호모그래피 행렬과 센서 데이터를 활용한 동영상 스티칭 방법

*김민우 **임용철 ***김상균

명지대학교

*ciyciygood@gmail.com

Moving Picture Stitching Method Using Homography Matrix & Sensor Data

*Minwoo Kim **Yong-Chul Lim ***Sang-Kyun Kim

Myongji University

요약

본 논문은 동영상 스티칭의 속도·정확도를 향상시키기 위해 호모그래피 행렬 생성과 센서 데이터 활용을 통한 동영상 스티칭 방법을 제안한다. 본 논문에서는 임의의 호모그래피 행렬을 선형으로 생성하여 이미지를 스티칭 하는 방법을 설명하고, 이 과정에서 스티칭 정확도가 낮아지는 단점을 센서 데이터 활용을 통해 보완하는 방법을 소개한다. 1만 쌍의 모든 프레임에서 호모그래피 행렬을 생성 시키는 방법과 본 논문에 제안한 임의의 호모그래피 생성 방법을 비교하였을 때 평균 2.6초 걸리는 스티칭 시간을 약 1.5초 단축시켜 빠른 스티칭을 가능하게 하였다. 또한 선형 호모그래피 행렬만을 사용한 스티칭 한 결과보다 선형 호모그래피 행렬과 센서데이터를 함께 사용하였을 때의 정확도가 28.2% 개선되었음을 확인하였다.

1. 서론

파노라마 이미지란 영상처리를 이용하여 여러 이미지를 하나의 이미지로 정합한 넓은 시야각을 제공하는 고해상도 이미지를 말하며 전시회, 스트리뷰(Street-view) 등에 널리 사용되고 있다.[1]

파노라마 이미지를 생성하기 위해서는 이미지를 붙이는 작업인 이미지 스티칭(Stitching) 기술과 색상 보정을 위한 블렌딩(Blending) 기술을 수행 하여야 한다. 이미지 스티칭은 이미지 특징점 추출, 추출된 특징점에서 매칭에 필요한 참인 점(inlier)을 선별, 참인 점을 호모그래피(homography) 행렬로 변환, 호모그래피 행렬을 사용하여 이미지를 왜곡(warping), 왜곡된 이미지와 다른 이미지를 합하는 과정으로 이루어져 있다.[2,3]

이미지의 특징점을 추출하기 위한 방법으로 SIFT(Scale Invariant Feature Transform)[4]와 SURF(Speeded Up Robust Feature)[5] 기법이 많이 사용되고 있다. SIFT의 경우 비교적 많은 특징점을 추출하여 다른 기법들에 비해 속도가 낮지만 스케일, 회전 등의 변화에 강인한 기법이다. SURF의 경우 속도를 개선한 만큼 빠르지만, 조도를 제외한 스케일, 회전 등의 변화에는 SIFT 기법보다 강인하지 못하다.[6,7]

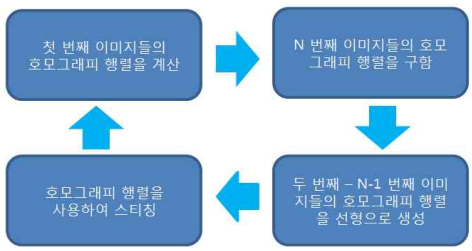
파노라마 동영상은 연속적인 파노라마 이미지의 집합이며 일반적으로 동영상의 프레임 비율(frame rate)은 29.97fps(frame per second)이다. 1초에 29.97개의 프레임을 사용하는 100초 분량의 동영상을 스티칭 하려면, 이미지 스티칭 작업을 2997번 수행하여야 한다. SIFT기기법을 이용할 경우, 시간이 너무 오래 걸리는 문제가 있고, SURF기법을 사용할 경우 시간을 줄일 순 있지만 스케일·회전의 강인함이 떨어질 수 있다.

이러한 문제점을 해결하기 위해서 본 논문에서는 파노라마 동영상을 생성하기 위하여 고속의 스티칭 기법을 제안한다. 동영상 스티칭 과정에서의 속도를 향상시키기 위해 선형 호모그래피 행렬을 생성하고, 정확도를 높이기 위해 센서데이터를 사용하였다.

본 논문의 구성은 다음과 같다. 2장에서는 속도를 향상시키기 위한 호모그래피 생성 방법을, 3장에서는 정확도를 높이기 위한 센서데이터를 사용하는 방법을 설명한다. 4장에서는 실험 방법 및 결과를 기술하고 5장에서는 결론을 내린다.

2. 호모그래피 행렬 생성 방법

본 논문에서 제안 하는 호모그래피 행렬 생성 방법이다.



[그림 1] 호모그래피 행렬 생성 방법 흐름도

이미지 스티칭 시, 특징점 추출과 호모그래피 행렬 추정 속도를 줄이기 위하여 첫 번째와 N 번째 프레임 이미지들의 특징점을 추출한 후 호모그래피 행렬을 구한다. N은 임의의 숫자이며 전체 프레임 이미지의 개수보다 작은 수이다.

$$\begin{bmatrix} h_1^1 h_2^1 h_3^1 \\ h_4^1 h_5^1 h_6^1 \\ h_7^1 h_8^1 h_9^1 \end{bmatrix} \quad H = \begin{bmatrix} h_1^N h_2^N h_3^N \\ h_4^N h_5^N h_6^N \\ h_7^N h_8^N h_9^N \end{bmatrix}$$

구해진 호모그래피 행렬 과 H 의 원소 $_1, h_2, h_3, \cdots, h_9$

를 모두 N 등분하여 , H_3, …, H $_{-1}$의 원소들을 구한다.

$$= h^1 + (h \ - h^1)/(\ \ -1)*(n-1) \quad (1)$$

h는 n 번째 호모그래피 행렬의 원소이다. 위 식(1)을 이용하여 호모그래피 행렬 H_2을 구하면 다음과 같다.

$$H_2 = \begin{bmatrix} h_1^1 + (h_1^N - h_1^1)/(N-1)*1 & h_2^1 + (h_2^N - h_2^1)/(N-1)*1 & h_3^1 + (h_3^N - h_3^1)/(N-1)*1 \\ h_4^1 + (h_4^N - h_4^1)/(N-1)*1 & h_5^1 + (h_5^N - h_5^1)/(N-1)*1 & h_6^1 + (h_6^N - h_6^1)/(N-1)*1 \\ h_7^1 + (h_7^N - h_7^1)/(N-1)*1 & h_8^1 + (h_8^N - h_8^1)/(N-1)*1 & 1 \end{bmatrix}$$

[그림 2] 식(1)을 이용한 호모그래피 행렬 H_2

위와 같은 방법으로 선형 호모그래피 행렬 H_{N-1}까지 구한 후, 구해진 호모그래피 행렬을 사용하여 이미지 스티칭을 진행한다. [그림 3]는 N=10 일 때, 호모그래피 행렬 H_1, H_{10}을 계산하고, H_1, H_{10}을 사용하여 H_2, …, H_9을 생성한 결과이다.

$$H_1 = \begin{bmatrix} 0.316439 & 0.180669 & 703.026 \\ -0.22353 & 0.900518 & -57.2087 \\ -0.00047269 & 0.000114387 & 1 \end{bmatrix} \quad H_2 = \begin{bmatrix} 0.316003 & 0.179586 & 702.058 \\ -0.223281 & 0.899005 & -55.5413 \\ -0.000472575 & 0.000113566 & 1 \end{bmatrix}$$

$$H_3 = \begin{bmatrix} 0.315568 & 0.178502 & 701.089 \\ -0.223032 & 0.897492 & -53.8739 \\ -0.00047246 & 0.000112746 & 1 \end{bmatrix} \quad H_4 = \begin{bmatrix} 0.315132 & 0.177419 & 700.121 \\ -0.222783 & 0.89598 & -52.2066 \\ -0.000472345 & 0.000111926 & 1 \end{bmatrix}$$

$$H_5 = \begin{bmatrix} 0.314697 & 0.176336 & 699.152 \\ -0.222534 & 0.894467 & -50.5392 \\ -0.00047223 & 0.000111106 & 1 \end{bmatrix} \quad H_6 = \begin{bmatrix} 0.314261 & 0.175252 & 698.184 \\ -0.222285 & 0.892955 & -48.8718 \\ -0.000472114 & 0.000110286 & 1 \end{bmatrix}$$

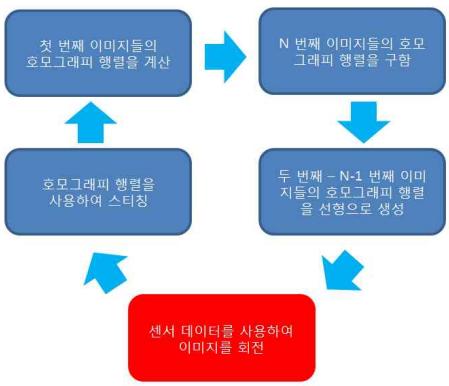
$$H_7 = \begin{bmatrix} 0.313826 & 0.174169 & 697.215 \\ -0.222035 & 0.891442 & -47.2044 \\ -0.000471999 & 0.000109466 & 1 \end{bmatrix} \quad H_8 = \begin{bmatrix} 0.31339 & 0.173085 & 696.246 \\ -0.221786 & 0.889929 & -45.5371 \\ -0.000471884 & 0.000108645 & 1 \end{bmatrix}$$

$$H_9 = \begin{bmatrix} 0.312955 & 0.172002 & 695.278 \\ -0.221537 & 0.888417 & -43.8697 \\ -0.000471769 & 0.000107825 & 1 \end{bmatrix} \quad H_{10} = \begin{bmatrix} 0.312519 & 0.170919 & 694.309 \\ -0.221288 & 0.886904 & -42.2023 \\ -0.000471654 & 0.000107005 & 1 \end{bmatrix}$$

[그림 3] N=10 일 때, 호모그래피 행렬 H_1, H_{10}과 생성된 호모그래피 행렬 H_2, …, H_9

3. 센서 데이터 활용

본 논문에서 제안한 선형 호모그래피 행렬 생성 방법을 이용하여 스티칭을 하면, 각 영상에서 호모그래피 행렬을 구해 스티칭하는 방법에 비해 스티칭 정확도가 낮아지게 된다. 이러한 문제를 보완하기 위하여 센서 데이터(촬영 시 촬영 기기의 3축 방향 정보)를 활용하는 방법을 제안한다. [그림4]은 [그림 1]에 센서 데이터를 사용하여 스티칭 정확도를 향상시키는 방법을 추가한 흐름도이다.



[그림 4] 센서 데이터를 사용하여 호모그래피 행렬의 정확도를 향상시키는 과정의 흐름도

센서 데이터를 활용한 3 DoF 이미지 스티칭 기법[8]에서 사용 하는 이미지 전처리 작업을 통하여 프레임 이미지 스티칭 전에 전처리를 수행하여 정확도를 향상시킬 수 있다. [그림 5]은 선형 호모그래피 행렬을 사용하여 스티칭한 결과 이미지와 센서 데이터로 이미지 전처리를 수행 한 후 스티칭한 결과를 보여준다. 선형 호모그래피 행렬을 사용한 이미지는 정면에 위치한 아파트가 부정확하게 스티칭 된 것을 볼 수 있다. 센서 데이터를 사용하여 이미지를 회전한 결과 완벽하지는 않지만 상당히 정확도가 향상된 것을 볼 수 있다.

[그림 5] 선형 호모그래피 행렬을 사용하여 스티칭(좌), 센서 데이터로 이미지 전처리를 수행 한 후 스티칭(우)

4. 실험 방법 및 결과

본 논문에서는 SURF 기법을 사용하여 스티칭을 진행하였다. [표 1]은 640×360 해상도의 Samsung사의 갤럭시 노트로 직접 획득한 1만 쌍의 프레임 이미지 데이터를 사용하여, 모든 프레임 이미지를 스티칭 했을 때와 호모그래피 행렬을 생성하여 스티칭 했을 때, 센서 데이터를 사용하여 생성된 호모그래피 행렬을 보정했을 때 걸린 시간이다.

일반적인 동영상의 프레임 비율은 약 30fps 이다. 5프레임 당 스티칭을 진행하면 속도 면에서 이득을 보기 힘들고, 15프레임 이상 임의의 호모그래피 행렬을 생성하면 정확도가 상당히 떨어졌다. 따라서 실험에서는 10장의 프레임 이미지 쌍을 대상으로 중간 8장의 이미지 쌍에 대한 호모그래피 행렬을 선형으로 생성하였다.

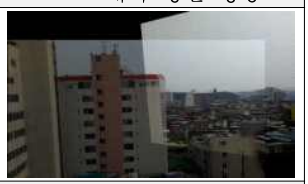
	모든 프레임	생성된 호모그래피 행렬	센서 데이터 보정
1만 쌍 스티칭(초)	25951.83	10463.64	11043.28
1쌍 평균 스티칭 시간(초)	2.60	1.05	1.10

[표 1] 모든 프레임 이미지, 호모그래피 행렬을 생성, 센서 데이터를 사용하여 보정했을 때 스티칭에 걸린 시간

모든 프레임 이미지를 스티칭 하는 방법은 매 이미지 마다 스티칭

을 수행하기 때문에 SURF 알고리즘 수행 속도인 약 2.6초만큼 소요되었다. 본 논문의 호모그래피 행렬 생성 방법을 사용하여 스티칭을 했을 때는 프레임 이미지 1쌍의 평균 스티칭 시간이 약 1.05초로 모든 프레임을 스티칭하는 방법 보다 약 40.4% 속도 향상이 있었다. 10장의 이미지 당 스티칭을 진행하여 90% 속도 향상을 기대하였지만, 호모그래피 행렬을 선형으로 계산하는 부분과 생성된 호모그래피 행렬을 C++ STL vector에 저장하는데 발생한 연산 속도 등에 의하여 기대한 속도에 미치지 못한 것으로 판단된다. 또한, 센서 데이터로 프레임 이미지를 보정하여 스티칭 하는 방법은 이미지 회전에 약 0.05초 정도 소요되어 약 1.1초가 걸렸다.

정확도 또한 동영상 스티칭의 성능에 중요한 요소이다. [표 2]는 모든 프레임 이미지, 호모그래피 행렬을 생성, 센서 데이터를 사용하여 보정했을 때의 프레임 이미지이다.

모든 프레임 이미지 스티칭	모든 프레임 이미지 스티칭
	
호모그래피 행렬 생성	호모그래피 행렬 생성
	
센서 데이터 보정	센서 데이터 보정
	

[표 2] 모든 프레임 이미지, 호모그래피 행렬을 생성, 센서 데이터를 사용하여 보정했을 때의 프레임 이미지

[표 2]의 왼쪽 영상에 대해 모든 프레임 이미지를 스티칭 한 결과는 상당히 양호하였다. 호모그래피 행렬을 생성하여 스티칭을 수행한 결과는 부정확한 호모그래피 행렬에 의하여 매끄럽게 스티칭이 되지 않았다. 하지만 센서 데이터를 사용하여 보정을 한 결과, 정면에 위치한 아파트는 여전히 흐릿하지만 중앙 하단의 병원 십자가 부분은 개선된 것을 볼 수 있다. [표 2]의 오른쪽 영상은 SURF 특징점 추출 알고리즘이 제대로 수행되지 않아, 모든 프레임 이미지를 스티칭 한 결과가 매우 부정확하다. 호모그래피 행렬을 생성한 이미지 역시 부정확하지만, 센서 데이터를 사용하여 보정을 한 결과는 상당히 개선되었다.

총 8쌍의 동영상 데이터를 사용하여 100개의 프레임 이미지를 추출한 후, 선형 호모그래피를 생성하여 스티칭을 진행하였을 때와 센서 데이터를 사용하여 보정을 하였을 때를 비교한 결과는 [표 3]과 같다. 센서 데이터를 사용하여 스티칭 결과가 개선된 비율은 평균 28.2%이고 개악된 비율은 평균 8.4%, 보정을 하였는데도 변화가 없는 이미지의 비율은 63.4%로 스티칭 정확도가 개선된 비율이 개악된 비율보다 높음을 볼 수 있다.

	같음(%)	개선(%)	개악(%)
동영상 쌍 1	70	16	14
동영상 쌍 2	89	10	1
동영상 쌍 3	72	28	0
동영상 쌍 4	58	15	27
동영상 쌍 5	28	72	0
동영상 쌍 6	56	22	22
동영상 쌍 7	63	18	19
동영상 쌍 8	62	28	10
평균	63.40	28.20	8.40

[표 3] 센서 데이터를 사용하여 보정한 결과

5. 결론 및 향후 연구

스티칭에 걸리는 시간을 단축시키기 위하여 본 논문에서는 호모그래피 행렬을 선형적으로 생성하였다. 또한 정확도를 높이기 위하여, 프레임 이미지 스티칭 전에 이미지를 전처리하였다. 제안한 방법은 평균 2.6초 걸리는 스티칭 시간을 약 1.5초 단축시켜 빠른 스티칭을 가능하게 하였다. 또한 선형 호모그래피 행렬만을 사용한 스티칭 한 결과보다 선형 호모그래피 행렬과 센서데이터를 함께 사용하였을 때 정확도가 28.2% 개선되었다. 하지만 예상과는 다르게 정확도가 낮아지는 경우도 발생하였다. 임의로 생성된 호모그래피 행렬을 사용하여 스티칭을 진행하였을 때, 스티칭 정확도가 낮은 경우 즉, 기준 이미지와 와핑된 이미지가 많이 틀어져 있을 때 센서 데이터를 사용하여 전처리를 하여도 정확도의 큰 변화가 없었다. 향후 이미지를 선별하여 센서 데이터를 적용하는 연구가 필요하다.

감사의 글

본 연구는 미래창조과학부 및 정보통신기술진흥센터의 정보통신·방송 연구개발사업의 일환으로 수행하였음. [B0126-15-1013, 퍼즐형 Ultra-wide viewing 공간 미디어 생성 및 소비 기술 개발]

참 고 문 헌 (References)

[1] Y. J. Cho, J. M. Seok, S. Y. Lim, S. W. An, J. I Seo, and J. H. Chan, “Post-UHD Realistic media, high quality panoramic AV technology”,Electronics and Telecommunications Trends, vol. 20, no. 3, pp. 33-46,June 2014.

[2] J. C. Bazin, I. Kweon, C. Demonceaux, and P. Vasseur, “Improvement of Feature Matching in Catadioptric Images Using Gyroscope,” 19th International Conference on Pattern Recognition, 1-5, 2008.

[3] R. Szeliski, “Image Alignment and Stitching - A Tutorial,” Foundations and Trends® in Computer Graphics and Vision, 2(1), 1-104, 2006.

[4] D. Lowe, “Distinctive image features from scale-invariant keypoints”, International Journal of Computer Vision, vol. 60, no. 2, pp. 91-110,November 2004

[5] L. M. J. Florack, B. M. Ter Haar Romeny, J. J. Koenderink, M. A.Viergever, “General intensity transformations and differential invariants”, Journal of Mathematical Imaging and Vision, Vol. 4, no. 2,pp. 171-187, May 1994.

[6] P. M. Panchal, S. R. Panchal, and S. K. Shah, “A Comparison of SIFT

and SURF," International Journal of Innovative Research in Computer and Communication Engineering, 1(2), 323-327, 2013.

[7] L. Juan and O. Gwun, "A Comparison of SIFT, PCA-SIFT and SURF," International Journal of Image Processing (IJIP), 3(4), 143-152, 2009.

[8] , 김상균 "관성 센서 데이터를 활용한 3 DoF 이미지 스티칭 향상", 방송공학회논문지 제22권 제1호, 2017.1, 51-61 (11 pages)