Journal of the Korea Industrial Information Systems Research Vol. 20 No.2, Apr. 2015 :45-56
http://dx.doi.org./10.9723/jksiis.2015.20.2.045 ISSN:1229-3741

고속 객체 검출을 위한 적분 히스토그램 기반 프레임워크

(Integral Histogram-based Framework for Rapid Object Tracking)

고 재 필[1], 안 정 호[2], 홍 원 기[3]*
(Jaepil Ko, Jung-Ho Ahn, and Won-Kee Hong)

요 약 본 논문에서는 스마트폰 카메라의 객체기반 자동초점 기능을 위해, 움직이는 물체의 고속 추적 방법을 제안한다. 사양이 낮은 플랫폼에서의 비-학습 제약을 고려하여 히스토그램 특징 기반의 슬라이딩 윈도우 검출 기법을 사용한다. 각 부분 윈도우에 대한 히스토그램의 계산 시간문제는 적분 히스토그램을 통해 해결한다. 본 논문에서는 지역적 후보 검출, 적응적 템플릿 크기 방법을 제안한다. 또한 추적 위치의 안정화를 위해 정합 함수에 안정화 항을 추가하는 기법을 제안한다. 자체 수집한 데이터에 대한 실험결과는 PC 환경에서 초당 100 프레임 수준의 높은 처리 속도 달성을 보여주었다.

핵심주제어 : 객체 검출, 객체 추적, 적분 히스토그램, 히스토그램 매칭

Abstract In this paper we propose a very rapid moving object tracking method for an object-based auto focus on a smart phone camera. By considering the limit of non-learning approach on low-performance platforms, we use a sliding-window detection technique based on histogram features. By adapting the integral histogram, we solve the problem of the time-consuming histogram computation on each sub-window. For more speed up, we propose a local candidate search, and an adaptive scaling template method. In addition, we propose to apply a stabilization term in the matching function for a stable detection location. In experiments on our dataset, we demonstrated that we achieved a very rapid tracking performance demonstrating over 100 frames per second on a PC environment.

Key Words : Object Detection, Tracking, Integral Histogram, Histogram Matching

1. 서 론

최근 스마트 폰을 이용한 동영상 촬영이 늘어나고 있는데, 동영상 촬영의 경우 연속적인 자동초점 기능이 중요한 쟁점이다. 일반적으로 자동초점 기능은 화면 중심부의 고정영역에 대한 초점 값을 기준으로 이루어지지만, 스마트폰의 경우 터치 화면을 통해 사용자가 즉각적으로 관심 대상을 지정할 수 있기 때문에, 관심 객체를 기준으로 자동초점이 가능하다. 그러나 스마트 폰을 움직이거나 관심 객체 자체가

* Corresponding Author
이 논문은 금오공과대학교학술연구비에 의하여 지원된 논문임.
Manuscript received February 19, 2015 / Revised April 17, 2015 / Accepted April 17, 2015
1) 금오공과대학교 컴퓨터공학과
2) 강남대학교 컴퓨터공학부
3) 대구대학교 멀티미디어학과, 교신저자(wkhong@daegu.ac.kr)

움직이는 경우, 비디오 영상에서 객체의 위치가 달라지므로 객체 기반의 자동초점을 위해서는 객체추적 기술이 요구된다.

영상인식에서 객체추적의 어려움은 물체의 갑작스런 움직임, 물체나 장면의 패턴변화, 물체의 변형, 다른 물체나 배경에 의한 가림, 카메라 움직임, 조명변화 등 다양한 요인에 기인한다[1]. 스마트 폰과 같은 모바일 플랫폼 사용자 환경은 이러한 요인에 대부분 노출되어 있으며, 연속적인 자동초점 적용을 위해서는 실시간 처리속도가 중요하다.

스마트 폰에서의 자동초점 상황을 고려하면, 사용자가 터치 화면을 통해 비교적 쉽게 검출 영역을 지정할 수 있기 때문에 초기 검출 문제는 고려하지 않아도 되지만, 정밀하게 구분되기 어려운 사용자 지정 객체에 대한 실시간 등록이 필요하므로 일반적으로 성능이 좋은 학습 기반의 추적 알고리즘은 사용할 수 없다.

히스토그램은 다양한 변화를 가진 광범위한 영상을 대상으로 하는 영상검색에서 널리 사용되는 전역 특징으로 심각한 모양변화, 회전변화에 강인하다. 추적 알고리즘 자체는 부드러운 움직임 가정에 벗어나는 상황에 매우 취약하여, 이러한 경우로 볼 수 있는 객체 사라짐에 대처하기 어렵다. 이에 비해 슬라이딩 윈도우 기반의 검출 방법은 이러한 문제점을 감소시킬 수 있다. 슬라이딩 윈도우 기반의 검출 방법은 간단하고 고전적인 방법임에도 최첨단 기술로 간주되고 있으며 최근까지도 대부분의 성공적인 검출 기법들이 이를 기반으로 하고 있다[2]. 이 방법의 단점은 매우 높은 계산 복잡도이다. 영상의 해상도가 인 경우 계산 복잡도는 이다. 이는 해상도가 인 경우 10억 개 이상의 부분 윈도우에 대한 계산을 요구하는 수준이다.

우리는 이러한 문제를 해결하기 이전 연구[3]에서 적분 히스토그램[4]을 적용한 방법을 제안하였다. 적분 히스토그램은 적분영상[5] 아이디어에 기반을 두어 어떤 부분 윈도우의 히스토그램을 히스토그램 빈(bin) 개수의 3배 만큼의 가감을 통해 임의 영역의 히스토그램을 빠르게 계산할 수 있는 기법이다. 우리의 이전 연구에서는 빈의 크기를 달리하여 적은 수의 빈으로 히스토그램을 효과적으로 표현할 수 있는 적응적 빈 방법을 새롭게 제안하여 속도향상을 달성하였다. 본 논문에서는 추가적인 속도향상과 성능향상을 위해 다음 방법을 제안한다. 프레임 전체 영역에 대해 적분 히스토그램을 계산하는 대신 이전 검출 영역에 한정하여 계산하고 템플릿의 크기도 적응적으로 조절할 수 있도록 한다. 또한 안정화 항을 제안하여 추적 위치의 불필요한 변동을 줄인다.

본 논문의 구성은 다음과 같다. 2장에서는 기반 연구로 객체 추적 모델, 적분 히스토그램, 히스토그램 매칭 방법을 설명하고, 3장에서는 적분 히스토그램을 이용한 히스토그램 기반의 고속 검출 방법을 제시한 후 제안하는 속도개선 및 성능향상 방법들을 상세히 설명한다. 4장에서는 자체 수집 데이터에 대한 실험 결과를 분석하고 5장에서는 결론을 기술한다.

2. 객체검출 기반 객체추적

일반적인 객체추적모델은 객체검출확률분포와 이전 프레임에서의 검출위치확률분포들의 곱으로 표현된다. 검출위치확률을 상수로 가정하면 객체추적모델은 객체검출 문제로 간주될 수 있다. 본 장에서는 먼저 일반적인 객체추적모델을 설명하고, 슬라이딩 윈도우 방식의 객체검출 방법에서 객체검출확률분포로 사용될 수 있는 적분히스토그램을 소개한다. 마지막으로 히스토그램 객체검출 기법의 성능에 영향을 주는 히스토그램 비교 방법을 설명한다.

2.1 객체추적 모델

본 절에서는 일반적인 객체추적 모델을 설명한다[6]. 먼저 슬라이딩 윈도우 기반의 객체 검출은 검출확률분포를 기반으로 영상에서 주어진 위치 X에서 X를 중심으로 하는 일정영역에 포함된 특정 픽셀 y과 이를 모두 포함하는 영역 Cx에 해당하는 이미지 패치 I가 찾으려는 물체일 확률이 가장 높은 위치 X*를 찾는 문제로 정의할 수 있다.

$$X^* = argmax_X p(I(y \in C_X)|X) \quad (1)$$

추적 물체의 변형을 고려한 일반적인 추적모델은 추적확률분포에 대한 다음 식으로 모델링 된다. t시간까지의 이미지 패치 집합 I_t가 주어졌을 때 변환(Transform) 파라미터 벡터 Xt가 최대가 되는 파라

미터를 찾는 문제로 정의할 수 있다. 이때 변환은 주로 파라미터가 6개인 어파인 변환(Affine Transform)을 가정한다.

$$X_t^* = \text{argmax}_{X_t} p(X_t \mid \Gamma_t)$$
$$\Gamma_t = \{I_1, \cdots, I_t\} \quad (2)$$

일반적인 추적모델을 직접 계산하는 것이 용이하지 않기 때문에 베이즈 규칙(Bayes'Rule)과 마르코프 모델(Markov Model)을 적용하면 수식 (2)의 분포함수는 이와 비례하는 다음 식으로 유도된다.

$$p(I_t|X_t) \int p(X_t|X_{t-1})\, p(X_{t-1}|\Gamma_{t-1}) dX_{t-1} \quad (3)$$

객체추적모델은 수식 (3)과 같이 검출확률분포와 적분 항의 곱으로 표현된다. 적분 항 내부는 이전 시간의 변환 파라미터가 주어졌을 때 현재 시간의 변환 파라미터에 대한 조건부 확률분포(동적모델)와 이전 시간까지의 추적확률분포의 곱으로 표현된다. 이때 문제를 쉽게 하기 위해 변환 파라미터에 대한 조건부 확률분포는 정규분포를 가정하며, 변환 파라미터 각각은 다시 통계적으로 독립임을 가정한다. 이 경우 변환 파라미터에 대한 조건부 확률밀도는 각 파라미터에 대한 정규분포의 곱으로 간단히 계산할 수 있다.

추적은 이전 위치가 주어졌을 때 이를 평균으로 하여 무작위로 변환 파라미터를 적용하여 후보 영역을 선정한다. 이때 변환 파라미터에 대한 조건부 확률분포가 계산된다. 그리고 후보 영역 각각에 대해 검출확률분포를 계산한다. 계산된 두 확률분포의 곱이 최대가 되는 후보 영역을 추적 영역으로 결정한다. 이전 시간까지의 추적확률분포는 모든 후보에 대해 공통이므로 구현과정에서는 고려하지 않는다.

2.2 적분 히스토그램 (Integral Histogram)

적분 히스토그램은 윈도우 마다 히스토그램 계산을 획기적으로 줄여주어 실시간 객체 검출을 가능하게 한다. 본 절에서는 적분 히스토그램에 대해 간략하게 설명한다.

적분 히스토그램은 적분 영상 아이디어에 기반을 두고 있다. 적분영상은 원점에서 특정 위치까지의 픽셀 합에 대한 영상이다. 이는 영상내의 임의 윈도우 영역에 대한 픽셀 합을 빠르게 계산할 수 있도록 얼굴검출 논문[5]에서 처음으로 소개되었다. 이 방법은 슬라이딩 윈도우 기반의 객체 검출 속도를 획기적으로 높여주었다.

적분 히스토그램[4]은 원점에서 특정 위치까지의 히스토그램을 빈별로 누적시킨 것이다. 적분 영상은 누적할 값이 하나인 반면, 적분 히스토그램은 누적할 값이 빈 개수로 확장된다. 적분 히스토그램의 특정 빈 b의 값은 다음과 같다.

$$H(x^p, b) = \bigcup_{i=0}^{p} Q(f(x^i)) \quad (4)$$

여기서 $H(\bullet)$는 x^p에서의 적분 히스토그램을 나타내며, $Q(\bullet)$는 픽셀 값 $f(x^i)$의 빈(bin) 값을 나타낸다. 그리고 $\bigcup_{i=0}^{p}$는 p 좌표까지의 빈 값 $Q(\bullet)$들의 합을 나타낸다.

적분 히스토그램은 추가되는 픽셀이 하나 생길 때마다 이전에 구해진 적분 히스토그램을 이용하여 빠르게 계산할 수 있다. 구하고자 하는 위치를 기준으로 행, 열로 각각 이전 좌표에서의 적분 히스토그램을 활용한다. 새로 추가되는 픽셀의 적분 히스토그램 $H(x_1, x_2, b)$은 이전 픽셀의 적분 히스토그램으로부터 다음과 같이 계산한다.

$$H(x_1, x_2, b) = H(x_1 - 1, x_2, b)$$
$$+ H(x_1 - 1, x_2 - 1, b)$$
$$- H(x_1 - 1, x_2 - 1, b) + Q(f(x_1, x_2)) \quad (5)$$

각 픽셀 위치에서의 적분 히스토그램이 주어지면 적분 영상과 마찬가지로 다음과 같이 빠르게 특정 영역의 히스토그램을 계산할 수 있다.

$$h(b) = H(x^+, y^+, b) - H(x^-, y^+, b)$$
$$- H(x^+, y^-, b) + H(x^-, y^-, b) \quad (6)$$

Fig. 1과 수식 (6)을 보면 지정영역 R의 빈 b에 대한 히스토그램 $h(b)$를 구하기 위해 R의 우측 하단에서의 적분 히스토그램 $H(x^+, y^+, b)$에서 R왼쪽의 $H(x^-, y^+, b)$과 위쪽의 $H(x^+, y^-, b)$를 빼준다. 그리고 두 개의 적분 히스토그램을 빼고, 중복하여 빼게 되는 $H(x^-, y^-, b)$를 더하면 R의 빈 b에 대한 히스

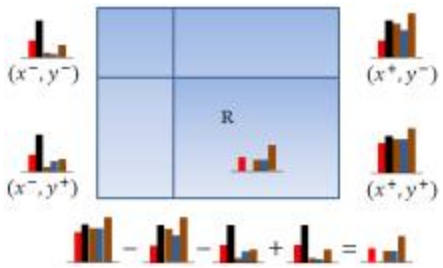


Fig. 1 Computation of histogram using Integral Histogram.

토그램 $h(b)$을 구할 수 있다. 이 과정을 빈의 개수만큼 반복하면 해당 영역의 히스토그램을 구할 수 있다.

2.3 히스토그램 매칭

히스토그램 표현은 전역특징을 표현하므로 회전에 강인하며 오류에 강인하여 영상검색 분야에서 널리 사용되어져 왔다. 슬라이딩 윈도우 기반의 객체 검출에서 히스토그램을 사용하는 경우, 등록한 영상패치의 히스토그램과 슬라이딩 윈도우의 히스토그램을 비교하는 것이 요구된다.

히스토그램 매칭은 두 히스토그램 간의 유사도 또는 거리를 측정하는 방법을 말한다[5]. 측정방법은 크게 두 그룹으로, 대응되는 빈끼리의 차이를 비교하는 Bin-by-Bin 방법과 빈들 간의 교차비교 Cross-Bin 방법으로 구분된다[6]. 일반적으로 전자는 비교적 간단하게 계산할 수 있지만 빈의 크기에 매우 민감하게 반응하는 단점이 있다. Cross-Bin의 단점은 최적의 교차 방안을 찾는 것이 쉽지 않다는 것으로 본 절에서는 최근 발표된 QC[7] 방법을 포함하여 대표적인 Bin-by-Bin 및 Cross-Bin 방법을 소개한다. 아래에 소개되는 매칭 방법에서 비교 대상 히스토그램을 각각 H와 K로, 이들의 i번째 빈의 값을 h_i, k_i로 표현한다.

○ 히스토그램 교차 (Histogram Intersection)

빈 간의 거리를 계산하는 대표적인 방법으로, 대응되는 빈의 값이 작은 쪽을 선택하여 합산하다. 간단하여 가장 널리 사용되는 방법 중 하나이다.

$$d_{\cap}(H,K) = 1 - \frac{\sum_i \min(h_i,k_i)}{\sum_i k_i} \tag{7}$$

히스토그램 교차 방법은 계산이 간단하고 성능이 비교적 우수하여 히스토그램 매칭에서 널리 사용되고 있다[7].

○ Chi-Square (χ^2) Statistic:

두 히스토그램 평균을 구하여 평균과의 거리를 계산한다. Bin-by-Bin 방법으로 분류된다.

$$d_{\chi^2}(H,K) = \sum_i \frac{(h_i - m_i)^2}{m_i}, \quad m_i = \frac{h_i + k_i}{2} \tag{8}$$

○ Quadratic Distance:

컬러영상 검색에서 처음 소개된 방법으로 Cross-Bin 방법으로 분류된다[7].

$$d_A(H,K) = \sqrt{(h-k)^T A (h-k)} \tag{9}$$

여기서, h와 k는 H와 K에 속한 벡터이고, $A = [a_{ij}]$는 빈 i와 j사이의 유사도를 담고 있다. 통산, A는 다음과 같이 계산하였다.

$$a_{ij} = 1 - \frac{d_{ij}}{d_{\max}} \tag{10}$$

여기서, d_{ij}는 빈 i와 j사이의 근간거리(ground distance)를, $d_{\max} = \max(d_{ij})$이다.

○ Match Distance:

누적 히스토그램의 거리를 계산하는 방법으로 Bin-by-Bin 방법으로 분류된다.

$$d_M(H,K) = \sum_i \left| \hat{h}_i - \hat{k}_i \right| \tag{11}$$

여기서 $\hat{h}_i = \sum_{j \leqq i} h_j$는 $\{h_i\}$의 누적 히스토그램이며, $\hat{k}_i = \sum_{j \leqq i} k_j$도 $\{k_i\}$의 누적 히스토그램이다.

○ Earth Mover's Distance (EMD)

EMD는 대표적인 Cross-Bin 거리 측정 방법으로 두 히스토그램을 같게 만드는 데 드는 일의 양을 거리로 사용한다.

$$EMD(P,Q)=\frac{\min(\sum_{i=1}^{m}\sum_{j=1}^{n}f_{ij}d_{ij})}{\sum_{i=1}^{m}\sum_{j=1}^{n}f_{ij}} \quad (12)$$

여기서 P와 Q는 서로 다른 두 개의 히스토그램을 나타내며, f_{ij}는 이동한 빈 값, d_{ij}는 빈 값의 이동거리를 나타낸다. EMD는 영상검색에 처음 사용되었으며 타 방법에 비해 월등히 높은 성능을 보여주었다[9]. 그러나 최적의 빈 간 이동량을 결정하기 위해서는 시간 복잡도가 높은 최적화 문제를 풀어야 한다. 최근 계산시간을 단축시킨 논문[10]이 발표되었으나 여전히 타 방법에 비해 계산시간이 많이 걸린다.

○ Quadratic-Chi Square Distance (QC)

QC 방법은 Cross-Bin 거리 측정 방법으로 Chi-Square 방법과 Quadratic 방법의 일반화 형태를 갖는다. P={Pi}와 Q={Qi} (i = 1, ... N)를 총 면적이 1인 정규화된 히스토그램이라 하자. N×N 행렬 A는 비음수 대칭 행렬로 빈 간의 유사도를 나타내며 모든 첨자 i에 대하여 Aii ≥ Aij (j = 1, ..., N)고 가정하자. 정규화 요소 m ∈ [0, 1)에 대하여 Quadratic-Chi(QC)는 다음과 같이 정의된다.

$$QC_m^A(P,Q)=\sqrt{\sum_{i=1}^{N}\sum_{j=1}^{N}\left(\frac{(P_i-Q_i)}{\left(\sum_{c=1}^{N}(P_c+Q_c)A_{ci}\right)^m}\right)\times\left(\frac{(P_j-Q_j)}{\left(\sum_{c=1}^{N}(P_c+Q_c)A_{cj}\right)^m}\right)A_{ij}} \quad (13)$$

이를 제곱하고 이중 합을 정리하면 다음과 같다.

$$\left(QC_m^A(P,Q)\right)^2=\sum_{i=1}^{N}\left(\frac{(P_i-Q_i)}{\left(\sum_{c=1}^{N}(P_c+Q_c)A_{ci}\right)^m}\times\sum_{j=1}^{N}\frac{(P_j-Q_j)A_{ij}}{\left(\sum_{c=1}^{N}(P_c+Q_c)A_{cj}\right)^m}\right)=\sum_{i=1}^{N}(B_i\,C_i) \quad (14)$$

여기서 $B_i=\frac{(P_i-Q_i)}{\left(\sum_{c=1}^{N}(P_c+Q_c)A_{ci}\right)^m}$,

$C_i=\sum_{j=1}^{N}\frac{(P_j-Q_j)A_{ij}}{\left(\sum_{c=1}^{N}(P_c+Q_c)A_{cj}\right)^m}$ 이다.

이 계산에서 0/0 = 0 으로 정의한다. 행렬 A가 positive-semidefinite일 때는 근호 안의 값이 늘 양수이나, 그렇지 않을 경우 근호 안의 값이 음수가 될 수 있다. 근호 안의 값이 음수일 때는 거리 QCmA(P, Q) 값은 0으로 정의한다. 이는 가장 최근에 발표된 방법[8]으로 논문에서는 EMD보다 우수한 실험 결과를 제시하였다.

3. 제안하는 고속 객체 검출 프레임워크

본 장에서는 적분 히스토그램 기반 객체검출에서 검출속도향상을 위한 제안 기법을 소개한다. 먼저 전체 검출 프레임워크를 설명하고, 속도향상을 위해 제안된 적응적-빈, 히스토그램의 지역적 계산, 적응적 크기의 템플릿 방법을 차례로 설명한다. 또한, 검출 위치의 흔들림 문제를 완화하기 위한 안정화 항을 소개한다.

3.1 적분 히스토그램기반 고속 객체 검출 프레임워크

적분 히스토그램을 이용한 고속 검출은 Fig. 2와 같이 추적 객체의 등록과 연속 프레임에서의 검출로 구성된다. 등록단계에서는 사용자의 입력을 받아 추적하고자 하는 객체를 포함하는 사각형 영역에 대한 히스토그램을 계산하여 저장한다. 이때 RGB 컬러를

YCbCr 컬러로 변경하여 히스토그램을 계산한다. 검출단계에서는 매 프레임별로 컬러변환을 수행하여 적분 히스토그램을 계산하고 슬라이딩 윈도우 기법을 적용하여 객체의 위치를 계산한다. 우리의 이전 연구[3]과는 달리 추가적인 처리속도 향상을 위해 현재 프레임에서의 검출위치 및 템플릿 크기를 다음 프레임에서의 적분 히스토그램 계산 영역을 제한하는데 사용하는 방법을 제안한다.

3.2 컬러변환 및 히스토그램 차원

스마트 기기로부터 획득되는 RGB 영상은 각 채널에 휘도성분이 들어있어 이들로부터 직접 계산한 히스토그램은 조명변화에 민감하므로 휘도와 색차로 구성된 YCbCr 컬러로 변경한다. Y는 RGB 성분의 가중치 평균으로 계산되고, Cb 및 Cr은 각각 (B-Y)와 (R-Y)를 기준으로 계산된다.

RGB에서 YCbCr로의 변환은 식 (15)와 같다.

이때 Y는 0~255 범위의 값을, Cb 및 Cr은 -128~128 범위의 값을 갖는다. 컬러 채널은 3개로 구성되므로 이들에 대한 히스토그램 계산 방법은 3가지로 분류된다. 첫째, 각 채널별로 3개의 히스토그램을 계산하는 방법, 둘째 3채널을 묶어 하나의 3D 히스토그램을 계산하는 방법, 셋째 1D과 2D 히스토그램을 조합하는 방법으로 나뉜다. 세 번째 방법은 Y채널에 대한 1D 히스토그램과 Cb 및 Cr 채널에 대한 2D 히스토그램을 사용한다.

3.3 적응적-빈(Adaptive-Bin) 기반의 적분 히스토그램

히스토그램 매칭의 성능 및 처리속도는 히스토그램 빈의 크기에 민감하다. 우리의 이전 연구[3]에서는 빈의 개수가 정해졌을 때 빈에 해당하는 픽셀의 수가 동일하도록 빈을 나눈 것으로 일종의 적응적 빈 방법을 제안하여, 적은 수의 빈으로 히스토그램을 효과적으로 표현하여 처리속도 향상을 가져올 수 있었다.

3.4 적분 히스토그램의 지역적 계산 및 적응적 크기의 템플릿을 이용한 속도 향상

등록단계에서 설정한 객체의 크기는 검출단계에서는 달라지기 때문에 검출단계에서는 슬라이딩 윈도우의 크기를 달리해 가며 히스토그램을 계산한다. 일반적으로 피라미드 영상을 통해 입력 영상의 크기를 달리하지만 제안 방법에서는 적분 히스토그램의 특성을 고려하여 슬라이딩 윈도우의 크기를 달리하여 피라미드 영상 계산을 피할 수 있다.

슬라이딩 윈도우 마다 히스토그램을 계산하여 등록단계에서 저장하여 둔 템플릿의 히스토그램과 비교하여 거리맵을 계산한다. 거리맵은 슬라이딩 윈도우의 크기별로 반복 계산되어 이 중 가장 거리가 짧은 위치를 검출 위치로 결정한다. 슬라이딩 윈도우의

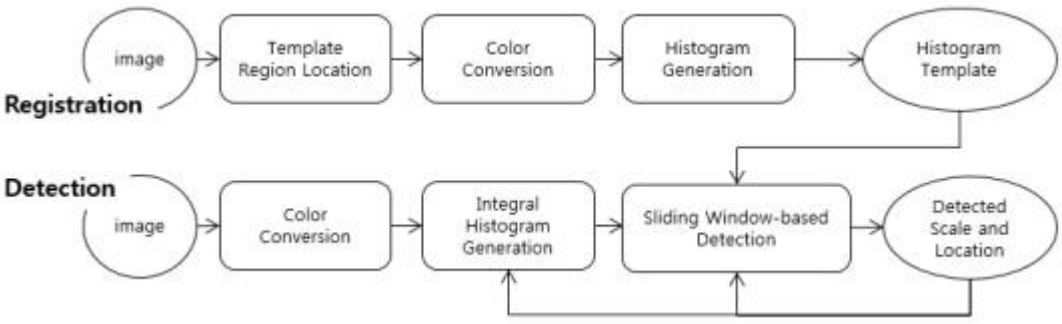


Fig. 2 Flow of Registration and Detection.

$$\begin{aligned} Y &= 0.299 \times R + 0.587 \times G + 0.114 \times B \\ Cb &= -0.16874 \times R - 0.33126 \times G + 0.5 \times B \\ Cr &= 0.5 \times R - 0.41869 \times G - 0.08131 \times B \end{aligned} \tag{15}$$

크기는 템플릿의 크기를 기준으로 0.5~1.5 사이의 5단계로 조절한다. Fig. 3은 5단계의 슬라이딩 윈도우 크기에 따른 거리맵과 최종 검출 위치를 보여준다.

슬라이딩 윈도우는 영상의 해상도가 $n \times n$인 경우 최대 계산 복잡도는 $O(n^4)$이다. 이를 줄이기 위한 일반적인 방법은 모든 픽셀을 검사하지 않고 몇 픽셀씩 건너뛰는 것이다. 이는 검출 성능저하를 가져올 수도 있고 검사할 템플릿의 크기가 많아질수록 효용성이 떨어진다.

추적모델의 빠른 처리속도는 부드러운 움직임 가정에 기반 하여 이전에 찾은 위치를 기준으로 그 주변에서만 후보를 선정하기 때문이다. 추적모델에서 변환 파라미터에 대한 분포인 동적모델이 이를 담당한다. 본 논문에서는 동적모델의 취지를 고려하여 이전에 찾은 위치를 기준으로 후보영역을 선정하고 후보영역에 한정해서 적분 히스토그램을 계산한다.

Fig. 4는 후보영역의 1.5배 영역에 한정해서 적분 히스토그램을 계산하고 이를 이용한 거리맵을 보여준다. 이를 통해 전체 영역에 대해 계산한 것과 같은 성능을 유지하면서 동일한 검출 결과를 얻을 수 있다.

추가적인 처리속도 향상을 위해 5개의 거리맵 계산 대신, 검출된 거리맵과 이의 주변 거리맵을 포함하여 총 3개의 거리맵만 계산한다. 주변 거리맵을 위한 슬라이딩 윈도우의 크기는 템플릿의 크기에 일정 비율을 곱하여 계산한다. 여기서 일정 비율은 우리가 5단계의 거리맵에 대해 미리 정해 둔 0.5, 0.8, 1.0, 1.2, 1.5에서 해당 거리맵에 +0.3, -0.3을 각각 더해서 결정한다.

3.5 검출위치 변동감소를 위한 안정화 항

히스토그램 기반의 거리맵에서 최소값의 위치는 프레임 차이에 민감하게 반응한다. 따라서 실제로 물체의 위치 변화가 없는 경우에도 검출 좌표 값은 요동치는 경우가 발생한다. 이를 안정화하기 위해 다음

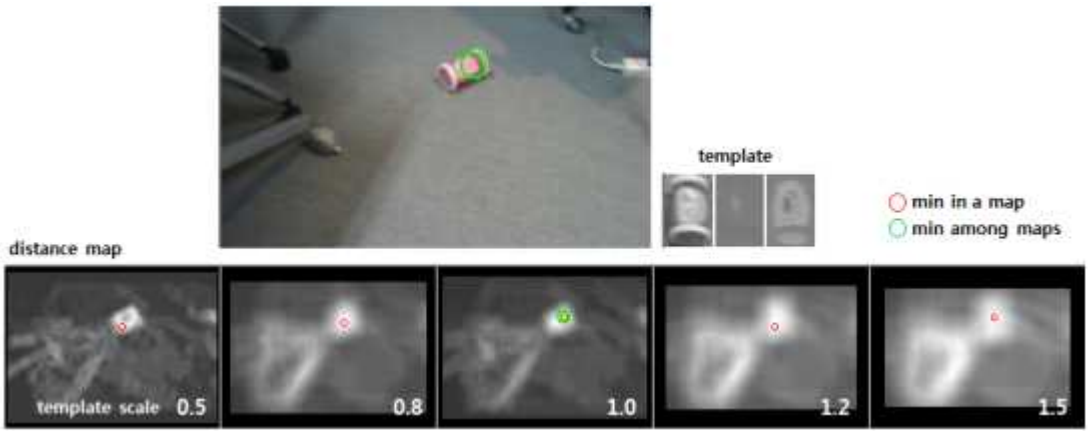


Fig. 3 Example of detection results and distance maps by the sliding-window size.

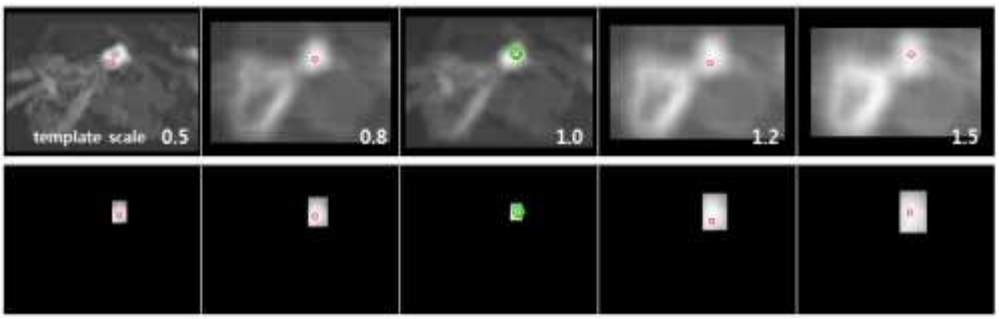


Fig. 4 Comparison of global and local distance maps.

과 같이 히스토그램 매칭을 위한 거리 함수에 안정화 항을 추가한다.

$$d = D(P,Q) + \alpha \left| \boldsymbol{x_p} - \boldsymbol{x} \right| + \beta \left| s_p - s \right| \qquad (16)$$

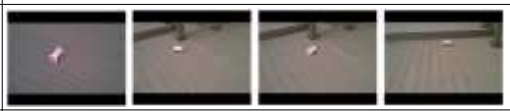
여기서 $D(P,Q)$는 히스토그램 P, Q간의 거리, $\boldsymbol{x_p}$는 이전 프레임에서 찾은 위치, $\boldsymbol{x}$는 현재 프레임에서 D의 최소값 위치, s_p는 이전 프레임에서의 템플릿 스케일, s는 현재 최소값을 갖는 템플릿 스케일을 나타낸다. α, β는 각각 위치 및 템플릿 스케일에 대한 안정화 계수로 본 논문에서는 각각 0.0001, 0.1의 값으로 설정하였다.

4. 실험결과

4.1 실험데이터 및 실험방법

실험을 위해 Intel i5-2.67 GHz PC 환경에서 C++ 언어를 사용하여 구현하였다. 실험 데이터는 Table 1과 같이 크기, 회전, 조명, 가려짐 등을 포함하는 사람 및 물체에 대한 320x240해상도의 다섯 가지 동영상을 사용하였다. 취득한 영상은 실제 적용하고자 하는 스마트 폰 카메라를 이용하여 인위적인 설정에 따라 다음과 같은 특징을 갖는다. M1은 조각모양의 물체가 지그재그로 움직이며 크기가 변하며 물체의 앞뒤가 바뀌는 경우를 포함한다. M2는 원통형의 물체가 크기가 변하면서 물체 자체의 회전과 한 점을 중심으로 원을 그리듯이 회전하는 경우를 포함한다. M4는 사람이 다양한 방향으로 움직여 가까워지다가 멀어지는 것을 반복하며 다양한 크기로 변화하는 경우를 포함하며 제자리에서 회전하는 경우를 포함한다. M5는 인형에 약간의 변형이 가해지며 움직이는데 비교적 빠르게 크기와 위치가 변화하는 경우를

Table 1 Test image sequences.

image	total frames	sample snapshots	note
M1	1,658		◦ view points variations
M2	1,697		◦ scale and rotation variations
M3	1,646		◦ out of scene object
			◦ scale and lightness variations
M4	2,217		◦ a big change of scales
			◦ view points variations
M5	1,346		◦ scale and location variations
			◦ partial occlusion variations

포함하며, 인형의 절반 이상이 화면 밖으로 사라진 경우 및 회전하여 뒷모습이 나타나거나 또는 위아래가 뒤바뀐 경우를 포함한다.

실험방법으로 Fig. 5와 같이 첫 프레임에서 추적 객체를 사용자가 등록한 후, 추적을 시작하면 연속되는 프레임에서 객체 추적을 수행하게 된다. 추적성능 계산 시 앞서 제시한 바와 같이 객체 위치에서의 초점 조절이 목적이므로 각 프레임마다 Fig. 6의 왼쪽 모습과 같이 각 프레임에서 그림과 객체에 포함된 좌표 추정 시 추적 성공, 오른쪽모습과 같이 벗어날 시에는 추적 실패로 기록하여 추적 성능을 계산한다.

Fig. 5 Example of object registration.

Fig. 6 Example of detection success or failure.

4.2 실험결과

□ 적분 히스토그램의 지역적 계산

컬러채널에 대해 4개의 빈을 갖는 적응적-빈 히스토그램 계산방법을 적용하고, 적분 히스토그램에 대한 전역적 및 지역적 계산 방법을 다르게 하여 추적성능과 처리속도 실험한 결과를 다음 Table 2 및 Fig. 7에 제시하였다.

Table 2 Tracking and processing time performance on the histogram computation methods.

image	integral histogram methods	tracking performance		FPS
		%	success/failure # of frames	
M1	local	99.52	1650/1658	58
	global	99.52	1650/1658	**68**
M2	local	99.94	1696/1697	63
	global	99.94	1696/1697	**115**
M3	local	99.88	1644/1646	63
	global	99.88	1644/1646	**123**
M4	local	99.73	2211/2217	62
	global	99.73	2211/2217	**108**
M5	local	99.03	1333/1346	53
	global	99.03	1333/1346	**57**

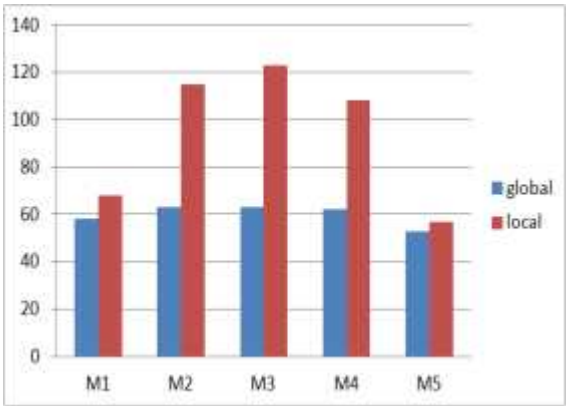


Fig. 7 Tracking and processing time performance on the histogram computation methods.

Table 2와 Fig. 7에서 보듯이 추적성능에는 변화가 없으며, 지역적으로 후보영역에 대해서만 적분영상을 계산함으로써 처리 속도가 향상되었다. 제안 방법이 M1, M5의 경우에는 추적객체의 크기가 크기 때문에 7~17%의 다소 적은 속도 향상을 보이지만 추적객체의 크기가 작은 M2, M3, M4의 경우 74~95%의 매우 높은 처리 속도의 향상을 확인하였다.

□ 적응적 크기의 템플릿

컬러채널에 대해 4개의 빈을 갖는 적응적-빈 히스토그램과 지역적 적분 히스토그램을 적용하고, 5개의 거리맵 계산하는 경우와 3개의 거리맵 만을 계산하는 경우에 대하여 추적성능과 처리속도 실험결과를 다음 Table 3 및 Fig. 8에 제시하였다.

Table 3 Tracking performance and processing time on the template scales

image	template scales	tracking performance		FPS
		%	success/failure # of frames	
M1	fixed	99.52	1650/1658	68
	adaptive	99.22	1645/1658	72
M2	fixed	99.94	1696/1697	115
	adaptive	99.94	1696/1697	117
M3	fixed	99.88	1644/1646	123
	adaptive	99.64	1640/1646	124
M4	fixed	99.73	2211/2217	108
	adaptive	94.14	2087/2217	111
M5	fixed	99.03	1333/1346	57
	adaptive	99.41	1338/1346	60

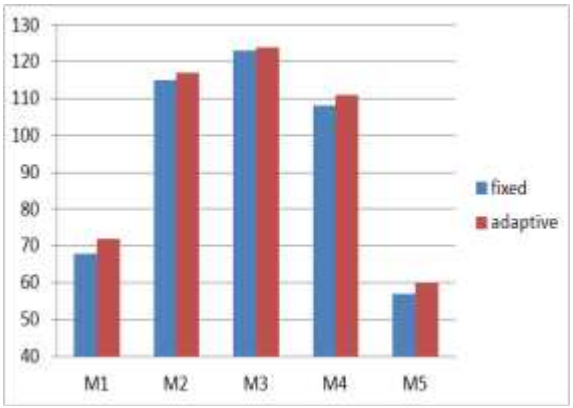


Fig. 8 Tracking performance and processing time on the template scales.

제안한 적응적 템플릿 크기를 적용한 경우 추적성능은 대등하면서 처리 속도는 1~5%의 향상을 보였다. 거리맵 계산을 5개에서 3개로 줄이는 경우는 속도개선이 미약한데, 이것은 이미 적분 히스토그램의 지역적 계산에 의해 거리맵 당 계산량이 충분히 줄어들었기 때문인 것으로 보인다.

□ 히스토그램 매칭 방법

컬러채널에 대해 4개의 빈을 갖는 적응적-빈 히스토그램과 지역적 적분 히스토그램, 적응적 크기의 템플릿을 적용하고, Intersection 방법과 Quadratic-Chi 방법을 비교하기 위하여 히스토그램 매칭 방법을 달리하였다. 관련 추적성능과 처리속도에 대한 실험결과를 Table 4 및 Fig. 9에 제시하였다.

Table 4 Tracking performance and processing time on the histogram matching methods.

image	histogram matching	tracking performance		FPS
		%	success/failure # of frames	
M1	Intersection	99.22	1645/1658	72
	Quadratic-Chi	98.43	1632/1658	49
M2	Intersection	99.94	1696/1697	117
	Quadratic-Chi	99.94	1696/1697	111
M3	Intersection	99.64	1640/1646	124
	Quadratic-Chi	99.57	1639/1646	120
M4	Intersection	94.14	2087/2217	111
	Quadratic-Chi	87.87	1948/2217	101
M5	Intersection	99.41	1338/1346	60
	Quadratic-Chi	99.48	1339/1346	37

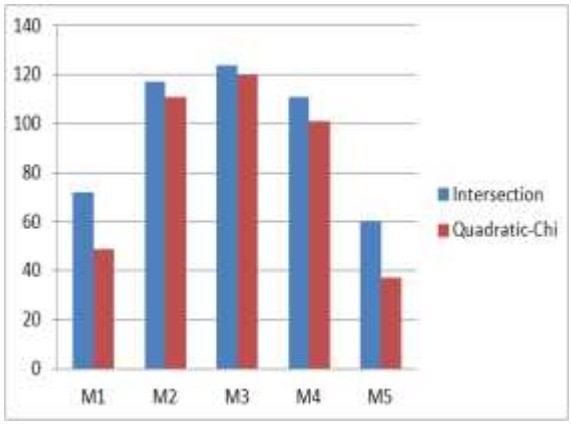


Fig. 9 Tracking performance and processing time on the histogram matching methods.

처리속도는 Quadratic-Chi 방법이 Intersection 방법보다 계산량이 많기 때문에 추적 속도가 낮아졌다. 추적객체의 크기가 작은 M2, M3, M4의 경우 3~9%, 크기가 큰 M1, M5의 경우 46~62%로 속도저하를 보였다. 추적 성능 또한 대등하거나 M4의 경우 오히려 Intersection 방법이 더 높게 나타났다.

5. 결 론

본 논문에서는 모바일 플랫폼 환경을 고려한 슬라이딩 윈도우 기반의 휘도성분과 색차성분의 히스토그램으로 표현된 추적 객체의 검출을 통한 추적 방법을 제안하였다. 우리의 이전연구[3]에서는 적분 히스토그램 프레임워크에서, 적응적-빈 방법을 제안하

여 추적성능 향상과 이를 통한 빈의 개수 감소를 통해 추가적인 처리속도 향상을 달성하였다. 또한 적분 히스토그램의 지역적 계산, 적응적 크기의 템플릿, 검출위치 변동감소를 위한 안정화 항을 새롭게 추가 제안하여, 성능은 유지하면서 이전의 추적속도인 초당 63 프레임 수준에서 초당 최대 124 프레임 수준으로 2배 향상시켰다. 마지막으로, Cross-Bin 방법으로 주목받은 QC 히스토그램 매칭은 본 실험에서는 전통적인 Intersection 방법보다 우수한 결과를 보여주지 못했다. 본 논문에서 제안하는 방법은 기존의 고속 객체 검출 및 추적 [11-13] 방법에 적용 가능하다.

References

[1] A. Yilmaz, O. Javed, and M. Shah, "Object Tracking: A Survey," ACM Computing Surverys, Vol. 38, No. 4, 2006.

[2] C. Lampert, M. Blaschko, T. Hofmann, "Beyond Sliding Windows: Object Localization by Efficient Subwindow Search," Proc. of Int'l Conf. on CVPR, 2008.

[3] J. Ko, J. Ahn, Y. Lee, and S. Kim, "A Histogram-based Object Tracking for Mobile Platform," Journal of Korea Multimedia Society, Vol. 15, No. 8, 2012.

[4] F. Porikli, "Integral Histogram: A Fast Way to Extract Histograms in Cartesian Spaces," Proc. of Int'l Conf. on CVPR, pp.829~836, 2005.

[5] P. Viola and M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features", Proc. of Int'l Conf. on CVPR, pp.511~518 2001.

[6] D. Ross, J. Lim, R. Lin, and M. Yang, "Incremental Learning for Robust Visual Tracking", International Journal of Computer Vision, 2007.

[7] Y. Rubner, C. Tomasi, and L. J. Guibs, "The Earth Mover's Distance as a Metric for Image Retrieval," Int'l Journal of Computer Vision, Vol. 40, pp.99-121, 2000.

[8] M. Swain and D. Ballard, "Color Indexing," Int'l Journal of Computer Vision, Vol. 7, No. 1, pp. 11-32, 1991.

[9] O. Pele and M. Werman, "The Quadratic-Chi Histogram Distance Family," Proc. of European Conf. on Computer Vision, 2010.

[10] O. Pele and M. Werman, "A Linear Time Histogram Metric for Improved SIFT Matching," Proc. of European Conf. on Computer Vision, 2008.

[11] In-Tae Jang, Dong-Woo Kim, Young-Jun Song, Hyeok-Bong Kown, and Jae-Hyeong Ahn, "Real Time Object Tracking Method using Multiple Cameras," Journal of the Korea Industrial Information System Society, Vol. 14, No. 4, pp. 51-59, 2012.8.

[12] Jong-Ho Kim, Sang-Kyoon Kim, Goo-Seun Hang, Sang-Ho Ahn, and Byoung-Doo Kang, "An Object Detection and Tracking System using Fuzzy C-mean and CONDENSATION," Journal of the Korea Industrial Information System Society, Vol. 16, No. 4, pp. 87-98, 2011.12.

[13] Se-Hyun Park, Kyung-Su Kwon, Eun-Yi Kim, Hang-Joon Kim, "ACMs-based Human Shape Extraction and Tracking System for Human Identification," Journal of the Korea Industrial Information System Society, Vol. 12, No. 5, pp. 39-46, 2007.12.

고 재 필 (Jaepil Ko)

- 정회원
- 연세대학교 전산학과 학사
- 연세대학교 컴퓨터과학과 석사
- 연세대학교 컴퓨터과학과 박사
- 금오공과대 컴퓨터공학과 교수
- 관심분야 : Object detection, Pattern recognition, Image processing

안 정 호 (Jung-Ho Ahn)

- 정회원
- 연세대학교 수학과 학사
- 연세대학교 수학과 석사
- 연세대학교 컴퓨터과학과 박사
- 강남대학교 컴퓨터공학부 교수
- 관심분야 : Object detection, Pattern recognition, Image processing

홍 원 기 (Won-Kee Hong)

- 정회원
- 연세대학교 전산과학과 학사
- 연세대학교 컴퓨터과학과 석사
- 연세대학교 컴퓨터과학과 박사
- 대구대학교 멀티미디어공학과 교수
- 관심분야 : Vehicular Ad hoc network, Realtime embedded OS, Wireless sensor network