1999년도 대한전기학회 정기총회 및 추계학술대회 논문집 1999. 11. 20

물체의 3차원 회전에 대응 가능한 영상 추적 알고리듬

조 영 주* 유 범 재** 임 준 홍* 오 상 록**
*:한양대학교 전자공학과 **:한국과학기술연구원 지능제어연구센터

Visual Tracking Insensitive to 3D Rotation of Objects

YoungJoo Cho*, Bum-Jae You**, JoonHong Lim*, Sang-Rok Oh**
*:Dept. of Electronics Eng. Han Yang Univ.
**:Intelligent System Control Research Center. KIST

Abstract - 영상 추적(visual tracking)은 로봇의 시각기반제어, 교통정보시스템, 무인감시시스템 등 다양한 분야에 적용 가능하기 때문에 지정된 혹은 운동이 감지된 물체를 지속적이고 빠르게 추적하는 데 목적을 둔다. 이 때 어려운 문제 중 하나는 시간이 지나면서 위치이동은 물론 회전에 의해 물체의 모양이 변한다는 것이다. 이에 본 논문에서는 물체의 3차원 회전에 대응 가능한 실시간 영상추적 알고리듬을 제안한다. 이 알고리듬은 SSD(sum-of-squared differences)를 기반으로 하되, 물체의 배경이 바뀔 때나 물체가 영상추적 윈도우보다 작은 경우에도 추적이 가능하고 3차원 회전에 대응 가능하다. 이것은 3차원 회전으로 인하여 추적목표를 잃어버리는 것을 막기 위하여 기준 영역이 회전할 때 제안된 성능지수에 따라 영상추적 영역과 기준 영상을 새롭게 설정해줌으로써 구현된다. 제안된 알고리듬은 PC기반 실시간 시각시스템에서 성공적으로 실험되었다.

1. 서 론

영상 추적(visual tracking)은 영상기반제어, 감시, 농업자동화, 의용 공학 등을 포함한 다수의 응용분야에서 시스템의 중요한 부분을 차지한다. 영상 추적은 카메라의 시점에서 지정된 또는 움직이는 물체의 정확한 위치를 지속적이고 빠르게 추적하는 데에 목적을 둔다.

추적에 있어서 어려운점은 시간이 지남에 따라 2차원 평면에 영사된 물체의 이미지가 변한다는 것이다. 이러한 변화는 3가지로 나눌 수 있는데 물체의 위치와 자세에 따른 변형과 조명에 의한 변화, 그리고 다른 물체와 겹치거나 배경이 바뀔 때이다. 또한 계산량이 많아짐에 따라 디지털신호처리소자(DSP chip)를 많이 사용하게 된다. 이러한 문제를 해결하기위해 T. Yamane등[4]은 optical flow, 단일밝기영역을 정보로하여 단일 카메라와 15개의 디지털신호처리소자(DSP chip)를 가진 이미지 프로세서를 통해 한사람을 추적하였다. Y. Shrai등[5]는 스테레오 카메라를 이용하여 optical flow, depth, 그리고 단일 밝기 영역(uniform brightness region)으로 여러 사람 추적을 하였다. 하지만 많은 연산량이 필요한 optical flow를 계산하기위해 여러개의 디지털신호처리소자(DSP chip)를 사용하였다. Natan Peterfreund[2]는 snake를 이용하여 복잡한 배경에서와 물체의 겹침문제를 해결하였다. 칼만필터의 단점인 갑작스러운 방향전환에도 잘 적응하는 뉴칼만필터기반의 contour 모델을 제안하였다. G. D. Hager[1]는 실시간 영상추적을 위해 많은 디지털신호처리소자(DSP chip)를 쓰는 대신에 성능지수함수를 최적화기법에 의해 풀어서 CPU만을 가지고도 실시간 영상추적이 가능하다는 것을 보여주었다. 축적된 파라미터 행렬을 가지고 어파인변환을 계산하였으며 물체의 회전, 신축, 이동을 추적하였다. G. D. Hager 와 P. N. Belhumeur[3]는 조명에 대한 변화와 다른 물체와의 겹침에 관한 문제를 해결하였다. 하지만 다른 물체는 지금의 물체와 평균값이 확연히 다른 것이어야하고 3차원 회전에 대해서는 추적목표를 놓칠 수 밖에 없었다.

그래서 본 논문은 실시간추적에 있어서 별도의 특수목적 하드웨어를 사용하지않고 SSD를 기반으로 하여 물체를 추적하기 위하여 화소의 특징(pixel identity)를 이용하고 성능지수함수를 재정의하여 물체의 3차원 회전에 대응가능한 알고리듬을 제안하였다.

본 논문은 다음과 같이 구성된다. 2.1장에서는 [1]에서 제안된 어파인변환(Affine Transform)에 대해 간략히 소개하고 2.2장에서는 물체의 배경이 바뀌거나 영상추적윈도우보다 작은 물체를 추적하기 위한 방법을 제안하였다. 2.3장에서는 3차원으로 회전하는 물체를 추적하기 위한 방법, 2.4장에서는 프로그램 개발환경과 구현 그리고 이전 방법과 차이를 보여주었다. 마지막으로 3장에서는 다음 연구방향에 대해서 이야기한다.

2. 영상추적 알고리듬

2.1 어파인변환(Affine Transform)을 이용한 영역 기반 추적

SSD란 sum-of-squared differences의 약자로 기준 템플레이트(Reference Template)와 실시간으로 들어오는 이미지의 정해진 영역의 차를 제곱하여 가장 차이가 작게 나오는 부분을 원 이미지로 본다. 원 추적윈도우 근처에서 최소 SSD를 찾는 것이 대부분이다[7][8]. 하지만 [1]에서는 최적화기법을 통하여 직접 그 위치를 구했는데, 영역기반 추적기(Region tracker)의 성능지수함수는 다음과 같다.

$$O(\mathbf{A};\mathbf{d}) = \sum_{x\in W}(I(\mathbf{A}x+\mathbf{d},t)-I(x,t_0))^2\omega(x), t > t_0 \qquad (1)$$

[3]에서는 물체가 신축(scale), 회전(rotation), 이동(transition)하는 것을 다른 디지털신호처리소자(DSP chip)를 사용하지 않고 직접 구할 수 있는 파라미터모델이 제시되었다. 최종성능지수 함수와 얼마나 추정한 값이 잘 되었지를 나타내는 r값은 다음과 같다.

$$O(\mathbf{A}';\mathbf{d}') = \sum_{\mathbf{x}\in W}(\nabla I(\mathbf{x},t)(\mathbf{A}'\mathbf{x}+\mathbf{d}')+(J(\mathbf{x},t)-I(\mathbf{x},t_0)))^2\omega(\mathbf{x}) \qquad (2)$$

$$r = \frac{\sqrt{\sum_{\mathbf{x}\in W} h_2(\mathbf{x})^2}}{|W|} \qquad (3)$$

이 때 I는 화소값(pixel value), J는 워핑(warping)한 값을 나타내며, **A**와 **d**는 신축, 비틀림(shear)값과 이동벡터를 나타내는 행렬이다.

2.2 배경이 변하거나 추적 윈도우보다 작은 물체추적방법

어파인변환의 가장 큰 단점은 물체가 윈도우보다 작은 경우에 물체 외 배경은 물체가 움직이거나 배경이 움직일 때 노이즈 값이 된다. 사람은 물체를 추적할 때

고정되어있는 물체인지 움직이는 물체인지를 인식하여 인식한 후에는 움직이는 물체만을 생각한다는 원리로 보면 각 화소의 특징이 움직이는 화소인가 고정되어있는 화소인가를 알아내어, 움직이는 화소만을 가지고 어파인 변환을 수행하면 추적목표를 놓치는 일은 없어질 것이며 그만큼 계산량이 줄어들어라 예상된다. 예를 들어 80×80크기 의 윈도우가 있을 때 배경이 차지하는 부분이 10%라면 한 화소당 곱하기가 4번씩 줄게 된다. 전체 윈도우로 생각하면 2560번 곱하는 시간이 줄게 되는 것이다.

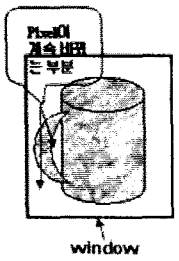


그림1:배경 값이 항상 변하여 추적목표를 놓침

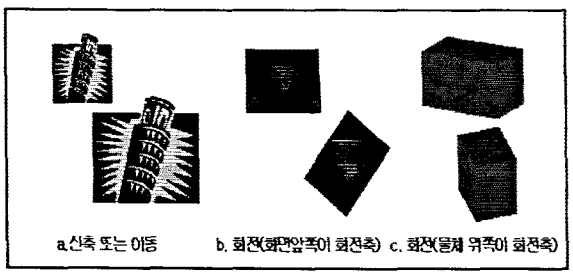
그림1의 컵 모양의 물체는 윈도우가 정사각형일 때 물체를 그랩해도 화소값이 계속 바뀌는 부분이 있으므로 (예를 들면 배경이 바뀌거나 물체가 움직일 때) 물체를 추적할 수가 없었다. 하지만 윈도우의 모양을 물체의 모양, 여기서는 컵의 모양, 과 같게 만들면 물체 외 영역이 변화가 있어도 상관이 없이 물체를 좇아갈 수 있게 되었다. 추적목표물을 구성하는 화소의 특징(pixel identity)을 알아내기 위한 방법에는 여러 가지가 있다. 첫째로 윤곽선추출(edge detection)을 이용하여 contour를 설정하는 법, 둘째로 optical flow를 이용한 학습방법, 셋째로 간편한 방법인 배경을 그랩한 후 물체를 놓아 차이를 추출하는 방법을 들 수 있다. 하지만 실제로 세 번째 방법의 문제점은 물체의 윤곽근처에서 화소값이 바뀌는 경우가 많으므로 물체보다 더 크게 윈도우가 설정될 수 있다. 이러한 문제점을 해결하기 위하여 첫째방법과 셋째방법을 같이 이용하면 윈도우의 윤곽선을 추출하여 그 부분을 빼주면 물체의 크기와 같은 윈도우를 얻을 수 있다. 화소의 특징을 결정하는 식은 다음과 같다.

$$m(\mathbf{x}) = \begin{cases} 0 & (I(\mathbf{x},t_0) - I(\mathbf{x},t_{-1})) > \alpha \quad \& \quad e(\mathbf{x}) \\ 1 & otherwise \end{cases} \qquad (4)$$

t_{-1}은 배경을 얻는 시간을 말하고, $e(\mathbf{x})$는 윈도우의 윤곽선을 나타낸다. 그리고 최종 성능지수함수는 다음과 같다.

$$O'(\mathbf{A}';\mathbf{d}') = \sum_{\mathbf{x}\in W} (\nabla I(\mathbf{x},t)(\mathbf{A}'\mathbf{x}+\mathbf{d}') + (J(\mathbf{x},t) - I(\mathbf{x},t_0)) \cdot m(\mathbf{x}))^2 \omega(\mathbf{x}) \quad (5)$$

2.3 3차원 물체의 회전을 개량된 어파인변환을 사용하여 추적



1림 2: 어파인변환(Affine Transform)을 설명

그림 2를 보면 어파인변환을 잘 이해할 수 있다. 어파인변환을 이용하면 물체의 이동, 회전(중심축이 화면의 나오는 방향)및 신축(scale)하는 것을 인식하고 추적할 수 있다. 하지만 어파인변환의 문제점은 그림2에서와 같이 3차원 물체의 한 면만을 추적할 수 있다. 그래서 물체의 앞면이 회전했을 경우 면적은 점점 작아지고 그 면이 사라지면 추적목표를 놓치게 된다. 이러한 추적을 3차원 물체의 모델링으로 추적할 수 있지만 많은 계산시간을 요구한다[6].

어파인변환에서 기준 템플레이트(reference template)를 다시 설정함으로써 3차원 회전에 대응가능한 영상추적이 가능하다. 이것의 물리적인 의미는 지금까지 축적되어왔던 에러를 다시 0으로 돌린다것이다. 기준 템플레이트를 설정하는 것은 어파인변환 추적에서는 오프라인에서 계산을 했었다. 그래서 온라인계산이 줄어들어 빠른 추적이 가능하였다.[1] 하지만 기준 템플레이트를 한번 계산하는 시간이 30ms정도 즉 한 프레임을 그랩하고 계산하고 화면에 보여주는 시간과 비슷하다. 그래서 회전이 많이 되면 윈도우의 종횡비(aspect ratio)가 변하고 어떠한 기준치를 넘을 때 기준 템플레이트를 재설정하면 된다. 이것은 매 프레임마다 하는 것이 아니기 때문에 계산하는 시간은 문제가 되지 않는다.

그러나 문제점은 어파인변환을 하면 원래의 이미지로 역변환할 수 있지만 이 방법은 원래의 이미지를 잃어버릴 수 있는 단점이 있다. 하지만 원래의 기준 템플레이트를 추적하는 것이 아니라 물체를 추적한다는데 관점을 둔다면 문제될 것이 없다. 물체 추적기(object tracker)의 상태방정식은 다음과 같이 정의된다.

$$S_t = (A_t, d_t, r_t, \eta_t) \qquad (6)$$

η_t는 종횡비(aspect ratio)를 말하고 r_t를 초기화(reset)시킨다.

2.4 실험결과

제안된 알고리듬은 펜티엄 200Mhz를 CPU로, Matrox Genesis-LC 보드를 가진 PC에서 실행하였다. 시스템의 OS는 Windows NT이고 소프트웨어는 Visual C++에서 작성하였다.

화소의 그래디언트(∇I)를 구하기위해서 소벨(sobel)을 사용하였다. 실험중 이동 벡터의 속도를 빠르게 하기위해 벡터를 다음과 같이 재정의하였다. d의 크기는 일반적으로 평균 0.5를 갖는데 30ms에 한 프레임을 그랩하고 계산한다면 1초에 33프레임이고 실제 5cm 윈도우가 80화소이라면 1초에 좇아갈 수 있는 거리는 1.65cm 이다.

Resolution을 움직임에 따라 다르게 할 수 있지만 [1] 계산상 오버로딩이 되므로 이동벡터 d의 값이 움직임이 작은 곳에서는 0에 가깝고 움직임이 큰 곳에서는 크기가 크므로 이동벡터 d를 재정의하여 만약 n값을 7로 했을 경우 1초에 12cm를 따라갈 수 있게 된다. 실제로는 평균 d값도 커지므로 15cm이상 좇아갈 수 있다.

$$\begin{aligned} d' &= n \cdot d \\ &= \beta \cdot r \cdot d \end{aligned} \qquad (7)$$

여기에서 $n=\beta \cdot r$이고 β는 r에 대한 이동벡터의 비례상수이다. 이 방법으로 빠른 추적과 오실레이션에 대한 문제도 해결할 수 있게 된다.

2.4.1. 어파인변환에 의한 얼굴추적

그림3은 얼굴의 회전, 이동, 신축을 나타낸다. 특별한 디지털신호처리소자(DSP chip)를 쓰지않고 샘플링을 2:1로 했을 때 한 프레임을 그랩하고 계산하는데 드는 시간은 약22~25ms정도 나왔고 40~45Hz의 속도

그림3 기준 템플레이트와 시간에 따른 얼굴추적

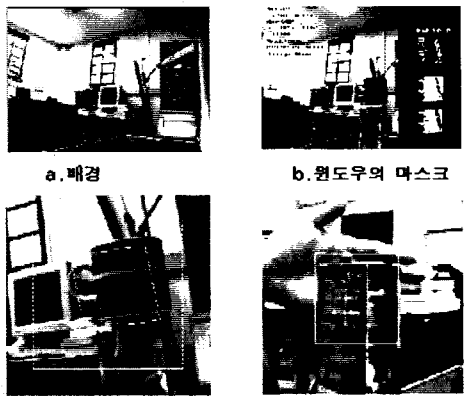


그림 4. 물체의 모양에 상관없는 추적

로 실시간 추적을 할 수 있다.

2.5.2. 물체와 같은 크기의 윈도우를 사용한 추적

그림4는 배경(a)과 윈도우의 물체에 따른 마스크(b), 물체의 회전(c), 물체의 배경변화에 상관없는 추적(d)을 보여주었다. b는 추적을 시작했을 때 모니터에 출력되는 화면이다. 왼쪽의 텍스트는 프레임수, 이동, 신축, 계산시간을 나타내는 양을 표시하고 오른쪽은 첫 번째 그림은 윈도우의 윤곽선을 나타내고, 두 번째 그림은 원래의 윈도우, 마지막은 윤곽선으로 에러를 제거한 최종윈도우를 나타낸다.

실험을 통해서 물체 외 부분이 너무 크면 이동을 계산할 수 있는 정보량이 줄어들게 되므로 너무 크지 않게 조절해야할 필요성을 느꼈다. 처리시간은 샘플링을 2:1로 했을 때 한 프레임당 시간이 약 19~22ms가 나와서 45~50Hz의 속도로 실시간 추적을 하였고 c와 같이 카메라를 이동하여 물체의 회전, 이동을 보였고 d에서는 손으로 배경을 변화하여도 추적윈도우를 잃어버리지 않는다는 것을 보여주었다

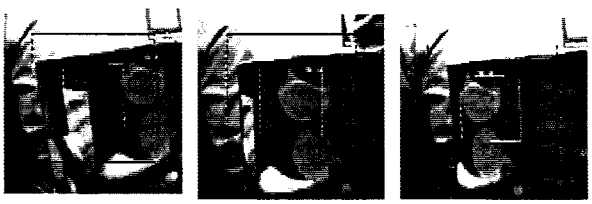


그림 5. 텍스쳐 원통의 회전을 추적

.2.4.3. 3차원 회전에 대응 가능한 추적

텍스쳐 이미지(texture image)를 가진 원통을 사용하여 회전을 시켜보았다. 그림 5는 일반적인 회전, 이동뿐 아니라 3차원회전에도 가능하다는 것을 보여주었다.

3. 결론 및 추후계획

본 논문은 2차원 영상추적알고리듬에서 물체의 배경이 바뀌거나 추적윈도우보다 작은 물체에 대한 알고리듬을 SSD의 관점에서 접근, 연구하였다. 그리고 어파인 변환을 개량하여 3차원회전에 대응가능한 알고리듬을 제안하였으며 그 예로서 회전하는 텍스쳐 원통을 추적하였다. SSD는 화소값을 가지고 계산하기 때문에 표면이 텍스쳐인 물체를 더욱 잘 추적한다. 그래서 얼굴의 옆면에 페인팅을 조금한다면 추적이 가능하겠지만 그냥 맨얼굴을 추적하기에는 어려운점이 있었다. 이것을 해결하기 위하여 정보가 훨씬 많은 컬러이미지를 이용한다면 얼굴 또한 충분히 추적 가능할 것이라 예상된다. 그리고 사람 눈과 같이 스테레오 카메라를 이용한다면 물체의 거리를 통해서 겹침에 대한 문제를 비롯하여 더욱 강인한 제어를 할 것이라 예상된다.

추후 연구에서는 컬러 스테레오 카메라를 이용하여 물체의 겹침 문제와 한 눈이 가리워졌을 때, 텍스쳐 원통이외의 더욱 복잡한 물체를 추적할 수 있는 더욱 강인한 영상 추적알고리듬을 제시하고자한다.

[참 고 문 헌]

[1]Gregory D. Hager and Kentaro Toyama, "X Vision: A Portable Substrate for Real-Time Vision Applications", Research Report YALEU/DCS/RR-1078, pp4-11, 1997

[2]Natan Peterfreund, "Robust Tracking of Position and Velocity With Kalman Snakes", IEEE Trans. PAMI, vol. 21, no. 6 pp564-569, 1999

[3] Gregory D. Hager, Peter N. Belhumeur, "Efficient Region Tracking With Parametric Models of Geometry and Illumination", IEEE Trans. Robotic and Automation, vol. 20, no. 10, pp1025-1038,1998

[4]T. Yamane, Y. Shirai, J. Miura, "Person Tracking by Integrating Optical Flow and Uniform Brightness Regions", Proc. IEEE Intl. Conf. on Robtics & Automation, pp3267-3272, 1998

[5]Y. Shirai, T. Yamane, R. Okada, "Robust Visual Tracking by Integrating Various Cues', IEICE Trans. Inf. &Syst, vol E81-D, no. 9, pp951-958, 1998

[6]S. Lee, B. You, G. Hager, "Model-based 3-D object Tracking using Projective Invariance", Proc. IEEE Intl. Conf. on R&A, pp1589-1594, 1999

[7]Nikolaos P. Papanikolopoulos, Pradeep K. Khosla, Takeo Kanade, "Visual Tracking of a Moving Target by a Camera Mounted on a Robot: A Combiantion of Control and Vision", IEEE Trans. Robotic and Automation, vol. 9, no. 1, 1993

[8]Kevin Nickels, Seth Hutchinson, "Measurement error estimation for feature tracking", Proc. IEEE Intl. Conf. on R&A, pp3230-3235, 1999