모바일 기기에서의 가속도 센서를 이용한 Rolling Shutter 왜곡 안정화 방법

최현철○ 김동철 박영민 우운택
광주과학기술원 정보통신공학부
{hchoi, dckim, ypark, wwoo}@gist.ac.kr

Rolling Shutter Distortion Stabilization using Accelerometer in Mobile Device

Hyeon-cheol Choi, Dong-chul Kim, Young-min Park, Woontack Woo
Gwangju Institute of Science and Technology

요 약

현대의 휴대기기들은 대부분 저비용 저전력의 CMOS를 채용한 카메라를 사용한다. 일반적으로 CMOS 센서는 영상의 왜곡을 발생시키는 Rolling Shutter를 채용하여 촬영을 한다. Rolling Shutter에 의한 왜곡현상을 보정하기 위해 기존의 방법들은 영상을 분석하여 카메라의 움직임을 추정한다. 하지만 증강현실과 같은 시스템에 있어 매우 중요한 실시간성을 보존하기 위해서는 더욱 빠르고 간결한 보정처리가 이루어져야 한다. 따라서 본 논문은 대부분의 모바일 기기에 장착되어있는 가속도 센서를 이용하여 왜곡현상을 고속 보정하는 방법을 제안한다.

1. 서 론

최근 스마트폰과 같은 휴대기기들이 대중화 되기 시작하면서, 이들 기기에 대한 이용율과 관심이 높아지고 있다. 특히 영상에 있어서, 저전력 고효율을 달성하기 위해 이러한 휴대기기들은 대부분이 낮은 소비전력과 가격을 가지는 CMOS(Complementary Metal-Oxide Semiconductor) 센서를 이용하고 있다. 모바일 기기에 사용되는 저가 CMOS 센서는 Rolling Shutter라는 전자식 셔터를 채용하는 것이 일반적인데, Rolling Shutter는 Global Shutter와 같이 한 프레임을 한꺼번에 찍는 것이 아니라 프레임을 상단에서 하단으로 한 라인씩 스캔하게 된다. 이 경우 위쪽 라인과 아래쪽 라인은 서로 다른 시간의 장면조각으로 볼 수 있으며 프레임에서의 노출, 이미지 스캔, 스캔된 이미지의 조합 과정이 비동기화 되어 이로인해 영상의 왜곡현상이 두드러지게 나타나는 문제점이 생긴다.

Rolling Shutter에 의한 왜곡현상을 보정하기 위한 방법은 크게 두 가지로 나눌 수 있다. 첫번째 방법은 센서 시프트 등의 하드웨어적인 방법으로 해결하는 것이다. 두번째 방법은 영상처리를 통한 디지털 보정이다. 하드웨어적으로 해결할 경우 높은 정확도를 보여 주지만 단가가 높아지는 문제점이 있고, 디지털 보정의 경우 내부적으로 처리하기 때문에 비용이 적게 들지만 정확도가 상대적으로 떨어지는 모습을 보인다.

디지털 보정에 있어서, Rolling Shutter에 의한 왜곡현상을 보정하기 위해 다양한 방법들이 제시되고 있다.[1],[2],[4],[5],[6],[7]. [4], [6]은 3D에 기반한 방법으로 연산량이 매우 크며, 따라서 실시간으로 처리하기에는 부담스러운 방법이다. [2]기법은 2D기반이므로 상대적으로 연산량의 부담을 덜었지만, 여전히 영상기반으로 속도를 추정하는 것은 실시간으로 처리하기에 무리가 있다. [3]의 경우 실시간으로 처리를 해야 하는 시스템이기 때문에 이러한 보상기법에서의 연산을 최소화하고 있다.

본 논문에서는 고속으로 안정화를 행하기 위해 모바일 기기 내부의 가속도 센서를 이용해 즉각적인 가속도를 인지하고, 이를 통해 입력영상을 보정하는 방법에 대한 알고리즘을 제안하고자 한다.

먼저, 왜곡현상에 대한 정의 및 가정을 2.1에서 소개한 뒤, 이를 보정하기 위한 알고리즘과 실험을 2.2와 2.3에서 설명하겠다. 마지막으로 실험에 대한 결과와 향후 연구방안에 대한 고찰을 3장과 4장에 나눠 기술하도록 하겠다.

2. 본 론

2.1. Rolling Shutter 왜곡 현상 정의 및 가정

CMOS 센서의 Rolling Shutter에 의한 왜곡현상은 크게 3가지로 나눌 수 있으며, 이는 각각 Skew, Wobbling, Partial Exposure 등으로 볼 수 있다. 이러한 현상은 모두 Rolling Shutter를 사용하는 경우에 발생하며, 수평적인 움직임에서 나타나는 Skew 현상이 가장 대표적인 왜곡형태이다.

Rolling Shutter에 의한 왜곡은 특히 근래들어 대중화 된 모바일 기기에서 자주 관찰되는데, 이미지의 왜곡은

단지 미관상의 문제뿐만 아니라 영상처리에 있어서도 어려움을 가져온다. 특히, 영상의 인식이나 추적에 있어서 이러한 왜곡은 정확도를 크게 떨어뜨리게 된다. 이러한 왜곡의 보정을 위해 [1]는 고해상도의 카메라에서, 작은 가속도가 있을때의 해결방안을 제시하고 있다. 하지만, [3]에서도 지적하고 있듯이, 이 방법은 모바일 환경에서 적용하기에는 무거운 연산이 들 수 있음을 감안해야 한다. 앞서 서론에서 밝힌 바 있지만, 본 논문은 모바일 기기의 실시간성을 보존하면서 Rolling Shutter에 의한 왜곡현상에 대한 보정을 행할 수 있는 방안을 제시하고자 한다. 현대의 스마트폰은 대부분 MEMS 가속도 센서를 내장하고 있기 때문에, 이 센서를 이용하여 그림 1에서 보이는 바와 같은 Rolling Shutter의 대표적인 왜곡현상인 Skew 왜곡을 보정하고자 한다.

제안한 방법을 적용하기 위해 몇 가지 조건을 가정한다. 먼저, 기기의 가속도 센서를 이용하므로, 기기의 속도만을 고려하며 물체 자체의 속도는 가정하지 않는다. 또한 이동속도를 구하기 위해 필요한 물체와의 거리값은 일정하지 않으므로, 제안하는 방법에서는 실시간 시스템에서의 상황을 고려하여 이 거리를 1m로 가정한다. 마지막으로, 매 프레임에서의 속도는 일정하다고 가정한다.

그림 1 Rolling Shutter로 인해 왜곡된 영상과 원본 영상

2.2. 알고리즘

먼저 수평방향으로의 이동을 가정해 본다. 오른쪽 방향으로의 모션이 생기는 것을 고려하면, CMOS의 스캔라인이 위쪽에서 아래로 진행된다고 가정할 때 모션에 따른 영상의 왜곡은 아래로 갈 수록 커질 것임을 예측할 수 있다.

그림 2에서 세로축 방향으로 CMOS Rolling Shutter가 스캐닝되어 내려가면, 수평방향으로의 속도 $\mathrm{V_x}$는 아래와 같이 정의된다.

$$\mathrm{V_x} = V_0 + a_x \Delta t \tag{1}$$

이는 매 프레임 마다의 속도이므로 Δt는 프레임 당 시간이 될 것이며, 이는 상수이거나, 또는 정확도를 위해 매번 계산될 수 있을 것이다. 또한 가속도 a_x는 센서에 의해 얻어진 값이다. 픽셀단위로 표현하기 위해 픽셀당 이동거리를 계산하고, 이에 따른 이동 거리 관계를 도출해 내야한다. 영상의 한 행에서 측정된 실제 길이 s와 한 행의 픽셀 수 W의 관계를 통해 픽셀단위의 속도 $\mathrm{PV_x}$를 아래와 같이 얻을 수 있다.

$$\mathrm{PV_x} = \mathrm{V_x}\frac{\mathrm{W}}{\mathrm{s}} \tag{2}$$

이제 전체 행의 수 H와 PV_x와의 관계를 통해 영상의 왜곡모델을 얻을 수 있다. 임의의 좌표 $\mathrm{P_{x,y}}$를 매트릭스 변환으로 표현하면 다음과 같다.

$$\begin{pmatrix} \mathrm{x}' \\ \mathrm{y}' \end{pmatrix} = \begin{pmatrix} 1 & \frac{PV_x}{H} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ \mathrm{y} \end{pmatrix} \tag{3}$$

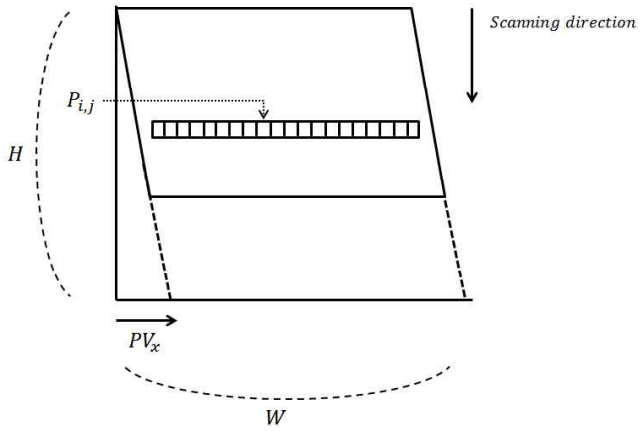

그림 2 수평방향으로의 모션에 따른 왜곡 모델

이제 이와 비슷하게, 수직방향으로의 이동을 가정해 보자. 이는 Rolling Shutter의 스캔 방향과 평행한 움직임이기 때문에, 스캔의 속도에 따라 영상이 축소되거나 확장될 것임을 알 수 있다. 모션에 대한 모델은 다음과 같이 표현 될 수 있다.

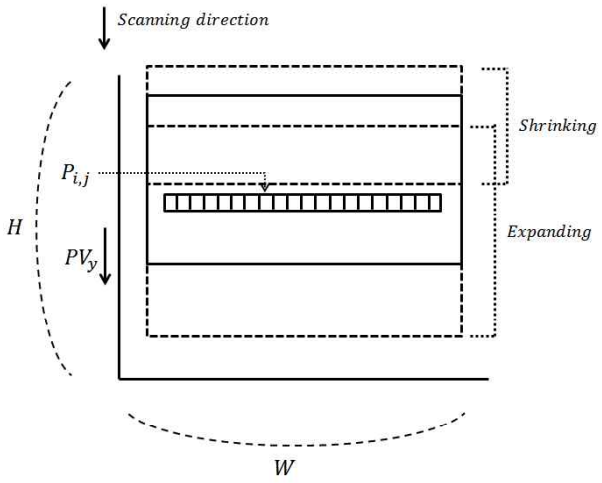

그림 3 수직방향으로의 모션에 따른 왜곡 모델

영상에 이미지가 출력되기 위해서는 스캐닝의 길이 S_{scan}가 상대적 좌표의 이동거리 S_p와 동일해야하므로, 아래와 같이 표현할 수 있다.

$$\mathrm{S_{scan}} = P_{x_0,y_0} + S_p \tag{4}$$

이를 속도에 관한 식으로 표현하고, 여기서 스캐닝 라인과 이동한 좌표의 위치가 동일할 때의 시간을 구할 수 있을 것이다. 일련의 과정은 아래와 같다.

$$\mathrm{S_{scan}} = Y_0 + S_p \tag{5.1}$$

$$\mathrm{V_{scan}}t = Y_0 + V_y t \tag{5.2}$$

$$(\mathrm{V_{scan}} - V_y)t = Y_0 \tag{5.3}$$

$$t = \frac{Y_0}{V_{scan} - V_y} \tag{5.4}$$

그러므로 이동한 점 $P'_{i,j}$는 다음과 같은 수식으로 도출된다.

$$\begin{aligned} P'_{i,j} &= P_{i,j} + \frac{P_{i,j}V_y}{V_{scan} - V_y} \\ &= \frac{P_{i,j}V_{scan}}{V_{scan} - V_y} \end{aligned} \tag{6}$$

수직방향과 수평방향으로의 매트릭스 변환을 함께 나타내면 최종적인 행렬은 아래와 같은 형태로 도출된다.

$$\begin{pmatrix} \mathrm{x}' \\ \mathrm{y}' \end{pmatrix} = \begin{pmatrix} 1 & \frac{PV_x}{H} \\ 0 & \frac{P_{i,j}V_{scan}}{V_{scan} - V_y} \end{pmatrix} \begin{pmatrix} x \\ \mathrm{y} \end{pmatrix} \tag{7}$$

입력 영상의 Skew 왜곡은 (7)에 의해 추정되어 이후의 처리에 이용된다.

2.3. 실험

실험은 iPhone4에서 직접 구현하여 수행하였으며, 480x360영상이 이용되었다. 수평방향의 경우 가속도 센서에 의해 구하여진 변환된 사각형의 각 꼭지점과 Rolling Shutter에 의해 왜곡된 실험 데이터간의 유클리드 거리를 픽셀 단위로 측정하였다. 수직방향의 경우 비율을 통해 차이를 측정하였다. 매트릭스 변환자체를 구하는 것은 센서값에서 추정되는 결과이므로 무거운 연산이 들지 않는다. 또한 실제 트래커에 적용하여 어느정도의 차이가 있는지 알아보았다.

3. 실험 결과

위의 방법으로 iPhone4에서 실제 구현된 알고리즘에 대한 오차결과는 일자 패턴에 대한 왜곡된 영상과, 매트릭스 변환사이의 관계를 통해 산출 할 수 있었다. 그림 4는 65번의 수평방향에 대해서의 모션에 따른 매트릭스 변환과 실제 왜곡된 이미지 상의 패턴과의 유사관계를 픽셀단위의 오차로 표현한 것이다. 두 왜곡의 픽셀단위 차이를 오차로 상정했을 때 평균은 은 3.738 픽셀이다. 모션의 정도를 점점 크게 해 가도록 정렬해 놓은 그래프에서, 오차는 증가하는 경향이 있음을 보인다. 또한 같은 레벨의 모션에서도 경우에 따라 차이를 보이는데, 이러한 오차가 발생하는 이유는 실험환경의 오차와 센서의 가속도가 급격히 증가할 때, $\mathrm{V_x}$를 구하는 과정에서 누적되는 오차에 의한 것으로 보인다. $\mathrm{V_x}$에 대한 추정값은 센서의 정확도에 의해 오차가 누적될 수 있는데, 이 경우 임계값을 통해 조정할 수 있을 것이다. 실험에서는 가속도의 임계값을 0.05로 둠으로써 모션의 끝에서 생길 수 있는 오차를 제거하였다.

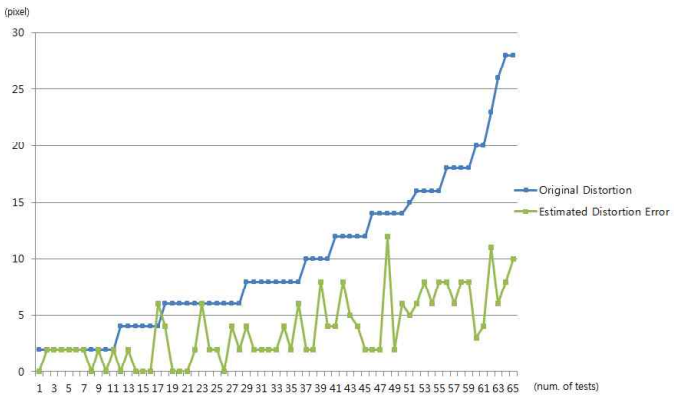

그림 4 수평 모션에 따른 왜곡 보정과 픽셀단위 오차

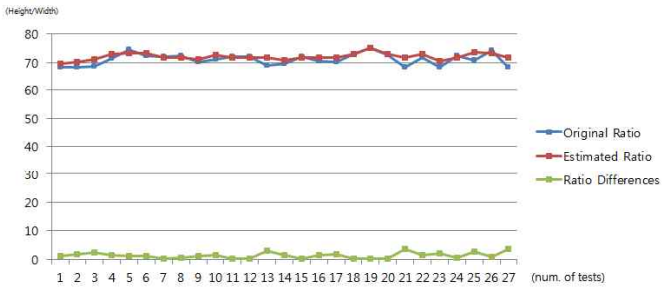

그림 5 수직 모션에 따른 왜곡 영상 보정과 오차

또한 수직방향의 모션에서는 영상의 ratio를 통해 오차를 측정하였는데, 그림 5에서와 같이 평균적으로 1.30의 오차가 발생하였음을 확인하였다. 그래프에서 나타나는 경향치들을 통해 수직방향으로의 확장이나 축소에 대해 대체적으로 좋은 추정을 보이고 있음을 확인할 수 있었다. 하지만 가속도 센서의 값은 고역

통과 필터를 사용하더라도 오차가 생겨 특정 케이스에서는 기대한 결과를 보여주지 못하는 경우도 있었다.

실제 템플릿 이미지와 일련의 왜곡된 이미지들을 ESM툴킷을 통해 매칭하여 나온 투사 영상과의 정규상관도는 표1과 같았다.

테스트 수	기존 영상과의 NCC(평균)	보정된 영상의 NCC(평균)
50	0.207915	0.272653

표 1 ESM 트래커를 이용해 구한 영상의 상관관계

그림 6 템플릿(왼쪽)과 기존 영상에서의 투사영상(중앙), 그리고 보정된 영상에서의 투사영상(오른쪽)

그림 6에서는 기존 영상을 이용해 트래킹을 했을 때와 보정된 영상을 통해 트래킹이 되었을 때의 차이를 보여준다.

4. 결론 및 향후 연구 방향

본 논문은 Rolling Shutter 현상 중 가장 두드러지는 왜곡현상인 Skew 왜곡 현상을 보정하기 위해 내장된 가속도 센서를 이용하였다. 측정된 값은 수평적인 모션과 수직적인 모션에 대해 적은 오차범위 이내에서 잘 표현됨을 확인했으며 이는 변환 모델로 추정되어 이후의 처리에 사용될 수 있다. 또한, 센서를 이용한 방식을 사용할 경우 영상의 변화에 대해 독립적인 장점을 가진다.

센서를 이용한 안정화는 기존의 무거운 연산을 들이는 방식에 비해 연산을 거의 들이지 않으면서 센서의 값을 기준으로 상대적으로 적정선의 보정을 유지할 수 있음을 확인했다. 하지만 제안하는 방법은 또한 가속도의 정밀도에 매우 의존적이고 실험에 의해 가속도 센서의 정밀도가 기대한 만큼 정확한 값을 도출해 내지는 못하기 때문에, 가속도 센서의 정확도를 향상시키기 위해 노이즈를 효과으로 필터링하는 방법을 알아내고 정확한 속도를 구해야 하는 문제를 해결해야 할 것이다. 이는 가속도 센서 자체의 정밀도의 개선을 통해 해결 될 수 있을 것이며, 그럴 경우 전체적인 보정의 정확성은 센서의 오차가 적을수록 더욱 증대될 것이다. 또한 향후에는 수평과 수직방향뿐만이 아닌 회전과 관련하여 자이로스코프 센서와 연동한 보정방법을 생각해 보는 것도 흥미로운 연구가 될 것이다.

5. Acknowledgement

본 연구는 문화체육관광부 및 한국콘텐츠진흥원으로 2011년도 문화콘텐츠산업기술지원사업의 연구결과로 수행되었음.

참고문헌

[1] 이윤구, 조경환, 박상규, “저비용 하드웨어 구현을 위한 롤링셔터 보상 알고리즘,” 대한전자공학회 2010년 하계종합학술대회, 33권, 1호, 2010

[2] C. Liang, L. Chang, H. H. Chen, “Analysis and Compensation of Rolling Shutter Effect,” IEEE Trans. on Image Processing, vol. 17, no. 8, 2008.

[3] Georg Klein, David Murray, “Parallel Tracking and Mapping on a Camera Phone,” ismar, pp.83-86, 2009 8th IEEE International Symposium on Mixed and Augmented Reality, 2009.

[4] Per-Erik Forseen, Erik Ringaby, “Rectifying Rolling Shutter Video from Hand-Held Devices.” cvpr, pp.507-514, 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2010.

[5] S. P. Nicklin, R. D. fisher, and R. H. Middleton, “Rolling Shutter Image Compensation,” in robocup 2006 LNAI 4434, pp.402-409, 2007.

[6] O. Ait-Aider, N. Andreff, J. M. Lavest, and P. Martinet, “Simultaneous Object Pose and Velocity Computation using a Single View,” in 9th European Conference on Computer Vision (ECCV 2006), pp.26-68, 2006.

[7] J.-B. Chun, H. Jung, C.-M. Kyung, “Suppressing Rolling Shutter Distortion of CMOS Image Sensors by Motion Vector Detection,” IEEE TCE, vol. 54, no. 4, pp. 1479-1487, 2008.