

옷 주름의 비사실적 실시간 표현*

권지용^o, 이은정[†], 노창환[‡] 이인권[§]

연세대학교 컴퓨터과학과^{o†‡§}

mage@cs.yonsei.ac.kr^o, angui@cs.yonsei.ac.kr[†], milleve@cs.yonsei.ac.kr[‡], iklee@yonsei.ac.kr[§]

Non-Photorealistic Rendering for Wrinkles of Clothes

Ji-yong Kwon^o, Eun-Jung Lee[†], Chang-Hwan Roe[‡], In-Kwon Lee[§]

Dept. of Computer Science, Yonsei University^{o†‡§}

요약

우리는 3차원 개체로 표현된 옷의 주름을 라플라시안 맵과 옷 주름 맵을 사용하여 실시간에 비사실적으로 표현하는 방법을 제안한다. 옷의 주름을 표현하기 위해, 옷의 주름을 디자이너가 직접 그린 옷 주름 맵과 옷이 변형된 상태에 대한 정보가 계산된 라플라시안 맵을 사용하여, 단순한 GPU 연산을 통해 최종적으로 옷의 주름을 표현한다. 실험 결과 우리가 제안한 방법은 옷의 주름을 실시간에 비사실적으로 표현할 수 있으며, 디자이너의 의도한 바대로 주름을 그릴 수 있다는 장점이 있음을 알 수 있었다.

1. 서론

현재 컴퓨터 그래픽스의 기술이 발전하면서 기존의 컴퓨터 그래픽스 기술의 화두였던 사실적 표현 기법에 대한 연구 이외에도 비사실적인 표현 기법에 대한 연구가 활발하게 진행되고 있다. 이는 비사실적 표현 기법이 사실적 표현 기법이 가질 수 없는 몇 가지 장점 때문인데, 이러한 비사실적 표현 기법이 가진 장점을 응용하는 대표적인 분야 중 하나로 3차원 개체의 만화적인 표현이다.

3차원 개체의 만화적인 표현을 위해 여러 가지 방법들이 제시되었다. 만화적 표현 기법 연구는 만화적 표현을 위한 실루엣 검출 기법, 만화적 셰이딩 기법, 그리고 예술적인 표현을 위한 스타일라이징 기법 등 다양한 측면에서 연구가 되었다. 특히 Gooch 등은 그의 저서에서 다양한 비사실적 표현 기법에 대해 정리하였다[2].

그러나 아직까지 옷의 주름을 효과적으로 비사실적인 표현을 하는 방법에 대한 연구는 나와 있지 않다. 다만 천의 움직임이나 이미 천의 움직임을 시뮬레이션하는 여러 가지 기법이 소개되어 있다. 특히 Choi[3]이 2002년에 발표한 연구 결과는 비교적 적은 연산량을 소모하면서도 매우 사실적인 천의 움직임을 보여준다. 그러나 이러한 천의 움직임을 시뮬레이션하는 기법들은 물리적인 법칙에 기반한 방법이며, 실제 만화적인 표현을 위해 이러한 시뮬레이션은 그다지 적합하지 않고 비용 또한 크다.

Redman은 캐릭터를 그려내는 데에 있어서 옷의 주름과 갈라진 부분을 그리는 기법에 대해 언급하고 있다[4]. 그림 1은 Redman이 제시한 옷 주름을 손으로 그리는 데 생각해 두어야 할 옷 주름의 흐름을 나타낸 것이다. 이러한 점에서 알 수 있듯이, 우리가 만화 작품을 보면서 기대하는 옷 주름은



그림 1: Redman의 옷 주름 흐름도.

사실적인 물리 법칙에 기반한 정교한 옷 주름이 아닌 단지 변형이 생긴 부분에 대해 주름의 존재 여부 정도만을 판가름할 수 있는 수준의 옷 주름이다.

이에 우리는 옷 주름의 만화적인 표현을 위한 실시간 기법을 제안한다. 우리가 제안하는 옷 주름의 생성 기법은 먼저 실제 디자이너가 3차원 개체의 맵핑 좌표에 맞춰 그려낸 옷 주름 맵이 필요하며, 이 옷 주름 맵과 현재 3차원 개체의 상태를 입력받아 비사실적인 옷 주름을 GPU 연산을 통해 빠르게 생성해내게 된다.

본 논문의 순서는 다음과 같다. 2절에서는 제안하는 옷 주름 기법의 전체적 흐름을 설명하고 알고리즘의 각 부분을 자세히 설명하겠다. 3절에서는 제안한 기법을 통해 생성해낸

*본 연구는 한국과학재단 특정기초연구 (R01-2004-000-10117-0 (2004))의 지원으로 수행되었음.

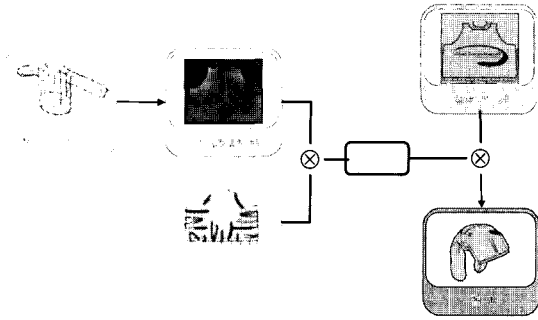


그림 2: 제안하는 알고리즘의 전체적 흐름도.

결과 화면을 보고 4절에서 결론을 맺겠다.

2. 옷주름 표현 기법

2.1 전체적인 알고리즘

그림 2는 우리가 제시하는 알고리즘의 전체적인 흐름을 보여주고 있다. 먼저 전처리 단계에서는 3차원 개체로 표현된 옷의 주름 없는 상태에서의 라플라시안 벡터의 크기를 계산한다. 라플라시안 벡터의 이론은 추후 설명하겠다. 그리고 유저가 3차원 개체의 맵핑 좌표에 맞추어 옷 주름을 직접 그린다. 여기서 바탕은 흰색, 옷 주름은 검은색으로 그려낸다. 이렇게 직접 그려낸 옷 주름 맵은 후에 보다 자연스러운 옷 주름 생성을 위해 약간의 가우시안 블러 처리를 해준다.

실시간 처리 단계에서는 다음과 같은 처리를 행한다. 먼저 현재 3차원 개체의 상태에서 라플라시안 벡터의 크기를 계산하여 미리 계산해둔 라플라시안 벡터와의 차이를 구하여 라플라시안 맵을 생성한다. 이 라플라시안 맵을 통해 우리는 현재 옷의 변형 정도를 알 수 있다. 라플라시안 맵과 옷 주름 맵의 값을 곱하여 현재 주름이 그려져야 할 영역을 선별하고 이를 톤 셰이딩 효과를 주기 위해 양자화(Quantization)한다. 최종적으로 3차원 개체의 원래 텍스처 맵과 곱하여 옷 주름이 입혀진 새로운 맵을 생성한다.

2.2 라플라시안 벡터와 라플라시안 맵

3차원 개체에서 한 정점에 대한 라플라시안 벡터는 해당 정점과 그 정점과 연결되어 있는 다른 모든 정점들의 중

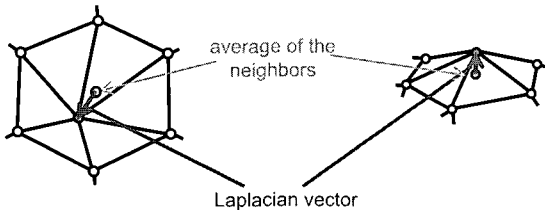


그림 3: 라플라시안 벡터의 계산.

심점(Center Of Mass)간의 차이로 정의되며, 자세한 설명은 [5]에 소개되어 있다. 그림 3은 임의의 정점에 대하여 해당 정점과 연결되어 있는 다른 모든 정점을 통해 라플라시안 벡터를 계산하는 과정을 설명하고 있다. 식으로 표현하면 다음과 같다.

$$L(v_i) = v_i - \frac{1}{d_i} \sum_{j \in N(i)} v_j \quad (1)$$

여기서 $L(v_i)$ 는 점 v_i 에 대한 라플라시안 벡터, v_i 는 해당 정점, $N(i)$ 는 해당 정점의 이웃점의 인덱스의 집합, d_i 는 이웃점의 개수를 의미한다. 식과 그림을 통하여 알 수 있듯이, 우리는 라플라시안 벡터를 통하여 해당 정점이 가진 곡률의 방향과 크기를 개략적으로 계산할 수 있다.

우리는 3차원 개체로 표현된 옷이 변형되었을 때, 변형이 심하게 된 점들을 찾아내기 위해 라플라시안 벡터를 이용하였다. 먼저 3차원 개체의 변형 전 상태에서 모든 정점에 대한 라플라시안 벡터의 크기의 제곱 $|L_0(v_i)|^2$ 를 미리 계산하여 저장해둔다. 다음 실시간 처리 단계에서 모든 정점에 대한 현재 상태에서의 라플라시안 벡터의 크기의 제곱을 계산하여 원상태 값과의 절대차 $LC(v_i)$ 를 계산한다.

$$LC(v_i) = ||L(v_i)|^2 - |L_0(v_i)|^2 \times c \quad (2)$$

여기서 상수 c 는 3차원 개체의 전체적인 크기에 따른 값의 차이가 나는 현상을 방지하기 위한 비례 상수이다. 위 값을 0과 1 사이의 값으로 제한한 뒤, 각 정점의 색상 형태로 대입하여 텍스처 형태로 렌더링하여 라플라시안 맵을 생성한다. 따라서 생성한 라플라시안 맵의 값이 크면 3차원 개체의 해당 부분에 대해 변형이 심하게 되었다는 것을 의미한다.

2.3 라플라시안 맵과 옷 주름 맵을 이용한 옷 주름 생성

앞에서 설명한 라플라시안 맵의 값은 해당 부분이 변형된 정도를 의미하므로, 우리는 라플라시안 맵의 값이 큰 부분에 미리 준비해둔 옷 주름을 그려 넣고자 한다. 이러한 처리는 단순하게 라플라시안 맵의 값 L_{map} 과 1.0에서 옷 주름 맵 값 W_{map} 을 뺀 값 간의 곱을 통해 해결할 수 있다. 식으로 표현하면 다음과 같다.

$$W_{now} = 1.0 - (L_{map} \times (1.0 - W_{map})) \quad (3)$$

위 식을 통해 구해진 W_{now} 값을 그대로 적용하면 너무 부드러워서 만화적인 표현에 적합하지 않으므로, 정해진 값으로 양자화한다. 양자화 방법은 기준값 a, b 를 정하여 해당 값이 0과 a 사이에 있을 때는 a 값으로, a 와 b 사이에 있을 때는 b 값으로, b 와 1 사이에 있을 때는 1값을 사용하는 단순한 방법을 적용하였다. 최종적으로 3차원 개체의 원래 텍스처 값 T_{map} 과 곱하여 최종 색상을 얻어낸다.

$$Result = W_{quantized} \times T_{map} \quad (4)$$

3. 실험 결과

실험을 위해 우리는 정점 326개, 삼각형 504개로 이루어진 저용량 상의 모델을 제작하였다. 모델의 제작과 애니메이션 제작에 3D Studio MAX 소프트웨어를 사용하였다. 그림 4는 우리가 실험에 사용한 3차원 개체 모델의 텍스처와 기본 상태 모습을 나타낸 것이다. 이 모델의 실제 처리를 위해 공개 애니메이션 라이브러리인 Cal3D[6]를 사용하였다. 또한 실제적인 렌더링 처리에 필요한 다중 텍스처 처리와 텍스처 맵으로의 렌더링 처리를 위하여 OpenGL에서 제공하는 저수준 웨이딩 언어를 사용하였다. 만화적 외곽선 표현 기법을 위해서 Raskar 등이 정리한 Image-Precision Algorithm 중에서 Wire-frame을 이용한 방법을 적용하였다[7].

그림 5는 우리가 제안한 기법을 통해 생성한 결과를 보여 주고 있다. (a)는 사용한 옷 주름 맵이며, 이를 적용하여 옷이 변형되었을 때의 결과가 각각 (b), (c), (d)에 제시되어 있다. (b)는 팔꿈치와 어깨부분, (c)는 복부 부분, (d)는 등 부분에 각각 옷 주름이 표현된 것을 볼 수 있다. 그림 6은 같은 변형에 대하여 옷 주름 맵을 달리 그렸을 경우의 모습을 보여 주고 있다. 실험을 통해서 알 수 있듯이, 우리가 제시한 방법은 변형이 심한 부분을 찾아내어 사용자가 원하는 형태의 주름을 자동적으로 생성하여 줄 수 있다.

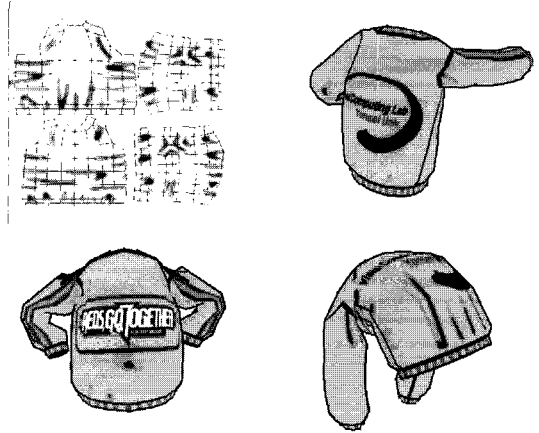


그림 5: 옷 주름 맵을 적용한 예.

4. 결론 및 향후 과제

우리는 옷 주름 맵과 라플라시안 맵을 사용하여 실시간에 비 사실적인 옷 주름을 표현하는 방법을 제시하였다. 제시한 방법은 실험 결과 실시간에 수행이 가능하면서 사용자가 원하는 주름을 옷의 변형에 맞추어 생성할 수 있었다.

그러나 우리가 제시한 방법에는 몇 가지 취약점이 있다. 먼저 비디오 카드가 다중 텍스처 맵핑 기법을 지원해야 한다. 또한 라플라시안 맵을 생성하는 데 텍스처 맵에 직접 렌더링 하기 때문에 추가적인 렌더링 타임이 소모되고, 또한 텍스처 맵의 저장에 위한 비디오 메모리가 소모된다. 또한 각 정점 당 라플라시안 맵을 계산하는데 들어가는 비용이 추가적으로 소모된다. 이러한 제약점을 해결하는 것이 추후 연구 방향이 될 것이다. 특히 각 정점 당 행해야 하는 라플라시안 벡터의 크기 계산을 정점 셰이더를 이용하여 해결하거나 라플라시안 벡터의 크기값을 대신할 수 있는 다른 값을 정점 셰이더를 이용해서 계산할 수 있다면 연산 속도 향상에 큰 도

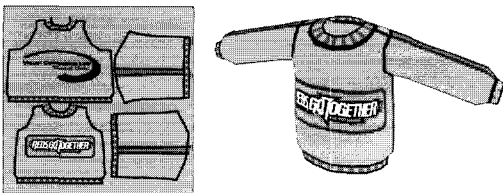


그림 4: 원본 텍스처 맵과 옷의 3차원 모델.

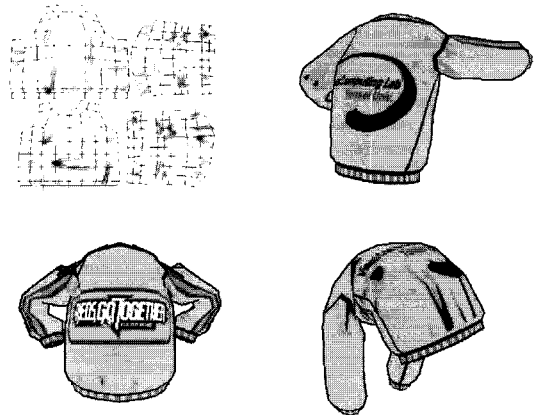


그림 6: 옷 주름 맵을 적용한 다른 예.

음이 될 것이다.

또한 제안한 방법에서 3차원 개체의 전체적인 크기 비율이 라플라시안 벡터 계산에 영향을 주기 때문에 이러한 부분에서 수식의 최적화가 필요하다. 그리고 현재 라플라시안 맵을 계산하는 데 사용되는 비례 상수를 최적화 하는 수식을 고려중이다. 마지막으로 옷의 주름을 표현할 옷 주름 맵을 디자이너의 손을 거치지 않고 옷의 움직임을 통해 자동적으로 생성해 내는 방법도 향후 고려할 과제이다.

참고 문헌

- [1] L. Itti, C. Koch and E. Niebur, A Model of Saliency-Based Visual Attention for Rapid Scene Analysis, IEEE Transactions on Pattern Analysis and Machine Intelligence, 20(11):1254-1259, 1998.
- [2] Bruce Gooch and Amy Gooch, Non-Photorealistic Rendering, A K Peters, 2001.
- [3] Kwang-Jin Choi and Hyeong-Seok Ko, Stable but Responsive Cloth, ACM Transactions on Graphics (SIGGRAPH 2002), 21(3):604-611, 2002.
- [4] Lenn Redman, How To Draw Caricatures, Contemporary Books, 1984.
- [5] Yaron Lipman, Olga Sorkine, Daniel Cohen-Or, David Levin, Christian R?ssl and Hans-Peter Seidel, Differential Coordinates for Interactive Mesh Editing, Proceedings of Shape Modeling International, IEEE Computer Society Press, 181-190, 2004.
- [6] Cal3D Library, <https://gna.org/projects/cal3d>
- [7] Ramesh Raskar and Michael Cohen, Image Precision Silhouette Edges, Symposium on Interactive 3D Graphics 1999, 1999.