입체영상변환을 위한 효율적인 줌 움직임 검출

박일권[O] 변혜란
연세대학교 컴퓨터과학과
{ikheart[O], hrbyun}@yonsei.ac.kr

Efficient Zooming Motion Detection for Stereoscopic Conversion

Ilkwon Park[O] , Hyeran Byun
Department of Computer Science, Yonsei University

요 약

본 논문에서는 단일 비디오 내에서 다양한 움직임 유형을 분석하여 각각의 움직임 유형에 따른 효율적인 입체 영상 변환 알고리즘을 제안한다. 움직임 정보는 단일 비디오 영상을 입체영상으로 변환하기 위해 가장 많이 사용되는 정보 중 하나이며 특히, 줌 인(zoom in), 줌 아웃 (zoom out) 움직임 발생시 상대적으로 높은 입체감을 제공한다. 따라서 본 논문은 수평, 수직 모션뿐만 아니라 줌 움직임(zooming motion)을 효율적으로 검출하기 위한 새로운 모션 결정 규칙을 제안 하였다. 실험에서는 MPEG 실험에 많이 사용되는 동영상 데이터를 이용하였으며 제안된 모션 결정 규칙은 순수한 줌 움직임뿐만 아니라 노이즈가 포함된 복잡한 줌 움직임에도 강인한 결과를 보였다.

1. 서 론

입체영상 제작은 일반적으로 입체영상 카메라를 통해 직접 생성하는 방법과 기존의 2차원 비디오 영상에 대한 3차원 입체 영상 변환알고리즘을 통해 생성하는 방법으로 나눌 수 있다. 2차원 비디오 영상에 대한 3차원 입체 영상 변환은 기존에 만들어진 비디오 컨텐츠를 활용할 수 있다는 점에서 오랫동안 많은 관심을 받아 왔다[1][2].

입체 영상 변환을 위해서는 단일 비디오 영상에서 각각 다른 시차정보를 갖는 좌측영상과 우측영상을 생성하게 된다. 단일 비디오 영상에서 시차정보를 추출하기 위해 대표적으로 특징 점들의 움직임 정보와 소실점을 이용한 원근 정보 등이 사용된다. 특히, 특징 점들의 움직임 정보는 간단하면서도 강력한 시차정보 추출 방법이다.

움직임 정보를 추출하기 위해서 광류(Optical flow)를 이용한 방법 또는 KLT추적 (Lucas-Kanade-Tomasi Tracker) 알고리즘 등이 사용된다[3]. 광류를 이용한 방법은 각각의 블록단위로 움직임 정보를 추출하기 때문에 영상 전체적인 움직임을 파악하기 쉬울 뿐만 아니라 알고리즘이 간단하여 구현하기 쉽다. 특히 MPEG 비디오 파일인 경우 비디오 데이터 안에 광류를 이용한 움직임 정보가 내재되어 있다.

그럼에도 불구하고 노이즈에 민감하여 정확한 움직임 정보를 추출하는 데는 어려움이 많다. 한편, KLT 추적 알고리즘은 광류를 이용한 움직임 정보 추출보다 상대적으로 노이즈에 강인하며 정교한 움직임 정보를 추출할 수 있다. 본 논문에서는 움직임 추출을 위해서 KLT 추적 알고리즘을 이용하였다.

한편, 입체 영상 변환에 사용된 움직임 정보는 움직임 방향과 이동 량에 따라 각각 다르게 적용하였다.

일반적으로 수평 움직임은 입체변환에 적용하기 용이하며 수직움직임은 눈의 피로를 가중시키는 경향이 있다. 더 나아가 줌 움직임은 독특한 움직임 정보를 가지고 있으며 입체 변환에 의해 관측자에게 높은 입체감을 제공해 주는 움직임이다.

따라서, 본 논문은 수평, 수직, 줌 인(Zoom in), 줌 아웃(Zoom out), 정지 상태로 움직임을 구분하였으며 각각에 대한 움직임 검출 규칙을 제안하였다. 특히, 높은 입체감을 제공하는 줌 움직임에 효율적인 검출 방법을 제안하였다. 그림 1은 제안한 움직임 검출 알고리즘을 이용한 입체영상 변환 시스템을 나타낸다.

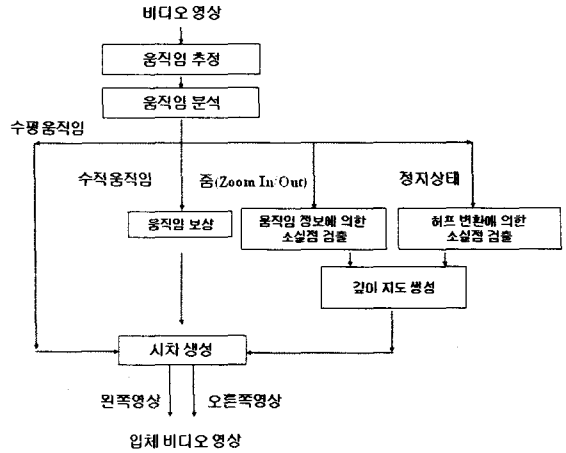

그림 1. 2차원 비디오 영상에 대한 3차원 입체영상 변환 시스템의 흐름도

2. 움직임 분석

입체 비디오 변환을 위한 대한 다양한 특징들 중에서 움직임 정보는 매우 유용한 정보이다. 2장에서는 움직임 추정 방법 과 제안한 결정 규칙을 이용한 움직임 분석방법을 기술한다.

2.1 움직임 추정

움직임 정보를 추정하기 위한 다양한 방법들인 보고되었으며 각각의 방법마다 장단점을 가지고 있다.[3],[4]. 대표적인 방법으로 광류를 이용한 방법 과 KLT 추적 알고리즘을 이용한 방법이 있다. 본 논문에서는 상당히 신뢰도가 높고 광류보다 노이즈에 덜 민감한 KLT 추적알고리즘을 이용하여 움직임 정보를 추정하였다.

추정된 움직임 정보는 각각의 화소 좌표를 기준으로 방향(orientation)와 크기(magnitude)로 표현되며 수식1 과 수식 2에 의해서 계산 되어 진다. 움직임 벡터 *MV*는 수평성분인 *MVx* 와 수직성분이 *MVy*로 구성된다.

$$\text{방향성분}\ (\theta) = \tan^{-1}\left(\frac{MV_y}{MV_x}\right) \qquad (1)$$

$$\text{크기}(M) = \left|\sqrt{MV_x^{\,2} + MV_y^{\,2}}\right| \qquad (2)$$

2.2 움직임 분석

추정된 움직임 정보는 움직임 분석을 통하여 움직임 유형을 판단하게 된다. 본 논문에서는 5가지 움직임 유형을 분류하며 각각은 수평 움직임, 수직 움직임, 줌 인(zoom in), 줌 아웃(zoom out), 정지(static) 상태로 구분한다.

움직임 분석을 위한 2가지 특징 추출 방법을 제안하였다. 첫 번째 특징은 그림 2와 같이 현재 프레임 내에서 각각의 움직임 벡터의 방향 성분을 8개의 방향은 나누어 누적 히스토그램을 작성하게 된다. 따라서, 누적 히스토그램을 통해서 현 프레임의 주요한 움직임 방향을 추출하게 된다.

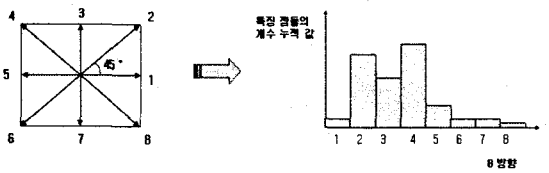

그림 2. 히스토그램을 이용한 8방향 성분의 누적

두 번째 특징은 입력 영상에서 움직임 벡터가 발생한 부분만을 나타내는 움직임 벡터 평면을 형성하고 움직임 벡터의 무게중심을 기준으로 4개의 영역으로 나누게 된다. 이때 각각의 영역에 대한 평균 벡터의 방향 성분을 특징 값으로 추출하게 된다. 그림3은 각각의 움직임 벡터 평면 위에 분할된 각 영역의 움직임 벡터들을 나타낸 것이다. CP는 움직임 벡터 평면상의 움직임 벡터의 무게 중심을 나타내며 R1~R4는 분할된 각각의 영역을 나타낸다.

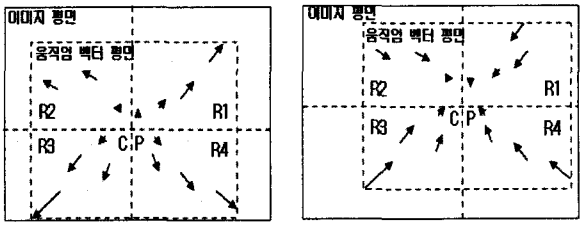

그림 3. 움직임 벡터의 무게 중심을 기준으로 나뉜 4개의 영역에 존재하는 움직임 벡터

2.3 결정 규칙

각각의 움직임 유형은 움직임 분석과정에서 추출한 2개의 특징 값을 기반으로 새롭게 제안한 결정 규칙을 이용하여 분류하게 된다. 먼저 결정 규칙을 적용하기 위하여 표 1과 같이 4가지 조건을 정의한다.

표 1 결정 규칙을 위한 조건 정의

조 건	표 현	
C1	Dnum : 주요 방향 성분	
C2	- CP : 움직임 벡터 평면의 무게 중심 - $\overline{MV}_m$: (R_i) 영역 내의 평균 움직임 벡터 - R_i (MV_m)≠ 0	
	MVI : Zoom In Rt: 0 < MV_m<=180° Rb:180<MV_m<=360° Rl:90° <MV_m<=270° Rr:270 <MV_m<=90°	MVO : Zoom Out Rt : 0° <MV_m = 180° Rb:180° <MV_m<= 360° Rl :90° <MV_m = 270° Rr :270° <MV_m<= 90°
	MVT : $\neg(MVI \cup MVO)$	
C3	MMV : 평균 움직임 벡터 크기	
C4	VM : 평균 움직임 수직 벡터	

C1은 누적 히스토그램을 이용하여 8방향 성분중 주요 방향성분을 나타내며 일정 임계 값을 넘는 모든 방향의 개수를 나타낸다. 또한 동시에 발생될 수 있는 움직임 벡터의 방향성분의 개수를 나타낸다. 예를 들어, 수평 움직임과 수직 움직임은 2개 이하의 방향이 발생되며 반면에 줌 모션은 2개 이상 발생한다. C2 는 각 영역에 포함되어 있는 평균 움직임 벡터의 방향 값이다. 조건 C2는 줌 인과 줌 아웃을 구별하는데 주요한 조건이다. C3는 전체 움직임 벡터의 평균 벡터의 크기를 나타내며 정지 상태를 검출하는데 사용된다. 마지막으로 C4는 평균 벡터의 수직성분을 나타내며 입체영상 변환 시 수직 움직임은 눈의 피로를 가중시키기 때문에 수직이 아닌 다른 움직임으로 전환되어야만 한다. 본 논문에서는 수직성분의 크기를 동일한 크기의 수평 성분으로 변환하였다.

표 1에서 정의된 4가지 조건은 모션 유형을 결정하기 위하여 표 2와 같이 결정 규칙을 생성하는데 사용된다. 제안한 5가지의 모션 유형은 4가지 조건의 조합을 이용하

여 결정하게 된다.

표 2 움직임 유형을 위한 결정 규칙

움직임 유형	C1	C2	C3	C4
수평	0<Dnum≤2	MMT	MMV > θ_{MMV}	VM< θ_{VM}
수직	0<Dnum≤2	MVT	MMV > θ_{MMV}	VM≥ θ_{VM}
줌인	Dnum≥2	MVI	MMV > θ_{MMV}	-
줌아웃	Dnum≥2	MVO	MMV > θ_{MMV}	-
정지	Dnum = 0;	-	MMV≤ θ_{MMV}	-

3. 깊이 지도 생성

결정 규칙에 의한 수평과 수직 움직임 유형을 제외한 3가지 움직임 유형(줌인, 줌 아웃, 정지 상태)에 대한 시차 생성을 위해 깊이 지도를 생성이 선행되어야만 한다. 수평과 수직움직임 유형은 움직임 정보를 바로 시차정보로 변환하여 사용하기 때문에 시차 생성부분에서 다룬다. 단일 비디오 영상에서 깊이 지도는 전형적으로 소실점 또는 소실 선을 이용한 원근 정보를 이용하여 생성한다.

소실선과 소실점을 단일 영상 내에서 추출하는 방법은 움직임 유형에 따라 다르게 적용하였다. 정지 영상의 경우 소실선의 추출은 허프 변환(Hough transform)을 이용하여 획득하였으며 특별히 본 논문에서는 줌인 과 줌 아웃은 움직임 벡터를 이용하여 획득하도록 제안하였다[5][6].

움직임 정보가 없는 정지 상태인 경우 움직임 정보를 이용하여 시차정보를 생성 할 수 없다. 따라서 영상내의 원근정보를 이용하는 방법이 타당하다. 일반적으로 허프 변환을 이용하여 영상내의 직선 성분을 추출하게 되며 추출된 직선 성분 중 주요 직선 성분의 교차점을 이용하여 소실점을 생성하게 된다. 한편, 영상내의 불충분한 직선 성분으로 인하여 소실점이 생성되지 않거나 잘못 생성되는 경우가 많다. 따라서 소실점이 생성되지 않는 경우는 영상의 중심을 가상의 소실점으로 간주한다.

또 다른 움직임 유형인 줌인과 줌 아웃의 경우 4개로 분할된 각각의 영역에서의 평균 움직임 벡터는 소실선과 일치하게 된다. 따라서 이러한 소실선을 연장시켰을 경우 소실선끼리 교차하는 지점이 바로 소실점이 된다.

따라서, 이러한 원근 정보를 이용하여 생성된 깊이 지도는 소실선을 따라 256단계의 밝기 값으로 표현된다. 소실점에 가까울수록 어둡고 영상 평면의 가장자리에 가까울수록 밝은 값을 가지게 된다. 즉, 어두울수록 멀리 떨어져 있으며 가까울수록 밝은 값을 가지게 된다.

4. 시차 생성

입체 영상을 생성하기 위해서는 시차를 갖는 두 영상이 필요하다. 본 논문에서는 움직임 유형에 따라 시차 생성 방법을 다르게 적용하였으며 3장에서 생성한 깊이 지도를 이용한 방법과 수평과 수직 유형에 대해서는 연속된 프레임상에 발생하는 특징 점들의 움직임 차이를 이용하여 시차를 생성하였다.

두 방법 모두 다음과 같은 두 가지 가정 상태에서 적용된다. 첫째, 관측자의 눈으로부터 영상이 맺히는 스크린까지의 대략적인 거리를 알고 있다. 둘째, 최대 시차는 사람의 두 동공 사이의 거리보다 작다[2].

4.1 움직임 기반 시차 생성

연속된 비디오에서 시차의 양은 평균 움직임 량에 비례하여 생성되며 움직임 량을 바로 시차로 바꾸어 적용할 수 있다. 한편 수직 움직임을 곧바로 시차로 변환하여 이동시키면 사람의 눈에 부적합한 시차가 생성된다. 따라서 적절한 수평 시차로 변환하는 과정이 필요하다. 본 논문에서는 수직 시차를 수평시차로 전환하고 일정 임계값 이상 시차가 발생할 경우에는 최대 시차 값으로 적용한다. 그림4 는 움직임 기반 시차 생성 의 개념을 도식화 한 것이다.

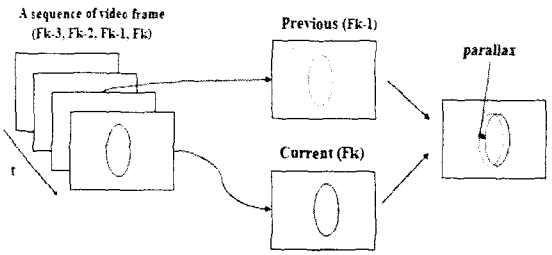

그림 4. 움직임 기반 시차 생성

4.2 깊이지도 기반 시차 생성

깊이 지도의 깊이 값은 수식3에 의해서 시차로 변환될 수 있다. 따라서 3장에서 구한 깊이 지도를 이용하여 특정 움직임 유형에 적절한 시차정보를 생성하게 된다.

수식 3에서 M은 최대 시차값을 나타내며 앞에 언급한 가정에 따라 두 동공 사이의 거리보다 작아야 한다. $Depth_value$는 깊이지도에서 각 화소가 나타내는 밝기 값을 나타낸다. 현재 프레임에서 각 화소는 parallax/2에 의해서 왼쪽영상을 위해 왼쪽으로 이동하고 오른쪽 영상 생성을 위해 parallax/2 만큼 오른쪽으로 이동하게 되며 시차를 적용하는 자세한 방법은 S. Battiato *et al.*[2]를 참조한다.

$$Parallax = M * \left(1 - \frac{depth\_value}{255}\right) \quad (3)$$

5. 실 험

본 논문을 위한 실험은 제한한 알고리즘의 성능을 평가하기 위하여 다양한 비디오 데이터를 사용하였다. 시스템 환경을 설정하기 위하여 먼저 관측자와 입체영상 상영이 가능한 LCD 모니터와의 거리는 대략 1.2m로 설정하였다. 표3은 제안한 알고리즘에 의해 검출된 움직임 유형의 결과를 나타낸 것이다.각 비디오에 대한 Ground truth

는 비디오 편집기를 이용하여 프레임별로 움직임 방향을 3명의 평균을 주관적으로 표기한 것이다. 대부분의 동영상은 순수한 줌 움직임만을 포함하지는 않는다. 다시 말해 다양한 움직임 유형이 섞인 상태에서 현저한 움직임을 가지고 있다. 이러한 복잡한 움직임 유형을 분석하는 일은 어려운 일이며 본 논문에서는 비교적 복잡성이 낮은 움직임 유형에 대해서만 실험하였다. 표3과 같이 상당히 정확히 움직임 유형을 파악할 수 있었다.

표 3. 제안된 시스템에 의해 검출된 움직임 유형

움직임 유형	Test Video					
	Highway		Tempete		Coastguard	
	Ground Truth	Detected frame	Ground Truth	Detected frame	Ground Truth	Detected frame
수평	0	93	0	31	287	276
수직	0	16	0	0	10	19
줌인	1998	1886	0	0	0	1
줌아웃	0	0	258	227	0	1
정지	0	3	0	0	0	0

6. 결 론

본 논문에서 제안한 결정 규칙은 비교적 복잡성이 낮은 움직임 동영상에서 강인한 움직임 유형 분석을 나타냈으며 특별히 카메라 줌 움직임을 상당히 정확히 분리해 냈다. 입체영상 변환에서 줌 움직임은 높은 입체감을 보장하기 때문에 제안한 움직임 분석을 통해서 움직임 유형에 따른 보다 세분화된 입체 영상 변환을 가능케 했다. 실험 결과는 입체영상 시스템에 움직임 유형을 결정하기 위한 신뢰할 만한 움직임 유형 검출 결과를 보였으며 앞으로 보다 복잡한 움직임이 섞여 있는 영상에 대해 움직임 유형을 결정할 수 있도록 본 알고리즘을 개선할 것이다.

감사의 글

본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT 연구센터 지원사업의 연구결과로 수행되었음 (IITA-2005-C1090-0502-0022)

참고 문헌

1. Lydia M. J. Meesters, Wijnand A. IJsselsteijn, and Pieter J. H. Seuntiëns, " A Survey of Perceptual Evaluations and Requirements of three-dimensional TV", IEEE Transctions on Circuits and systems for Video technology, Vol. 14, No. 3, March, 2004.
2. S. Battiato, A. Capra, S. Curti, and M. La Cascia, "3D Stereoscopic Image pairs by Depth-Map Generation", Proceeding of the 2nd International Symposium on 3D Data Processing, Visualization, and Transmission pp.124-131, 2004
3. Carlo Tomasi and Takeo Kanade. "Detection and Tracking of Point Features", Carnegie Mellon University Technical Report CMU-CS-91-132, April 1991.
4. Manbae Kim, Sanghoon Park, Haksoo Kim, and Ignatov Artem, "Automatic conversion of two-dimensional video into stereoscopic video", Proceeding of SPIE, Vol. 6016, Nov. 15, 2005.
5. Andrea Matessi and Luca Lombardi, "Vanishing Point Detection in the Hough Transform Space", Lecture Notes in Computer Science, 1685, pp. 987-994, 1999.
6. M. V. Srinivasan, S. Venkatesh, and R. Hosie, "Qualitative Estimation of Camera Motion Parameters from Video Sequences", Pattern Recognition Vol. 30. No. 4 pp 593-606, 1997.
7. Virginio Cantoni, Luca Lombardi, Marco Porta, and Nicolas Sicard, "Vanishing Point Detection: Representation Analysis and New Approaches", Proceeding of the 11th International Conference on Image Analysis and Processing, Sept. 26-28, 2001.