JKIICE Journal of the Korea Institute of Information and Communication Engineering

한국정보통신학회논문지(J. Korea Inst. Inf. Commun. Eng.) Vol. 18, No. 11 : 2733~2737 Nov. 2014

실시간 영상에서 물체의 색/모양 정보를 이용한 움직임 검출 알고리즘 구현

김남우[1] · 허창우[2*]

The motion estimation algorithm implemented by the color / shape information of the object in the real-time image

Nam-Woo Kim[1] · Chang-Wu Hur[2*]

[1]Department of Electronic Engineering, Mokwon University, Daejon 302-318, Korea
[2*]Department of Electronic Engineering, Mokwon University, Daejon 302-318, Korea

요 약

실시간 영상을 이용하여 움직임 검출을 하는데 사용하는 배경 차영상 기법에 의한 움직임 및 변화 영역 검출 방법과 움직임 히스토리에 의한 움직임 검출법, 광류에 의한 움직임 검출법, 움직임 추적을 위한 추적하려는 물체의 히스토그램의 역투영을 이용하면서 물체의 중심점을 추적하는 MeanShift와 물체의 중심, 크기, 방향을 함께 추적하는 CamShift, Kalman 필터에 의한 움직임 추적 알고리즘 등이 있다. 본 논문에서는 물체의 색상과 모양 정보를 이용한 움직임 검출 알고리즘을 구현하고 검증하였다.

ABSTRACT

Motion detection according to the movement and the change area detection method according to the background difference and the motion history image for use in a motion estimation technique using a real-time image, the motion detection method according to the optical flow, the back-projection of the histogram of the object to track for motion tracking At the heart of MeanShift center point of the object and the object to track, while used, the size, and the like due to the motion tracking algorithm CamShift, Kalman filter to track with direction. In this paper, we implemented the motion detection algorithm based on color and shape information of the object and verify.

키워드 : 움직임 검출, 움직임 추적, 배경 차영상, 움직임 히스토리 이미지, 광류. MeanShift, CamShift, 칼만 필터

Key word : Motion detection, Motion Tracking, Background Substraction, Motion History Image, Optical Flow.MeanShift, CamShift, Kalman Filter

접수일자 : 2014. 10. 01 심사완료일자 : 2014. 10. 31 게재확정일자 : 2014. 11. 05

* **Corresponding Author** Chang-Wu Hur(E-mail:chang@mokwon.ac.kr, Tel:+82-42-829-7655)
Department of Electronic Engineering, Mokwon University, Daejeon 302-318, Korea

Open Access http://dx.doi.org/10.6109/jkiice.2014.18.11.2733 print ISSN: 2234-4772 online ISSN: 2288-4165

Ⅰ. 서 론

객체 추적은 연속한 영상들로부터 관심 객체의 위치와 크기 등을 알아내는 것으로서 추적대상을 사람으로 한 응용 분야는 감시시스템, HCI(Human Computer Interaction), 가상 현실 등이 있다. 특히 최근에는 신분확인, 표정인식, 행동인식 등의 인식분야에 대한 연구가 많아지면서 인식을 적용할 영역의 분할(segmentation)의 목적으로 추적 알고리즘이 필수적으로 요구된다.

기존의 객체 추적 알고리즘들은 움직임정보를 이용한 방법, 모양(shape) 정보를 이용한 방법, 칼라 정보를 이용한 방법을 많이 사용된다.

움직임 정보를 이용한 방법은 이동 물체의 색깔이나 모양에 관계없이 검출할 수 있는 장점이 있지만 영상에는 움직임 벡터의 모호성이 존재하고 팬-틸트 회전 및 배율제어가 빈번히 발생하는 경우에는 정확한 이동체의 움직임 영역을 검출하기 어렵다.

칼라 히스토그램과 모양 정보를 같이 사용하면 두 방법이 가지는 단점을 보안하여 더욱 강인한 추적 결과를 제공할 수 있으며 대표적으로 Birchfield가 제안 한 방법[1]과 Comaniciu가 제안한 방법[2, 3]가 있다.

Birchfield가 제안한 방법[1]은 정해진 검색 영역에서 타원의 크기와 위치를 바꿔가면서 타원 내부의 컬러 히스토그램이 모델 히스토그램과 유사한 정도와 타원 경계에 존재하는 경사도 벡터(gradient vector)가 타원의 모양을 이루는 정도를 측정하여 가장 타당한 후보를 선택하는 전역 탐색 방법(full search)을 사용한다. 두 가지 정보를 동시에 고려하기 때문에 보다 정확한 검출 결과를 제공하지만 전역검색(full search)방법을 사용하여 많은 계산량이 요구된다는 단점이 있다.

Comaniciu가 제안한 방법[3]은 칼라 히스토그램의 유사성을 측정하기 위해 Bhattacharyya 계수[4]를 사용하고 mean shift[5] 라는 최적화 방법을 도입함으로써 몇 번의 반복(iteration)만으로 추적 대상의 중심 위치로 수렴하여 매우 빠른 추적 속도를 제공한다. 하지만 추적 대상의 칼라 히스토그램이 변할 경우 정확한 추적 대상의 위치와 크기를 찾아내지 못 한다는 단점이 있다.

Ⅱ. 본 론

색이나 모양 특징값 등을 이용해 이동 물체를 모델링하고 이 모델을 이용하여 가장 적합한 영역을 이동 물체의 영역으로 결정한다. 이 방법은 비교적 간단하지만 정확한 초기 이동물체 분할이 어렵고 시간에 따라 이동물체의 색이나 모양이 변화하는 경우 추적에 실패하기 쉬울 뿐 아니라 한번 잘못 예측된 모델이 다음 추적 오류의 원인이 되는 오류 전파의 문제를 발생시킨다.

본 논문에서는 특정 색을 지정한 뒤 화면에서 그 색의 원만 검출한다. 알고리즘 구현을 위해 intel사의 OpenCV를 사용하였으며, 순서는 다음과 같다.

1. 특정 포인트의 색을 지정
2. 원영상의 Smoothing
3. 색 공간 변환 RGB → HSV
4. 지정한 포인트의 H,S,V 값 추출
5. 검출할 H,S,V 값의 범위지정
6. HSV 이미지마스킹
7. 모폴로지
8. 이미지에서 원검출

2.1. 특정 포인트의 색을 지정

cvSetMouseCallback("PreviewImage",on_eventhandle,(void*)img); 를 사용하였는데 img 영상에서 마우스 이벤트가 발생하면 on_eventhandle 라는 함수를 호출하도록 Setting하였다.

작성한 on_eventhandle은 다음과 같다.

```
void on_eventhandle(int event, int x ,int y, int flag, void*
param)
{
  switch(event)
  {
    case CV_EVENT_LBUTTONDOWN:
    if(getFlag == 0)
    {
      getx = x;
      gety = y;
     getFlag = 1;
    }
    else if(getFlag == 1)
    {
      getFlag = 2;
```

```
    }
    break;

     case CV_EVENT_RBUTTONDOWN;
    getFlag=0;
    break;
   }
}
```

2.2. 색 공간 변환 RGB → HSV

색공간을 기존의 RGB (Red, Green, Blue) color space에서 HSV(Hue, Saturation, Value - Brightness) 로 바꾼다. OpenCV 에서는 cvCvtColor(src, dest, CV_BGR2HSV); 로 간단하게 바꿀수 있다.[6]

2.3. 지정한 포인트의 H,S,V 값 추출

cvGet2D()함수를이용하여 cvScalar로저장하는방법을사용 cvGet2D()는아래와같이설명하고 있다.

C: CvScalar cvGet1D(const CvArr* arr, int idx0)
C: CvScalar cvGet2D(const CvArr* arr, int idx0, int idx1)
C: CvScalar cvGet3D(const CvArr* arr, int idx0, int idx1, int idx2)
C: CvScalar cvGetND(const CvArr* arr, const int* idx)
Python: cv.Get1D(arr, idx) → scalar Python: cv.Get2D(arr, idx0, idx1) → scalar
Python: cv.Get3D(arr, idx0, idx1, idx2) → scalar
Python: cv.GetND(arr, indices) → scalar Return a specific array element.

Parameters:
arr – Input array
idx0 – The first zero-based component of the element index
idx1 – The second zero-based component of the element index
idx2 – The third zero-based component of the element index
idx – Array of the element indices
The functions return a specific array element. In the case of a sparse array the functions return 0 if the requested node does not exist (no new node is created by the functions).
link
:
http://docs.opencv.org/modules/core/doc/old_basic_structures.html?highlight=cvget2d#CvScalar cvGet2D(const CvArr* arr, int idx0, int idx1)

2.4. 검출할 H,S,V 값의 범위지정

같은 H,S,V 값만가지는픽셀을선택하면 원의 색상자체가 완전히 일정하지 않으므로 일정범위를 줄 필요가 있다. H,S,V 값의 범위는 H = 0~179, S = 0~255, V = 0~255로 정하였다.

```
//calculate HSV Boundary
lowH = H * LOW_BOUND;
highH = H * HIGH_BOUND;
if(highH >= 180) highH = 179;

lowS = S * LOW_BOUND;
highS = S * HIGH_BOUND;
if(highS >= 256) highS = 255;

lowV = V * LOW_BOUND;
highV = V * HIGH_BOUND;
if(highV >= 256) highV = 255;
```

2.5. HSV 이미지마스킹

지정한범위의 색외에 나머지는 모두 마스킹시켜 원을 검출하기 용이하도록 한다. 마스킹은 IplImage 가 아닌 Mat를 이용한다.

cvCreateMat(int rows, int cols, int type);

cvInRange(const CvArr* src, const CvArr* lower, const CvArr* upper, CvArr* dst)

```
mask = cvCreateMat(size.height, size.width,CV_8UC1);
cvInRangeS(HSVImg,
cvScalar(lowH,lowS,lowV),cvScalar(highH,highS,highV),mask);
```

이와 같이 mask Mat를 만들고 cvInRanges 하면 원래 image에 정하진 범위안의 H,S,V 값 만을 남기는 Masking 이 완료된다.

2.6. 모폴로지

Masking한 이미지는 도형내부의 그림자나 외부와의 간섭으로 인한 불필요한 것들을 보정해줄 필요가 있다. 사용한 cvErode()와 cvDilate()외에 OpenCV에서는 그 외에 다른 모폴로지함수(cvMorphologyEx와같은)를 많이 제공한다.

```
cvErode(mask,mask,NULL);
cvDilate(mask,mask,NULL);
```

2.7. 이미지에서 원검출

cvHoughCircles를 이용하여 원을 검출하였다.

C++: void HoughCircles(InputArray image, OutputArray circles, int method, double dp, double minDist, double param1=100, double param2=100, int minRadius=0, int maxRadius=0)

C: CvSeq* cvHoughCircles(CvArr* image, void* circle_storage, int method, double dp, double min_dist, double param1=100, double param2=100, int min_radius=0, int max_radius=0)

그림 1. 지정한 원모양만 검출
Fig. 1 targeted circle detection

여기서 dp 1이면 입력해상도와 동일한해상도, 2면 인데, 1을 사용하였다. minDist는 검출된 원의 중심 사이의 거리를 설정하는 것인데, 너무 좁으면 가까운영역에서 많은원들이 검출될수있기 때문에 적당히 조절할 필요가 있다. minRadius는 검출할 원의 최소반지름, maxRadius는 검출할 원의최 대반지름 이다. 이 값들의 의 범위를좁히면 cvHoughCircles()의 수행속도가 빨라진다. 그러므로 필요에 맞게 범위조절하면 더 빠른 결과를 얻을 수 있다.

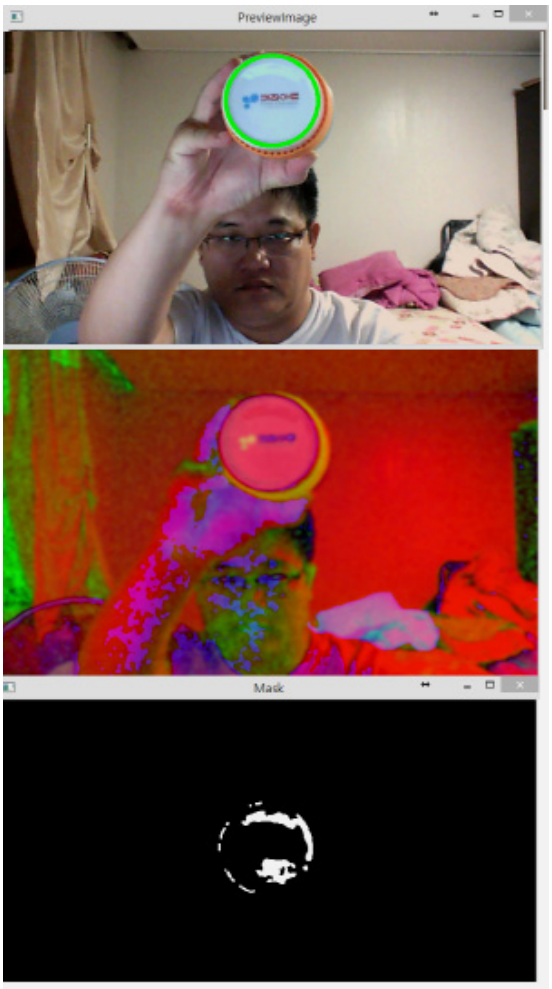

그림 2. 지정한 색상의 원모양만 검출
Fig. 2 targeted color circle detection

Ⅲ. 결 론

본 논문에서는 실시간 영상에서 움직임 검출에 관련한 기본 기술들을 이용하여 특정색상과 모양을 검출하는 알고리즘을 구현하였다. 구현된 알고리즘의 구현 순서는 특정 포인트의 색을 지정 하고 원영상을 Smoothing한 후 색 공간을 RGB에서 HSV로 변환하여 인식율을 높이는 작업을 진행한다. 이후 지정한 포인트의 H,S,V 값 추출하고, 검출할 H,S,V 값의 범위지정 하여 검출 범위를 높인 후에 HSV 이미지마스킹을 통해서 정확도를 높이는 방법으로 이미지에서 원검출 하였다.

시험 결과 색이나 모양 특징값 등을 이용해 이동 물체를 모델링하고 이 모델을 이용하여 가장 적합한 영역을 이동 물체의 영역으로 결정하는 방식은 비교적 간단하지만 정확한 초기 이동물체 분할이 어렵고 시간에 따라 이동 물체의 색이나 모양이 변화하는 경우 추적에 실패하기 쉬울 뿐 아니라 한번 잘못 예측된 모델이 다음 추적 오류의 원인이 되는 오류 전파의 문제를 발생시킬 수 있다는 것을 확인 하였다.

감사의 글

이 논문은 2013년도 목원대학교 연구년 지원에 의하여 연구되었음

REFERENCES

[1] Birchfield, S., “Elliptical Head Tracking Using Intensity Gradient and Color Histograms”, *IEEE Conf. Computer Vision and Pattern Recognition*, Santa Barbara, California, 1998. 6

[2] Comaniciu, D., “Robust Detection and Tracking of Human Faces with an Active Camera”, *IEEE Int'l Workshop on Visual Surveillance*, 2000. 7, pp. 11-18

[3] Comaniciu, D., “Kernel-Based Object Tracking”, *IEEE Trans. Pattern Analysis and Machine Intelligence*. vol. 25, No. 5 2003, pp. 564-575

[4] Kailath, T., “The Divergence and Bhattacharyya Distance Measures in Signal Selection”, *IEEE Trans. Communication Technology*. vol. 15, No. 1 1996. 2, pp. 52-60

[5] Comaniciu, D. and Meer, P., “Mean Shift: A Robust Approach Toward Feature Space Analysis”, *IEEE Trans. Pattern Analysis and Machine Intelligence*. vol. 24, No.5 2002. 5

[6] http://ko.wikipedia.org/wiki/HSV_%EC%83%89_%EA%B3%B5%EA%B0%84

김남우(Nam-Woo Kim)

1998년~1999년 한국전자통신연구원 위촉연구원
2000년 목원대학교 공학석사
1999년~2004년 ㈜에스엠아이티 과장
2004년~2006년 ㈜힘스코리아 전임연구원
2006년~2008년 ㈜휴인스 선임연구원
2009년~현재 ㈜파인텔레콤 책임연구원
※관심분야 : SoC설계 및 검증, 임베디드 시스템, 통신용 반도체

허창우(Chang-Wu Hur)

1991년 연세대학교 전자통신공학과 공학박사
1986년~1994년 LG 중앙연구소
1994년 ~ 현재 목원대학교 IT공학부 교수
1996년 ~ 현재 연세대학교 ASIC연구소 자문교수
2005년 호주 Griffith 대학교 초빙교수
2011년 ~ 2012년 한국정보통신학회 회장
※관심분야 : 반도체공학 및 VLSI설계