공학석사학위논문

ASIFT 기반 객체 인식의 정확도 향상에 관한 연구

A Study on Accuracy Improvement for Object Detection with ASIFT

2015년 2월

서울대학교 대학원

전기 컴퓨터 공학부

이 주 형

ASIFT 기반 객체 인식의 정확도 향상에 관한 연구

A Study on Accuracy Improvement for Object Detection with ASIFT

지도교수 김 재 하

이 논문을 공학석사 학위논문으로 제출함

2015 년 2 월

서울대학교 대학원

전기 컴퓨터 공학부

이 주 형

이주형의 공학석사 학위논문을 인준함

2015 년 2 월

위 원 장 : 최 진 영 (인)

부위원장 : 김 재 하 (인)

위 원 : 이 혁 재 (인)

초록

최근에 디지털 기기의 증가와 함께 다양한 촬영기기들의 도입으로 인하여 많은 영상과 이미지들이 유통되고 다루어지고 있다. 이러한 영상과 이미지의 증가는 자연스럽게 다양한 부분에서 이미지 인식, 분류 시스템의 필요성과 수요의 증가를 가져왔다.

Local feature를 이용한 영상인식은 local feature detector들이 가지는 여러 장점에 의해 많이 사용되어 왔다. 이러한 local feature detector중 SIFT는 뛰어난 성능으로 인해 많이 사용된다. 하지만 SIFT는 연산량이 높고 검출하려는 객체의 기울임 정도에 따라 그 성능이 떨어진다. 이에 affine SIFT(ASIFT)가 제안되었다. ASIFT는 이미지로부터 다시점의 여러 이미지들을 생성한 뒤 SIFT를 적용하여 fully affine invariant 한 SIFT를 구현하였다.

본 논문에서는 ASIFT 알고리즘을 이용하여 이를 실제 실생활의 영상에서 객체 검출을 시도해 본다. 이때 ASIFT가 만드는 많은 keypoint로 인한 오매칭과 오검출을 줄이는 방법을 제안한다. 제안하는 방법에는 오검출을 제거할 수 있는 distance filter와 angle filter 그리고 미검출을 재검출하는 local object update 방식의 재검출 방법을 제안한다. Distance filter는 객체를 구성하는 각 점 사이의 거리와 상대적인 위치를 이용하여 오검출을 제거한다. Angle filter는 검출된 객체의 각을 이용하여 오검출을 제거한다. 미검출시 이전 영상에서 검출된 object를 이용 재검출을 시도하는 local object

update 방식으로 객체의 검출 율을 증가시킨다.

이러한 방법을 통해 기존 검출에 비해 잘못된 검출 결과를 0%에 가깝게 유지하고 검출률은 90% 근처로 유지할 수 있다.

주요어 : SIFT, ASIFT, Object detection

학 번 : 2013-20857

목차

표 목차

그림 목차

제 1 장 서론

1.1 연구 배경 및 목표

최근에 디지털 기기의 증가와 함께 다양한 촬영기기들의 도입으로 인하여 많은 영상과 이미지들이 유통되고 다루어지고 있다. 이러한 영상과 이미지의 증가는 자연스럽게 다양한 부분에서 이미지 인식, 분류 시스템의 증가를 가져왔다.

Local feature를 이용한 영상 인식, 분류는 local feature detector들이 가지는 view point change, noise에 강인한 특성 덕분에 영상 인식에서 많이 사용되고 있다.

Scale Invariant Feature Transform(SIFT)는 이미지내에서 object 검출에 사용되는 local feature detector중 하나이다. SIFT는 scale, rotation에 강인한 특성 덕분에 많은 object 검출 상황에서 사용되고 있다. 그러나 SIFT는 찾고자 하는 object와 검출하려는 이미지 사이에 각도가 다른 경우 object 검출률이 떨어지는 단점이 있다. Affine SIFT(ASIFT) 알고리즘은 이러한 문제를 해결하기 위해 object 와 object를 찾으려는 이미지를 affine transform을 통해 다 시점의 이미지들을 생성하여 각 이미지들끼리 SIFT 매칭을 시도함으로써 각도차이가 있는 object도 검출할 수 있는 방법이다.

본 논문에서는 정해진 객체의 오검출시 객체의 특성을 이용하여 검증할 수 있는 distance and angle filter를 제안한다. 또한 이미지의 blur와 filtering으로 인해 미검출 되는 경우에 미리 검출하여 저장한 local object 이미지를 이용하여 ASIFT 알고리즘으로 미검출 되는 이미지에서 객체를 다시 검출하는 방법을 제안한다.

1.2 논문 구성

본 논문의 구성은 다음과 같다. 2장에서는 SIFT 알고리즘과 ASIFT 알고리즘을 살펴보고 한계와 응용방법에서 대해서 다룬다. 3장에는 ASIFT 알고리즘을 이용 동영상에서 객체 검출을 시도할 때 어려움을 알아보고 이를 보정하기 위한 Distance and Angle filter와 Local object update 방식의 재검출을 제안한다. 4장에는 제안하는 방법의 실험 결과를 보이고 마지막으로 5장에서 결론을 맺는다.

제 2 장 관련연구

2.1 SIFT 알고리즘

이미지 매칭은 컴퓨터 비전 분야에서 기초적인 부분으로 영상분석, 3D sturture 구성, tracking, image stitching등에 자주 사용된다[1][2]. 이미지 매칭을 위해 이미지 내에서 corner 혹은 edge와 같이 특징이 명확한 점을 이용하여 이미지 매칭을 하는 방법을 feature detector를 이용한 이미지 매칭이라고 한다. Feature에는 local feature와 global feature가 있다. Local feature detector에는 SURF[3]를 비롯해 FAST[4] 등이 있으며 각각의 장점과 단점을 가지고 있다. Lowe[5]가 제안한 SIFT(Scale Invariant Feature Transform) 는 이미지의 scale, rotation에 강인한 특성과 타 feature들 보다 우수한 성능으로[6] 인해 3d reconstruction, panorama, image stabilization에 다양하게 사용되고 있다[7][8][9].

Scale Invariant Feature Transform 알고리즘은 크게 아래의 4개의 stage로 이루어 진다.

a. Scale-space extrema detection
b. Keypoint localization
c. Orientation assignment

d. Keypoint descriptor

아래의 장들에서 SIFT의 구현 원리를 알아본다.

2.1.1 Scale-Space Extrema Detection

먼저 제대로 된 scale invariant한 keypoint를 추출하기 위해서 SIFT알고리즘은 입력된 이미지 내에서 Gaussian function[10]을 이용 extrema를 먼저 추출한다. 이를 위해 아래 식 2-1 과 같이 input 이미지 $I(x, y)$에 대하여 $G(x, y, \sigma)$의 convolution을 취해 결과 $L(x, y, \sigma)$를 얻는다.

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{2-1}$$

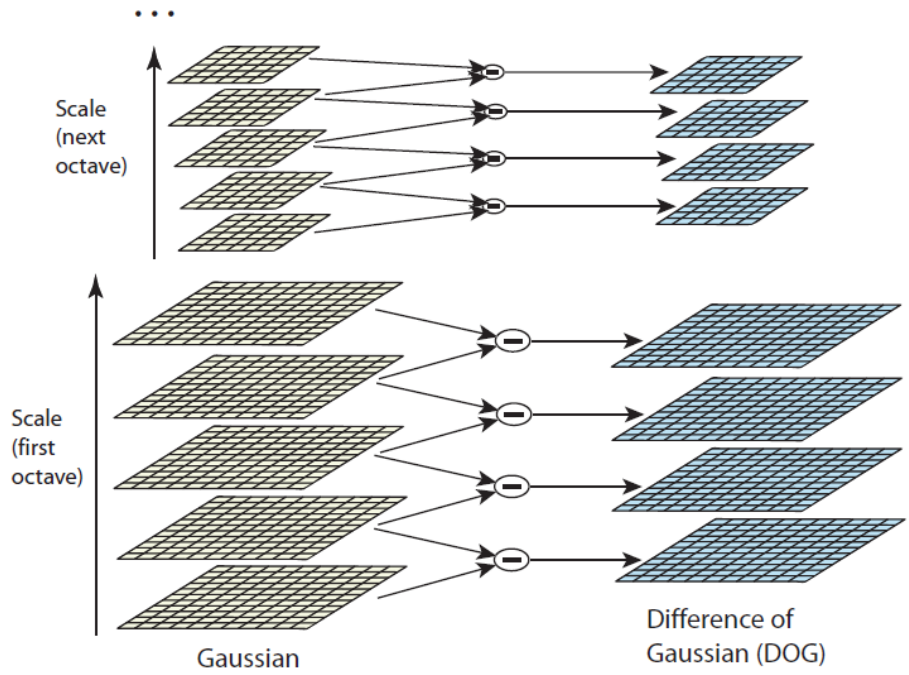
이 때 Gaussian function $G(x, y, \sigma)$는 아래의 식 2-2로 표현할 수 있다.

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2-2}$$

Scale에서 stable한 extrema를 얻기 위해 위와 같이 이미지를 서로 다른 σ 값을 이용하여 Gaussian convolution한 결과의 차를 이용하여 Difference-of-Gaussian(DOG) 을 구한다. 이는 아래와 같이 표현 할 수 있다.

$$\mathrm{D}(\mathrm{x},\mathrm{y},\ \sigma) = \big(G(x,y,k\sigma) - G(x,y,\sigma)\big) * I(x,y) \qquad (2\text{-}3)$$
$$= L(x,y,k\sigma) - L(x,y,\sigma)$$

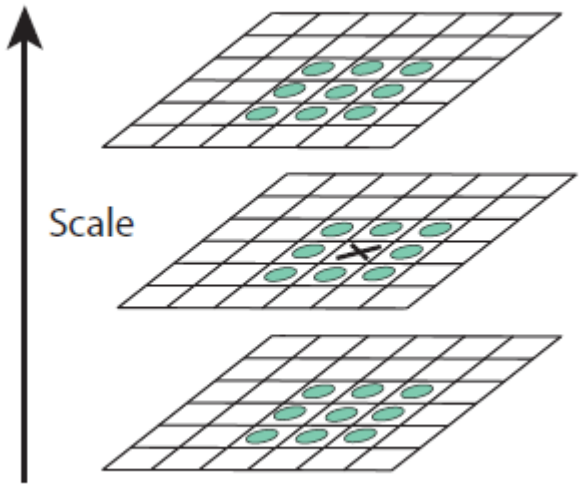
DOG를 만들 때 scale invariant하게 feature를 구하기 위해 octave를 다르게 한다. 3개의 octave를 가지고 각 octave에서는 5개의 다른 σ를 이용하여 DOG를 생성한다.



[그림 2. 1 DOG 생성 방법[5]]

이렇게 구한 DOG를 이용하여 각 pixel에서 extrema를 찾는다. 각 pixel과 같은 DOG안에서 주변 8 pixel과 위 아래의 DOG의 각각 9개의 pixel를 비

교한다. 그 pixel 값이 비교 대상의 26개의 pixel의 비해 maximum 혹은 minimum일 경우 extrema로 추출한다.



[그림 2. 2 Extrema 추출 방법[5]]

2.1.2 Keypoint Localization

DOG를 이용하여 extrema를 찾은 후 찾은 keypoint를 localization을 통해 low contrast를 가진 keypoint와 edge에서 추출된 keypoint를 제거한다. 이는 keypoint의 stability를 유지하기 위한 작업이다.

먼저 low contrast를 가지는 keypoint를 제거한다. Low contrast 가지는 keypoint는 noise에 취약하다.

$$\mathrm{D(x)} = D + \frac{\partial D^T}{\partial x} x + \frac{1}{2} x^T \frac{\partial^2 D}{\partial x^2} x \tag{2-4}$$

$$\hat{x} = -\frac{\partial^2 D^{-1}}{\partial x^2} \frac{\partial D}{\partial x} \tag{2-5}$$

를 대입하면

$$\mathrm{D}(\hat{x}) = D + \frac{1}{2} \frac{\partial D^T}{\partial x} \hat{x} \tag{2-6}$$

를 얻게 된다. [5]에서는 $|D(\hat{x})| < 0.03$ 인 extrema는 low contrast를 가지는 경우로 보고 모두 제거하였다.

또한 edge responses를 가지는 keypoint도 제거해야 한다. DOG를 통해 구한 keypoint들은 edge 주변에서 keypoint가 많이 나오게 된다. 이러한 keypoint중에 잘 추출되지 않은 keypoint는 noise에 약한 특성을 보이게 되어 stability를 떨어뜨리게 되므로 제거해야 한다.

$$\mathrm{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix} \tag{2-7}$$

$$\begin{aligned} \mathrm{Tr(H)} &= D_{xx} + D_{yy} = \alpha + \beta \\ \mathrm{Det(H)} &= D_{xx} D_{yy} - \left(D_{xy}\right)^2 = \alpha\beta \\ \frac{\mathrm{Tr(H)}^2}{Det(H)} &= \frac{(\alpha + \beta)^2}{\alpha\beta} = \frac{(\mathrm{r}+1)^2}{r} \end{aligned} \tag{2-8}$$

$$\frac{\mathrm{Tr(H)}^2}{Det(H)} < \frac{(\mathrm{r}+1)^2}{r} \tag{2-9}$$

위의 식을 이용하여 r = 10으로 놓고 모든 edge에 대해 위의 비율을 계산하여 keypoint를 제거하였다.

2.1.3 Orientation assignment

각각의 keypoint에 대하여 orientation에 대한 정보를 부여해야 orientation invariant한 feature를 얻을 수 있다. SIFT 알고리즘에서는 Gaussian smoothing을 거친 이미지의 한 pixel에 대하여 magnitude, orientation을 pixel간의 차이를 이용하여 구한다. 아래 식은 Gaussian smoothing 을 거친 pixel L(x,y)에 대하 gradient magnitude m(x,y)과 orientation θ(x,y)를 구하는 계산식이다.

$$\mathrm{m}(\mathrm{x},\mathrm{y}) = \sqrt{\big(L(x+1,y) - L(x-1,y)\big)^2 + \big(L(x,y+1) - L(x,y-1)\big)^2} \quad (2\text{-}10)$$

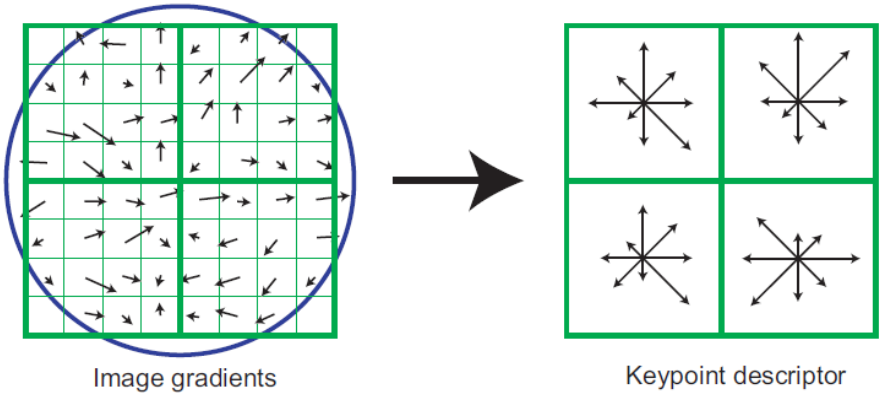
$$\theta(\mathrm{x},\mathrm{y}) = \tan^{-1}\left(\frac{L(x,y+1)-L(x,y-1)}{L(x+1,y)-L(x-1,y)}\right) \quad (2\text{-}11)$$

이렇게 계산된 각 pixel sample들을 gradient magnitude와 Gaussian-weighted circular window를 이용하여 가중되어 histogram에 저장된다. 360도에 대응하기 위해 36개의 bin을 이용하여 저장한다.

2.1.4 Local image descriptor

Descriptor를 생성하기 위해서는 image에서 추출된 keypoint 주변에서

image gradient 의 magnitude와 orientation을 구한다. 두 값을 계산하기 위해 Gaussian blur를 취한 image에서 연산하며 2.4에서 orientation을 구할 때 각 피라미드에서 미리 연산해 놓았던 값을 재사용한다.



[그림 2. 3 Descriptor 생성 방법[5]]

그림 2.3은 descriptor의 생성에 관련된 그림이다. 위의 그림은 8x8 samples set을 가지는 2x2 descriptor array를 나타낸다. 위의 그림의 왼쪽에서 각 칸의 화살표들은 image gradient를 나타낸다. 같은 그림에서 동그란 원은 circular Gaussian weighting function를 나타낸다. Image gradient가 오른쪽과 같이 bin에 저장되기 전에 각각의 위치에 따라 Gaussian weighting function을 통과한 뒤 결과가 저장된다. 이때 σ 값은 descriptor width의 1.5배인 값을 사용한다. 그림의 오른쪽과 같이 descriptor array는 8개의 gradient orientation을 가지고 각 화살표의 크기는 각 orientation의 magnitude를 나타낸다. 왼쪽의 각 image gradient는 오른쪽과 같이 bin에 더해지기 전에 각 dimension에 대해 1-d 만큼 곱해져서 저장된다. 이때 d는 각 image 에서 중심까지의 거리를 나타낸다. SIFT에서는 기본적으로 4x4의 descriptor array

를 사용하며 8개의 orientation bin을 사용한다. 따라서 하나의 keypoint에 대해 128개의 element를 가지는 descriptor vector를 생성하게 된다. 이렇게 생성된 vector는 빛 변화에 강하게 하기 위해 normalize한다.

2.1.5 Keypoint matching

Descriptor를 이용하여 database에 있는 descriptor와 현재 keypoint의 descriptor를 매칭한다. 각각의 descriptor는 128개의 vector로 되어 있으므로 유사한 정도를 판단하기 위해 Euclidean distance를 아래와 같이 계산 할 수 있다.

$$\mathrm{d}(\mathrm{p},\mathrm{q}) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \tag{2-12}$$

이렇게 계산된 Euclidean distance가 작을수록 유사성이 높다고 할 수 있다. 하지만 이렇게 distance를 각각의 descriptor에 대해 계산을 할지라도 distance가 0이 되는 경우는 극히 드물고 또한 간혹 keypoint가 많은 경우 descriptor가 우연히도 비슷한 모양을 가질 수 있기 때문에 distance만을 이용해 descriptor가 알맞게 matching되었다는 것을 판단하는 기준이 필요하다.

Distance를 이용하여 matching 판단을 하는 대표적인 방법에는 Nearest Neighbor distance ratio 방법이 있다.

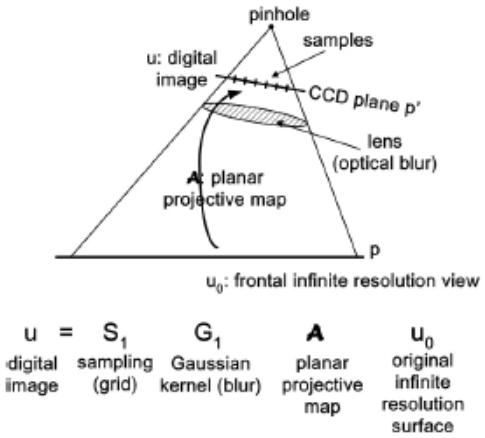
$$\mathrm{NNDR} = \frac{\text{first shortest euclidean distance}}{\textit{second shortest euclidean distance}} \tag{2-13}$$

이는 제일 distance가 작은 값과 두 번째로 distance가 작은 값의 ratio를 비교해서 shortest distance가 두번째로 작은 값보다도 훨씬 작은 경우 첫번째 descriptor와 database의 descriptor가 정확하게 matching 되었다고 가정한다.

2.2 ASIFT 알고리즘

SIFT의 성공이후에 많은 feature detector가 개발되었다. SIFT와 비슷한 SURF[3], 속도를 강조한 FAST[4], SIFT 방식을 유지하면서 속도와 affine에 강인한 PCA-SIFT[11] 등이 연구되었다. 그러나 fully affine invariant한 특성을 가지는 feature detector는 없었다. Morel이 제안한 ASIFT[12]는 다른 feature detector가 6개의 affine parameter를 normalize하는 데 비해 3개의 parameter를 simulate하고 나머지는 parameter는 normalize해서 사용한다. 따라서 fully affine invariant 하다.

2.2.1 Camera model



[그림 2. 4 Camera model[12]]

위의 그림 은 평평한 물체의 화상이 카메라에 들어오기까지의 모습을 나타낸 모형도 이다. 이러한 그림은 다음의 식으로 표현 될 수 있다.

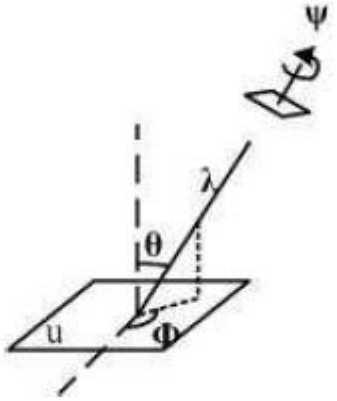
$$u = S_1 G_1 A T u_0 \tag{3-1}$$

이때 u는 카메라로부터 얻은 화상이며 u_0 는 평평한 물체의 ideal한 resolution view이다. A와 T는 카메라의 움직임으로부터 생긴 plane translation 과 planar projective map이다. G_1 은 lens의 blur때문에 생기는 Gaussian convolution modeling 이며 S_1은 CCD에서 sampling할 때 얻게 되는 standard sampling operator이다.

위의 모델을 좀더 simplify 하여 affine map $A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$를 만들면 이는 다시 아래와 같이 나눌 수 있다.

$$A = H_{\lambda} R_1(\psi) T_t R_2(\varphi) = \lambda \begin{bmatrix} cos\psi & -sin\psi \\ sin\psi & cos\psi \end{bmatrix} \begin{bmatrix} t & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} cos\varphi & -sin\varphi \\ sin\varphi & cos\varphi \end{bmatrix} \tag{3-2}$$

이때 $\lambda > 0$ 이고 λt 는 A의 determinant이다. R_i 는 rotations T_t는 tilt를 의미하며 T_t의 $t > 1$ 이다.



[그림 2. 5 Affine map decomposed camera model[12]]

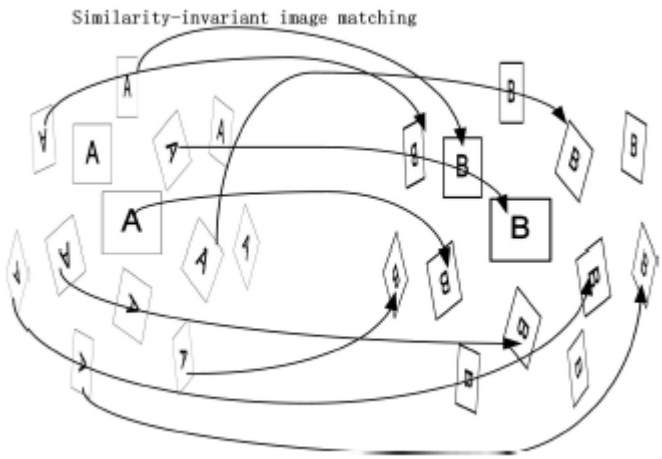
위의 그림은 식 3-2를 그림으로 modeling 한것이다. 이때 θ와φ = arcos 1/t로 정의되는 viewpoint angle이다. ψ와 λ는 각각 camera spin과 zoom을 의미하게 된다. 만일 카메라가 어떤 물체의 충분히 떨어진 위치에서 정면을 촬영하고 있다고 가정하면 λ = 1, t =1, $\varphi = \psi$ =0 이 될 것이다.

2.2.2 ASIFT Algorithm Process

먼저 이미지들에 대해 affine transform을 이용하여 camera optic axis에 따른 parameter인 longitude φ 와 latitude θ를 변화시켜 가며 다양한 transform된 이미지를 생성한다. 이미지를 생성할 때 tilt parameter인 t는 t = $\left|\frac{1}{cos\theta}\right|$인 값을 사용한다. Gaussian blur에 사용할 표준편차는 $c\sqrt{t^2-1}$ ($\therefore c = 0.8$) 인 값을 사용한다.

Affine parameter중 latitude θ 는 $a = \sqrt{2}$인 geometric series $1, a, a^2, a^3 \ldots a^n$

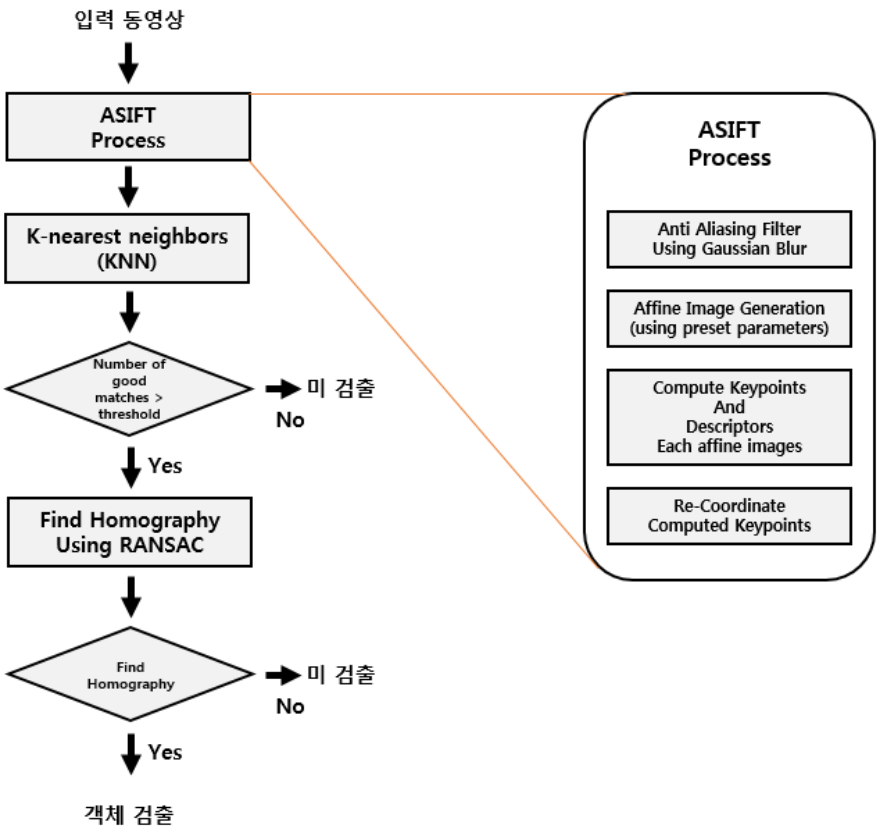
를 사용하며 n = 5로 사용한다. Longitude φ 는 각각의 tilt에 대해 $0, \frac{b}{t}, \ldots \frac{kb}{t}$를 사용하며 $b \cong 72°$ 를 사용하고 k는 $\frac{kb}{t} < 180°$ 인 값으로 정한다.



[그림 2. 6 ASIFT 알고리즘 수행 모식도[12]]

위의 그림과 같이 parameter를 이용하여 affine 된 이미지를 생성한 후 모든 생성된 이미지에 SIFT를 적용하여 keypoint와 descriptor를 생성한 뒤 각각을 Euclidean distance를 이용하여 비교한 뒤 good matching keypoint를 찾는다.

2.2.3 ASIFT의 동영상 내 객체인식 구현



[그림 2. 7 ASIFT 알고리즘 구현]

그림 2.7 과 같이 ASIFT 알고리즘을 구현하였다. 먼저 인식할 객체에 대해 Affine transform을 통해 16개의 affine 된 이미지를 생성한다. 생성된 16개의 이미지에 대해 keypoint와 descriptor를 추출하여 저장한다. 이때 추출된 keypoint들은 매칭 후 homography를 제대로 찾기 위해 다시 inverse affine transform을 통해 원래의 이미지의 keypoint위치로 변환을 시킨후 저장한다. 이렇게 한번 추출된 객체의 keypoint와 descriptor를 영상내 검출이

끝날 때까지 저장하여 사용한다.

객체에 대한 연산이 끝난 후에는 동영상의 매 프레임에 대하여 객체과 동일한 방법을 사용하여 keypoint와 descriptor를 추출한다. 이렇게 추출된 프레임 descriptor와 객체에서 추출된 descriptor를 nearest neighbors[13] 방식 중 하나인 k-nearest neighbors와 threshold 0.6을 이용하여 distance를 비교한다. 이렇게 knn을 이용하여 매칭된 keypoint들에 한해서 Random Sample Consensus(RANSAC)[14] 알고리즘을 이용하여 homography를 찾는다.

찾은 homography를 이용하여 프레임 내부에 객체가 있는지 확인한다. homography가 나오면 객체를 검출하였다고 판단한다. 하지만 매칭된 keypoint의 위치 특성이 명확하지 않으면 homography값이 제대로 나오지 않으며 이 경우 객체를 발견하지 못했다고 판단한다.

제 3 장 ASIFT 기반 객체 인식 정확도 향상 방법

3.1 기존의 ASIFT의 문제

ASIFT를 구현하여 실험하면 영상에 따라 차이가 있으나 검출되지 않아야 하는 이미지내에서도 많은 오매칭이 일어난다. 이러한 오매칭을 RANSAC을 이용하여 제거해야 함에도 불구하고 keypoint의 오매칭이 많으면 알고리즘 자체에서는 검출했다고 판단하는 오검출 현상이 발생한다.

아래 그림 3.1을 보면 원래 planetb라는 간판을 인식하여야 하지만 실제는 전혀 엉뚱한 부분을 간판으로 인식하여 녹색선으로 표시한 것을 알 수 있다.

[그림 3. 1 ASIFT Planetb 객체 검출 결과 이미지]

그 외에도 오검출은 검출되야 하는 이미지에서도 많이 일어나는 것을 볼 수 있는데 이러한 오검출은 전체적인 검출 안정성을 저하시키게 된다. 또한 이미지의 blur와 fucus의 문제는 keypoint의 숫자를 줄이고 정상적인 검출을 막게되어 검출되지 않는 미검출을 유발한다.

[그림 3. 2 ASIFT 검출 안정성 저하]

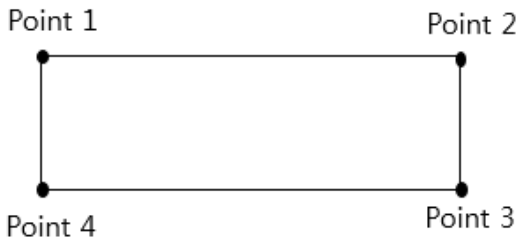
이러한 오검출과 미검출 상황들을 전체적인 동영상 내에서 객체 검출시 지속적인 검출을 어렵게 하고 엉뚱한 검출을 발생하게 하여 전체적인 검출 신뢰도를 낮게하는 단점으로 작용한다.

본 논문에서는 오검출과 미검출 현상을 개선하고자 distance and angle filter와 local object update를 통한 객체 인식 정확도 향상 방법을 제안한다.

3.2 Distance and Angle Filter

ASIFT를 수행하면 확률적으로 false detection이 발생할 가능성이 있다. 이때 검출하려는 객체의 특성을 반영하여 distance and angle filter를 사용하여 false detection을 제거 한다.

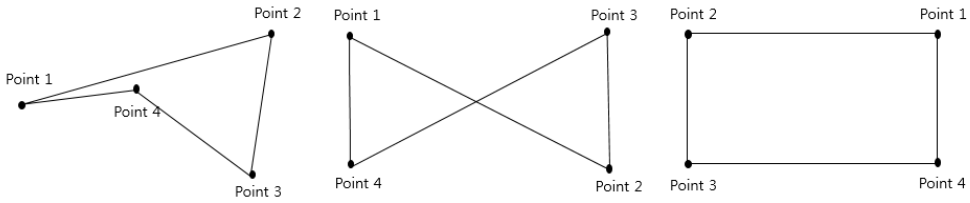
실험의 경우에 예를 들면 객체 검출에 사용한 간판은 모두 직사각형에 위치도 올바른 간판이다. 따라서 만일 간판이 검출 된다고 하면 검출된 간판은 틀어진 정도에 따라 다를 수 있으나 직사각형 모양을 유지하며 뒤집힘 현상이 없을 것이다.



[그림 3. 3 객체를 구성하는 각 point 표시]

이런 특성을 감안하여 먼저 distance filter를 제안한다. 그림과 같이 간판 object에서 제일 위쪽부터 시계방향으로 point 1~4까지라고 하면 point 1의 경우 point 2에 비해 상대적으로 왼쪽에 있어야 한다. 비슷하게 point 2의 경우도 point 3에 비해서는 항상 위에 있어야 하면서 point 1에 비해서는 오

른쪽에 있어야 한다. Point 3의 경우는 point 2보다는 아래에 point 4에 비해서는 위에 있어야 하며 point 4의 경우에는 point 1보다 아래에 point 3보다는 왼쪽에 있어야 한다. 이러한 검출된 사각형을 구성하는 점들을 위치적 특성을 반영하여 homography를 거친 검출결과에서 filtering을 수행한다. 이러한 filtering을 통해 아래 그림과 같인 뒤집혀서 검출 되는 것, 꼬인 형태의 검출 형태를 제거할 수 있다.



[그림 3. 4 비정상적인 객체 검출 상황 모식도]

[그림 3. 5 비정상적인 객체 검출 상황]

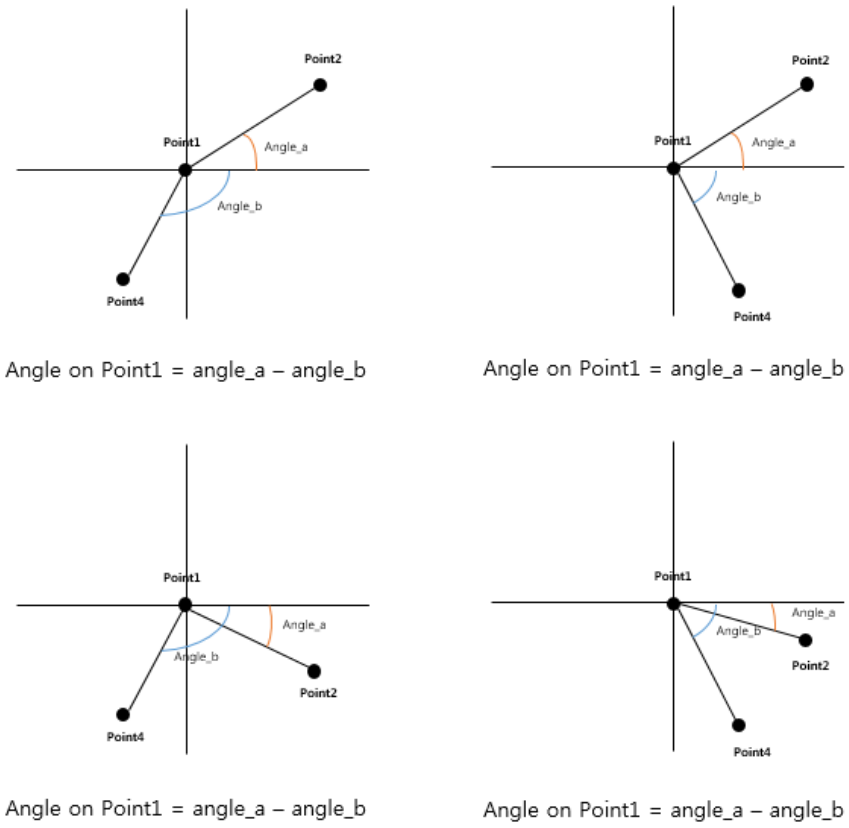
또한 잘못된 검출결과 중에는 단순히 꼬인 형태가 아니더라도 아래 그림 3.6과 같이 작은 점과 같은 결과를 출력하는 경우가 있다. 이런 경우를 제거하기 위해서 각 point 사이의 최소 거리를 정해 이를 넘지 비정상적이지 않더라도 크기가 너무 작은 homography 결과를 제거하는 효과를 얻는다. 실험에서는 10px보다 작은 point사이의 거리를 모두 기각하였다.

[그림 3. 6 너무 작은 객체 검출 예]

Angle filter는 distance filter를 통과한 검출 후보들을 대상으로 filtering을 수행한다. Distance filter를 통과한 검출 후보들은 각각의 point를 대상으로 각도를 계산한 뒤 각도가 정상적인 사각형이 가질 수 있는 한계를 넘는 값을 제거한다.

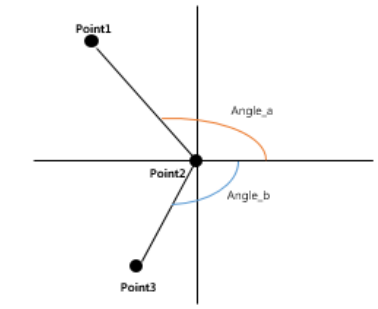
Distance filter로 인해 각 point에 대한 다른 point들을 상대적인 위치가

결정 되었으므로 arctangent 함수를 이용하여 각 point의 각도를 정할 수 있다.

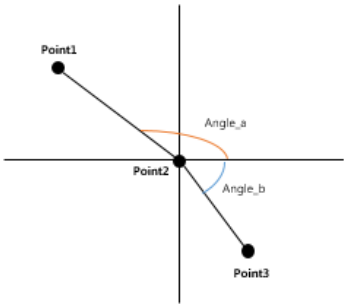


[그림 3. 7 Point1에 대한 angle 측정 방법]

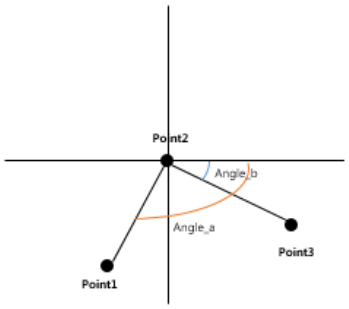
예를 들어 위의 그림과 같이 point1의 경우 point 2와 point 3의 위치가 point 1을 기준으로 고정되어 있으나 그림과 같이 총 4가지의 경우가 발생할 수 있다. 하지만 point1에 해당하는 각도를 구하는데 모든 경우에 대해 angle_a – angle_b 를 구하면 각을 구할 수 있다.



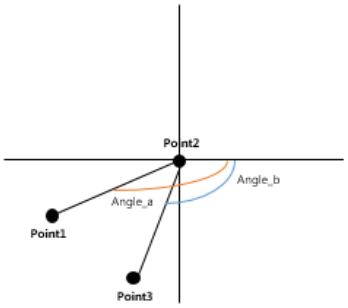
Angle on Point2 = 2π - angle_a + angle_b



Angle on Point2 = 2π - angle_a + angle_b



Angle on Point2 = angle_b - angle_a



Angle on Point2 = angle_b – angle_a

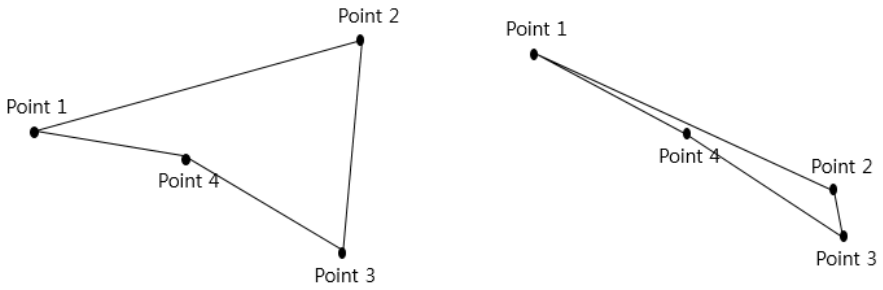
[그림 3. 8 Point2에 대한 angle 측정 방법]

하지만 point 2처럼 각각의 경우에 따라 다르게 각도를 계산해야 하는 경우가 생긴다. Angle filter를 만드는 과정에서 point1 과 point4는 같은 특성을 보여주었고 point2와 point3의 경우 위의 그림과 같이 경우에 따라 각각 계산해야 한다.

각각의 point의 각도 계산을 통해 연산을 수행하고 나면 그 결과값이

나온다. 이를 통해 그림 3.9과 같은 distance filter를 통과하는데는 문제가 없었으나 실제 검출은 불가능 한 형태의 검출결과를 제거 할 수 있다.

먼저 검출 환경에서는 사각형의 모양이 나와야 함으로 각 point에서 한 각의 크기는 180도를 넘을 수 없다. 이를 이용하여 아래 그림 3.7과 같은 오목한 형태의 사각형 결과들을 제거할 수 있다.



[그림 3. 9 Angle filter로 제거되는 모양]

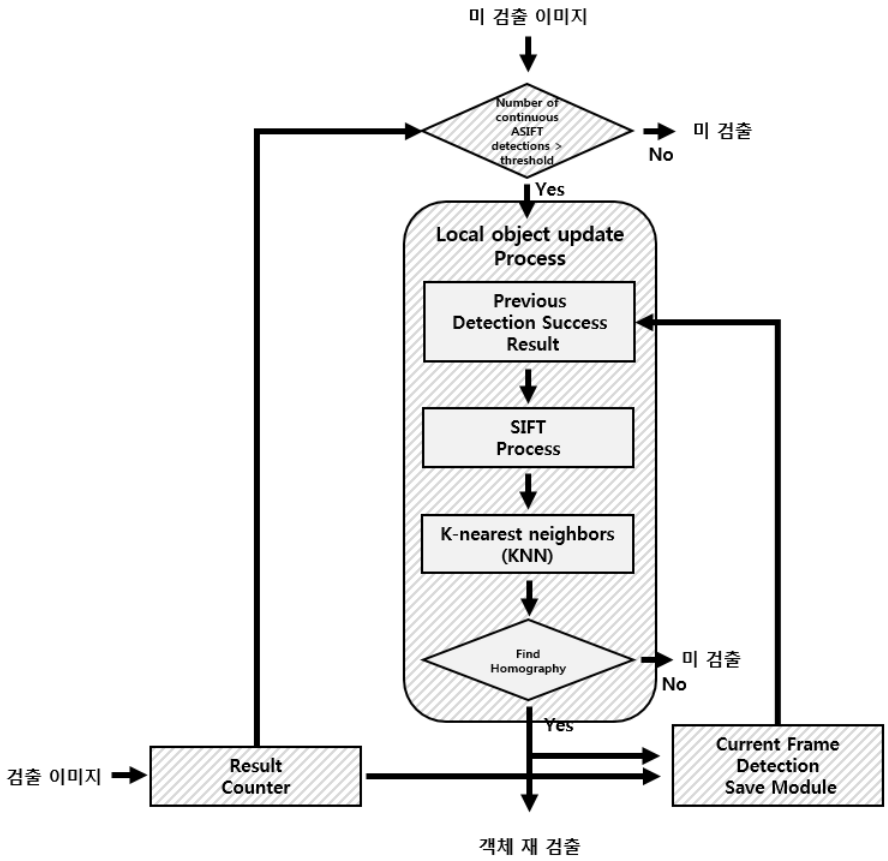
[그림 3. 10 Angle filter로 제거되는 모양 예시]

이와 같이 angle filter를 만들어 distance filter는 통과하였지만 사각형 모양이 검출되어야 하나 검출되지 않는 위의 그림과 같은 모양을 제거하였다.

3.3 Local object update 방식의 재검출

3.2절에서 제안한 방법으로 오매칭으로 인해 잘못 검출된 객체들과 검출 안정성이 낮은 객체들을 제거할 수 있다. 하지만 오매칭으로 인한 filtering과 오매칭으로 인한 미검출로 인해 검출이 되지 않는 경우가 발생한다. 특히 미검출의 경우 검출해야 하는 객체의 keypoint 숫자가 전체적으로 떨어지는 경우에 많이 발생한다. 예를 들어 camera의 광량 부족, 혹은 camera의 motion으로 인한 blur현상이 동영상에서는 자주 일어나는데 이런 경우는 전제 이미지 속에서 keypoint 수가 급감하는 현상이 일어난다. 또한 camera의 초점이 주변사물에 맞고 원하는 부분에 맞지 않아서 특정 영역 부분에서는 blur효과가 일어날 때에는 오검출이 많이 발생할 수 있다.

이 때 기존의 객체 이미지를 사용하여 영상내에서 객체를 검출하려는 시도는 good match가 충분히 발생하지 않아서 검출돼야 하는 상황에서도 미 검출이 될 수 있다. 이러한 미검출 상황에서도 검출할 수 있도록 이전 영상에서 검출된 object 를 저장하고 update하는 방식으로 검출하는 방법을 제안한다.



[그림 3. 11 local object update 재검출 방법]

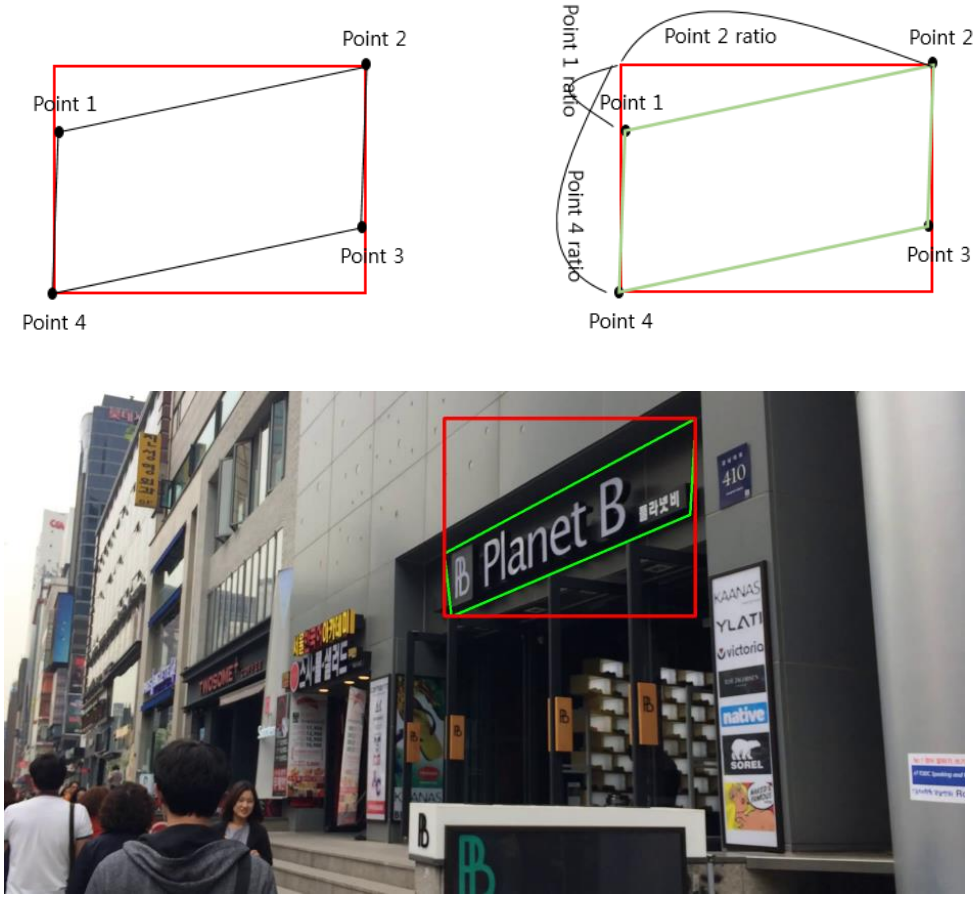
위의 그림 3.12는 local object update 재검출 방법의 흐름에 대한 모식도이다. 이를 통해 보면 먼저 기존의 ASIFT와 distance and angle filter를 이용하여 제대로 검출된 이미지를 구한다. 이때 검출된 이미지에 대해 counter를 이용하여 counting을 하게 된다. 이는 ASIFT의 오검출이 많기 때문에 distance and angle filter를 통과하더라도 우연히도 오검출 되는 경우가 있을 수 있기 때문이다. 이렇게 오검출된 이미지를 저장해서 미검출시 사용한

다면 local object update 방법은 SIFT를 이용하기 때문에 한번 오검출된 부분은 계속 오검출이 될 가능성이 있다. 이를 방지하기 위해 ASIFT 와 제안한 distance and angle filter를 이용한 검출이 연속적으로 n개 이상 누적 되었을 때에만 미검출시 local object update 재검출 방법을 이용하도록 한다. 이때 사용한 threshold n은 실험적으로 검증한 3을 사용하였다. 이것을 위해 각각의 ASIFT 검출 결과가 만들어 졌을 때 이 검출이 지속적인 검출인지 result counter 부분에서 counting을 하게 된다. Counting은 다음과 같은 규칙을 counting된다.

- ASIFT 검출 결과가 들어오면 기존 count에 1을 추가한다.
- ASIFT 미검출 결과가 들어오면 재검출을 할 수 있는 threshold가 되었고 재검출이 된다면 현재 count는 유지한다. 만일 미검출 결과가 들어오고 기존 local object를 이용한 재검출조차 실패하면 count는 0으로 리셋된다.

Counting이 끝난 결과는 frame내에서 추출되어 Current Frame Detection Success Save Module에 저장된다. 특히 이때 추출된 영역보다 적당하게 padding을 주어 주변의 영역까지 같이 잘라서 검출 영역보다 크게 저장하게 된다. 이는 padding부분의 keypoint와 descriptor를 이용하여 좀 더 local object를 이용한 재검출을 정확하게 하기 위해서 이다. 이 때 padding은 검출된 영역을 감싸는 최소의 직사각형을 구성하게 만들어 진다. Padding을 이용하여 영역을 저장하여 놓음으로 제대로 검출 결과를 표시하기 위해서는 저장된 영역과 원래 검출된 point 사이에 비율을 알고 저장하여야 한

다. 이때 저장된 비율은 나중에 재검출 후 frame에 표시될 때 검출 영역을 정확하게 표현하는데 사용된다.



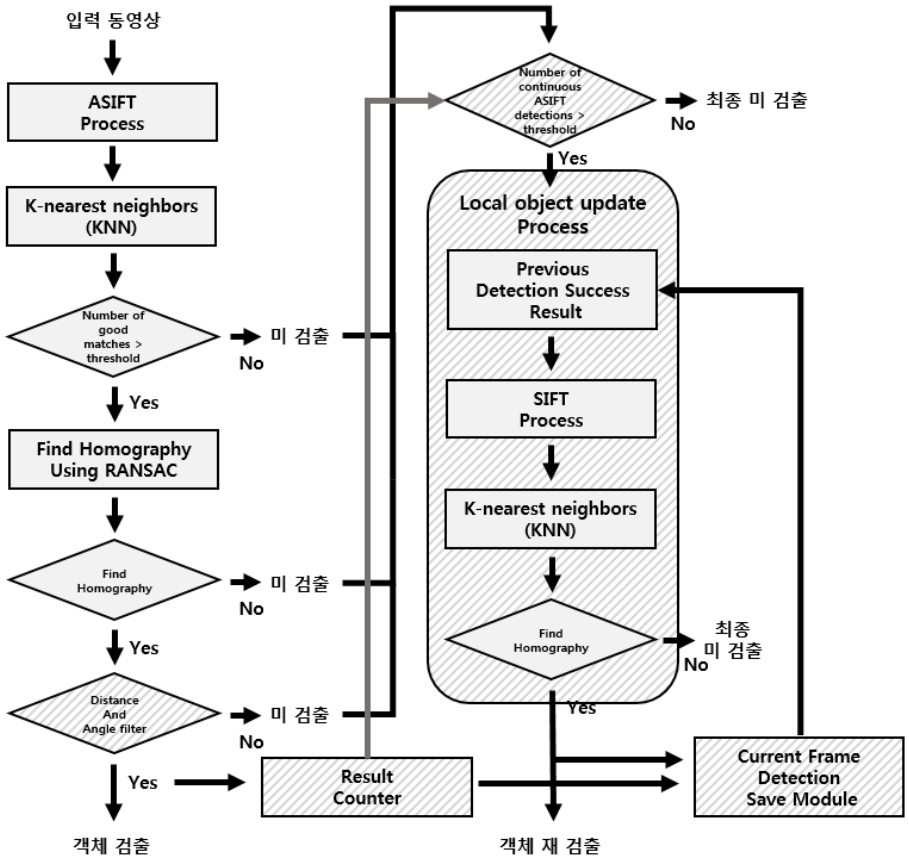
[그림 3. 12 Padding을 이용한 검출 저장방법]

저장된 local object 검출 결과는 매번 새로운 ASIFT 검출 결과가 들어올 때 마다 갱신되어 저장된다. 이는 최대한 현재 검출된 local object 결과 그대로를 재 검출에 이용하기 위해서이며 정확성을 높이기 위해서 갱신하여

저장한다.

저장된 local object를 이용한 재 검출은 ASIFT 대신 SIFT를 이용하여 수행된다. 이는 local object를 이용한 검출시 보통은 프레임간의 차이가 심하지 않은 경우에 수행되게 되는데 이 경우에는 상당히 검출하고자 하는 객체가 affine이 되어있는 상태라 할지라도 검출에 사용되는 object 정보 자체에 이미 affine되어 있기 때문에 SIFT만 사용하더라도 정확하게 검출하면서도 빠르게 검출할 수 있다. 그림 3.12에서 모식도와 같이 저장된 예전 검출 이미지를 불러온 후 keypoint와 descriptor를 추출한 후 knn match를 이용하여 good match keypoint를 찾는다. 찾은 keypoint를 RANSAC방식을 이용하여 homography를 구하고 이를 통해 재검출 결과를 표시한다.

3.4 제안한 방법의 ASIFT 구현과 분석



[그림 3. 13 Local object update와 distance and angle filter를 이용한 검출 process]

위의 그림 3.13은 최종 설계한 distance and angle filter와 local object update 재검출을 사용한 객체 인식 정확도 향상 알고리즘의 흐름도 이다. 제안

한 방식으로 인해 추가된 기능들은 빗금으로 표시하였다. 먼저 왼쪽을 보면 기본적인 ASIFT 방식을 유지하는 것은 같지만 최종적으로 homography를 찾고 나서 distance and angle filter를 이용해 오검출이 될 수 있는 이상한 검출들을 모양을 통해서 제거하였다. Distance and angle filter를 제대로 통과한 객체검출을 제대로 검출하였다고 판단하며 제대로 검출된 결과들은 만일 local object update 재검출을 위해 current frame detection success save module 에 저장되게 된다. 매 프레임마다 저장된 검출 데이터들은 previous detection 모듈에 갱신되어 저장된다.

만일 ASIFT 검출 과정 중에 homography를 제대로 구하지 못하거나 distance and angle filter에서 검증되지 못하여 미 검출 된 프레임의 영상은 local object 재검출 방식을 이용해서 처리된다. 먼저 미 검출된 프레임 이전에 threshold를 넘는 충분한 숫자의 ASIFT의 검출이 이루어졌는지 확인한다. 만일 이때 threshold를 넘지 못하면 더 이상 진행하지 않고 미 검출로 판단한다. 하지만 threshold를 넘으면 저장된 예전의 local object 검출 결과를 이용하여 새롭게 검출을 시도한다. 이 때는 예전 local object의 검출 결과를 이용하므로 ASIFT를 사용할 필요가 없이 SIFT 알고리즘만을 이용하여 연산량과 오매칭의 가능성을 줄인다. 저장된 예전 검출 결과와 SIFT 알고리즘을 이용하여 검출되면 객체를 검출했다고 판단하며 만일 그렇지 않다면 최종적으로 검출되지 않았다고 판단한다.

제 4 장 실험 결과 및 분석

4.1 실험 구현

동영상에서 객체를 검출하는 실험을 수행하였다. 촬영된 동영상은 모바일 기기를 이용하여 촬영하였고 일반적인 도심내 길거리 모습을 영상에 담았다. 검출하려는 객체는 간판으로 이를 이용하여 실내와 실외에서 사용할 수 있는 위치정보를 보정하려고 하였다. 실험을 위해 시간과 날씨의 조건이 다른 영상을 촬영하였고, 간판 촬영은 동영상 촬영과 다른 시간에 간판사진을 찍어와 사용하였다. 모든 간판 사진은 최대한 정면에서 촬영한 사진을 사용하였다.

실험에 필요한 software는 모두 OpenCV[15] 2.4.6버전을 기반으로 구현되었다. 실험에서 good match keypoint를 검증하는 threshold는 4를 사용하였으며 이는 사각형의 homography를 찾기위해 RANSAC을 돌리기 위해서는 최소한의 사각형의 구성 점의 수인 4개가 필요하기 때문이다.

결과 분석을 위해 설정한 ground truth는 ASIFT 수행 시 첫 번째로 검출된 간판으로부터 간판이 프레임 내에서 벗어날 때 까지를 눈으로 확인하여 설정하였다. 결과 비교를 위하여 True Positive Rate, False Positive Rate의 경우 아래와 같이 정의하였다.

$$\text{True Positive Rate} = \frac{\Sigma\text{True test outcome positive}}{\Sigma\text{Ground truth positive}}$$

$$\text{False Positive Rate} = \frac{\Sigma\text{False test outcome positive}}{\Sigma\text{Ground truth negative}}$$

강남역, 서울대입구역과 삼성역의 2개지점에서 동영상을 촬영하였으며 각 동영상에서 5개씩 간판을 검출을 시도하였으며 각각의 실험 결과를 아래 장들에서 나타내었다.

4.2 Distance and Angle Filter 실험 결과

4.2.1 Distance Filter 실험 결과

먼저 강남역에서 촬영한 영상을 기준으로 5개의 객체 각각 Planetb, Sushi, Twosome place, OST, YBM 간판을 검출하였다. 영상은 기본적으로 걸으면서 촬영된 영상이다.

로고	Distance filter				
	ASIFT 오검출	Distance filter 이용 제거된 오검출 수	오검출 제거율	Filter false negative rate	Filter false positive rate
Planetb	764	527	68.98%	8.90%	2.88%
Sushi	693	407	58.73%	12.38%	2.45%
Twosome	713	450	63.11%	10.80%	0.98%
OST	846	663	78.37%	19.08%	7.08%
YBM	916	812	88.65%	2.38%	2.40%
Average	786.4	571.8	72.71%	11.35%	3.25%

[표 4. 1 강남역 간판검출 Distance filter 성능 평가]

위의 표는 Distance filter의 성능을 평가한 표이다. 첫 번째 열에는 각각 사용한 로고를 표시하였다. 두번째 열에는 ASIFT만 수행하였을 때 검출된 결과 중 Ground truth 대비 오 검출인 frame의 수를 나타내었다. 세번째

열에는 Distance filter를 통해 제거된 오검출 frame 수를 나타내었으며 네 번째 열에는 실제 distance filter를 통해 제거한 오검출의 비율을 나타내었다. 각 로고의 특성에 따라서 다르나 일반적으로 최저 59%, 최대 89% 정도의 오검출 결과를 distance filter를 통해 제거할 수 있음을 알 수 있다.

다섯번째 열에는 Ground truth 상 검출이 되었어야 하지만 distance filter에 의해 기각된 비율을 나타내고 있다. 이를 통해 실제 기각하지 말았어야 하는 것도 distance filter에 의해 기각 당하는 것을 확인 할 수 있으나 실제 확인 결과는 Ground truth임에도 정확하지 않은 모양으로 검출되는 경우에 주로 기각 당한 것을 아래 그림 4.1에서 확인해 볼 수 있다. 따라서 Ground truth 상 검출이 되었어야 하지만 distance filter가 기각하였다고 하여 그것이 distance filter의 문제라고 보기는 어렵다.

[그림 4. 1 Ground Truth 인 distance filter 기각 예제]

여섯번째 열에는 Ground truth false 대비해 distance filter를 통과한 잘못된 검출의 경우의 비율을 나타낸다. 평균 3%정도의 잘못된 검출 결과가 우연히도 distance filter를 통과하며 이는 결과적으로 오검출을 유발할 수 있다.

로고	ASIFT		ASIFT with Distance filter	
	True Positive Rate	False Positive Rate	True Positive Rate	False Positive Rate
Planetb	100.00%	100.00%	91.10%	2.88%
Sushi	100.00%	100.00%	87.62%	2.45%
Twosome	100.00%	100.00%	89.20%	0.98%
OST	100.00%	99.76%	80.92%	7.08%
YBM	100.00%	100.00%	97.62%	2.40%
Average	100.00%	99.95%	88.65%	3.25%

[표 4. 2 강남역 간판검출 ASIFT 와 ASIFT with Distance filter 성능 비교]

위의 표는 previous ASIFT의 결과에 대비해 Distance filter 만 도입하였을 때 검출결과의 변화를 나타낸 표이다.

로고를 제외한 둘째, 셋째 열은 기존의 ASIFT 기반일 때 True Positive Rate와 False Positive Rate를 나타낸다. 이 두 열을 놓고 보았을 때 거의 모든 경우에서 ASIFT 알고리즘은 알맞은 검출을 수행했다고 하지만 실제로는 모든 Ground truth 상 true, false에 상관없이 모든 frame에 객체가 있다고 검출하는 것을 알 수 있다.

실제 ASIFT 와 distance filter를 함께 사용한 결과 True Positive Rate가 떨어진 것을 볼 수 있다. 기존에 100% 에서 88.65%으로 11.35%가량 떨어졌

다. 그러나 다섯째 열을 보면 distance filter의 도입으로 false positive가 어느 정도 줄었는지를 확인 할 수 있다. 기존 99.95%에서 3.25%로 떨어졌으며 96.75%의 false positive 감소한 것을 확인할 수 있다.

서울대 입구역에서 촬영된 동영상으로 간판 NHBank, Gongcha, Hakoya, Orange 안경점, Apgujung 김밥에 대한 검출하였다. 그 결과는 다음과 같다.

로고	Distance filter				
	ASIFT 오검출	Distance filter 이용 제거된 오검출 수	오검출 제거율	Filter false negative rate	Filter false positive rate
NHBank	837	527	83.15%	26.38%	2.51%
Gongcha	836	407	86.12%	39.63%	2.03%
Hakoya	808	450	76.49%	11.46%	2.48%
Orange	823	663	78.49%	14.08%	6.41%
Apgujung	735	812	66.80%	10.25%	3.31%
Average	807.8	634.2	78.51%	19.34%	3.37%

[표 4. 3 서울대입구역 간판검출 Distance filter 성능 평가]

표 4.3은 서울대 입구역 간판에 대한 Distance filter의 성능을 평가한 표이다. 표를 통해 보면 Distance filter는 평균적으로 78.51%의 오검출을 제거하며 Ground truth 대비 약 3.37% 정도의 잘못된 검출 결과만이 distance filter를 통과하는 것으로 나타났다.

로고	ASIFT		ASIFT with Distance filter	
	True Positive Rate	False Positive Rate	True Positive Rate	False Positive Rate
NHBank	100.00%	100.00%	73.62%	2.51%
Gongcha	100.00%	100.00%	60.37%	2.03%
Hakoya	100.00%	100.00%	88.54%	2.48%
Orange	100.00%	95.92%	85.92%	6.41%
Apgujung	100.00%	97.22%	89.75%	3.31%
Average	100.00%	98.63%	80.66%	3.37%

[표 4. 4 서울대입구역 간판검출 ASIFT와 ASIFT with Distance filter 성능 비교]

위의 표는 서울대 입구역 간판 검출시 ASIFT만 사용한 결과 대비 ASIFT에 Distance filter만 사용했을 때 검출 결과를 표로 나타낸 것이다. ASIFT만 이용하였을 때는 의미 없는 True positive와 False Positive가 95%이상인 결과가 나온 것에 비해 Distance filter를 사용한 후에는 잘못된 검출의 경우는 3.37%로 줄어 정확성이 향상 된 것을 볼 수 있다. 단 Distance filter에 의해 True Positive rate역시 평균적으로 80.66%로 감소한 것을 알 수 있다.

4.2.2 Angle Filter 실험 결과

먼저 강남역 동영상에서 간판 검출시 Distance filter를 통과한 검출 결과만 놓고 filtering을 수행한다. 결과는 아래 표 4.5과 같다.

로고	Angle filter			
	Distance filter 오검출	Angle filter가 제거한 오 검출	Angle filter 제거 비율	Angle filter 추가 제거
Planetb	22	22	100.00%	4
Sushi	17	17	100.00%	0
Twosome	7	7	100.00%	14
OST	60	60	100.00%	3
YBM	22	22	100.00%	2
Average	25.6	25.6	100.00%	4.6

[표 4. 5 강남역 간판검출 Angle filter 성능 평가]

두번째 열에는 distance filter를 통과한 Ground truth 상 오검출 frame의 개수가 나타나 있고 세번째 열에는 angle filter가 이를 몇 개나 제거하였는지를 알 수 있다. 이를 통해서 angle filter의 제거 비율을 보면 angle filter를 모든 distance를 통과한 오검출된 결과를 제거한다.

마지막 열에는 Ground truth 상 검출되어야 하나 angle filter에서 제거된 frame의 수를 나타낸다. 이는 angle filter자체가 검출 모양 자체가 이상한 검출 결과들을 제외하기 때문에 생기는 것으로 실제 검출 모양이 아래 그림 4.2와 같이 비정상적인 경우가 많다.

또 이후에 제안하는 Local object update 방식의 재 검출을 위해서 ASIFT 결과는 상당히 좋아야 하기 때문에 angle filter에서 어느 정도 강한 방식으로 재검출에 사용하기 부적절하거나 불안정한 검출 결과들을 많이 제거하게 된다.

[그림 4. 2 Angle filter에서 제거된 Ground truth 사진 예제]

로고	ASIFT		ASIFT with Proposed Filter	
	True Positive Rate	False Positive Rate	True Positive Rate	False Positive Rate
Planetb	100.00%	100.00%	89.41%	0.00%
Sushi	100.00%	100.00%	87.62%	0.00%
Twosome	100.00%	100.00%	84.32%	0.00%
OST	100.00%	99.76%	78.95%	0.00%
YBM	100.00%	100.00%	95.24%	0.00%
Average	100.00%	99.95%	86.49%	0.00%

[표 4. 6 강남역 간판검출 ASIFT 와 ASIFT with Proposed filter 성능 비교]

표 4.6 는 기존 ASIFT와 제안된 filter를 사용한 ASIFT 알고리즘의 비교이다. 표를 통해서 보면 기존 ASIFT 알고리즘의 높은 False Positive Rate가

문제가 되었었는데 Proposed Filter에서는 잘못된 검출을 완벽하게 제거한 것을 볼 수 있다. 이는 Distance filter와 Angle filter가 보수적으로 오검출들을 제거한 점에서 그 이유를 찾을 수 있다. 대신 True Positive Rate역시 이러한 보수적인 오검출 제거방법에 의해 제거되어 기존 100% 검출에 비해 최소 79% 에서 최대 95%까지, 평균적으로는 86.49%의 True Positive Rate가 나타났다. 이는 기존에 비해 아쉬운 결과이다. 물론 Ground truth 결과가 나타나야 하는 frame 내에서도 오검출이 존재한다는 사실을 감안해야 한다.

서울대 입구역 동영상 내에서 간판을 찾는 경우에 대해서도 Angle filter에 대한 실험을 수행 하였다. 그 결과는 다음과 같다.

로고	Angle filter			
	Distance filter 오검출	Angle filter가 제거한 오 검출	Angle filter 제거 비율	Angle filter 추가 제거
NHBank	21	21	100.00%	20
Gongcha	17	17	100.00%	7
Hakoya	20	20	100.00%	7
Orange	55	49	89.09%	23
Apgujung	25	24	96.00%	3
Average	27.6	26.2	94.93%	12

[표 4. 7 서울대입구역 간판검출 Angle filter 성능 평가]

기존 강남역 테스트 결과와 같이 angle filter의 경우 Distance filter를 통과한 오 검출을 최소 89%까지 평균적으로 94.93%까지 제거하였으며 추가제거를 통해 평균적으로 12개의 Ground truth내의 검출을 제거하였다. 이러한

검출 중 상당수는 검출 모양이 깔끔하지 않는 검출 결과들을 포함하고 있다.

로고	ASIFT		ASIFT with Proposed Filter	
	True Positive Rate	False Positive Rate	True Positive Rate	False Positive Rate
NHBank	100.00%	100.00%	61.35%	0.00%
Gongcha	100.00%	100.00%	56.10%	0.00%
Hakoya	100.00%	100.00%	84.90%	0.00%
Orange	100.00%	95.92%	69.72%	0.70%
Apgujung	100.00%	97.22%	88.52%	0.13%
Average	100.00%	98.63%	74.03%	0.17%

[표 4. 8 서울대입구역 간판검출 ASIFT 와 ASIFT with Proposed filter 성능 비교]

기존 강남역 결과와 유사하게 Proposed filter를 사용시 기존 ASIFT만 사용했던 경우에 비해 False positive 검출의 경우 0에 가깝게 줄일 수 있었다. 다만 filter에 의해 제거되는 True positive 결과가 있어 평균적으로 검출률은 74.03%로 떨어지는 것을 알 수 있다.

4.3 Local object update 재 검출 실험 결과

Local object update 방식의 재 검출은 기존의 frame에서 검출되었던 object의 정보를 저장해 놓았다가 미 검출 시 다시 재 검출을 시도하는 방법이다. 이를 통해 미 검출되는 객체 없이 프레임 내에서 연속적으로 객체를 검출 할 수 있도록 하였다.

	Local Object Update 재검출		
로고	재검출 시도	재검출 성공율	재검출 사용율
Planetb	20	90.00%	80.00%
Sushi	27	92.59%	71.05%
Twosome	42	92.86%	93.33%
OST	32	96.88%	100.00%
YBM	6	100.00%	100.00%
Average	25.4	93.70%	86.99%

[표 4. 9 강남역 간판검출 Local object update 재검출 성능 평가]

표 4.9에서 첫 번째 열을 강남역 동영상에서 사용한 로고를 나타낸다. 두 번째 열은 실제 Local object update 재검출 방식에서 실제 재검출을 시도한 횟수를 표시하고 있다. 세번째 열은 각 재검출 시도에서 얼마나 많은 재검출이 성공적으로 이뤄졌는지 나타낸다. 또한 마지막 열에서는 실제 ASIFT가 검출 하지 못하였거나 Filtering에 의해 제거되어 검출하지 못한 간판에 대해 어느 비율로 재검출 시도가 이루어 졌는지를 나타내었다.

이를 통해 볼 때 Ground truth 대비 간판 검출이 되야하지만 실제

proposed filter를 사용한 ASIFT 간판검출에서 간판 검출이 일어나지 않는 경우에 86.99% 정도 재검출이 시도된다. 각 재검출 시도는 local object update 재검출 방식을 이용하여 평균 93.70%의 재검출에 성공한다.

로고	Local Object Update 재검출 최대 연속 검출수		
	ASIFT with Filter 최대 연속 검출수	Local object update 재검출 최대 연속 검출수	증가율
Planetb	106	192	81.13%
Sushi	168	176	4.76%
Twosome	98	142	44.90%
OST	50	150	200.00%
YBM	66	84	27.27%
Average	97.6	148.8	52.46%

[표 4. 10 강남역 간판검출 Local object update 재검출 최대 연속 검출]

영상내에서 검출의 경우에는 끊기지 않고 지속적으로 객체를 검출하는 것이 중요한데 local object update 방식을 사용하면 위의 표 4.10과 영상내에서 지속적으로 검출되는 객체의 수가 증가한 것을 볼 수 있다. 최소 5%에서 최대 200%까지 평균적으로 52.46% 정도 연속 검출되는 비율이 늘어난 것을 확인 할 수 있다.

로고	Local Object Update 재검출		
	재검출 시도	재검출 성공율	재검출 사용율
NHBank	41	95.12%	65.08%
Gongcha	48	97.92%	66.67%
Hakoya	22	95.45%	75.86%
Orange	10	90.00%	75.86%
Apgujung	23	95.65%	82.14%
Average	28.8	95.83%	61.28%

[표 4. 11 서울대입구역 간판검출 Local object update 재검출 성능 평가]

표 4.11은 서울대 입구역 동영상에 대한 간판 검출시 Local object update 재검출 사용 결과이다. 서울대 입구역 동영상이 조도가 낮은 상황에서 촬영되었기 때문에 검출이 정확하지 않은 경우가 있어 많은 재검출 시도가 있었다. 그러나 재검출 사용율은 떨어졌는데 이는 이전 프레임에서 지속적인 간판 검출 결과가 있어야 미검출 시 재검출 시도가 되는데 실제 조도 부족으로 지속적인 검출이 이루어 지지 않은 경우가 있기 때문이다.

로고	Local Object Update 재검출 최대 연속 검출수		
	ASIFT with Filter 최대 연속 검출수	Local object update 재검출 최대 연속 검출수	증가율
NHBank	31	117	277.42%
Gongcha	46	136	195.65%
Hakoya	49	183	273.47%
Orange	70	95	35.71%
Apgujung	159	234	47.17%
Average	71	150	115.49%

[표 4. 12 서울대입구역 간판검출 Local object update 재검출 최대 연속 검출]

표 4.12와 같이 동영상 내에서 Local object update 방식 재검출은 연속 간판 검출을 평균 115.9% 정도로 증가시켰다.

4.4 Proposed Algorithm 실험 결과

실험에 사용된 검출 영상에 대해 Distance and Angle filter와 Local object update 재검출을 모두 사용한 결과를 아래 표에 나타내었다.

Overall						
로고	Previous		Proposed			
	ASIFT		ASIFT with Proposed filter		ASIFT with Proposed filter and Local Object Update 재검출	
	True Positive Rate	False Positive Rate	True Positive Rate	False Positive Rate	True Positive Rate	False Positive Rate
Planetb	100.00%	100.00%	89.41%	0.00%	96.61%	0.00%
Sushi	100.00%	100.00%	87.62%	0.00%	94.14%	0.00%
Twosome	100.00%	100.00%	84.32%	0.00%	97.21%	0.00%
OST	100.00%	99.76%	78.95%	0.00%	99.34%	0.00%
YBM	100.00%	100.00%	95.24%	0.00%	100.00%	0.00%
Average	100.00%	99.95%	86.49%	0.00%	96.72%	0.00%

[표 4. 13 강남역 간판검출 Overall 결과]

두 번째 열과 세 번째 열에는 기존 ASIFT만 사용했을 경우의 true positive rate와 false positive rate를 나타내었다. 이를 통해 보면 기존의 ASIFT만 사용하였을 때에는 모든 프레임에서 간판을 검출했다라고 인식하여 ASIFT를 이용한 간판검출이 정확하지 않아 의미가 없다라고 판단할

수 있다.

세 번째 열과 네 번째 열에는 ASIFT와 proposed filter인 Distance and Angle filter를 사용하여 검출한 결과를 나타낸다. 이 값을 보면 filtering을 통해 훌륭하게 false positive를 제거하였음을 알 수 있다. 하지만 실제 filtering 규칙으로 인해 true positive도 조금 제거되어 평균적으로 기존 100% 검출에서 86.49%로 감소한 것을 볼 수 있다.

다섯 번째 열과 마지막 열에서는 ASIFT를 poposed filter와 local object update 재검출 방식을 이용하여 실험한 결과를 나타내고 있다. 이를 통해 local object update 재검출을 이용하여 false positive 검출 결과 없이 true positive 검출을 증가 시켰음을 확인 할 수 있다. 이를 통해 proposed filter와 ASIFT만 사용시 86.49%의 검출률을 가졌던 알고리즘을 평균적으로 96.72% 검출률을 가지도록 향상시켰음을 확인할 수 있다.

Overall						
로고	Previous		Proposed			
	ASIFT		ASIFT with Proposed filter		ASIFT with Proposed filter and Local Object Update 재검출	
	True Positive Rate	False Positive Rate	True Positive Rate	False Positive Rate	True Positive Rate	False Positive Rate
NHBank	100.00%	100.00%	61.35%	0.00%	82.21%	0.00%
Gongcha	100.00%	100.00%	56.10%	0.00%	84.15%	0.00%
Hakoya	100.00%	100.00%	84.90%	0.00%	93.75%	0.00%
Orange	100.00%	95.92%	69.72%	0.70%	80.28%	0.70%
Apgujung	100.00%	97.22%	88.52%	0.13%	97.54%	0.13%
Average	100.00%	98.63%	74.03%	0.17%	88.84%	0.17%

[표 4. 14 서울대입구역 간판검출 Overall 결과]

표 4.14는 서울대입구역 간판검출 영상을 이용하여 ASIFT와 Proposed Algorithm을 사용한 결과를 나타내고 있다. 강남역 결과와 마찬가지로 ASIFT만 이용하였을 때 false positive가 100%에 육박하여 모든 프레임에 대해 간판을 찾았다고 인식한 데에 비해 filtering을 통해 false positive를 평균 0.17%가까이 낮추었다. filtering으로 인해 낮아진 true positive 결과를 local object update 재검출을 이용하여 기존 평균 74.03%에서 평균 88.84%으로 검출률을 향상 시켰다.

제 5 장 결론

본 논문에서는 ASIFT를 이용하여 영상 내에서 객체인식을 할 때 오매칭으로 인해 검출의 어려움을 확인하고 이를 통해 적절한 방식을 이용하여 검출을 판단해야 함을 밝혔다. 동시에 오매칭으로 인한 오검출과 threshold 값과 영상 내 blur와 같은 제한으로 인해 미검출이 되는 상황을 밝히고 정확도를 향상 시킬 수 있는 방안에 대해 두 가지 방면으로 제안했다.

첫째로 검출해야 하는 객체의 모양적 특성을 이용한 distance and angle filter를 제안했다. 이는 객체가 검출되어야 하는 일반적인 구조와 그 구조를 이루는 점들의 각도를 이용하여 오검출을 방지하는 기술이다. 실험 결과 거의 모든 오검출의 경우에 대해 오검출임을 filter를 통해 판단하였다. 또한 검출이지만 검출 모양이 적절하지 않은 결과에 대해서도 검출 결과를 판단함으로 좀 더 정확한 모양만 검출결과로 인정하였다.

둘째로 local object update 재검출 방식을 제안하였다. 이는 이미 이전에 ASIFT를 통해 검출된 영상 내 local object를 저장하였다가 미검출 시 저장된 local object 데이터를 이용하여 재검출을 시도하는 방안이다. 이를 통해 영상의 blur 와 같이 검출을 방해하는 영상 특성과 알맞지 못한 검출 결과에서도 정확하게 객체를 재검출 할 수 있었다.

제안된 두 가지 알고리즘을 이용하여 잘못된 검출 결과는 제거하여 0%에 가깝게 유지하고 검출률은 90% 근처로 유지할 수 있다.

참고문헌

[1] S. Battiato, G. Gallo, G. Puglisi and S. Scellato, "SIFT Features Tracking for Video Stabilization," in *Proc. ICIAP*, Modena, Italy, Sep. 10-14, 2007, pp. 825-830.

[2] M. Brown and D. G. Lowe, "Automatic Panoramic Image Stitching Using Invariant Features," *International Journal of Computer Vision*, vol. 74, no. 1, pp. 59-73, Aug. 2007.

[3] H. Bay, A. Ess, T. Tuytelaars and L.V. Gool, "Speeded-Up Robust Features (SURF)," *Elsevier Journal on Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346-359, Jun. 2008

[4] E. Rosten, R. Porter and T. Drummond, "Faster and better: A Machine Learning Approach to Corner Detection", *IEEE Trans. On Pattern Analysis and Machine Intelligence*, vol. 32, no. 1, pp. 105-119, Jan. 2010.

[5] D.G Lowe, "Distinctive image features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91-110, Nov. 2004.

[6] T. Tuytelaars and K. Mikolajczyk, "Local Invariant Featue Detectors: A Survey," *Journal on Foundations and Trends® in Computer Graphics and Vision*, vol. 3, no. 3, pp. 177-280, Jan. 2008.

[7] N. Snavely, S. M. Seitz and R. Szeliski, "Photo Tourism: Exploring Photo

Collections in 3D," *ACM Transactions on Graphics*, vol. 25, no. 3, pp. 835-846, July. 2006

[8] R. Hu, R. shi, I. Shen and W. Chen, "Video Stabilization Using Scale-Invariant Features," in *Proc. Information Visualization*, Zurich, Switzerland, July. 4-6, 2007, pp871-877.

[9] X. Deng, F. Wu, Y. Wu and C. Wan, "Automatic spherical panorama generation with two fisheye images," in *Proc. Intelligent Control and Automation*, Chongqing, China, June. 25-27, 2008, pp 5955- 5959.

[10] T. Lindeberg, "Scale-space theory: A basic tool for analyzing structures at different scales," *Journal of applied statistics*, vol. 21, no. 2, pp. 225-270, 1994

[11] K. Yan, and R. Sukthankar, "PCA-SIFT: A More Distinctive Representation for Local Image Descriptors," in *Proc. IEEE CVPR*, June. 27 - July. 2, 2004, pp. 506-513.

[12] J. M. Morel and G. Yu. "ASIFT: A New Framework for Fully Affine Invariant Image Comparison," *SIAM Journal on Imaging Sciences*, vol. 2, issue 2, 2009.

[13] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. On Information Theory*, Vol. 13, no. 1, pp. 21-27, Jan. 1967

[14] M. Fischler and R. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Application to Image Analysis and Automated Cartography," *Magazine on Communications of the ACM*, vol. 24, no. 6, Jun. 1981.

[15] Open Source Computer Vision (OpenCV)[online]. Available: http://opnecv.org

Abstract

A Study on Accuracy Improvement for Object Detection with ASIFT

Object detection using local features is widely used because of the advantage of local features. SIFT, a well-known local feature, achieves better performance than o other local feature detectors, but, it suffers from degraded accuracy in affine object detection. Affine SIFT (ASIFT) overcomes SIFT and achieves fully affine invariant SIFT. To guarantee fully affine invariant SIFT, ASIFT uses affine parameters to create many affine images.

In this research, an ASIFT algorithm is used for object detection in real-life video streams. A good matched threshold is used before extracting homography to reduce false matched keypoints and false detected objects. Furthermore, local object update detection, distance and angle filter are proposed. Local object update is used for decreasing the undetected rate by redetecting the previous detected object on an

undetected frame. Distance and angle filter is used to decrease the false object detection rate. The proposed method keeps the false positive rate to almost 0% and the true positive detection rate near 90%, better than a previous ASIFT detection algorithm.

Keywords : SIFT, ASIFT, Object detection

Student Number : 2013-20857