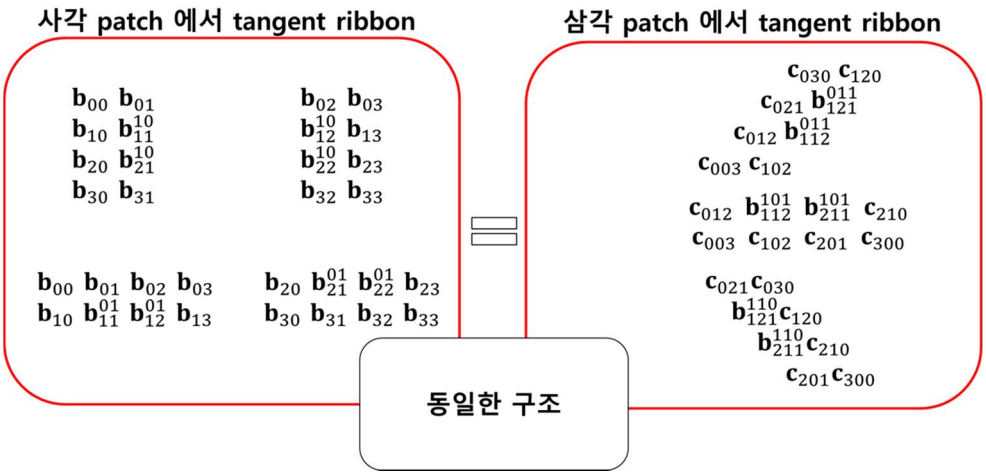
[그림 2-14] 사각 Gregory 곡면의 tangent ribbon

$$\begin{matrix} & & \mathbf{c}_{030} & \mathbf{c}_{120} \\ & \mathbf{c}_{021} & \mathbf{b}_{121}^{011} & \\ \mathbf{c}_{012} & \mathbf{b}_{112}^{011} & & \\ \mathbf{c}_{003} & \mathbf{c}_{102} & & \end{matrix} \qquad \begin{matrix} \mathbf{c}_{021} & \mathbf{c}_{030} & & \\ & \mathbf{b}_{121}^{110} & \mathbf{c}_{120} & \\ & & \mathbf{b}_{211}^{110} & \mathbf{c}_{210} \\ & & & \mathbf{c}_{201}\mathbf{c}_{300} \end{matrix}$$

$$\begin{matrix} \mathbf{c}_{012} & \mathbf{b}_{112}^{101} & \mathbf{b}_{211}^{101} & \mathbf{c}_{210} \\ \mathbf{c}_{003} & \mathbf{c}_{102} & \mathbf{c}_{201} & \mathbf{c}_{300} \end{matrix}$$

[그림 2-15] 삼각 Gregory 곡면의 tangent ribbon

2.2.1.과 2.2.2.에서 살펴본 사각, 삼각 Gregory 곡면의 tangent ribbon 은 **[그림 2-14]**, **[그림 2-15]**에서 보는 것과 같다. 여기서 중요한 점은 **[그림 2-16]**을 보듯이 사각 곡면과 삼각 곡면의 tangent ribbon 의 구조가 같다는 것이다. 즉 삼각, 사각 곡면의 종류에 상관없이 G^1 연속 조건을 만족시킬 수 있으므로 "agnostic[2]" 이다.

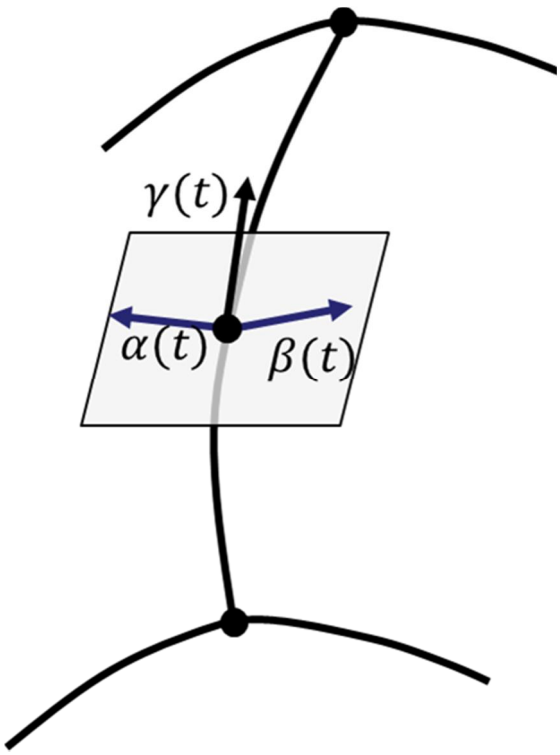


[그림 2-16] "Agnostic G^1 Gregory 곡면" 에서 agnostic 의 의미

[2] Agnostic 은 '불가지론의'란 형이상학적인 의미 외에도 '~에 상관없이 사용 가능한'이라는 뜻이 있다.

2.3. G^1 연속 조건

2.3.1. 일반적인 G^1 연속 조건



$\boldsymbol{\alpha}(t), \boldsymbol{\beta}(t)$ =공통 경계 곡선에서의 cross boundary derivative
$\boldsymbol{\gamma}(t)$ = 공통 경계 곡선에서의 접선 벡터

[그림 2-17] 일반적인 G^1 연속 조건 설명

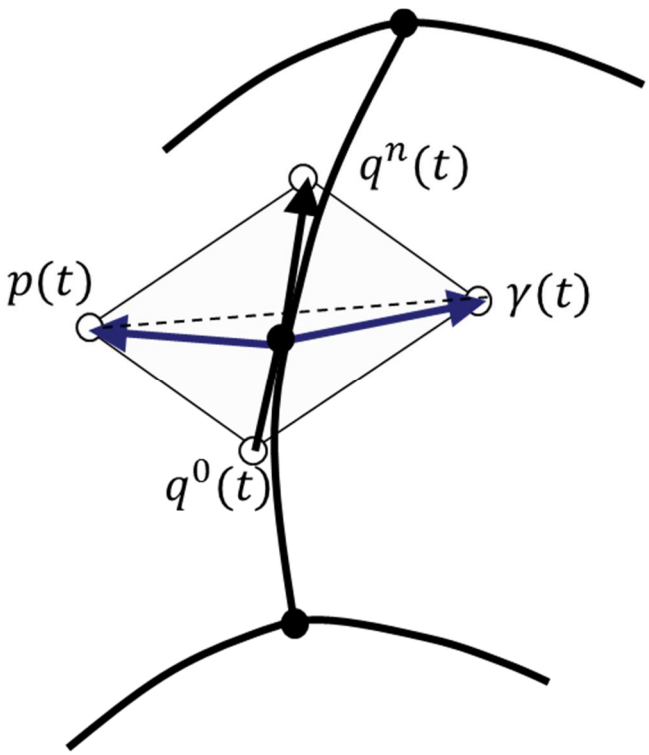
$$\boldsymbol{\alpha}(t)\lambda(t) + \boldsymbol{\beta}(t)\mu(t) + \boldsymbol{\gamma}(t)\nu(t) = 0 \qquad \text{식 2-12}$$

G^1에서 G는 'geometric continuity'(기하학적 연속성)에서의 G를 의미한다. 두 곡면 사이의 G^1 연속 조건의 정의는 다음과 같다.

정의 1. 두 개의 곡면이 한 경계 곡선을 공유하고 그 곡선을 따라 연속적으로 변하는 접평면을 가질 때, 두 곡면은 $\boldsymbol{G^1}$이라고 한다.

[그림 2-17]에서 보듯이 경계 곡선 위의 점과 각 벡터 $\boldsymbol{\alpha}(t), \boldsymbol{\beta}(t), \boldsymbol{\gamma}(t)$ 가 가리키는 점들이 모두 하나의 평면에 있고, 이는 하나의 접평면을 이룬다. 이를 식 2-12 를 통해 보면, 임의의 t 에 대하여 좌변이 0 이 되므로 세 벡터가 한 평면 위에 있다. 이 평면은 cross boundary derivative 와 접선 벡터에 의해 이루어지는 평면이므로 이는 접평면이다. 따라서 두 곡면은 경계 곡선에서 연속적으로 변하는 공통된 접평면을 가지므로 G^1이라 할 수 있다.

2.3.2. G. Farin 의 G^1 연속 조건 [2]



[그림 2-18] G. Farin 의 G^1 연속 조건 설명

$$\mathbf{p}(t) = \sum_{i=0}^{n-1} \mathbf{p}_i B_i^{n-1}(t) \qquad \text{식 } 2-13$$

$$\mathbf{q}^0(t) = \sum_{i=0}^{n-1} \mathbf{q}_i B_i^{n-1}(t) \qquad \text{식 } 2-14$$

$$\mathbf{q}^n(t) = \sum_{i=0}^{n-1} \mathbf{q}_{i+1} B_i^{n-1}(t) \qquad \text{식 } 2-15$$

$$\mathbf{r}(t) = \sum_{i=0}^{n-1} \mathbf{r}_i B_i^{n-1}(t) \qquad \text{식 } 2-16$$

$$\big(1-\lambda(t)\big)\mathbf{p}(t) + \lambda(t)\mathbf{r}(t) = \big(1-\mu(t)\big)\mathbf{q}^0(t) + \mu(t)\mathbf{q}^n(t) \qquad \text{식 } 2-17$$

G. Farin 의 G^1 연속 조건은 **2.3.1.**에서의 일반적인 G^1 연속 조건과 같은 의미이나 식 2-17 에서 보듯이 표현 방식에 차이가 있다. G. Farin 의 G^1 정의는 다음과 같다.

정의 2. 네 점 $\mathbf{p}(t)$, $\mathbf{q}^0(t)$, $\mathbf{q}^n(t)$, $\mathbf{r}(t)$ 가 임의의 t 에 대하여 같은 평면 상에 있을 때, 즉 **선분** $\overline{\mathbf{pr}}$ 과 $\overline{\mathbf{q}^0\mathbf{q}^n}$ 이 항상 교차하여 만날 때 두 곡면을 $\boldsymbol{G^1}$이라 한다.

이는 식 2-13 ~ 식 2-16 에서 정의된 네 점 $\mathbf{p}(t)$, $\mathbf{q}^0(t)$, $\mathbf{q}^n(t)$, $\mathbf{r}(t)$이 식 2-17 을 만족하도록 하는 $\lambda(t)$와 $\mu(t)$가 존재함을 의미한다.

앞으로 쓰일 식은 식 2-17 을 기반으로 한 식이다. 이를 위해 식 2-17 을 좀 더 살펴보겠다.

2.3.2.1. 두 삼각 곡면 사이의 G^1 연속 조건

본 논문에서 삼각 곡면의 경계 곡선은 4 차 Bezier 곡선이고, 사각 곡면의 경우 3 차 Bezier 곡선이다. 사각 곡면의 경우 degree elevation 을 해주지만 삼각 곡면은 경계 곡선의 degree elevated 조종점이 이미 주어져 있다. 삼각 곡면 사이의 G^1 연속 조건을 만족하려면 정의 2.에 의하여 식 2-17 을 만족하여야 한다. 이 식을 구체화 하기 위해선 우선 $\lambda(t)$와 $\mu(t)$를 구해야 한다. **[그림 2-18]**에서 보듯이 양 끝의 꼭짓점은 근방의 조정점과 함께 동일 평면 상에 있어야 한다. 이를 coplanar 조건이라 하며, 각 꼭짓점에서 즉 $t=0$과1 일 때 $\lambda(t)$와 $\mu(t)$의 값을 λ_0, μ_0 과 λ_1, μ_1라 하면 이 조건은 다음과 같이 쓸 수 있다.

$$\overline{\lambda_0}\mathbf{p}_0 + \lambda_0\mathbf{r}_0 = \overline{\mu_0}\mathbf{q}_0 + \mu_0\mathbf{q}_1 \qquad \text{식 2-18}$$

$$\overline{\lambda_1}\mathbf{p}_{n-1} + \lambda_1\mathbf{r}_{n-1} = \overline{\mu_1}\mathbf{q}_{n-1} + \mu_1\mathbf{q}_n \qquad \text{식 2-19}$$

여기서, $\bar{\lambda} = 1-\lambda,\ \bar{\mu} = 1-\mu.$

이 때, $\lambda(t)$와 $\mu(t)$가 t에 대해 선형적으로 변한다고 가정하자. $t=0$과 1 일 때 $\lambda(t)$와 $\mu(t)$의 값을 λ_0, μ_0 과 λ_1, μ_1이므로 $\lambda(t)$와 $\mu(t)$는 다음과 같이 표현 된다.

$$\lambda(t) = (1-t)\lambda_0 + t\lambda_1 \qquad \text{식 2-20}$$

$$\mu(t) = (1-t)\mu_0 + t\mu_1 \qquad \text{식 2-21}$$

또한 이는 아래 식 2-22 와 식 2-23 으로도 쓸 수 있다.

$$1-\lambda(t) = (1-t)\overline{\lambda_0} + t\overline{\lambda_1} \qquad \text{식 2-22}$$

$$1-\mu(t)=(1-t)\overline{\mu_0}+t\overline{\mu_1} \qquad \text{식 } 2-23$$

구한 $\lambda(t)$ 와 $\mu(t)$ 와 네 점 $\mathbf{p}(t)$, $\mathbf{q^0}(t)$, $\mathbf{q^n}(t)$, $\mathbf{r}(t)$ 을 식 2−17 에 대입하면 G^1 연속 조건은 다음과 같다.

$$\begin{aligned}&\left((t)\overline{\lambda_0}+t\overline{\lambda_1}\right)\sum_{i=0}^{n-1}\mathbf{p}_i B_i^{n-1}(t)\\&\quad+\left((1-t)\lambda_0+t\lambda_1\right)\sum_{i=0}^{n-1}\mathbf{r}_i B_i^{n-1}(t)\\&=\left((1-t)\overline{\mu_0}+t\overline{\mu_1}\right)\sum_{i=0}^{n-1}\mathbf{q}_i B_i^{n-1}(t)\\&\quad+\left((1-t)\mu_0+t\mu_1\right)\sum_{i=0}^{n-1}\mathbf{q}_{i+1} B_i^{n-1}(t)\end{aligned} \qquad \text{식 } 2-24$$

여기서 Bernstein 다항식에 대한 degree elevation 공식

$$(1-t)B_i^n(t)=\frac{n+1-i}{n+1}B_i^{n+1}(t) \qquad \text{식 } 2-25$$

$$tB_i^n(t)=\frac{i+1}{n+1}B_{i+1}^{n+1}(t) \qquad \text{식 } 2-26$$

식 2−25, 식 2−26 을 이용하여 B_i^{n+1}항에 대하여 index 를 변형시키면 다음과 같은 식을 얻을 수 있다.

$$\begin{aligned}&\frac{i}{n}\left(\overline{\lambda_1}\mathbf{p}_{i-1}+\lambda_1\mathbf{r}_{i-1}\right)+\frac{n-i}{n}\left(\overline{\lambda_0}\mathbf{p}_i+\lambda_0\mathbf{r}_i\right)\\&\quad=\frac{i}{n}\left(\overline{\mu_1}\mathbf{q}_{i-1}+\mu_1\mathbf{q}_{i-1}\right)+\frac{n-i}{n}\left(\overline{\mu_0}\mathbf{q}_i+\mu_0\mathbf{q}_{i+1}\right)\\&\qquad\text{여기서, } i=0,\ldots,n.\end{aligned} \qquad \text{식 } 2-27$$

2.3.2.2. 사각 곡면이 있을 경우의 G^1 연속 조건

사각 곡면의 경계 곡선의 경우 $\mathbf{q}(t)$를 차수 n 에서 n+1 로 degree elevation 을 시행한다. 이때 degree elevation 이란 곡선의 형태에 변화를 주지 않고 degree 와 조정점의 개수를 늘리는 것을 말하는 데, 이에 대한 식은 다음 식 2-28 과 같다. 사각 곡면의 tangent ribbon 의 구조는 삼각 곡면의 tangent ribbon 의 구조와 같다. 따라서 사각 곡면이 있을 때의 G^1 연속 조건은 2.3.2.1.에서 유도한 식 2-27 을 그대로 이용하되 $\mathbf{q}(t)$ 의 차수를 높인 값을 대입 해준다. 이를 식 2-27 에 대입하여 정리하면 식 2-29 와 같은 결과가 나온다.

$$\hat{\mathbf{q}}_i = \frac{i}{n+1}\mathbf{q}_{i-1} + \left(1 - \frac{i}{n+1}\right)\mathbf{q}_i$$

$$\hat{\mathbf{q}}_0 = \mathbf{q}_0, \hat{\mathbf{q}}_{n+1} = \mathbf{q}_0$$

식 2-28

여기서, $i = 1, \dots, n.$

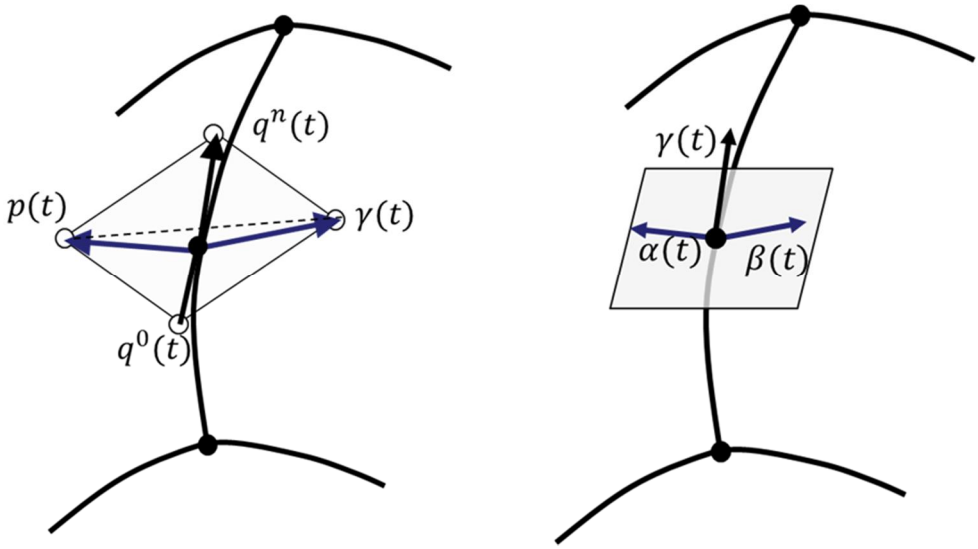
$$\frac{i}{n+1}\left[\left(\overline{\lambda_1}\mathbf{p}_{i-1} + \lambda_1\mathbf{r}_{i-1}\right) - \left(\overline{\mu_1}\hat{\mathbf{q}}_{i-1} + \mu_1\hat{\mathbf{q}}_i\right)\right]$$

$$= -\left(1 - \frac{i}{n+1}\right)\left[\left(\overline{\lambda_0}\mathbf{p}_i + \lambda_0\mathbf{r}_i\right) - \left(\overline{\mu_0}\hat{\mathbf{q}}_i + \mu_0\hat{\mathbf{q}}_{i+1}\right)\right]$$

여기서, $i = 0, \dots, n+1.$

식 2-29

2.3.3. 두 G^1 연속 조건의 비교



[그림 2-19] 두 G^1 연속 조건의 비교

G. Farin 의 G^1 연속 조건과 일반적인 G^1 연속 조건은 동일한 식이다. 이를 식 2-17 에서 식 2-12 를 유도함으로써 보일 수 있다.

식 2-17 을 다시 써보면 다음과 같다.

$$\big(1-\lambda(t)\big)\mathbf{p}(t)+\lambda(t)\mathbf{r}(t)=\big(1-\mu(t)\big)\mathbf{q}^0(t)+\mu(t)\mathbf{q}^n(t)$$

이는 다음과 같이 쓸 수 있다.

$$\begin{aligned}\big(1-\lambda(t)\big)\big(\mathbf{p}(t)-\mathbf{q}(t)\big)+\lambda(t)\big(\mathbf{r}(t)-\mathbf{q}(t)\big)+\big(1-\lambda(t)\big)\mathbf{q}(t)\\ +\lambda(t)\mathbf{q}(t)=\big(1-\mu(t)\big)\mathbf{q}^0(t)+\mu(t)\mathbf{q}^n(t)\end{aligned} \qquad \text{식 2-30}$$

이 때, $\mathbf{p}(t),\boldsymbol{\gamma}(t)$는 경계 곡선에서의 cross boundary derivative 이므로 다음과 같이 쓸 수 있다.

$$\mathbf{p}(t) - \mathbf{q}(t) = \boldsymbol{\alpha}(t) \qquad \text{식 } 2-31$$
$$\boldsymbol{\gamma}(t) - \mathbf{q}(t) = \boldsymbol{\beta}(t)$$

$$\begin{aligned} &\big(1-\lambda(t)\big)\boldsymbol{\alpha}(t) + \lambda(t)\boldsymbol{\beta}(t) \\ &= -\mathbf{q}(t) + \big(1-\mu(t)\big)\mathbf{q}^0(t) + \mu(t)\mathbf{q}^n(t) \end{aligned} \qquad \text{식 } 2-32$$

식 2−31 을 식 2−30 에 대입하면 식 2−32 를 얻을 수 있다. 이때, 식 2−32 의 우변을 RHS라 하면

$$RHS = -\mathbf{q}(t) + \mathbf{q}^0(t) + \mu(t)\big(\mathbf{q}^n(t) - \mathbf{q}^0(t)\big) \qquad \text{식 } 2-33$$

여기서, $\mathbf{q}^n(t) - \mathbf{q}^0(t) = n\mathbf{q}'(t) = \boldsymbol{\gamma}(t)$ 이고, $\mathbf{q}(t) - \mathbf{q}^0(t) = t\big(\mathbf{q}^n(t) - \mathbf{q}^0(t)\big) = tn\mathbf{q}'(t) = t\boldsymbol{\gamma}(t)$ 이므로 식 2−33 은 다음과 같이 쓸 수 있다.

$$\begin{aligned} RHS &= -\mathbf{q}(t) + \mathbf{q}^0(t) - \mu(t)\boldsymbol{\gamma}(t) \\ &= -t\boldsymbol{\gamma}(t) - \mu(t)\boldsymbol{\gamma}(t) \\ &= -\big(t+\mu(t)\big)\boldsymbol{\gamma}(t) \end{aligned} \qquad \text{식 } 2-34$$

$$\big(1-\lambda(t)\big)\boldsymbol{\alpha}(t) + \lambda(t)\boldsymbol{\beta}(t) + \tilde{\mu}(t)\boldsymbol{\gamma}(t) = 0 \qquad \text{식 } 2-35$$

$$\therefore \boldsymbol{\alpha}(t)\lambda(t) + \boldsymbol{\beta}(t)\mu(t) + \boldsymbol{\gamma}(t)\nu(t) = 0 \qquad \text{식 } 2-36$$

식 2−34 에서 $\big(t+\mu(t)\big) = \tilde{\mu}(t)$라 하고 식 2−32 를 다시 쓰면 식 2−35 와 같다. 이를 매개 변수를 달리하여 표현하면 식 2−36 이 되어 일반적인 G^1 연속 조건과 G. Farin 의 G^1 연속 조건이 같음을 증명하였다.

2.3.4. G. Farin 의 G^1 연속 조건 식 유도

$$\mathbf{p}_0\ \mathbf{q}_0\ \mathbf{r}_0$$
$$\mathbf{p}_1\ \mathbf{q}_1\ \mathbf{r}_1$$
$$\mathbf{p}_2\ \mathbf{q}_2\ \mathbf{r}_2$$
$$\mathbf{p}_3\ \mathbf{q}_3\ \mathbf{r}_3$$

(1)

두 3 차 Bezier 곡면 사이의 G^1 연속 조건을 적용해 볼 것이다. (1) 에서 보듯이 두 곡면의 공통 경계 곡선은 $\mathbf{q}_0, \mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3$을 조정 다각형(control polygon)으로 한다. 곡면 1 의 경계 곡선과 인접한 조정 다각형을 $\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3$이라 하자. 곡면 2 의 경우, 이를 $\mathbf{r}_0, \mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3$라 한다. G. Farin 의 G^1 연속 조건을 적용하기 위해서는 내부 점들에 관하여 식이 3 개가 필요하므로 경계 곡선에서 degree elevation 을 시행한다. Degree elevation 후의 조정점을 $\hat{\mathbf{q}}_0$, $\hat{\mathbf{q}}_1$, $\hat{\mathbf{q}}_2$, $\hat{\mathbf{q}}_3$, $\hat{\mathbf{q}}_4$라 하면 이를 구하는 식은 다음과 같다.

$$\hat{\mathbf{q}}_i = \frac{i}{4}\mathbf{q}_{i-1} + \left(1 - \frac{i}{4}\right)\mathbf{q}_i$$
$$\hat{\mathbf{q}}_0 = \mathbf{q}_0, \hat{\mathbf{q}}_4 = \mathbf{q}_3$$
여기서, $i = 1,2,3$

식 2−37

식 2−37 에서 다음 식 2−38 을 얻을 수 있다.

$$\hat{\mathbf{q}}_1 = \frac{1}{4}\mathbf{q}_0 + \frac{3}{4}\mathbf{q}_1$$
$$\hat{\mathbf{q}}_2 = \frac{1}{2}\mathbf{q}_1 + \frac{1}{2}\mathbf{q}_2$$
$$\hat{\mathbf{q}}_3 = \frac{3}{4}\mathbf{q}_2 + \frac{1}{4}\mathbf{q}_3$$

식 2−38

이를 이용하여 식들을 구성한다. 앞서 2.2.2.에서 유도한 식 2-29 는 다음과 같다.

$$\frac{i}{n+1}\left[\left(\overline{\lambda_1}\mathbf{p}_{i-1}+\lambda_1\mathbf{r}_{i-1}\right)-\left(\overline{\mu_1}\hat{\mathbf{q}}_{i-1}+\mu_1\hat{\mathbf{q}}_i\right)\right]$$
$$=-\left(1-\frac{i}{n+1}\right)\left[\left(\overline{\lambda_0}\mathbf{p}_i+\lambda_0\mathbf{r}_i\right)-\left(\overline{\mu_0}\hat{\mathbf{q}}_i+\mu_0\hat{\mathbf{q}}_{i+1}\right)\right]$$

이 식에서 $\mu_0,\lambda_0,\mu_1,\lambda_1$은 각각 꼭짓점에서의 매개변수 값이고, 또한 3 차 Bezier 곡선이므로 n=3 이다. $i=0$과 4를 대입하면 다음과 같이 구할 수 있다.

$$i=0\ \rightarrow 0=\left(\overline{\lambda_0}\mathbf{p}_0+\lambda_0\mathbf{r}_0\right)-\left(\overline{\mu_0}\hat{\mathbf{q}}_0+\mu_0\hat{\mathbf{q}}_1\right) \qquad \text{식 2-39}$$

$$i=4\rightarrow 0=\left(\overline{\lambda_1}\mathbf{p}_3+\lambda_1\mathbf{r}_3\right)-\left(\overline{\mu_1}\hat{\mathbf{q}}_3+\mu_1\hat{\mathbf{q}}_4\right) \qquad \text{식 2-40}$$

식 2-39 는 미지수가 μ_0,λ_0 으로 2 개 이나 방정식이 3 개인 overdetermined system 이다. 이를 풀면 μ_0,λ_0 을 구할 수 있다. 마찬가지로 식 2-40 을 풀면 μ_1,λ_1을 구할 수 있다.

위에서 구한 $\mu_0,\lambda_0,\mu_1,\lambda_1$을 토대로 식 2-29 에 $i=1,2,3$을 대입하면 다음과 같은 세 식이 나온다.

$$i=1\ \rightarrow\frac{1}{4}\left[\left(\overline{\lambda_1}\boldsymbol{p}_0+\lambda_1\boldsymbol{r}_0\right)-\left(\overline{\mu_1}\hat{\boldsymbol{q}}_0+\mu_1\hat{\boldsymbol{q}}_1\right)\right]$$
$$=-\frac{3}{4}\left[\left(\overline{\lambda_0}\boldsymbol{p}_1+\lambda_0\boldsymbol{r}_1\right)-\left(\overline{\mu_0}\hat{\boldsymbol{q}}_1+\mu_0\hat{\boldsymbol{q}}_2\right)\right] \qquad \text{식 2-41}$$

$$i=2\ \rightarrow\frac{2}{4}\left[\left(\overline{\lambda_1}\mathbf{p}_1+\lambda_1 r_1\right)-\left(\overline{\mu_1}\hat{\mathbf{q}}_1+\mu_1\hat{\mathbf{q}}_2\right)\right]$$
$$=-\frac{2}{4}\left[\left(\overline{\lambda_0}\mathbf{p}_2+\lambda_0\mathbf{r}_2\right)-\left(\overline{\mu_0}\hat{\mathbf{q}}_2+\mu_0\hat{\mathbf{q}}_3\right)\right] \qquad \text{식 2-42}$$

$$i=3\ \rightarrow\frac{3}{4}\left[\left(\overline{\lambda_1}\mathbf{p}_2+\lambda_1\mathbf{r}_2\right)-\left(\overline{\mu_1}\hat{\mathbf{q}}_2+\mu_1\hat{\mathbf{q}}_3\right)\right]$$
$$=-\frac{1}{4}\left[\left(\overline{\lambda_0}\mathbf{p}_3+\lambda_0\mathbf{r}_3\right)-\left(\overline{\mu_0}\hat{\mathbf{q}}_3+\mu_0\hat{\mathbf{q}}_4\right)\right] \qquad \text{식 2-43}$$

식 2-41, 2-42, 2-43 을 행렬로 표현하면 이는 underdetermined system 식 2-44 가 된다.

$$\boldsymbol{A}\mathbf{x} = \boldsymbol{u} \qquad \text{식 2-44}$$

$$\mathbf{x}^{\boldsymbol{e}} = [\mathbf{p}_1^{\boldsymbol{e}}, \mathbf{r}_1^{\boldsymbol{e}}, \mathbf{p}_2^{\boldsymbol{e}}, \mathbf{r}_2^{\boldsymbol{e}}]^{\boldsymbol{T}} \qquad \text{식 2-45}$$

이때 $\mathbf{x} = [\mathbf{p_1}, \mathbf{r_1}, \mathbf{p_2}, \mathbf{r_2}]$ 이고 식 2-44 는 underdetermined system 이므로, 이를 풀기 위해 미리 estimation[3]한 값인 식 2-45 를 식 2-44 에 대입하면 식 2-44 는 다음 식 2-46 이 된다

$$\begin{bmatrix} -3(1-\lambda_0) & -3\lambda_0 & 0 & 0 \\ (1-\lambda_1) & \lambda_1 & (1-\lambda_0) & \lambda_0 \\ 0 & 0 & -3(1-\lambda_1) & -3\lambda_1 \end{bmatrix} \begin{bmatrix} \mathbf{p}_1^e \\ \mathbf{r}_1^e \\ \mathbf{p}_2^e \\ \mathbf{r}_2^e \end{bmatrix} =$$

$$\begin{bmatrix} ((\mathbf{1}-\boldsymbol{\lambda_1})\mathrm{p_0} + \boldsymbol{\lambda_1}\mathrm{r_0}) - \big((\mathbf{1}-\boldsymbol{\mu_1})\hat{\mathrm{q}}_\mathbf{0} + \boldsymbol{\mu_1}\hat{\mathrm{q}}_\mathbf{1}\big) - \mathbf{3}((\mathbf{1}-\boldsymbol{\mu_0})\hat{\mathrm{q}}_\mathbf{1} + \boldsymbol{\mu_0}\hat{\mathrm{q}}_\mathbf{2}) \\ \big((\mathbf{1}-\boldsymbol{\mu_1})\hat{\mathrm{q}}_\mathbf{1} + \boldsymbol{\mu_1}\hat{\mathrm{q}}_\mathbf{2}\big) + ((\mathbf{1}-\boldsymbol{\mu_0})\hat{\mathrm{q}}_\mathbf{2} + \boldsymbol{\mu_0}\hat{\mathrm{q}}_\mathbf{3} \\ ((\mathbf{1}-\boldsymbol{\lambda_0})\mathrm{p_3} + \boldsymbol{\lambda_0}\mathrm{r_3}) - \mathbf{3}\big((\mathbf{1}-\boldsymbol{\mu_1})\hat{\mathrm{q}}_\mathbf{2} + \boldsymbol{\mu_1}\hat{\mathrm{q}}_\mathbf{3}\big) - \mathbf{3}((\mathbf{1}-\boldsymbol{\mu_0})\hat{\mathrm{q}}_\mathbf{3} + \boldsymbol{\mu_0}\hat{\mathrm{q}}_\mathbf{4}) \end{bmatrix}$$

식 2-46

식 2-46 을 풀면 $\mathbf{x}^{\boldsymbol{e}}$를 구할 수 있고 이를 토대로 auxiliary linear system 인 식 2-47 을 이용하여 해를 구한다. 식 2-47 에서 $\mathbf{A^T d}$를 구하면 구하고자 하는 해 $\mathbf{x}$는 식 2-48 과 같이 구할 수 있다.

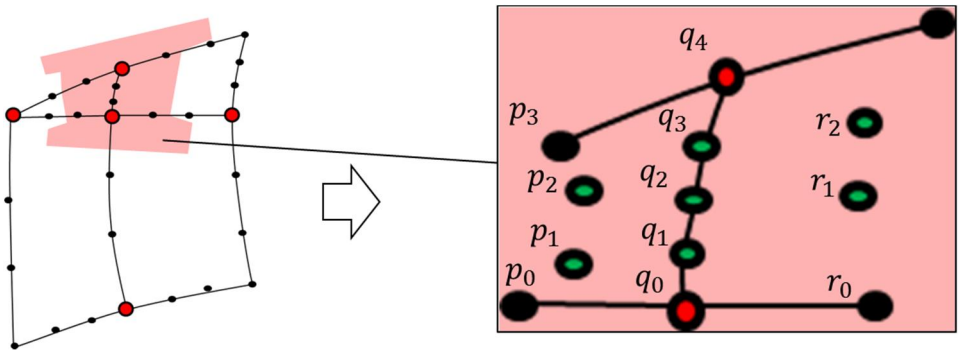
$$AA^T\mathbf{d} = \mathbf{u} - A\mathbf{x}^{\boldsymbol{e}} \qquad \text{식 2-47}$$

$$\mathbf{x} = \mathbf{x}^{\boldsymbol{e}} + A^T\mathbf{d} \qquad \text{식 2-48}$$

---

[3] 값의 추정을 말한다. 이 방법은 제 3 절에 자세히 다루었다.

2.4. G^1 곡면 fitting



[그림 2-20] 삼각 곡면과 사각 곡면 사이의 tangent ribbon

2.3.에서 보았던 G. Farin 의 G^1 연속 조건을 실제 선박의 곡면에 적용하여 보았다. 각 곡면의 경계 곡선의 조정점이 주어졌다고 가정한다. [그림 2-20]에서 보듯이 삼각 곡면 하나와 사각 곡면 3 개의 각 경계 곡선에서 G^1 연속 조건을 만족하게 해야 한다. 우선 G. Farin 의 G^1 연속 조건을 적용하기 위해 앞서 설명했듯이 경계 곡선에서 degree elevation 을 시행한다.

2.4.1. 경계 곡선을 따라 tangent ribbon estimation

Underdetermined system $A\mathbf{x}=u$를 풀기 위해 $\mathbf{x}$에 해당하는 값들을 estimation 한다. 즉, $\mathbf{p_0}, \mathbf{p_1}, \mathbf{p_2}, \mathbf{p_3}, \mathbf{r_0}, \mathbf{r_1}, \mathbf{r_2}, \mathbf{r_3}$ estimation 을 시행한다. 이때, 삼각 곡면의 경계 곡선과 사각 곡면의 경계 곡선은 차수가 다르므로 각 경우 estimation 을 달리한다.

2.4.1.1. 사각 곡면의 경계 곡선일 경우

우선, 좌측 곡면의 경우 다음과 같이 estimation 을 시행한다.

$$\mathbf{p}_0^e = \mathbf{p}_0 \qquad \text{식 } 2\text{-}49$$

$$\mathbf{p}_3^e = \mathbf{p}_3 \qquad \text{식 } 2\text{-}50$$

$$\mathbf{p}_1^e = \mathbf{q}_1 + \frac{2(\mathbf{p}_0^e - \mathbf{q}_0)}{3} + \frac{\mathbf{p}_3^e - \mathbf{q}_3}{3} \qquad \text{식 } 2\text{-}51$$

$$\mathbf{p}_2^e = \mathbf{q}_2 + \frac{\mathbf{p}_0^e - \mathbf{q}_0}{3} + \frac{2(\mathbf{p}_3^e - \mathbf{q}_3)}{3} \qquad \text{식 } 2\text{-}52$$

마찬가지 방식으로 우측의 곡면의 경우,

$$\mathbf{r}_0^e = \mathbf{r}_0 \qquad \text{식 } 2\text{-}53$$

$$\mathbf{r}_3^e = \mathbf{r}_3 \qquad \text{식 } 2\text{-}54$$

$$\mathbf{r}_1^e = \mathbf{r}_1 + \frac{2(\mathbf{r}_0^e - \mathbf{r}_0)}{3} + \frac{\mathbf{r}_3^e - \mathbf{r}_3}{3} \qquad \text{식 } 2\text{-}55$$

$$\mathbf{r}_2^e = \mathbf{q}_2 + \frac{\mathbf{r}_0^e - \mathbf{r}_0}{3} + \frac{2(\mathbf{r}_3^e - \mathbf{r}_3)}{3} \qquad \text{식 } 2\text{-}56$$

위와 같은 방식으로 식 2-45 $\mathbf{x}^e = [\mathbf{p}_1^e, \mathbf{r}_1^e, \mathbf{p}_2^e, \mathbf{r}_2^e]^T$를 구할 수 있다.

2.4.1.2. 삼각 곡면의 경계 곡선일 경우

이 경우, 경계 곡선이 4 차 식이므로 경계 곡선의 양 끝에서 tangent ribbon 의 길이를 맞춰 줘야 한다. 이 점에 있어서 2.3.1.1.에서 설명했던 $\mathbf{p}_0^e$, $\mathbf{p}_3^e$, $\mathbf{r}_0^e$, $\mathbf{r}_3^e$를 추정하는 방식을 다음과 같이 달리한다. 좌측 곡면의 경우 다음과 같이 estimation 을 시행한다.

$$\mathbf{p}_0^e = \frac{\mathbf{q}_0 + 3\mathbf{p}_0}{4} \qquad \text{식 2-57}$$

$$\mathbf{p}_3^e = \frac{\mathbf{q}_3 + 3\mathbf{p}_3}{4} \qquad \text{식 2-58}$$

마찬가지 방식으로 우측의 곡면의 경우,

$$\mathbf{r}_0^e = \frac{\mathbf{q}_0 + 3\mathbf{r}_0}{4} \qquad \text{식 2-59}$$

$$\mathbf{r}_3^e = \frac{\mathbf{q}_3 + 3\mathbf{r}_3}{4} \qquad \text{식 2-60}$$

위와 같은 방식으로 식 2-45 $\mathbf{x}^e = [\mathbf{p}_1^e, \mathbf{r}_1^e, \mathbf{p}_2^e, \mathbf{r}_2^e]^T$ 를 구할 수 있다. $\mathbf{p}_1^e$, $\mathbf{r}_1^e$, $\mathbf{p}_2^e$, $\mathbf{r}_2^e$를 추정하는 방식은 2.4.1.1. 방식과 동일하다.

2.4.2. 꼭짓점에서의 매개변수 결정

식 2-39 와 식 2-40 을 푼다. 이 방정식은 양 끝 꼭짓점에서 coplanar 조건을 적용하는 것과 동일하다. Coplanar 조건을 만족한다면 overdetermined system 일지라도 정확한 해가 나온다. 식 2-39 를 풀어 μ_0, λ_0을 구하고, 식 2-40 을 풀어 μ_1, λ_1을 구한다.

2.4.3. 내부 경계 곡선을 따라 G^1 연속 조건 적용

구한 $\mu_0, \lambda_0, \mu_1, \lambda_1$을 토대로 $A\mathbf{x}=\boldsymbol{u}$ 행렬을 풀되, 2.4.1.에서 추정했던 항들을 사용하여 행렬을 구성한다. 즉 $\mathbf{p}_0, \mathbf{p}_3, \mathbf{r}_0, \mathbf{r}_3$ 대신 estimation 후의 값인 $\mathbf{p}_0^e, \mathbf{p}_3^e, \mathbf{r}_0^e, \mathbf{r}_3^e$를 사용한다. 즉, 식 2-46 은 다음과 같이 다시 쓸 수 있다.

$$\begin{bmatrix} -3(1-\lambda_0) & -3\lambda_0 & 0 & 0 \\ (1-\lambda_1) & \lambda_1 & (1-\lambda_0) & \lambda_0 \\ 0 & 0 & -3(1-\lambda_1) & -3\lambda_1 \end{bmatrix} \begin{bmatrix} \mathbf{p}_1^e \\ \mathbf{r}_1^e \\ \mathbf{p}_2^e \\ \mathbf{r}_2^e \end{bmatrix} =$$

$$\begin{bmatrix} ((1-\lambda_1)\mathbf{p}_0^e + \lambda_1\mathbf{r}_0^e) - \left((1-\mu_1)\hat{\mathbf{q}}_0 + \mu_1\hat{\mathbf{q}}_1\right) - 3((1-\mu_0)\hat{\mathbf{q}}_1 + \mu_0\hat{\mathbf{q}}_2) \\ \left((1-\mu_1)\hat{\mathbf{q}}_1 + \mu_1\hat{\mathbf{q}}_2\right) + ((1-\mu_0)\hat{\mathbf{q}}_2 + \mu_0\hat{\mathbf{q}}_3 \\ ((1-\lambda_0)\mathbf{p}_3^e + \lambda_0\mathbf{r}_3^e) - 3\left((1-\mu_1)\hat{\mathbf{q}}_2 + \mu_1\hat{\mathbf{q}}_3\right) - 3((1-\mu_0)\hat{\mathbf{q}}_3 + \mu_0\hat{\mathbf{q}}_4) \end{bmatrix}$$

식 2-62

이때 2.4.1.에서 살펴보았듯이 사각 곡면의 경계 곡선일 경우 $[\mathbf{p}_0^e, \mathbf{p}_3^e, \mathbf{r}_0^e, \mathbf{r}_3^e] = [\mathbf{p}_0, \mathbf{p}_3, \mathbf{r}_0, \mathbf{r}_3]$ 이므로, 식 2-46 과 식 2-62 는 동일한 식이 된다. 이 후 식 2-47

$$AA^T\mathbf{d} = \mathbf{u} - A\mathbf{x}^e$$

을 least squares 를 이용하여 푼 후 $\mathbf{d}$를 구한다. 그러면 식 2-48

$$\mathbf{x} = \mathbf{x}^e + A^T\mathbf{d}$$

에서 $A^T\mathbf{d}$ 를 미리 추정한 값인 식 2-45

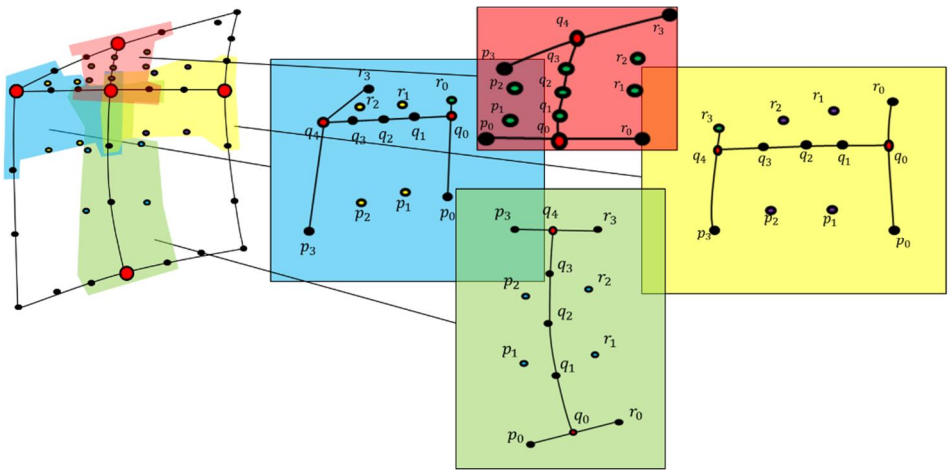
$$\mathbf{x}^e = [\mathbf{p}_1^e, \mathbf{r}_1^e, \mathbf{p}_2^e, \mathbf{r}_2^e]^T$$

에 더하여 $\mathbf{x} = [\mathbf{p}_1, \mathbf{r}_1, \mathbf{p}_2, \mathbf{r}_2]^T$를 구할 수 있다.

2.3.4. 계산된 tangent ribbon 으로 Gregory 곡면 생성

2.4.3.에서 구한 $[\mathbf{p}_1, \mathbf{r}_1, \mathbf{p}_2, \mathbf{r}_2]^T$ 로 Gregory 곡면의 내부 조정점을 update 한다. 이 때 곡면 사이의 경계 곡선이 아니므로, G^1 연속 조건을 적용하지 못하는 모서리가 있다. 이 모서리에 해당하는 tangent ribbon 의 내부 조정점은 2.4.1.에서 시행하였던 estimation 과정을 통해 구한다. 이를 토대로 사각 곡면의 경우 내부 8 개의 조정점을

블렌딩하여 곡면을 생성하고, 마찬가지로 삼각 곡면도 내부 6 개의 조정점을 블렌딩하여 곡면을 생성한다.



[그림 2-21] 각 곡면 사이 경계에서의 tangent ribbons

2.4.5. 순서 또는 방향과 무관한 G^1 연속 조건

마지막으로 [그림 2-21]와 같이 다른 모든 tangent ribbon 에도 앞서 2.4 에서 했던 과정을 적용한다. 이때, Gregory 곡면의 특징은 앞서 말했듯이 내부 8 개의 조정점으로 각 모서리에서의 cross boundary derivative 를 각기 달리 표현할 수 있다는 점이다. 즉 각 모서리마다 tangent ribbon 이 존재하고, 이들은 서로 독립적인 관계이다. [그림 2-21]처럼 네 개의 Gregory 곡면의 G^1 연속 조건을 만족하게 하려면 순서에 상관없이 모든 tangent ribbon 에서 linear system 을 풀어야 한다. 따라서 G^1 연속 조건을 적용시키는 것은 중간

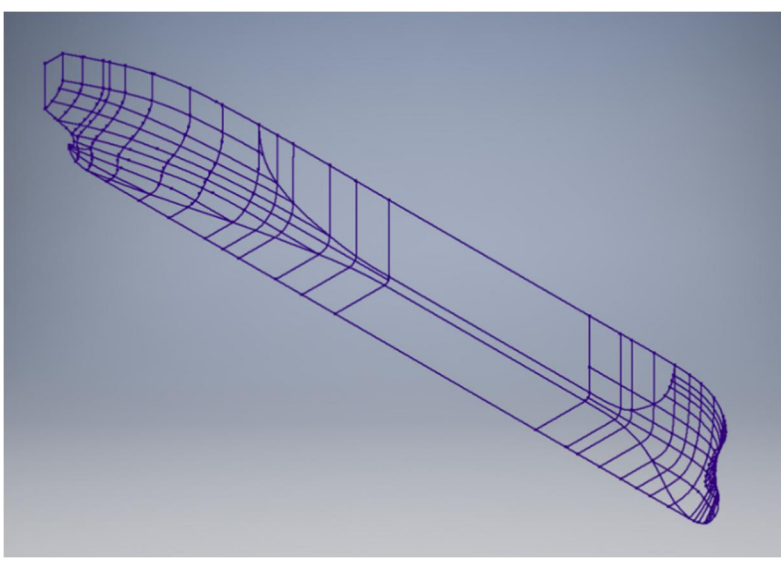
교차점을 기준으로 시계방향이든 반시계방향이든 순서에 상관이 없다. 이를 각 곡면의 네 개 경계에 모두 적용하면, 부드럽게 이어진 Gregory 곡면을 구할 수 있다.

3. 실제 선박에서의 G^1 곡면 fitting

3.1. 입력 데이터의 전처리

3.1.1. 입력 데이터 전처리 과정의 필요성

본 연구에서 원하는 입력 데이터는 곡면 사이의 경계 곡선이 3 차 Bezier 곡선인 데이터이다. 실제 입력 데이터는 이와 달라, 이를 바로 G^1 연속 조건에 적용하기에는 여러 가지 어려움이 많았다.



[그림 3-1] VLBC 의 선형

3.1.1.1. 입력 데이터의 구조

우선, 입력 데이터는 VLBC(Very Large Bulk Carrier)의 선형으로 이는 **[그림 3-1]**에서 볼 수 있다. G^1 연속 조건을 적용하기 위해서 필요한 데이터는 각 곡면에 있는 경계 곡선의 조정점이다. 또한, 경계 곡선은 3 차 Bezier 곡선으로 이루어져야만 한다. 그러나 본 연구에서 주어진 데이터는 3 차 Bezier 곡선으로 이루어진 B-spline 데이터이다.

3.1.1.2. 곡선들 간의 교차점 부재

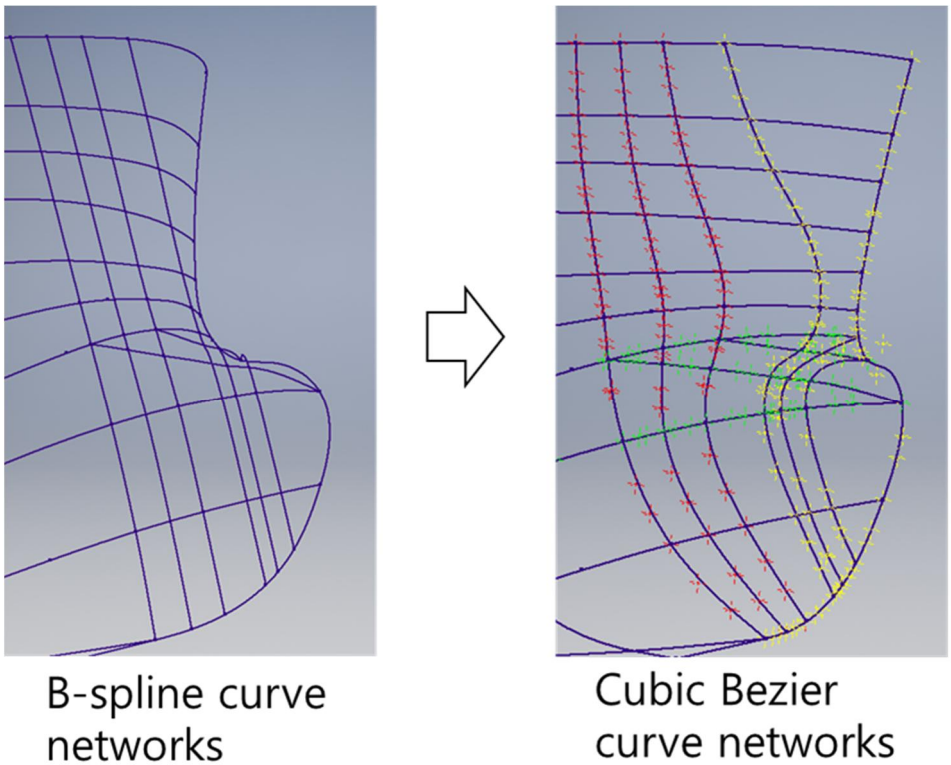
우선 입력 데이터인 B-spline 은 piece wise 3 차 Bezier 곡선이다. 하지만 각 3 차 Bezier 곡선의 시작과 끝의 조정점이 곡선들 사이의 교차점이 아닌, 임의의 위치에 불규칙하게 배열 되어 있다. 따라서 곡면을 정의하기 위해서는, 교차점에 대한 정의 및 이를 토대로 한 곡선의 재정의가 필요했다.

3.1.1.3. Coplanar 조건의 불만족

각 교차점을 구했을지라도, 각 꼭짓점을 중심으로 조정점이 coplanar 하지 않는 경우가 있으면 G^1 연속 조건을 만족하게 할 수 없다. 따라서 데이터의 전처리 과정은 필수라 할 수 있다.

3.1.2. 입력 데이터 전처리 과정

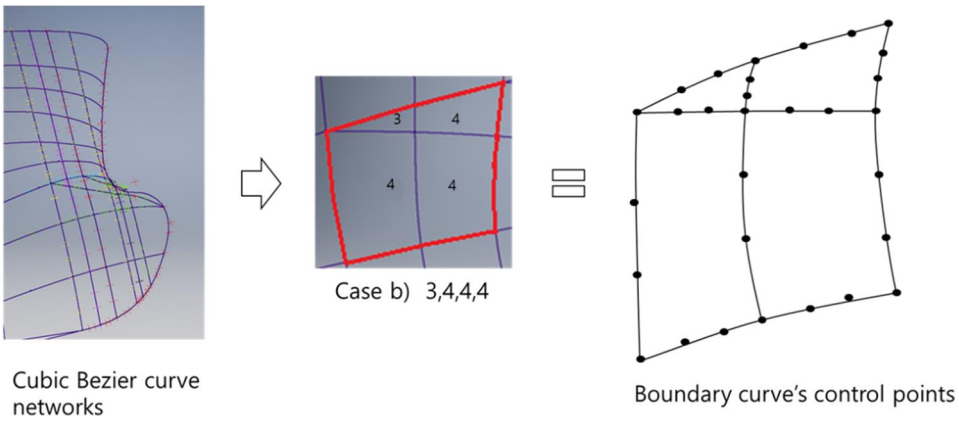
3.1.2.1. 교차점을 지나는 곡선 구하기



[그림 3-2] 교차점을 지나는 곡선 구하기

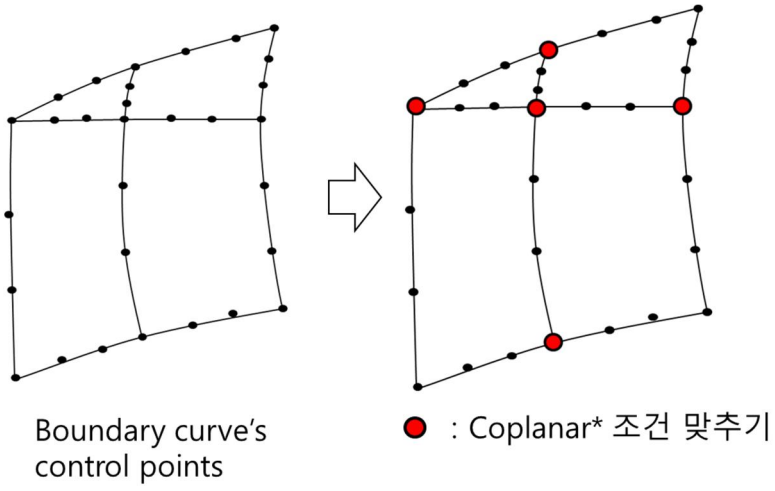
우선 곡선들 간의 교차점을 구한다. 새로이 B-spline 을 정의할 때, 이 곡선의 놋트(knot) 벡터를 교차점을 지나도록 설정하고, 교차점 사이의 곡선은 3 차 Bezier 곡선으로 한다. 교차점들을 포함하여 곡선 위의 점들을 샘플링(sampling)하고 이를 통해 곡선을 근사한다. 이 과정을 모든 곡선에 대하여 적용한다. 그리고 각 곡면의 경계 곡선에 해당하는 조정점을 곡면 사이의 연결 관계에 맞게 설정해준다.

3.1.2.2. Coplanar 조건 만족시키기



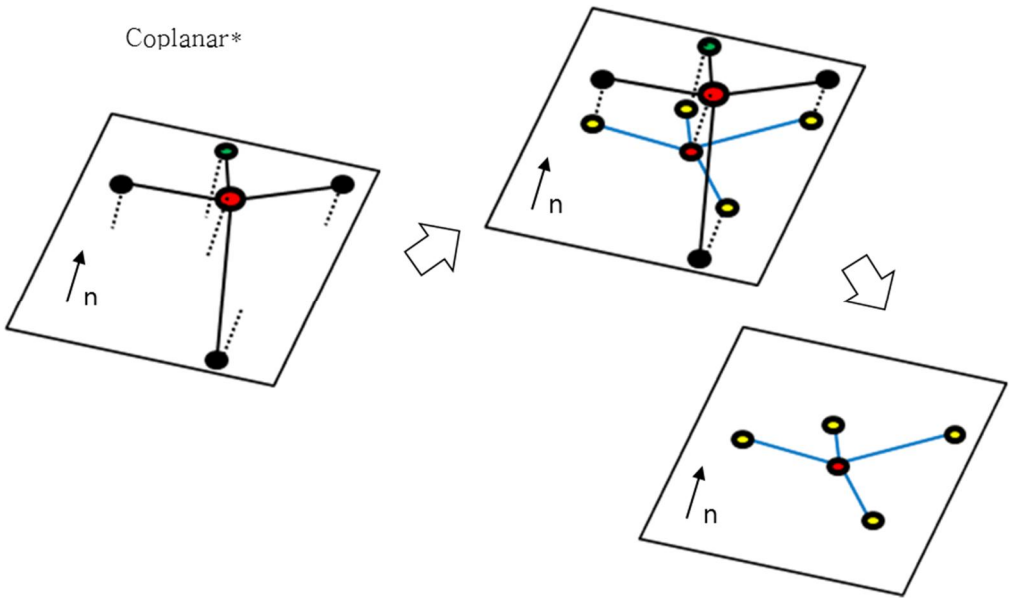
[그림 3-3] Coplanar 조건 적용을 위한 예

3.1.2.1.에서 각 곡면 사이 경계 곡선의 조정점이 모두 주어졌다. 각 꼭짓점에서, 꼭짓점을 포함한 근방의 조정점들은 coplanar 조건을 만족하여야 G^1 연속 조건을 적용시킬 수 있다. 가령 [그림 3-3]과 같이 삼각 곡면 하나와 사각 곡면 3 개로 이루어진 메시를 생각해보자.



[그림 3-4] Coplanar 조건에 해당하는 꼭짓점

꼭짓점 중 곡면 사이의 교선 위에 있는 점을 **[그림 3-4]**에서 보듯이 큰 점으로 표시했다. 이 점들 각각의 근방에서 coplanar 조건을 만족하도록 해야 한다. 이 점들 중 중앙에 있는 점을 보면, 이 점 주위의 조정점은 4 개가 있으므로 이 점을 포함한, 총 5 개의 점이 동일 평면 상에 있어야 한다. 본 논문에서는 위 조건을 만족하게 하기 위해 각 점들을 특정 평면에 투영 시키는 방법을 이용하였다.

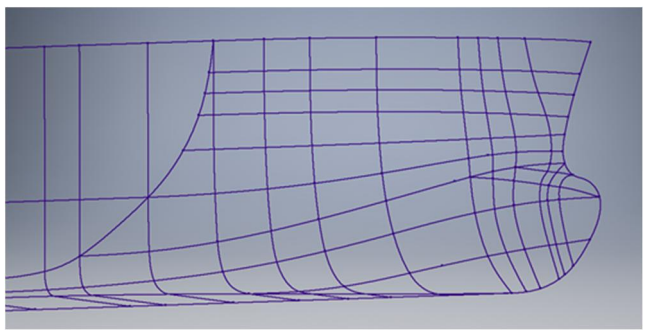


[그림 3-5] 투영하여 coplanar 조건 맞추기

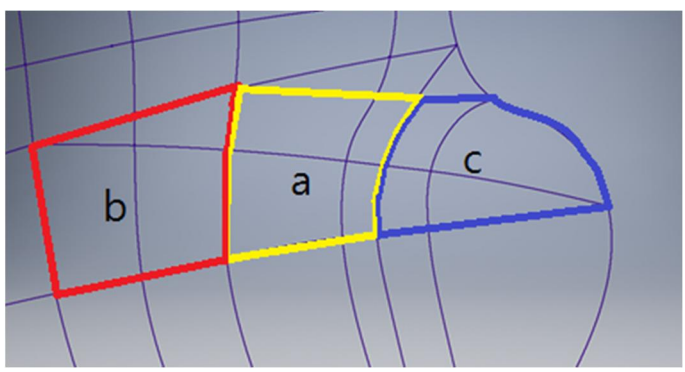
[그림 3-5]에서 보듯이, 꼭짓점을 기준으로 4 개의 조정점 있다. 이때 2 개의 조정점과 하나의 꼭짓점은 하나의 평면을 이룰 수 있고 4 개의 평면이 존재하므로, 이러한 평면을 구성하는 법선 벡터들을 총 4 개를 구할 수 있다. 구한 법선 벡터들을 평균을 낸 후, 이를 새로운 공통 평면의 법선 벡터로 설정한다. 이후 각 점들을 이 벡터를 법선

벡터로 하는 평면에 투영시킨다. 따라서 5 개의 점은 모두 동일 평면에 위치하여 coplanar 조건을 만족하게 된다.

3.2. 예제

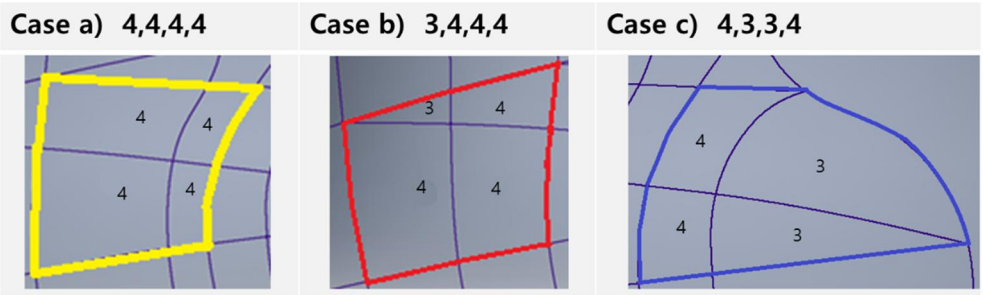


[그림 3-6] VLBC 의 선수 부분



[그림 3-7] 선수 부분 중 특정 case 의 곡면

본 논문에서는 VLBC 의 선수 부분에서 특정 곡면에 agnostic G^1 연속 조건을 적용시켜 보았다.



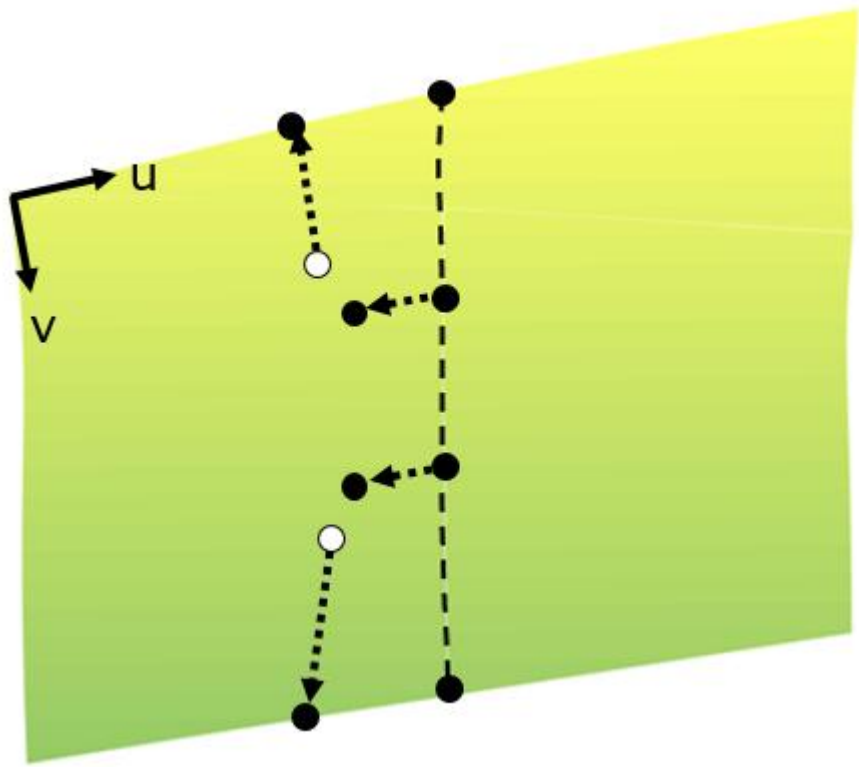
[그림 3-8] case a), b), c)

우선 case a)의 경우 4 개의 곡면 모두 사각 곡면으로 이루어진 경우, case b)는 삼각 곡면 1 개와 3 개의 사각 곡면으로 이루어진 경우, 그리고 case c)는 삼각 곡면 2 개와 사각 곡면 2 개로 이루어진 경우이다.

3.3. 곡면의 가시화 및 곡면 사이의 각도 측정

Gregory 곡면 사이의 각도를 정의하기 위하여 각 모서리의 cross boundary derivative(CBD)가 tangent ribbon 만의 함수임을 증명하고 이후 이를 이용하여 각도를 측정하는 방법을 설명하였다. 이를 토대로, 각 case 별로 agnostic G^1 연속 조건을 적용시킨 후 Gregory 곡면을 shading 을 주어 가시화하였으며, 곡면 사이의 각도를 측정하여 결과를 검증하였다.

3.3.1. Gregory 곡면의 cross boundary derivative



[그림 3-9] 사각 Gregory 곡면

두 Gregory 곡면이 **[그림 3-9]**와 같이 주어져 있다고 하자. 이때, 두 곡면 사이의 각도를 측정하기 위해선 경계 곡선 상에서 각 곡면의 법선 벡터가 필요하다. 이는 CBD 와 경계 곡선의 접선 벡터를 외적하여 구할 수 있다. 이를 위해 경계 곡선에서의 CBD 를 구한다. 앞서 2 장에서 설명했듯이 각 모서리마다 CBD 가 다르다. Gregory 곡면의 내부 조정점은 총 8 개로 각 꼭짓점 부근의 조정점이 블렌딩되어 곡면을 생성하지만, 각 모서리의 CBD 는 이들의 블렌딩과 무관하게 결정된다.

즉 각도의 측정에 쓰이는 조정점은 각 모서리의 tangent ribbon 에만 영향을 받는다. 이를 증명해보기로 한다.

$$\mathbf{b}(u,v)=\sum_{i=0}^{3}\sum_{j=0}^{3}\mathbf{b}_{ij}B_i^3(u)B_j^3(v) \qquad \text{식 } 3-1$$

$$\mathbf{b}_{11}=\frac{u}{u+v}\mathbf{b}_{11}^{10}+\frac{v}{u+v}\mathbf{b}_{11}^{01} \qquad \text{식 } 3-2$$

$$\mathbf{b}_{21}=\frac{1-u}{1-u+v}\mathbf{b}_{21}^{10}+\frac{v}{1-u+v}\mathbf{b}_{21}^{01} \qquad \text{식 } 3-3$$

$$\mathbf{b}_{12}=\frac{u}{1-u+v}\mathbf{b}_{12}^{10}+\frac{1-v}{1-v+u}\mathbf{b}_{12}^{01} \qquad \text{식 } 3-4$$

$$\mathbf{b}_{22}=\frac{1-u}{2-u-v}\mathbf{b}_{22}^{10}+\frac{1-v}{2-u-v}\mathbf{b}_{22}^{01} \qquad \text{식 } 3-5$$

식 3−1 을 전개하면 다음과 같다.

$$\begin{aligned}\mathbf{b}(u,v)=\ &\mathbf{b}_{00}(1-u)^3(1-\mathrm{v})^3+3\mathbf{b}_{01}(1-u)^3v(1-v)^2\\&+3\mathbf{b}_{02}(1-u)^3v^2(1-v)+\mathbf{b}_{03}(1-u)^3v^3\\&+3\mathbf{b}_{10}u(1-u)^2(1-v)^3+9\mathbf{b}_{11}u(1-u)^2v(1-v)^2\\&+9\mathbf{b}_{12}u(1-u)^2v^2(1-v)\ +3\mathbf{b}_{13}u(1-u)^2v^3\\&+3\mathbf{b}_{20}u^2(1-u)(1-v)^3+9\mathbf{b}_{21}u^2(1-u)v(1-v)^2\\&+9\mathbf{b}_{22}u^2(1-u)v^2(1-v)+3\mathbf{b}_{23}u^2(1-u)v^3+\mathbf{b}_{30}u^3(1-v)^3\\&+3\mathbf{b}_{31}u^3v(1-v)^2+3\mathbf{b}_{32}u^3v^2(1-v)+\mathbf{b}_{33}u^3v^3\end{aligned} \qquad \text{식 } 3-6$$

식 3−6 에 식 3−2 ~ 3−5 를 대입하면 다음과 같다.

$$\mathbf{b}(u,v) = \mathbf{b_{00}}(1-u)^3(1-\mathbf{v})^3 + 3\mathbf{b_{01}}(1-u)^3v(1-v)^2$$
$$+ 3\mathbf{b_{02}}(1-u)^3v^2(1-v) + \mathbf{b_{03}}(1-u)^3v^3$$
$$+ 3\mathbf{b_{10}}u(1-u)^2(1-v)^3 + 9(\frac{u}{u+v}\mathbf{b_{11}^{10}}$$
$$+ \frac{v}{u+v}\mathbf{b_{11}^{01}})u(1-u)^2v(1-v)^2 + 9(\frac{u}{1-u+v}\mathbf{b_{12}^{10}}$$
$$+ \frac{1-v}{1-v+u}\mathbf{b_{12}^{01}})u(1-u)^2v^2(1-v) + 3\mathbf{b_{13}}u(1-u)^2v^3$$
$$+ 3\mathbf{b_{20}}u^2(1-u)(1-v)^3 + 9(\frac{1-u}{1-u+v}\mathbf{b_{21}^{10}}$$
$$+ \frac{v}{1-u+v}\mathbf{b_{21}^{01}})u^2(1-u)v(1-v)^2 + 9(\frac{1-u}{2-u-v}\mathbf{b_{22}^{10}}$$
$$+ \frac{1-v}{2-u-v}\mathbf{b_{22}^{01}})u^2(1-u)v^2(1-v) + 3\mathbf{b_{23}}u^2(1-u)v^3$$
$$+ \mathbf{b_{30}}u^3(1-v)^3 + 3\mathbf{b_{31}}u^3v(1-v)^2 + 3\mathbf{b_{32}}u^3v^2(1-v)$$
$$+ \mathbf{b_{33}}u^3v^3$$

식 3−7

[그림 3−9]에서 경계 곡선은 $v=1$에서의 값이므로 CBD 를 구하기 위해 식 3−7 을 v에 대하여 미분한 후 $v=1$을 대입한다. 미분 후에 $v=1$을 대입하면 $(1-v)$가 포함된 항은 모두 소거되므로 식 3−7 은 다음과 같이 간단히 정리할 수 있다.

$$\frac{\partial}{\partial v}\mathbf{b}(u,v)|_{v=1} = -3\mathbf{b}_{02}(1-u)^3 + 3\mathbf{b}_{03}(1-u)^3 - 9\mathbf{b}_{12}^{10}u(1-u)^2$$
$$+ 9\mathbf{b}_{13}u(1-u)^2 - 9\mathbf{b}_{22}^{10}u^2(1-u) + 9\mathbf{b}_{23}u^2(1-u) - 3\mathbf{b}_{32}u^3$$
$$+ 3\mathbf{b}_{33}u^3$$

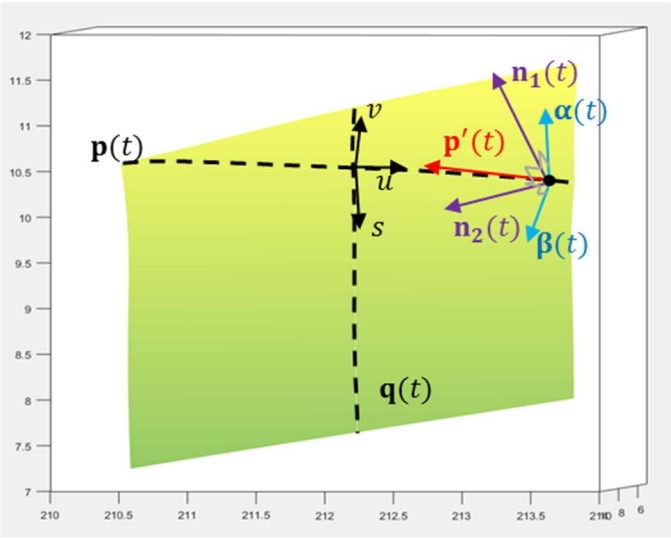
식 3−8

$$\frac{\partial}{\partial \boldsymbol{v}}\mathbf{b}(u,v)|_{\boldsymbol{v=1}} = 3(1-u)^3(\mathbf{b_{03}} - \mathbf{b}_{02}) + 9u(1-u)^2(\mathbf{b}_{13} - \mathbf{b}_{12}^{10})$$
$$+ 9u^2(1-u)(\mathbf{b}_{23} - \mathbf{b}_{22}^{10}) - 3u^3(\mathbf{b}_{33} - \mathbf{b}_{32})$$

식 3-9

식 3-9 에 의하면 $v=1$ 에서 CBD 는 블렌딩되는 다른 모서리의 Gregory 내부 조정점($\mathbf{b}_{12}^{01}$, $\mathbf{b}_{22}^{01}$)과는 상관없이 ($\mathbf{b}_{12}^{10}$, $\mathbf{b}_{22}^{10}$)에 영향을 받음을 알 수 있다. 또한, 삼각 Gregory 곡면의 tangent ribbon 의 구조가 사각 Gregory 곡면과 같으므로 삼각 Gregory 곡면도 동일한 방식으로 증명할 수 있다. 따라서 Gregory 곡면 사이의 각도 측정은 각 모서리의 tangent ribbon 정보만을 필요로 한다. 3.3.2.에선 이를 토대로 각도를 측정하였다.

3.3.2. 곡면 사이의 각도 측정 방법



[그림 3-10] 각도 측정 방법

곡면 사이의 각도는 경계 곡선 위의 모든 점에서 측정하였다. 우선 그림 경계 곡선 $\mathbf{p}(t)$ 위의 점에서 곡선 방향으로의 접선 벡터를 $\mathbf{p}'(t)$라 하자. 곡선 방향을 u, 이와 수직한 방향을 v, s라하고, $\mathbf{p}'(t)$는 v=0 일 때의 곡선을 u 방향으로 미분한 것이므로 $\mathbf{p}'(t) = \mathbf{p}'(u)$라 할 수 있다. 이때의 접선 벡터 $\mathbf{p}'(t)$는 식 2-1 을 미분하여 구할 수 있고 이는 다음과 같다.

$$\mathbf{p}'(t) = \mathbf{p}'(u) = \frac{\partial}{\partial u}\mathbf{b}(u,0) \qquad \text{식 3-9}$$

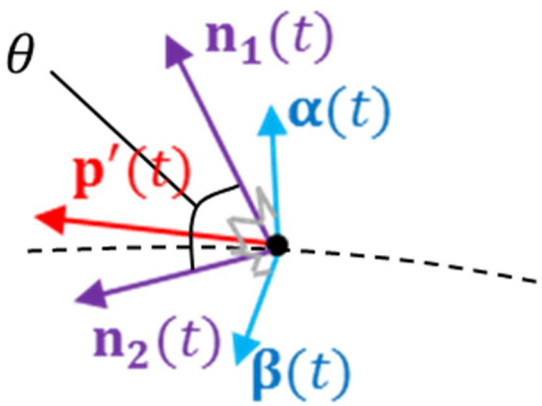
특정 $t = t_0$ 에서의 u를 u_0이라 하면 이때의 접선 벡터는 식 3-9 에 $u = u_0$을 대입한 값과 같다.

같은 점에서 각 곡면으로의 CBD 를 구하기 위해서 $u = u_0$ 에서의 곡선 $\mathbf{b}(u_0, v)$ 을 각각 v 방향, s 방향으로 미분한다. $\boldsymbol{\alpha}(t) = \boldsymbol{\alpha}(v), \boldsymbol{\beta}(t) = \boldsymbol{\beta}(s)$라 쓸 수 있으며 이는 다음과 같다.

$$\boldsymbol{\alpha}(t) = \boldsymbol{\alpha}(v) = \frac{\partial}{\partial v}\mathbf{b}(u_0, v) \qquad \text{식 } 3-10$$

$$\boldsymbol{\beta}(t) = \boldsymbol{\beta}(s) = \frac{\partial}{\partial s}\mathbf{b}(u_0, s) \qquad \text{식 } 3-11$$

이때 구하고자 하는 벡터는 경계 곡선 $\mathbf{p}(t)$ 위의 점에서 CBD 이므로 식 3-10, 식 3-11 에 $v = 0,\ s = 0$을 대입한 값과 같다.



[그림 3-11] 각도 측정 방법 확대

위 값들을 구한 후 그 점에서 각 곡면의 법선 방향 벡터를 구한다. $\mathbf{p}'(t)$과 $\boldsymbol{\alpha}(t)$를 외적한 벡터를 $\mathbf{n}_1$, $\mathbf{p}'(t)$과 $\boldsymbol{\beta}(t)$를 외적한 벡터를 $\mathbf{n}_2$라 하자.

$$\mathbf{n_1}(t) = \mathbf{p}'(t) \times \boldsymbol{\alpha}(t) \qquad \text{식 } 3-12$$

$$\mathbf{n_2}(t) = \mathbf{p}'(t) \times \boldsymbol{\beta}(t) \qquad \text{식 } 3-13$$

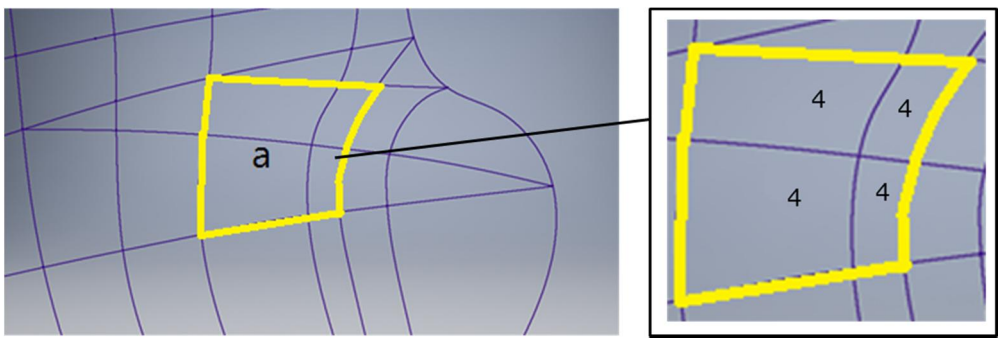
두 법선 벡터 사이의 각도 θ를 측정함으로써 두 곡면 사이의 각도를 알 수 있고 θ는 다음과 같다.

$$\theta = \cos^{-1}\frac{\mathbf{n}_1 \cdot \mathbf{n}_2}{|\mathbf{n}_1||\mathbf{n}_2|}$$

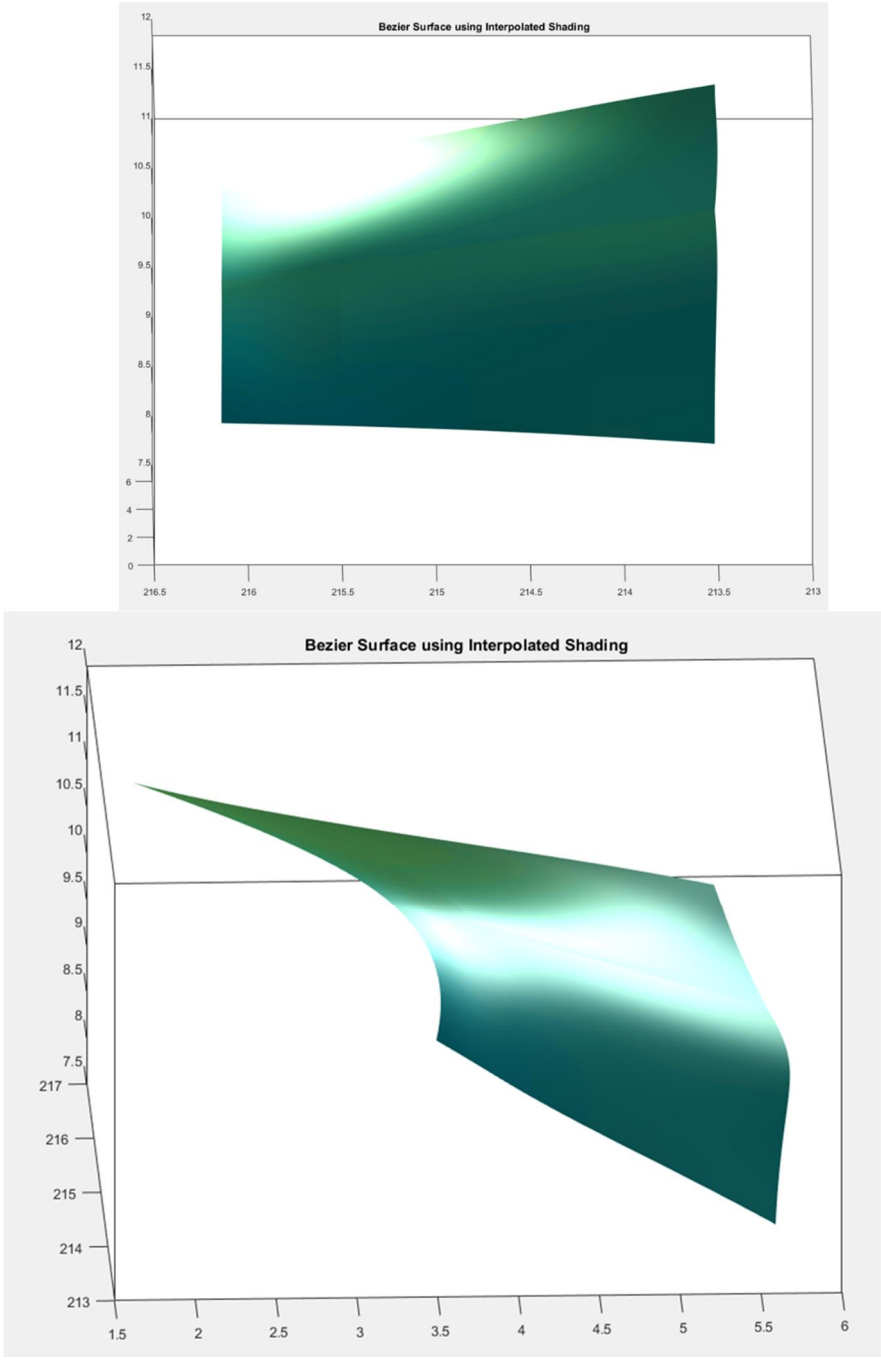
식 3-14

이를 토대로 각 case 별로 각도를 측정한 결과 그 값이 10^{-5} ° 로 0 에 가까움을 알 수 있다. 결과는 다음과 같다.

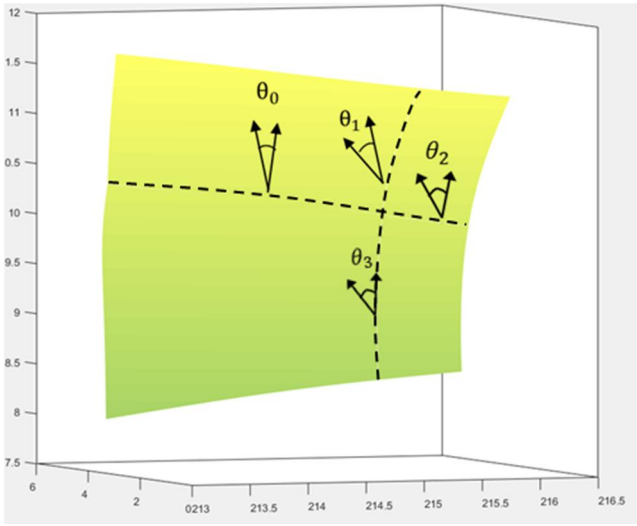
3.3.3. Case a)



[그림 3-12] Case a) 4,4,4,4



[그림 3-13] Case a) 곡면 모델링 결과

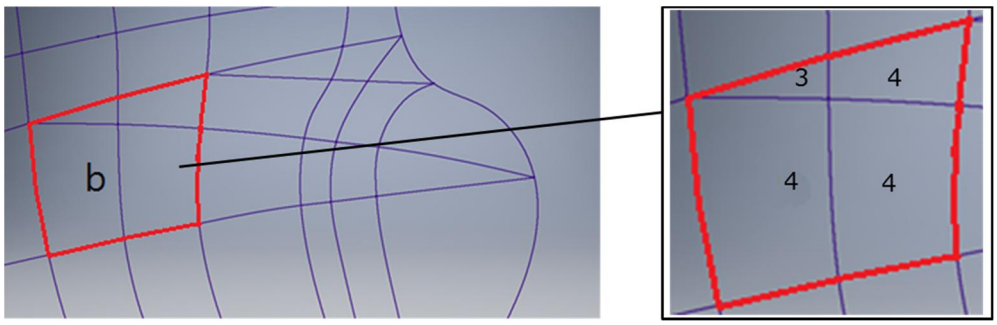


[그림 3-14] Case a)의 각도 측정

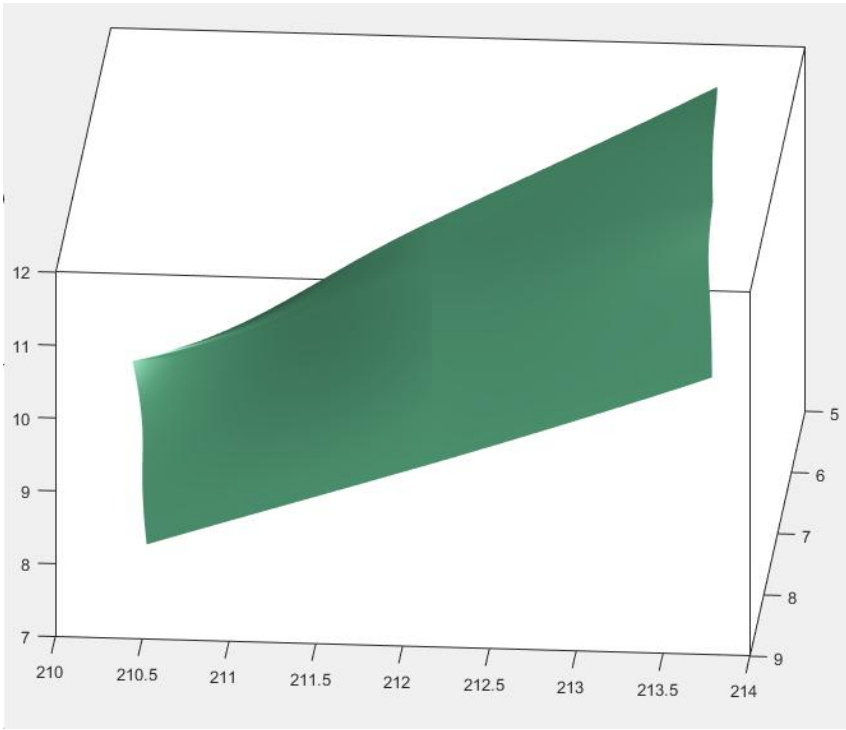
[표 3-1] Case a)의 각도 측정 결과

$\boldsymbol{\theta}$	min ~ max
$\boldsymbol{\theta_0}$	$0.0000 \sim 0.1708 * 10^{-5}$
$\boldsymbol{\theta_1}$	$0.0000 \sim 0.1479 * 10^{-5}$
$\boldsymbol{\theta_2}$	$0.0000 \sim 0.1207 * 10^{-5}$
$\boldsymbol{\theta_3}$	$0.0000 \sim 0.1708 * 10^{-5}$

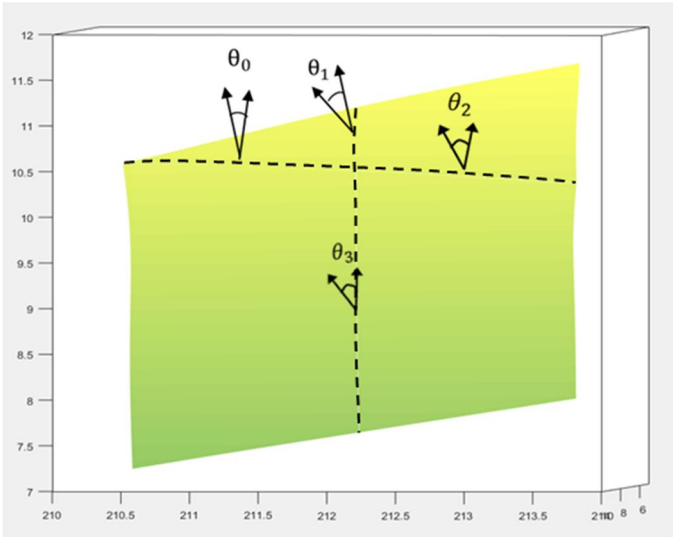
3.3.4. Case b)



[그림 3-15] Case b) 3,4,4,4



[그림 3-16] Case b) 곡면 모델링 결과

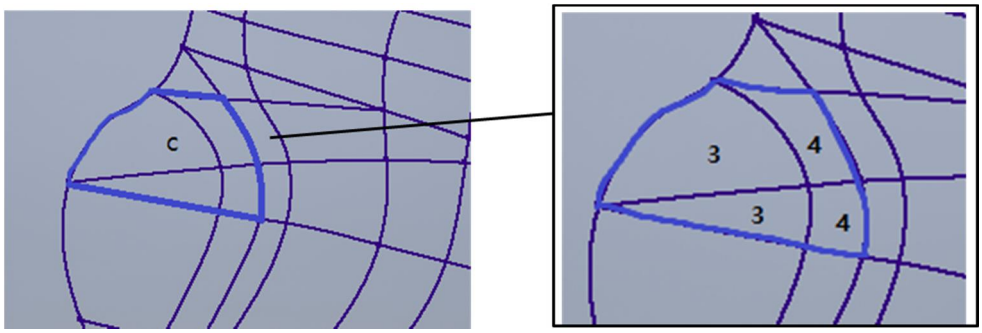


[그림 3-17] Case b)의 각도 측정

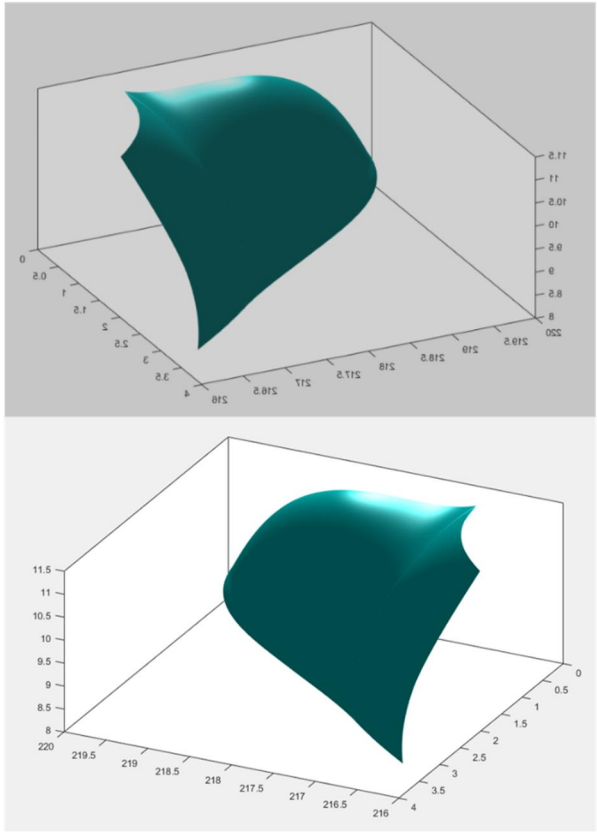
[표 3-2] Case b)의 각도 측정 결과

$\boldsymbol{\theta}$	min ~ max
$\boldsymbol{\theta_0}$	$0.0000 \sim 0.1708 * 10^{-5}$
$\boldsymbol{\theta_1}$	$0.0000 \sim 0.1479 * 10^{-5}$
$\boldsymbol{\theta_2}$	$0.0000 \sim 0.1207 * 10^{-5}$
$\boldsymbol{\theta_3}$	$0.0000 \sim 0.1708 * 10^{-5}$

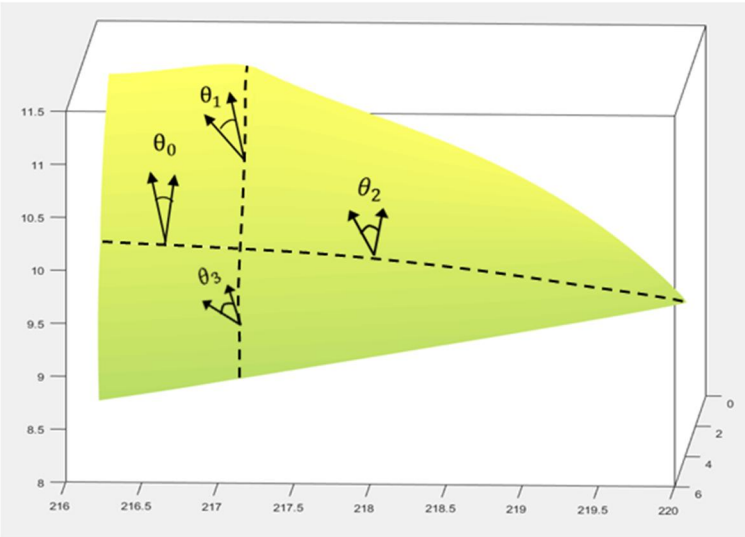
3.3.5. Case c)



[그림 3-18] Case c) 3,4,4,3



[그림 3-19] Case c) 곡면 모델링 결과



[그림 3-20] Case c)의 각도 측정

[표 3-3] Case c)의 각도 측정 결과

$\boldsymbol{\theta}$	min ~ max
$\boldsymbol{\theta_0}$	$0.0000 \sim 0.1708 * 10^{-5}$
$\boldsymbol{\theta_1}$	$0.0000 \sim 0.1479 * 10^{-5}$
$\boldsymbol{\theta_2}$	$0.0000 \sim 0.1207 * 10^{-5}$
$\boldsymbol{\theta_3}$	$0.0000 \sim 0.1708 * 10^{-5}$

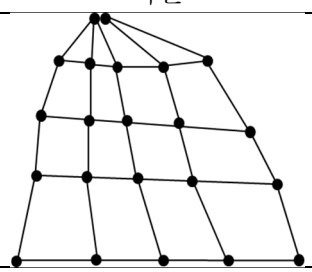
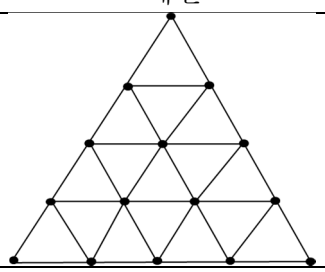
4. 결론

정리하면, 본 연구는 다음과 같은 차별성 및 장점을 가지고 있다.

• 삼각 곡면, 사각 곡면 모두 같은 구조의 tangent ribbon 을 갖게 되어 삼각/사각 구분 없이 G^1 연속 조건을 적용할 수 있다..

• Gregory 곡면의 조정점이 모서리 방향마다 cross boundary derivative 를 각각 표현할 수 있음을 이용하여, 곡면 사이의 G^1 은 순서에 상관없이 적용 가능하다.

• Local method 로서 global method 에 비해 간단하다..

	기존	제안
삼각 곡면		
	퇴화된 사각 기저 함수	삼각 기저 함수

[표 4-1] 삼각 곡면 생성 시 차이점

• **[표 4-1]**에서 보듯이 삼각 곡면을 표현할 때, 기존의 곡면 모델링 방식은 퇴화된 사각 기저 함수를 이용하였으나, 본 연구에서는 삼각 기저 함수를 이용하였다.

결론적으로는 선체의 곡면 중 특정 경우들에 “agnostic G^1 Gregory 곡면” 을 이용하여 모델링 한 결과, G^1 연속 조건을 만족하여 추후 선체 곡면 모델링에 있어서 제안한 본 연구를 활용 가능할 것이다.

또한 다음과 같은 사항들이 더 추가될 필요가 있다.

- T junction 을 가진 선체 전체를 본 연구 내용을 토대로 G^1 연속 조건을 만족하도록 곡면을 모델링
- Gregory 곡면을 상용 가능한 IGES 파일로 전환
- 전체적인 과정의 모듈화

참고 문헌

1. Farin, G., & Hansford, D. (2012). Agnostic G1 Gregory surfaces. *Graphical Models*, *74*(6), 346-350.

2. Farin, G. E. (2002). *Curves and surfaces for CAGD: a practical guide*. Morgan Kaufmann.

3. Chiyokura, H., & Kimura, F. (1984, September). A new surface interpolation method for irregular curve models. In *Computer Graphics Forum* (Vol. 3, No. 3, pp. 209-218). Blackwell Publishing Ltd.

4. Du, W. H., & Schmitt, F. J. (1990). On the G1 continuity of piecewise Bezier surfaces: a review with new results. *Computer-Aided Design*, *22*(9), 556-573.

5. Peters, J. (1991). Smooth interpolation of a mesh of curves. *Constructive Approximation*, *7*(1), 221-246.

6. Liu, Q., & Sun, T. C. (1994). G1 interpolation of mesh curves. *Computer-Aided Design*, *26*(4), 259-267.

7. Shi, X., Wang, T., & Yu, P. (2004). A practical construction of G1 smooth biquintic B-spline surfaces over arbitrary topology. *Computer-Aided Design*, *36*(5), 413-424.

8. Cho, D. Y., Lee, K. Y., & Kim, T. W. (2006). Interpolating G 1 Bézier surfaces over irregular curve networks for ship hull design. *Computer-Aided Design*, *38*(6), 641-660.

9. 임중현, 조두연, 이규열, & 김태완. (2005). 가상 등매개변수 곡선을 이용한 곡선그물망을 보간하는 discrete G^1 B-spline 선박형상곡면 생성. *한국 CAD/CAM 학회 학술발표회 논문집*, 542-549.

10. 조두연, 이규열, 김태완. (2005). 곡선그물망을 보간하는 discrete G^1 선박형상곡면생성 및 fairing. 한국 CDE 학회 학술발표회 논문집, 1114-1125.

11. Sebastian H. G & Robert B. (2016, September) "Using subdivision surfaces to address the limitations of B-spline surfaces in ship hull form modeling." In *Proceedings of PRADS2016 Copenhagen, Denmark*

12. Farin, G. (1982). A construction for visualC1 continuity of polynomial surface patches. *Computer Graphics and Image Processing*, *20*(3), 272-282.

부록. Source code of important functions

```
function [ y ] = coplanar_check(x1,x2,x3,x4 )
%COPLANAR 유무를 CHECK.
y = dot((x3-x1),cross((x2-x1),(x4-x3)));
end

function [ updated_center,updated_p ] =
%COPLANAR 조건을 맞추기 위해 center를 기준으로 평면의 normal vector
들을 구하고 평균을 낸 뒤 그 평면에 사영시킨다.
make_coplanar( center,p1,p2,p3,p4 )
center = squeeze(center)
p1 = squeeze(p1)
p2 = squeeze(p2)
p3 = squeeze(p3)
if nargin < 5
normal = makenormal_average(center,p1,p2,p3);
updated_center = makenormal_projection(normal,center);
updated_p(1,:) = makenormal_projection(normal,p1);
updated_p(2,:) = makenormal_projection(normal,p2);
updated_p(3,:) = makenormal_projection(normal,p3);
else
p4 = squeeze(p4)
normal = makenormal_average(center,p1,p2,p3,p4);
updated_center = makenormal_projection(normal,center);
updated_p(1,:) = makenormal_projection(normal,p1);
updated_p(2,:) = makenormal_projection(normal,p2);
updated_p(3,:) = makenormal_projection(normal,p3);
updated_p(4,:) = makenormal_projection(normal,p4);
end
end
```

```
function [ ribbon ,not_elevated_ribbon] = tangentRibbon( LP,RP )

%%output ribbon -> [5x3x3] double 빈공간은 (0,0,0)
% p33 q4 r33
%       q3
%       q2
%       q1
% p00 q0 r00
%% degree elevation of q
q = LP(:,4,:)
q0_ = q(4,:)
q1_ = q(3,:)
q2_ = q(2,:)
q3_ = q(1,:)
not_ele_q = [q3_;q2_;q1_;q0_]
not_ele_q = cat(3,not_ele_q(:,1),not_ele_q(:,2),not_ele_q(:,3))
q0 = q0_
q1 = (q0_+q1_*3)/4
q2 = (q1_+q2_)/2
q3 = (3*q2_+q3_)/4
q4 = q3_
elevated_q = [q4; q3; q2; q1; q0]
elevated_q = cat(3,elevated_q(:,1),elevated_q(:,2),elevated_q(:,3))
p33 = LP(1,3,:);
p22 = LP(2,3,:);
p11 = LP(3,3,:);
p00 = LP(4,3,:);
r33 = RP(1,2,:);
r22 = RP(2,2,:);
r11 = RP(3,2,:);
r00 = RP(4,2,:);
q3_ = not_ele_q(1,:,:)
q2_ = not_ele_q(2,:,:)
q1_ = not_ele_q(3,:,:)
q0_ = not_ele_q(4,:,:)
q4 = elevated_q(1,:,:)
q3 = elevated_q(2,:,:)
q2 = elevated_q(3,:,:)
```

```
q1 = elevated_q(4,:,:)
q0 = elevated_q(5,:,:)
temp = cat(3,0,0,0);
not_elevated_ribbon_(:,:,:) = [p33 p22 p11 p00;q3_ q2_ q1_ q0_;r33 r22
r11 r00]
ribbon_(:,:,:) = [p33,p22,temp,p11,p00;q4 q3 q2 q1
q0;r33,r22,temp,r11,r00];
not_elevated_ribbon(:,:,:) =
cat(3,not_elevated_ribbon_(:,:,1)',not_elevated_ribbon_(:,:,2)',not_elevated_
ribbon_(:,:,3)')
ribbon(:,:,:) = cat(3,ribbon_(:,:,1).',ribbon_(:,:,2).',ribbon_(:,:,3).')
end
function [ lamda, mu ] = lamda1( ribbon )

%%input P : patch's ribbon ( q is not elevated yet!)
%% so input must be 'not_elevated_ribbon' of tangentRibbon function
%%output lamda0,lamda1,mu0,mu1
p = ribbon(:,1,:)
q = ribbon(:,2,:)
r = ribbon(:,3,:)
q0 = q(5,:)
q1 = q(4,:)
q2 = q(3,:)
q3 = q(2,:)
q4 = q(1,:)
p0 = p(5,:)
p3 = p(1,:)
r0 = r(5,:)
r3 = r(1,:)
A = [r0(:)-p0(:), q0(:)-q1(:)]
b = [q0(:)-p0(:)]
x1 = A\b
A2 = [r3(:)-p3(:), q3(:)-q4(:)]
b2 = [q3(:)-p3(:)]
x2 = A2\b2
lamda = [x1(1),x2(1)]
mu = [x1(2),x2(2)]
end
```

```
function [ A,u ] = MatrixCompu2( lamda, mu, ribbon )

%% G1 condtion's matrix expression
%% Ax = u -> x = [p1,r1,p2,r2]
p = ribbon(:,1,:)
p33 = p(1,:)
p00 = p(5,:)
q = ribbon(:,2,:)
q4 = q(1,:)
q3 = q(2,:)
q2 = q(3,:)
q1 = q(4,:)
q0 = q(5,:)
r = ribbon(:,3,:)
r33 = r(1,:)
r00 = r(5,:)
lamda0 = lamda(1);
lamda1 = lamda(2);
A = [-3*(1-lamda0) -3*lamda0 0 0;1-lamda1 lamda1 1-lamda0
lamda0;0 0 -3*(1-lamda1) -3*lamda1]
mu0 = mu(1);
mu1 = mu(2);
u = [((1-lamda1)*p00+lamda1*r00)-(((1-mu1)*q0+mu1*q1))-3*((1-
mu0)*q1+mu0*q2);
((1-mu1)*q1+mu1*q2)+((1-mu0)*q2+mu0*q3);
((1-lamda0)*p33+lamda0*r33)-3*((1-mu1)*q2+mu1*q3)-((1-
mu0)*q3+mu0*q4)]
end
```

```
function [updated_ribbon, x ,not_elevated_ribbon ] = computeX( A, u,
ribbon,not_elevated_ribbon )
%% x_e = [p1_e,r1_e,p2_e,r2_e]
A
u
x_e = [ribbon(4,1,:) ribbon(4,3,:) ribbon(2,1,:) ribbon(2,3,:)]
x_e =[x_e(:,:,1);x_e(:,:,2);x_e(:,:,3)]
x_et = x_e'
Axe = A * x_et
u_Axe = u - Axe
d = inv(A*A')*u_Axe
A'*d
x_et
x = x_et + A'*d
% %
% Moore- Penrose pseudoinverse
% x = A'*inv(A*A')*u
%update ribbon's inner points
updated_ribbon = ribbon;
updated_ribbon(4,1,:) = x(1,:) %p1
updated_ribbon(2,1,:) = x(3,:) %p2
updated_ribbon(4,3,:) = x(2,:) %r1
updated_ribbon(2,3,:) = x(4,:) %r2
if nargin > 3
not_elevated_ribbon(3,1,:) = x(1,:) %p1
not_elevated_ribbon(2,1,:) = x(3,:) %p2
not_elevated_ribbon(3,3,:) = x(2,:) %r1
not_elevated_ribbon(2,3,:) = x(4,:) %r2
end
end
```

```
function [ Theta,Theta_sin,lGra,rGra,lrGra ] = ThetaEstimate( LCP,RCP )
%input
%LCP: Left patch's control points , RCP : Right patch's control points
%% 수학적으로 각도 측정. 3 공통 curve에서 법선벡터를 각각 구한다
% @u_B(1,v) = 3*(1-v)^3*(P30-P20)+3(1-v)^2*v*(P31-P21) +
% 3*(1-v)*v^2*(P32-P22)+v^3*(P33-P23)
% r0,r1,r2,r3 를 지나는 bezier curve위의 점들을 지나는 점들을 구하고 공통
커브에서 그 점들을 빼서 벡터를 구한다.
% curve에서 처음 두 control points를 이은 벡터가 바로 tangent vector이므
로...
% | | | -> 우선 세곡선의 control points들을 뽑은 후 curve를 구축한다.
% | | |
% | | |
% | | |
% | | |
% 각각 공통, 왼쪽, 오른쪽 ControlPoints 정의
cCP(:,:) = LCP(:,4,:); %center's Control Points
lCP(:,:) = LCP(:,3,:); %left's Control Points
rCP(:,:) = RCP(:,2,:); %right's Control Points
length = 100;
for t=1:length+1
i = (t-1)/length;
%left Gradient , Right Gradient
lTan(t,:) = (cCP(4,:)-lCP(4,:))*(1-i)^3+(cCP(3,:)-lCP(3,:))*3*(1-
i)^2*i+(cCP(2,:)-lCP(2,:))*3*(1-i)*i^2+(cCP(1,:)-lCP(1,:))*i^3;
rTan(t,:) = (cCP(4,:)-rCP(4,:))*(1-i)^3+(cCP(3,:)-rCP(3,:))*3*(1-
i)^2*i+(cCP(2,:)-rCP(2,:))*3*(1-i)*i^2+(cCP(1,:)-rCP(1,:))*i^3;
v_directionTan(t,:) = -3*(1-i)^2*cCP(4,:)+(3*(1-i)^2-6*(1-
i)*i)*cCP(3,:)+(6*(1-i)*i-3*i^2)*cCP(2,:)+3*i^2*cCP(1,:);
lGra(t,:) = cross(lTan(t,:),v_directionTan(t,:));
rGra(t,:) = cross(rTan(t,:),v_directionTan(t,:));
test2(t,:) = cross(lGra(t,:),rGra(t,:));
Theta_sin(t) = asin(norm(test2(t,:))/norm(lGra(t,:))/norm(rGra(t,:)));
lrGra(t,:) = [lGra(t,:) rGra(t,:)];
end
for i = 1:length+1
lnorm(i) = norm(lGra(i,:));
rnorm(i) = norm(rGra(i,:));
```

```
dotOfGrad(i) = dot(lGra(i,:),rGra(i,:));
if dotOfGrad(i)/lnorm(i)/rnorm(i) > 1
b(i) = 1;
else
Theta(i) = acos(dotOfGrad(i)/lnorm(i)/rnorm(i));
Theta(i) = min(Theta(i),pi-Theta(i));
Theta(i) = Theta(i) * 180 / 3.141592;
end
end
end
```

```
% % Function to interpolate 20 control points of a Gregory patch.
% % -> A Gregory patch is defined by 20 control points
% % -> dim is the dimension, e.g., for 3D a control point
% % has three coordinates (x,y,z)
% % Details
% % -> Input Matrix P stores 16 control points of a patches
% % -> Size of P is 4 x 4 x dim
% % -> P(:,:,k) holds control points of kth dimension
% % -> Size of P(:,:,k) is 4 x 4 i.e., 16 control points.
% % -> For example for 3D (dim=3, k=1..3) size of P(:,:,k) is 4 x 4 x 3
% % i.e., 16 control points for in each dimension
% % P(:,:,1): x-coordates of control points as 4 x 4 matrix
% % P(:,:,2): y-coordates of control points as 4 x 4 matrix
% % P(:,:,3): z-coordates of control points as 4 x 4 matrix
% % -> Input Matrix G stores 4 control points of a patches
% % -> Size of G is 2 x 2 x dim
% % -> G is composed of inner points of control points
% % which are the other two edge's cross boundary derivatives
% % -> Output matrix Q stores interpolated values between control points
% % Q is similar to P in format but has more values, i.e., it stores
% % end control points and interpolated values.
% % -> u, v and optional argments that specify number of interpolated
% % points between control points. Default values of u and v are 101
function Q = Gregorypatchinterp(P,G,varargin,n)
if nargin < 4
n = 101
end
%%% Default Values %%%
u=linspace(0.001,0.999,n); % uniform parameterization
v=u;
defaultValues = {u,v};
%%% Assign Valus %%%
[u,v] = deal(defaultValues{:});
% % ------------------------------
Q=[];
[r c dim]=size(P);
M = [
1 0 0 0;
```

```
-3 3 0 0;
3 -6 3 0;
-1 3 -3 1
];
RP = P;
OP = P;
MT = M'; % transform of matrix M
for i = 1:length(u)
for j = 1:length(v)
U = [1 u(i) u(i)^2 u(i)^3];
VT = [1 v(j) v(j)^2 v(j)^3]';
RP(2,2,:) = (P(2,2,:)*u(i)+G(1,1,:)*v(i))/(u(i)+v(i));
RP(3,2,:) = (P(3,2,:)*(1-u(i))+G(2,1,:)*v(i))/(1-u(i)+v(i));
RP(2,3,:) = (P(2,3,:)*u(i)+G(1,2,:)*(1-v(i)))/(1-v(i)+u(i));
RP(3,3,:) = (P(3,3,:)*(1-u(i))+G(2,2,:)*(1-v(i)))/(2-u(i)-v(i));
if i==1
U = [1 0 0 0];
RP = OP;
end
if j==1
VT = [1 0 0 0]';
RP = OP;
end
if i==length(u)
U = [1 1 1 1];
RP = OP;
end
if j==length(v)
VT = [1 1 1 1]'
RP = OP;
end
for k = 1:dim % interpolation for each dimension of control points
Q(i,j,k) = U * M * RP(:,:,k) * MT * VT;
end
end
end
end
```

```
function [ estimated_triangle ,elevated_boundary,boundary ] =
triangleEstimate( Triangle_boundary )
% input -> boundary
% output은 estimated 된 삼각형 patch
% -> [a b c d;e f g h;i j k l]
% 1번 모서리 1234 estimate 하기 %p00,p33,p1e,p2e
p1_1 = pev(Triangle_boundary(1,:),Triangle_boundary(5,:))
p4_1 = pev(Triangle_boundary(4,:),Triangle_boundary(6,:))
p2 =
pone(Triangle_boundary(2,:),p1_1,Triangle_boundary(1,:),p4_1,Triangle_b
oundary(4,:))
p3 =
ptwo(Triangle_boundary(3,:),p1_1,Triangle_boundary(1,:),p4_1,Triangle_b
oundary(4,:))
% 2번 모서리 9864 estimate 하기
p9_2 = pev(Triangle_boundary(9,:),Triangle_boundary(7,:))
p4_2 = pev(Triangle_boundary(4,:),Triangle_boundary(3,:))
p8 =
pone(Triangle_boundary(8,:),p9_2,Triangle_boundary(9,:),p4_2,Triangle_b
oundary(4,:))
p6 =
ptwo(Triangle_boundary(6,:),p9_2,Triangle_boundary(9,:),p4_2,Triangle_b
oundary(4,:))
% 3번 모서리 1579 estimate 하기
p1_3 = pev(Triangle_boundary(1,:),Triangle_boundary(2,:))
p9_3 = pev(Triangle_boundary(9,:),Triangle_boundary(8,:))
p5 =
pone(Triangle_boundary(5,:),p1_3,Triangle_boundary(1,:),p9_3,Triangle_b
oundary(9,:))
p7 =
ptwo(Triangle_boundary(7,:),p1_3,Triangle_boundary(1,:),p9_3,Triangle_b
oundary(9,:))
for i = 1:3
estimated_triangle(:,:,i) =
[p1_1(i),p2(i),p3(i),p4_1(i);p9_2(i),p8(i),p6(i),p4_2(i);p1_3(i),p5(i),p7(i)
,p9_3(i)]
end
%%%%% degree elevation of boundary.
```

```
% 1번 모서리 1234 -> 01234 estimate 하기
q0_ = Triangle_boundary(1,:)
q1_ = Triangle_boundary(2,:)
q2_ = Triangle_boundary(3,:)
q3_ = Triangle_boundary(4,:)
q_temp = [q0_; q1_; q2_; q3_;]
boundary(:,:,1) = cat(3,q_temp(:,1),q_temp(:,2),q_temp(:,3))
q0 = q0_
q1 = (q0_+q1_*3)/4
q2 = (q1_+q2_)/2
q3 = (3*q2_+q3_)/4
q4 = q3_
elevated_q_temp = [q0; q1; q2; q3; q4]
elevated_boundary(:,:,1) =
cat(3,elevated_q_temp(:,1),elevated_q_temp(:,2),elevated_q_temp(:,3))
% 2번 모서리 9864 -> 9864~
q0_ = Triangle_boundary(9,:)
q1_ = Triangle_boundary(8,:)
q2_ = Triangle_boundary(6,:)
q3_ = Triangle_boundary(4,:)
q_temp = [q0_; q1_; q2_; q3_;]
boundary(:,:,2) = cat(3,q_temp(:,1),q_temp(:,2),q_temp(:,3))
q0 = q0_
q1 = (q0_+q1_*3)/4
q2 = (q1_+q2_)/2
q3 = (3*q2_+q3_)/4
q4 = q3_
elevated_q_temp = [q0; q1; q2; q3; q4]
elevated_boundary(:,:,2) =
cat(3,elevated_q_temp(:,1),elevated_q_temp(:,2),elevated_q_temp(:,3))
% 3번 모서리 1579 -> 1579~
q0_ = Triangle_boundary(1,:)
q1_ = Triangle_boundary(5,:)
q2_ = Triangle_boundary(7,:)
q3_ = Triangle_boundary(9,:)
q_temp = [q0_; q1_; q2_; q3_;]
boundary(:,:,3) = cat(3,q_temp(:,1),q_temp(:,2),q_temp(:,3))
q0 = q0_
```

```
q1 = (q0_+q1_*3)/4
q2 = (q1_+q2_)/2
q3 = (3*q2_+q3_)/4
q4 = q3_
elevated_q_temp = [q0; q1; q2; q3; q4]
elevated_boundary(:,:,3) =
cat(3,elevated_q_temp(:,1),elevated_q_temp(:,2),elevated_q_temp(:,3))
end


function [elevated_tangent_ribbon, tangent_ribbon] =
triangleRectangleRibbon( estimated_triangle,Rectangle_patch ,index )

%TRIANGLERECTANGLERIBBON 이 함수의 요약 설명 위치
% 자세한 설명 위치
% input estimated_triangle 은 외부 boundary 9개를 estimated한 점. 12개임
% 모서리 3개 저장된 data
% estimated 된 삼각형 patch
% output은 estimated 된 삼각형 patch
% 4 d h 4
% 3 c g 6
% 2 b f 8
% 1 a e 9
% i j k l
% 1 5 7 9
% -> [a b c d;e f g h;i j k l]
% 1번 모서리 -> 1행, 2번 모서리 -> 2행, 3번 모서리 -> 3행
% input Rectanle_patch는 (m,n,d,t) m은 행의수 n은 열의수 d는 x,y,z 좌표의
지표, t는 patch의 number.
% output은 이웃한 사각형과 삼각형으로 이루어진 tangent ribbon
% index -> ( N x 2 ) matix ( i = 1...N, N 은 사각 patch의 갯수)
% index(i,:) = [j,k] -> i번째 사각 patch와 삼각형사이에서 ~ j,k 는 각각 공
통 boundary curve에서 삼각형, 사각형의 edge number
% index(1,:) = [1,4] -> 1번째 사각 patch와 공통 경계 곡선의 모서리 숫자
는 삼각형 모서리 1번, 사각현 모서리
% 4번-> 사각형 모서리 숫자는 아래에 설명. 4번일 경우 4행의 점들
%
% % % % % % % % % % 삼각형 모서리 숫자. edge number
% 젤 왼쪽 밑에 꼭짓점을 1번이라하면
```

```
% 2
% 1 3 -> 이런식으로 12 가 1번 모서리, 23 이 2번 모서리 13이 3번 모서리
%
% % % % % % % % % % % % % % % 사각형 모서리 숫자 .edge number
% 이웃한 사각 PATCH와 공통 BOUNDARY가 4행일 경우 index = 4
% 이웃한 사각 PATCH와 공통 BOUNDARY가 1행일 경우 index = 1
% 즉, 이웃한 사각 PATCH와 공통 BOUNDARY가 i행일 경우 index = i
% 이웃한 사각 PATCH와 공통 BOUNDARY가 i열일 경우 index = 4 + i
[np,k] = size(index)
for i=1:np
triangle_edge_number(i) = index(i,1)
boundary_rectangle_edge_number = index(i,2)
rectangle_edge_number_ = 1
if(boundary_rectangle_edge_number == 4 ||
boundary_rectangle_edge_number == 8)
rectangle_edge_number_ = boundary_rectangle_edge_number - 1
elseif(boundary_rectangle_edge_number == 1
||boundary_rectangle_edge_number == 5)
rectangle_edge_number_ = boundary_rectangle_edge_number + 1
end
if(boundary_rectangle_edge_number < 5)
Midpatch(:,:,i) = Rectangle_patch(boundary_rectangle_edge_number,:,:,i)
Sidepatch(:,:,i) = Rectangle_patch(rectangle_edge_number_,:,:,i)
else
Midpatch(:,:,i) = Rectangle_patch(:,boundary_rectangle_edge_number,:,i)
Sidepatch(:,:,i) = Rectangle_patch(:,rectangle_edge_number_,:,i)
end
% 삼각patch-> 1번 모서리는 왼쪽 patch, 2번 모서리는 오른쪽 patch
tripatch(:,:,i) = estimated_triangle(triangle_edge_number(i),:,:)
for j=1:3
if ( i==1 )
tangent_ribbon(:,:,j,i) = [tripatch(:,j,i)';Midpatch(:,j,i)';Sidepatch(:,j,i)']'
end
if (i == 2)
tangent_ribbon(:,:,j,i) = [Sidepatch(:,j,i)';Midpatch(:,j,i)';tripatch(:,j,i)']'
end
end
q = tangent_ribbon(:,2,:,i);
```

```
q0_ = q(4,:);
q1_ = q(3,:);
q2_ = q(2,:);
q3_ = q(1,:);
q0 = q0_;
q1 = (q0_+q1_*3)/4;
q2 = (q1_+q2_)/2;
q3 = (3*q2_+q3_)/4;
q4 = q3_;
elevated_q = [q4; q3; q2; q1; q0];
elevated_q = cat(3,elevated_q(:,1),elevated_q(:,2),elevated_q(:,3))
elevated_q(:,:,:)
elevated_Midpatch(:,:,i) =
[elevated_q(:,:,1)';elevated_q(:,:,2)';elevated_q(:,:,3)']'
for j=1:3
elevated_Sidepatch(:,j,i) =
[Sidepatch(1,j,i),Sidepatch(2,j,i),0,Sidepatch(3,j,i),Sidepatch(4,j,i)]
elevated_tripatch(:,j,i) =
[tripatch(1,j,i),tripatch(2,j,i),0,tripatch(3,j,i),tripatch(4,j,i)]
end
for j=1:3
if ( i==1 )
elevated_tangent_ribbon(:,:,j,i) =
[elevated_tripatch(:,j,i)';elevated_Midpatch(:,j,i)';elevated_Sidepatch(:,j,i)']
'
elseif (i == 2)
elevated_tangent_ribbon(:,:,j,i) =
[elevated_Sidepatch(:,j,i)';elevated_Midpatch(:,j,i)';elevated_tripatch(:,j,i)']
'
end
end
end
end
```

```
function [ Q,b,x ] = triangularpatchinterp( tri_control )
%TRIANGULARPATCHINTERP 이 함수의 요약 설명 위치
% 자세한 설명 위치
% % tri_control(:,:.:,:) -> [2 x 5 x 3 x 3] 2 row x 5 colum , 3 -> x,y,z좌
표 , 3 -> 모서리 순서
% 1번 모서리 b004 b013 b022 b031 b040
%             b103 b112    0    b121 b130
%
% 2번 모서리 b400 b310 b220 b130 b040
%             b301 b211    0    b121 b031
%
% 3번 모서리 b004 b103 b202 b301 b400
%             b013 b112    0    b211 b310
%
% b040                                  1
% b031 b130                        2 3
% b022 b121 b220                   4 5 6
% b013 b112 b211 b310              7 8 9 10
% b004 b103 b202 b301 b400         11 12 13 14 15
%
%
% b040                             b040
% b031 b103                 b031 b130
% b022 *b121               *b121 b220
% b013 *b112               *b211 b310
% b004 b103                 b301 b400
%
%        b013 *b112 *b211 b310
%        b004 b103 b202 b301 b400
%
% *들은 각 쌍이서 서로 blending 되어서 5,8,9 control point가 된다?.
%
for np=1:3
for i=1:3
tri_control2(:,:,i,np) = tri_control(:,:,i,np)'
end
end
same_or_not(1,:) = tri_control2(1,5,:,1)
```

```
same_or_not(2,:) = tri_control2(1,5,:,2)
same_or_not2(1,:) = tri_control2(1,1,:,1)
same_or_not2(2,:) = tri_control2(1,1,:,3)
same_or_not3(1,:) = tri_control2(1,1,:,2)
same_or_not3(2,:) = tri_control2(1,5,:,3)
b(1,:) = tri_control2(1,5,:,1) % b040
b(2,:) = tri_control2(1,4,:,1) % b031
b(3,:) = tri_control2(1,4,:,2) % b130
b(4,:) = tri_control2(1,3,:,1) % b022
b(5,:) = 1/2 * tri_control2(2,4,:,2) + 1/2 * tri_control2(2,4,:,1) % b121
b(6,:) = tri_control2(1,3,:,2) % b220
b(7,:) = tri_control2(1,2,:,1) % b013
b(8,:) = 1/2* tri_control2(2,2,:,3) + 1/2 * tri_control2(2,2,:,2) % b112
b(9,:) = 1/2 * tri_control2(2,4,:,3) + 1/2 * tri_control2(2,2,:,2) % b211
b(10,:) = tri_control2(1,2,:,2) % b310
b(11,:) = tri_control2(1,1,:,1) % b004
b(12,:) = tri_control2(1,2,:,3) % b103
b(13,:) = tri_control2(1,3,:,3) % b202
b(14,:) = tri_control2(1,4,:,3) % b301
b(15,:) = tri_control2(1,5,:,3) % b400
k = 1;
n = 400;
for i = 1:n*2+1
for j = 1:n+1
Q(i,j,1) = b(11,1);
Q(i,j,2) = b(11,2);
Q(i,j,3) = b(11,3);
end
end
index = 0;
for i = 1:n+1
for l = 1:n+2-i
w = (i-1)/n;
v = (l-1)/n;
u = 1 - w - v;
if(u==0 & v==0)
Q(i,l,:) = b(11,:)
end
```

```
if(u==0 & w==0)
Q(i,l,:) = b(15,:)
end
if(v==0 & w==0)
Q(i,l,:) = b(1,:)
end
b(8,:) = u/(u+v) * tri_control2(2,2,:,3) + v/(u+v) * tri_control2(2,2,:,1);
b(9,:) = w/(w+v) * tri_control2(2,4,:,3) + v/(w+v) * tri_control2(2,2,:,2);
b(5,:) = u/(u+w) * tri_control2(2,4,:,2) + w/(u+w) * tri_control2(2,4,:,1);
Q(i,l,:) = b(1,:) * v^4 +b(2,:)*4*v^3*w + b(3,:)*4*u*v^3 +
b(4,:)*6*v^2*w^2 + b(5,:)*12*u*v^2*w +
b(6,:)*6*u^2*v^2+b(7,:)*4*v*w^3
+b(8,:)*12*u*v*w^2+b(9,:)*12*u^2*v*w+b(10,:)*4*u^3*v + b(11,:)*w^4
+ b(12,:)*4*u*w^3+b(13,:)*6*u^2*w^2 + b(14,:)*4*u^3*w + b(15,:)*u^4;
index = index + 1;
x(index,:) = Q(i,l,:);
end
end
end
```

Abstract

Ship Hull Surface Modeling using Agnostic G^1 Gregory Surfaces

Dohwan Kim
Department of Naval Architecture and Ocean Engineering
The Graduate School
Seoul National University

The necessity of ship hull surface satisfying the G^1 continuous condition has been steadily increasing in order to reduce the difference of the design results between ship lines designers and worker using ship lines. Due to the difference between the ship lines design result and the modeled hull surface, the accurate performance evaluation of the structure / resistance and propulsion performance is not made, and it affects the reduction of the automation rate of the production work such as the expansion of curved plate.

Various studies have been carried out for the generation of the hull surface satisfying the continuous condition, but no algorithms have been developed that can be applied to various triangular / rectangular surface boundary conditions generated by complex ship lines which are due to the characteristics of the ship's geometry.

In this paper, we apply agnostic G^1 Gregory surface generation algorithm to ship and verify it. The algorithm is an improved surface generation algorithm that uses the existing Gregory surface and satisfies the G^1 continuity condition between curved surfaces in the area where triangular / rectangular surfaces are mixed.

We produce ship hull surface by applying agnostic G^1 Gregory surface generation algorithm to ship lines. And then verified that the G^1 continuity condition is satisfied by measuring the angle between the normal vectors of surfaces on the boundary of them.

Keywords : G^1 condition, Agnostic G^1 Gregory surfaces, Ship hull surface, Ship hull surface modeling

Student number : 2015-22861