논문번호 13-01-08

Journal of IKEEE.Vol.17,No.1,051～056,March 2013
http://dx.doi.org/10.7471/ikeee.2013.17.1.051

실시간 다중 객체인식 알고리즘 구현

Implementation of Real time based Multi-object recognition algorithm

박 태 룡*

Tae-Ryong Park*

Abstract

This thesis propose a improved matching method for implementing an ORB algorithm based multi-object recognition. SURF algorithm that is well known in the object recognition algorithms is robust in object recognition. However, there is a disadvantage for real time operation because, SURF implemention requires a lot of computation. Therefore we propose a modified ORB algorithm which shows the result of almost 70% speed improvement by improving matching part to recognize multi object on real time.

요 약

본 논문에서는 ORB 알고리즘을 기반으로 하는 다중객체 인식 구현을 위하여 개선된 매칭 기법을 제안한다. 객체 인식 알고리즘으로 잘 알려진 SURF 알고리즘은 객체인식에 강인하지만 연산량이 많아 실시간으로 구현하기에는 어려운 단점이 있다. 따라서 ORB 알고리즘을 활용하여 객체를 인식하였고, 실시간 다중객체인식을 위해 매칭 단계를 개선하여 속도를 약 70% 향상 시켰다.

Key words : ORB, SURF, Matching, Hamming Distance, OpenMP

Ⅰ. 서론

컴퓨터 비전 및 패턴 분석을 비롯한 다양한 분야에서 이미지 상에 존재하는 특정 객체를 자동으로 인식하고 추적하기 위한 많은 연구가 진행되어 왔다.[1]

많은 연구 중 최근 스마트 폰과 같은 임베디드 환경에서 인식 알고리즘 연구가 활발히 이루어지고 있다. 대표적인 예로 증강현실을 들 수가 있다. 객체를 인식하고 인식한 화면에 3D 객체를 보여 주어 광고, 게임 등 여러 가지 분야로 사용될 수 가 있다. 인식 알고리즘들 중 대표적인 알고리즘으로는 SIFT[2]와 SURF[3]가 있다. 하지만 SIFT와 SURF는 연산량이 많아 실시간으로 물체인식 하기에는 어려운 단점을 가지고 있다. 이러한 단점을 보완한 알고리즘이 ORB[4]이다. ORB는 특징점 추출인 빠른 FAST Corner Detection 알고리즘과 Descriptor 생성이 빠른 BRIEF알고리즘을 사용한다.

ORB 알고리즘은 특징점 추출과 Descriptor 생성이 빠르지만, 매칭 단계에서 특징점을 비교하기 때문에 특징점이 많아 질 수 록 인식을 수행하는 속도가 느려지는 단점이 있다. 특히, 특징점이 많은 다중객체들을 실시간으로 인식하기가 어렵다.

본 논문에서는 이 점을 보완하기 위해 ORB 알고리즘을 통하여 추출된 특징점과 Descriptor 데이터를 정렬하고, 매칭 단계에서 근처에 있는 데이터를 비교하여 속도를 개선 시켰으며, 또한 OpenMP를 활용한 병렬 처리로 수행 속도를 개선 시켰다.

* Professor, Dept. of Computer Engineering, Seokyeong University, trpark@skuniv.ac.kr, 02-940-7742

Manuscript received Mar. 12.2013; revised Mar. 22.2013; accepted Mar.25.2013

실험 환경으로는 인텔 CPU 와 Cortex-A9이 탑재된 ODROID-X 보드에서 C로 구현된 ORB알고리즘을 기존 매칭 기법과 제안된 매칭 기법을 비교하였으며 그 결과 매칭단계의 수행속도가 약 1.7배 향상되었다.

본 논문의 구성은 다음과 같다. 2장에서는 기존 매칭 기법을 소개하고, 3장에서는 개선된 매칭을 적용한 ORB 알고리즘을 소개한다. 4장에서는 실험 결과 및 분석으로 ORB 성능, 매칭 성능, 그리고 개선된 매칭 수행 성능을 보여 준다.

II. 기존 매칭 기법

객체를 인식하기 위해서는 대상이 되는 객체에 ORB 알고리즘을 사용하여 특징점과 Descriptor를 생성한다. 그 후 같은 객체인지 확인하기 위하여 매칭 기법을 사용하게 된다. ORB 알고리즘을 사용하여 생성한 Descriptor는 256 bit string description으로 구성 되어 있다. 이로 인해 매칭 알고리즘으로는 binary matching 알고리즘을 사용 한다. 이 때 사용되는 알고리즘이 Hamming Distance 알고리즘[11]이다.

같은 특징점을 찾기 위해서 Hamming Distance 알고리즘을 사용한다. Hamming Distance 알고리즘은 같은 길이의 두 String형 데이터 사이가 같은 위치에 속하는 특징점인지 측정하는 방법이다. 그림 1은 두 개의 String형 데이터를 나타낸다. 이 두 개 데이터를 XOR 연산 방식을 사용하여 비교한 후, 틀린 부분을 더하게 되면 distance를 구할 수가 있다. distance가 작으면 같은 특징점이라고 판별한다.

1	1	0	0	1	0	1	0

1	0	0	0	1	0	0	0

Fig. 1. Two Sring Data
그림 1. 두 개의 String 형 데이터

그림 2는 DB 영상에서 추출된 데이터와 카메라 입력 영상에서 생성된 데이터를 비교하는 그림이다.

그림 2와 같은 기존의 매칭 기법의 단점은 Input 영상에서 추출된 특징점 개수와 Database 영상에서 추출된 개수가 많아 질 수 록 속도가 저하된다는 점이다.

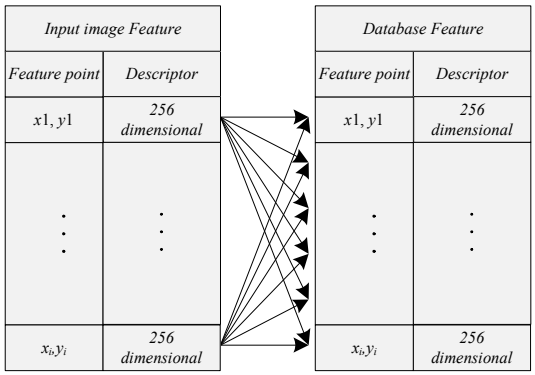

Fig. 2. Conventional Matching method
그림 2. 기존 매칭 기법

III. 개선된 매칭 기법을 적용한 ORB 알고리즘

1. ORB 알고리즘

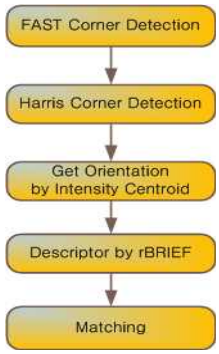

Fig. 3. Flow for ORB algorithm
그림 3. ORB 알고리즘 흐름도

ORB 알고리즘은 그림3과 같이 FAST Corner Detection 알고리즘[5] 사용하여 특징점을 추출한다.

FAST 알고리즘은 그림4와 같이 윈도우 영역을 잡고 중심점의 밝기 값과 16개의 밝기 값들을 비교한다. 우선 1, 5, 9, 13에 위치한 밝기 값을 비교하여 어둡거나 밝으면 1, 비슷하면 0을 주어 3 이상 되면 후보 특징점으로 선택한다. 그 후 나머지 위치한 밝기 값과 비교를 하여 12 이상이면 특징점으로 결정하게 되고, 각 특징점 주위로 이웃한 8개 픽셀 밝기 값과 비교하여 제일 큰 값을 가지고 있는 픽셀을 최종 특징점으로 결정한다. 추출된 특징점들에서 blob 특징

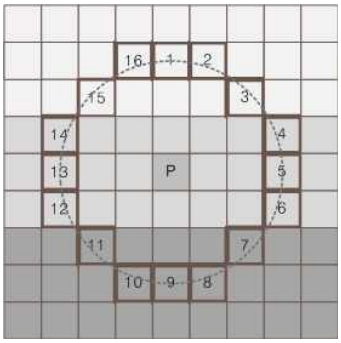

Fig. 4. FAST algorithm
그림 4. FAST 알고리즘

점을 제거하고, corner 특징점을 추출하기 위해 Harris Corner Detection[6]을 활용한다. Harris Corner Detection은 Corner 영역을 인식 할 수 있어 Corner만 추출이 가능 하다. 주 방향을 구하기 위해 Intensity Centroid[7]를 이용하여 방향을 결정해 준다.

$$m_{pq} = \sum_{x,y} x^p y^q I(x,y) \quad (1)$$

식 1은 Rosin이 정의한 momnets of a patch로 Intensity Centroid를 찾는데 사용된다.

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}}\right), \theta = atan2(m_{01}, m_{10}) \quad (2)$$

식 2의 C는 Intensity Centroid 위치를 뜻하고, θ는 방향 각도를 뜻한다. 둘 다 1차 모멘트를 활용하여 구하게 된다.

Descriptor 생성 단계는 rBRIEF 알고리즘을 사용하여 생성하게 된다. rBRIEF(rotation BRIEF)는 회전에 약한 BRIEF 알고리즘을 보완한 알고리즘이다.

BRIEF(Binary Robust Independent Elementary Features)[8] 알고리즘은 bit string description으로 구성된다.

$$\tau(p;x,y) := \begin{cases} 1 : p(x) < p(y) \\ 0 : p(x) \geq p(y) \end{cases} \quad (3)$$

식 3의 p는 smoothed image patch을 나타내고, τ는 binary test를 뜻 한다. p(x)와 p(y)는 x, y에 위치한 픽셀의 밝기 값을 나타낸다. 한마디로 두 위치의 밝기 값과 비교를 하여 작으면 1, 크거나 같으면 0인 binary test가 구해진다.

$$f_n(p) := \sum_{1 \leq i \leq n} 2^{i-1} \tau(p; x_i, y_i) \quad (4)$$

식 4의 n은 vector length을 뜻 한다. BRIEF 논문에서는 128, 256 그리고 512 vector length를 사용한다.

각 특징점 주위로 부분영상을 만들고, 주 방향을 이용하여 모든 비교점을 회전 시킨 후 bit string description을 구성하면, 유익성과 차별성이 떨어진다. 이 문제를 해결하기 위해 p(x)와 p(y)를 임의로 정하지 않고, 유익성과 차별성이 좋은 두 개의 픽셀 선택 패턴을 결정하여 학습데이터로 만들게 된다. 학습 데이터를 바탕으로 256개의 binary test를 구하여 descriptor를 생성한다.

2. Proposed Matching

매칭 알고리즘으로는 Hamming distance 알고리즘을 사용한다. DB 영상에서 생성된 특징점 과 Descriptor 그리고, Input 영상에서 생성된 특징점 과 Descriptor를 전체 비교할 경우, 다중 객체 인식하기에는 검사 할게 많아 실시간으로 구현하기에는 어려운 단점을 가지고 있다. 이 점을 보완하기 위해 ORB에서 추출된 특징점과 256bit 벡터 정보를 낮은 수부터 높은 수로 정렬한 후에 비교하게 된다. 그림 5는 정렬한 그림을 나타낸다. 왼쪽의 데이터를 오른쪽 데이터처럼 낮은 수부터 높은 수로 정렬 하면 근처에 있는 데이터들만 가지고 비교를 해도 일치하는 부분을 찾을 수 있어 속도가 향상 된다.

Feature point	*Descriptor*
70,50	...0000001010
⋮	⋮
200,400	···0010010100
400,500	···0000000010

Feature point	*Descriptor*
400,500	···0000000010
70,50	...0000001010
⋮	⋮
200,400	···0010010100

Fig. 5. Sorting Algorithm
그림 5. 정렬 알고리즘

또한 그림 6은 특징점들을 반으로 나눠 코어 별로 할당한 후 매칭단계를 병렬적으로 수행하는 그림이다[9].

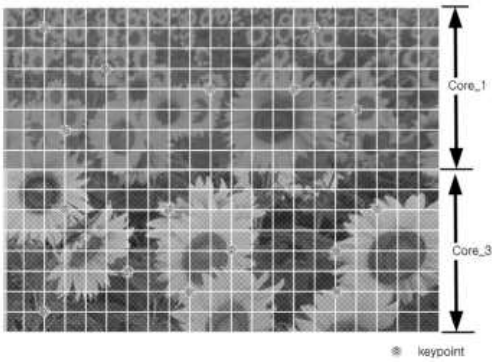

Fig. 6. Parallel Processing Technique
그림 6. 병렬처리 기법

그림 7은 위의 두 가지 방법을 적용한 방법이다. 특징점을 코어 별로 나눈 다음 근처의 데이터만 비교하여 속도를 개선 시켰다.

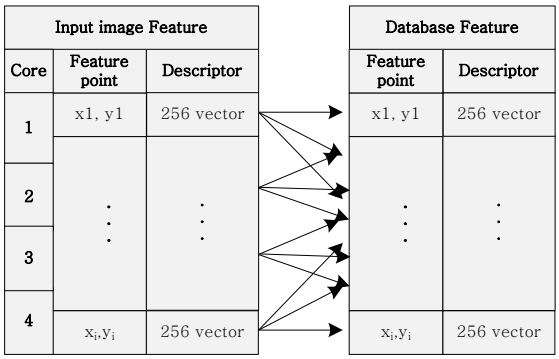

Fig. 7 Proposed Matching method
그림 7 제안하는 매칭 기법

Ⅳ.실험 결과 및 분석

그림 8은 ORB 다중객체인식 결과화면으로 여러 가지 물체들을 인식하는 것을 볼 수가 있다.

그림 9는 병렬처리 시 다중객체 매칭 단계 수행속도를 보여준다. 환경으로는 Intel i7 3.4GHz에서 수행하였다. DB의 이미지는 그림 8과 같이 꽃, 카드, 그래피티를 사용하였다. 추출 된 특징점 개수는 845, 861, 862로 총 2568개의 특징점과 카메라 영상의 특징점 450개를 비교하였다. 정렬과 매칭 알고리즘을 같이 사용했을 경우 쿼드 코어로 병렬 처리하면 19ms가 걸리고, 매칭만 했을 경우 41ms시간이 걸려 약 2배 정도 속도가 개선이 되었다.

Fig. 8 Result Screens for ORB
그림. 8 ORB 결과 화면

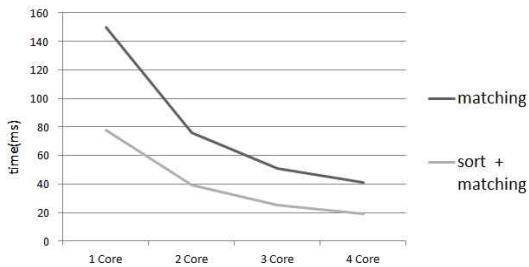

Fig. 9 Speed for Matching algorithm
그림 9 매칭 알고리즘 수행 속도

그림 10은 Cortex-A9 1.4GHz CPU에서 그림 8의 다중객체에 대하여 ORB 알고리즘과 매칭 단계를 수행한 결과이다. 논문 [9]와 비교한 결과 개선된 매칭 알고리즘으로 인해 수행 속도가 58ms로 향상된 것을 확인 할 수 있다.

그림 11은 논문 [10]과 비교한 결과이다. 논문 [10]은 Intel 2.8GHz 환경에서, 본 논문은 Intel 3.4GHz에서 수행 하였다. 논문 [10]은 OpenCV 라이

브러리를 이용한 매칭으로 DB 특징점이 361개, Input 특징점이 374일 때, 27.77ms 걸리고, 본 논문은 DB 특징점이 795개, input 특징점이 450 일 때, 8ms의 수행 속도를 보여준다.

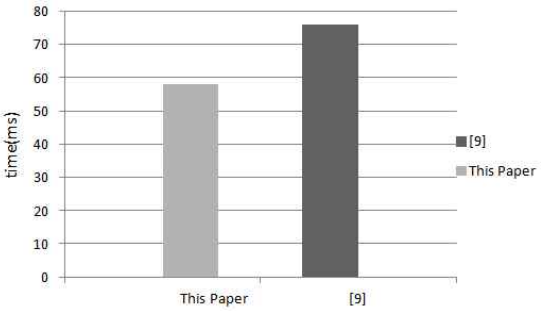

Fig. 10. Performance compare
그림 10. 수행 속도 비교

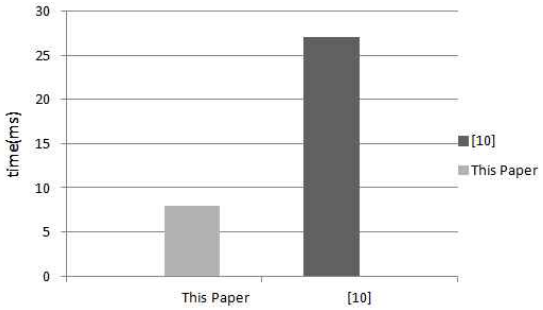

Fig. 11. Performance compare
그림 11. 수행 속도 비교

그림 12는 Intel i7 3.4GHz에서 수행한 결과와 Exynos4421 Cortex-A9 Quad Core 1.4GHz 프로세서가 장착된 ODROID-X 플랫폼에서 수행한 결과이다. 인텔 CPU 에서는 매칭 단계를 쿼드코어로 병렬 처리할 경우 초당 34 프레임을 처리 하고, Cortex-A9에서는 초당 18 프레임을 처리 한다.

V. 결 론

본 논문에서는 실시간으로 다중객체인식 알고리즘을 구현하였다. 매칭 기법의 수행 속도를 줄이기 위하여 몇 가지 방법을 적용하였고, 적용 결과를 다중객체에 대하여 인텔 CPU와 임베디드 CPU에서 비교 검증하였다. 그 결과 매칭 단계의 수행속도 시간이 2배 빠른 것을 확인 하였다.

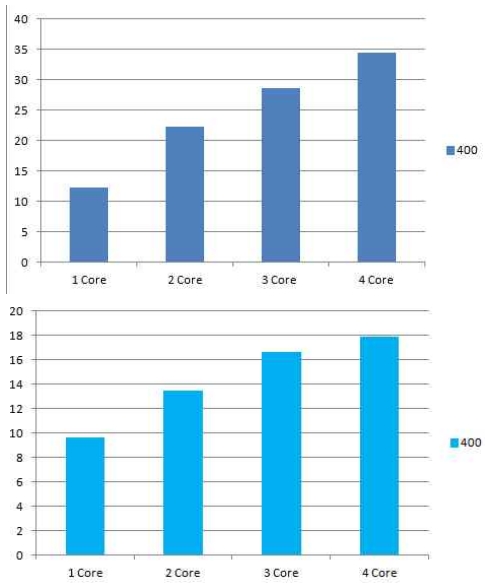

Fig. 12. Performing speed under PC and Cortex-A9
그림 12. PC 와 Cortex-A9 환경에서의 수행속도

참고문헌

[1] Dae Hoon Kim, Seung Min Rho, Een Jun Hwang, “Real-time Multi-objects Recongnition and Trancking Scheme”, Journal of KONI, Vol 16, No 2, 2012. 4

[2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints", International Journal of Computer Vision, 60(2):91-110. 1, 2

[3] H. Bay, E. Andreas, T. Tuytelaars and L. V. Gool, "Speeded-up roubst features", Computer Vision and Image Understanding, Vol 110, Issue 3, pp 346-359, June 2008

[4] E. Rublee, V. Rabaud, K. Konolige, "ORB: An efficient alternative to SIFT or SURF", Computer Vision and 2011 IEEE International Conference, pp 2564-2571, Nov 2011

[5] E. Rosten and T. Drummond. "Machine learning for high-speed corner detection", In European Conference on Computer Vision, volume 1, 2006. 1

[6] C. Harris and M. Stephens. A combined corner and edge detector. In Alvey Vision Confernce, pp 147-151, 1988. 2

[7] P. L. Rosin. Measuring corner properties. Computer Vision and Image Understanding, pp 291-307, 1999. 2
[8] M. Calonder, V. Lepetit, C. Strecha, and P .Fua, "Brief: Bi-nary robust independent elementary features". In European Conference on Computer Vision. 2010.
[9] , Heo Hoon, Kyung Gyu Taek, Jae Chang Kwak, Kwang Yeob Lee"Implementation of a robust direct object recognition and tracking algorithm", Proceeding of KCS2013, 2013. 1
[10] Ding Dajun, Chanho Lee, "Fast algorithm for Traffic Sign Recognition", Journal of IKEEE, pp 356-363, 2012. 12
[11] Hamming, Richard W. "Error detecting and error correcting codes", Bell System Technical Journal, pp 147-160, 1950.

BIOGRAPHY

Park Tae Ryong (member)

1985 : Hangyang University, Dept. of Mathmatics(BS)
1987 : Hangyang University, Dept. of Mathmatics(MS)
1995 : Hangyang University, Dept. of Mathmatics(Ph.D)
1994~ : Seokyeong Univeristy, Dept. of Computer Engineering, Professor
<Research interests> Crypto Algorithm, Computer Security, Computer Arithmetic, Recongnition Algorithm