Journal of Institute of Control, Robotics and Systems (2015) 21(1):21-27
http://dx.doi.org/10.5302/J.ICROS.2015.14.9055 ISSN:1976-5622 eISSN:2233-4335

Ego-Motion 보정기법을 적용한 쿼드로터의 화재 감지 알고리즘

Fire Detection Algorithm for a Quad-rotor using Ego-motion Compensation

이 영 완, 김 진 황, 오 정 주, 김 학 일*
(Young-Wan Lee[1], Jin-Hwang Kim[1], Jeong-Ju Oh[1], and Hakil Kim[1,*])
[1]Department of Information & Communication Engineering, Inha University

Abstract: A conventional fire detection has been developed based on images captured from a fixed camera. However, It is difficult to apply current algorithms to a flying Quad-rotor to detect fire. To solve this problem, we propose that the fire detection algorithm can be modified for Quad-rotor using Ego-motion compensation. The proposed fire detection algorithm consists of color detection, motion detection, and fire determination using a randomness test. Color detection and randomness test are adapted similarly from an existing algorithm. However, Ego-motion compensation is adapted on motion detection for compensating the degree of Quad-rotor's motion using Planar Projective Transformation based on Optical Flow, RANSAC Algorithm, and Homography. By adapting Ego-motion compensation on the motion detection step, it has been proven that the proposed algorithm has been able to detect fires 83% of the time in hovering mode.

Keywords: quad-rotor, ego-motion compensation, optical flow, RANSAC algorithm, homography, planar projective transformation

I. 서론

국토의 70%이상이 산악지형인 우리나라는 매년 크고 작은 산불로 인하여 인명피해 및 연간 250억 원의 금전적 피해를 입고 있다. 그래서 이러한 문제를 해결하기 위한 다양한 화재감시기술들이 개발되어 왔다. 온도센서, 후각센서, 연기센서, 열감지센서 등과 같이 센서기반의 화재 감지시스템[1,2]이 있다. 이 시스템은 광범위하게 센서노드들을 설치하여 이 센서노드들로부터 계측된 데이터를 이용하여 화재를 감시하는 것이다. 그러나 이러한 센서기반의 시스템은 산불이 커져야만 감지가 되기 때문에 산불이 커지는 것을 조기에 진압하지 못하는 단점을 갖고 있다.

한편 센서기반이 아닌 카메라를 이용한 영상 기반의 화재 감지시스템이 있다. 센서기반의 감시시스템과 비교했을 때 CCD 카메라와 같은 영상기반의 화재감시시스템[3-5]이 가진 장점은 여러 가지가 있다. 첫째, 열이나 연기가 커질 때까지 기다릴 필요 없이 화재를 감지할 수 있다. 둘째, 다른 센서들보다 CCD 카메라의 비용이 더 저렴하고, 이미 곳곳에 설치되어 있어서 여러 센서들을 설치하는 것보다 경제적으로 더욱 이득이다. 그러나 CCD 카메라는 카메라의 촬영 가능 구역 이외에서 발생하는 화재를 감지할 수 없는 한계를 갖고 있다. 이러한 한계를 극복하기 위해 더 많은 CCTV를 곳곳에 설치하기에도 경제적, 기술적 어려움이 크다.

이러한 고정 카메라 기반의 화재 감지시스템의 한계를 극복하려는 방법으로 본 연구는 카메라가 탑재된 쿼드로터를 이용한 화재 감지시스템을 제안한다. 이 시스템은 자유롭게 비행하며 촬영할 수 있기에 기존의 촬영 반경의 한계를 개선할 수 있다. 그러나 쿼드로터를 통해서는 움직이는 영상만을 촬영할 수 있는데, 이는 고정된 화면을 이용하는 기존의 화재 감지 알고리즘에 적용할 수가 없기 때문에 알고리즘의 개선이 요구된다.

그림 1은 기존의 화재 감지 알고리즘[4,5]을 쿼드로터를 이용한 화재 감지시스템에 맞게 보완한 흐름도이다. 기존의 화재 감지 알고리즘은 크게 3 단계로 구성된다. 먼저, 화재색상(color detection)을 감지하고 다음으로, 움직임을 감지(motion detection)한 후 마지막으로 화재진위여부판정을 한다. 두 번째 단계인 움직임감지 단계는 고정카메라의 특징을 이용한다. 다음 영상에서 이전 영상과의 차이로 움직임을 검출할 수 있는데, 이는 고정된 객체나 배경은 다음 영상이나 이전 영상에서 같은 위치에 존재하기에 이를 뺌으로써 픽셀값이 0이 된다. 반면에 움직임이 있는 객체는 0이 아닌 픽셀값을 갖게 되어 움직임이 감지되는데, 이를 차영상(difference image)이라 한다.

그러나 움직이는 카메라에서 촬영된 영상은 고정된 물체와 배경도 마치 움직이는 것처럼 인식된다. 때문에 차영상 기법을 적용했을 때 움직이는 실제 물체(그림 2의 불)뿐만 아니라 고정된 배경과 물체까지도 그림 2(d)와 같이 0이 아닌 차이값이 존재하게 된다. 그 결과로 실제로 움직인 물체를 감지할 수가 없게 된다.

* Corresponding Author
Manuscript received August 30, 2014 / revised September 15, 2014 / accepted September 22, 2014
이영완, 김진황, 오정주, 김학일: 인하대학교 정보통신공학과
(sun008sun@naver.com/gskjhyoo@naver.com/mandu45@gmail.com/hikim@inha.ac.kr)
※ 본 논문은 인하대학교 공학교육혁신센터에서 지원하여 연구하였음.

따라서 본 논문은 쿼드로터를 이용한 화재 감지시스템에 적용되는 화재 감지 알고리즘을 연구하는 데 목적이 있다. 앞서 기술한 문제를 해결하기 위해 본 논문은 Ego-motion 보정기법[6]을 움직임 검출 단계(motion probability estimation)에 도입한다. 이전 영상에 쿼드로터의 움직임 정도를 보정함으로써 마치 카메라가 움직이지 않은 것 같은 효과를 발휘하게 된다. 그 결과, 고정된 배경과 객체는 이전 영상과 다음 영상에서 동일한 위치에서 같은 화소값을 갖게 된다. 그림 2(e)는 Ego-motion 보정기법을 적용한 후 차영상기법이 정상적으로 적용되어 불씨가 움직이는 객체로 검출됨을 확인할 수 있다.

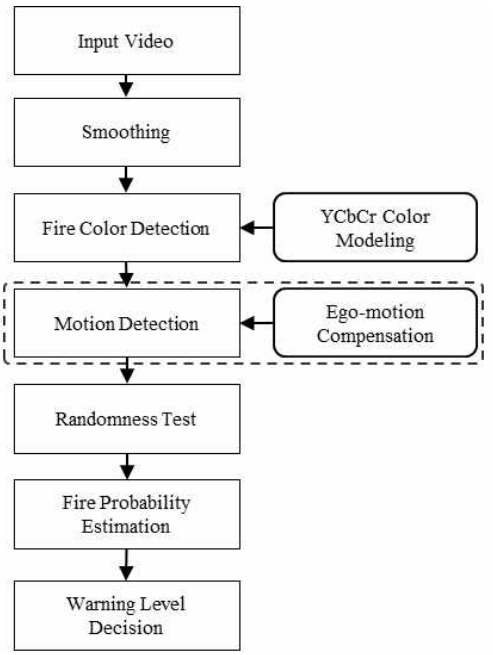

그림 1. 화재 감지 알고리즘 전체과정(점선, 본 논문의 개선사항).

Fig. 1. Fire Detection Algorithm(Dotted line is improvement part).

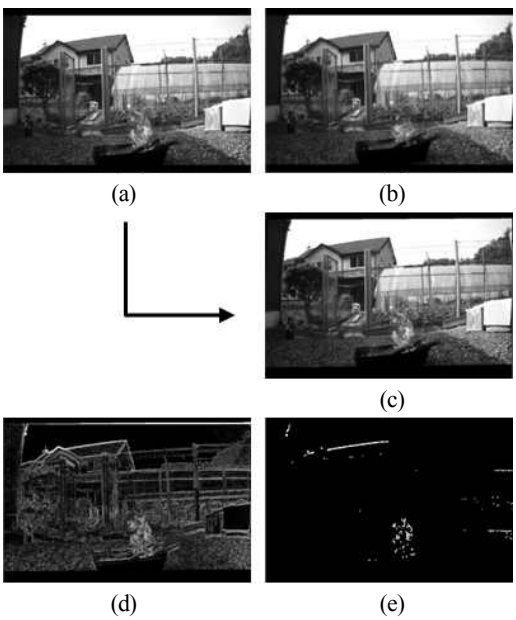

그림 2. (a) 이전 영상, (b) 다음 영상, (c) 보정된 이미지, (d) (b)와 (a)의 차영상, (e) (b)와 (c)의 차영상.

Fig. 2. (a) image at *t*-1, (b) image at *t*, (c) compensated image of (a), (d) difference image between (a) and (b), (e) difference image between (c) and (b).

본 논문의 구성을 살펴보면 다음과 같다. II 장에서는 쿼드로터를 이용한 화재 감지시스템에 대해서 기술하고, III 장에서부터 본격적으로 화재 감지 알고리즘을 위한 Ego-motion 보정기법에 관하여 논한다. IV 장에서는 실험결과 분석을 하며 V 장에서 결론을 맺는다.

II. 화재 감시 시스템

본 연구에 사용한 쿼드로터는 Wi-Fi 통신을 통해 컴퓨터와 실시간으로 영상정보와 비행제어를 위한 센서취득값 및 제어신호값을 송수신한다. 쿼드로터를 이용한 화재 감시 시스템의 흐름도는 그림 3과 같다. 쿼드로터는 감시하고자 하는 위치의 GPS 좌표들을 입력 받고 이륙을 한다. 이후 지정된 경로들을 순차적으로 자동비행하는 Way-point 비행모드를 수행한다. 이때 쿼드로터에 장착된 전방 카메라를 통해 수집된 2차원 영상으로부터 화재 색상 검출 단계를 수행한다. 이 단계는 YCbCr Color Modeling을 통하여 비행 중에 색상으로만 화재후보를 검출하는 단계이다. 만일 색상이 검출 되지 않을 경우 계속해서 Way-point 비행을 하며 색상검출을 수행하고, 화재 색상이 검출될 경우 Way-point 비행모드에서 Hovering 비행(제자리 비행)모드로 전환하여 기체가 제자리에서 화재를 촬영하도록 한다. Hovering 비행모드에서 본격적으로 그림 1의 화재 감지 알고리즘(fire

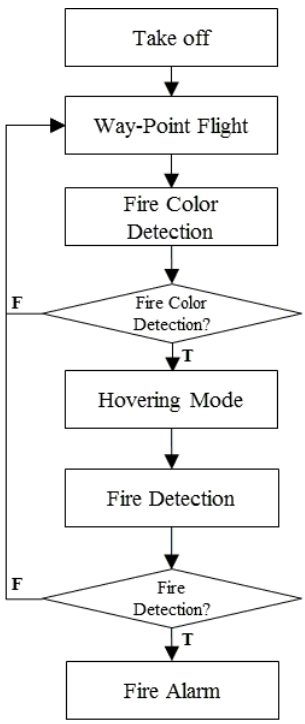

그림 3. 화재 감시 시스템 흐름도.

Fig. 3. Fire Detection System Flow Chart.

detection algorithm)이 적용된다. 가장 먼저 화재 색상을 다시 한 번 검출하고, 그 다음으로 화재에 대한 움직임을 검출한 후에 Randomness Test를 통해 최종적으로 화재 판정을 하게 된다. 만약 판정 단계에서 화재로 판정될 경우 컴퓨터가 이를 알리게 된다. 그러나 화재로 판정되지 않을 경우 다시 Way-point 비행모드로 전환되어 남은 순찰 경로를 비행한다.

III. Ego-Motion 보정기법

Ego-motion 보정기법은 4 단계로 구성되어 있다. 그림 4는 이 단계들을 보여준다. 첫째, 영상에서 특징점을 추출한다. 둘째, 옵티컬플로우(optical flow)를 이용하여 다음 영상에서 특징점이 어디로 이동했는지 추적한다. 셋째, 추적한 점들 중에 벗어난 점(outliers)을 제거하고 Homography 행렬을 구하기 위해 RANSAC 알고리즘을 적용한다. 넷째, Planar Projective Tranformation [7]을 통해 움직임 정도(Homography 행렬)를 이전 영상에 보정한다.

1. Feature Tracking

연속된 영상 간의 움직인 정보를 파악하기 위해 특징 점을 추출하고, 그 점을 추적한다. 특징점을 추출하는 방법으로는 Shi와 Tomasi의 방법[8]을 사용하고, 특징점을 추적하는 방법으로는 계산의 효율이 높은 Lucas-Kanade [9]의 OpticalFlow Algorithm을 사용한다.

1.1 Lucas-Kanade Algorithm

각 픽셀의 움직임을 추적하는 옵티컬플로우는 다양한 방식이 있다. 그 중 연산량이 적어 널리 사용되는 방법은 1981년에 제안된 루카스-카나데(LK: Lucas-Kanade) 알고리즘[9,10]으로, 아래 세 가지 가정에 기초를 둔다.

- 밝기 항상성(brightness constancy)은 어떤 객체의 픽셀은 프레임이 바뀌어도 밝기값이 변하지 않는다는 것을 말한다.
- 시간 지속성(temporal persistence)은 영상에서 객체의 움직임에 비하여 시간의 변화가 더 빠르게 진행되어 연속된 프레임 사이에서 객체의 이동량이 많지 않음을 의미한다.

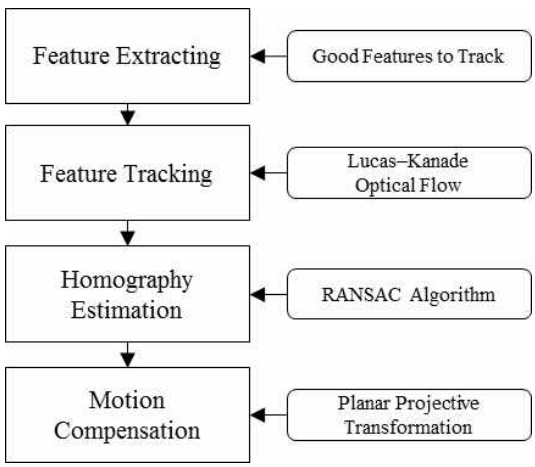

그림 4. Ego-motion Compensation 흐름도.
Fig. 4. Ego-Motion Compensation Flow Chart.

- 공간 일관성(spatial coherence)은 공간적으로 인접한 점들은 동일한 객체에 속할 가능성이 높아 일관된 움직임을 갖는 것을 의미한다.

1.2 Image Pyramid

쿼드로터에 탑재된 카메라는 24Hz로 동작하는데 크고 불규칙한 움직임이 빈번하게 발생하여 앞에서 제시한 작고 일관된 움직임이라는 가정을 위배하게 된다. 이러한 커다란 움직임을 검출하기 위해 큰 윈도우를 사용해야 하지만 이는 일관된 움직임이라는 가정을 깨기 쉽다. 일련의 문제들을 해결하기 위해 영상 피라미드(image pyramid)를 사용할 수 있다. 영상 피라미드는 하나의 원본 영상을 원하는 단계까지 다운샘플링(downsampling)하여 생성한 영상들의 집합이다. 먼저 피라미드의 최상위 계층부터 옵티컬플로우의 계산을 시작함으로써 작고 일관된 움직임이라는 가정에 위배되는 문제를 완화시키며, 상위 계층에서 측정된 정보를 하위 계층에서의 움직임 추정 시 시작점으로 사용한다. 이러한 연산을 피라미드 최하위 계층에 도달할 때까지 반복적으로 수행함으로써 앞서 제시한 움직임 가정을 위반하는 경우를 최소화하고, 빠르고 큰 움직임도 추적할 수 있게 된다.

1.3 Tracking

이전 영상에서 추출된 특징점들이 다음 영상에서 어디로 이동했는지 추적한다. 이 추적된 점과 이전 영상에서의 특징점을 잇는 붉은색 선을 그려 (그림 5(a)) 움직임 정도를 파악할 수 있다. 여기서 붉은색 선들의 방향을 관찰해보면, 현재 카메라가 어떤 방향으로 이동하고 있는지 추측할 수 있다. 붉은 선들 중에 대부분이 아래 방향으로 향하고 있음을 확인할 수 있다. 이는 촬영하고 있는 쿼드로터가 아래 방향으로 이동하고 있기 때문이다. 한편, 아래 방향으로 향하는 선이 지배적인 반면, 곳곳에 전혀 엉뚱한 방향으로 향하는 선들이 있음을 확인할 수 있는 데, 이는 추적에서 벗어난 점이다. 이는 RANSAC("RANdom SAmple Consensus") 알고리즘[11]을 이용해 제거할 수 있다(그림5(b)).

2. Homography Estimation

연속된 두 영상 간의 관계를 나타내는 Homography를 통하여 움직임 정도를 나타낼 수 있다. 이 Homography는 앞서 추적한 특징점을 기반으로 추정할 수 있다. RANSAC 알고리즘을 통해 특징점들 중 벗어난 점을 제거하고, Homography 행렬의 최적의 모델을 추정할 수 있다.

2.1 RANSAC Algorithm

Fischler와 Bolles [11]에 의해서 제안된 RANSAC 알고리즘은 측정 노이즈가 심한 원본 데이터로부터 모델 파라미터를 예측하는 방법이다. RANSAC은 전체 원본 데이터 중에서 모델 파라미터를 결정하는 데 필요한 최소의 데이터를 랜덤하게 샘플링한다. 샘플링 된 데이터를 이용하여 반복적으로 해를 계산함으로써 최적의 해를 찾는다. 이 방법은 전통적인 통계적 방법과는 반대의 개념을 가진다. 대부분의 전통적인 방법들은 초기의 해를 획득하기 위해서 가능한 많은 데이터를 사용하고 그 결과로부터 유효하지 않은 데이터(outliers)를 제거한다. 반면에 이 방법은 가능한 적은 양의 초기 데이터를 사용해서 일관된 데이터의 집합(consensus set)을 확장해가는 방식을 사용한다.

옵티컬플로우를 통해 이전 점과 추적된 점의 쌍이 하나의 대응점을 이룬다. 이 대응점들을 이용하여 RANSAC 알고리즘은 영상의 움직임 정도를 뜻하는 Homography 행렬을 예측하게 된다. RANSAC 알고리즘은 전체의 대응점 후보들 중에서 Homography 행렬을 결정하는 데 반드시 필요한 최소한의 대응점을 랜덤하게 반복적으로 샘플링하면서 최적의 모델 파라미터인 Homography 행렬을 구하게 된다. RANSAC 알고리즘의 수행 순서는 다음과 같다.

Step 1: 전체 대응점 후보들로부터 N개의 샘플 대응점 획득

Step 2: 획득한 샘플 대응점을 참값으로 가정하고 Homography 행렬(모델 파라미터)을 예측

Step 3: 예측된 Homography 행렬을 이용하여 Consensus Set을 획득 (Outliers 제거)

Step 4: 예측된 Homography 행렬이 옳은지 판단하고, 참값이 아닐 경우 Step 1 ~ 3 반복

위의 과정 Step 1에서 N은 영상 보정 단계에서 사용되는 Homography 행렬을 구하는 데 필요한 4개의 대응점이다.

2.2 Homography 행렬 추정

Homography 행렬(H)는 Planar Projective Transformation을 통해 식을 세울 수 있다.

$$\begin{bmatrix} x'_1 \\ x'_2 \\ x'_3 \end{bmatrix} = \begin{bmatrix} h_{11} h_{12} h_{13} \\ h_{21} h_{22} h_{23} \\ h_{31} h_{32} h_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \tag{1}$$

$$X' = HX$$

H 행렬은 8자유도이기 때문에 식 (1)에서 h_{33}을 스케일로 보고 1로 설정한다. 식 (2)에서 u_1은 이전 영상의 x좌표, v_1는 y좌표이다.

$$\begin{bmatrix} x_{r1} \\ x_{r2} \\ x_{r3} \end{bmatrix} = \begin{bmatrix} h_{11} h_{12} h_{13} \\ h_{21} h_{22} h_{23} \\ h_{31} h_{32} \ 1 \end{bmatrix} \begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix} \tag{2}$$

(2)를 이용하여 (3), (4)의 x_1, y_1을 구할 수 있다.

$$x_1 = \frac{x_{r1}}{x_{r3}} = \frac{h_{11}u_1 + h_{12}v_1 + h_{13}}{h_{31}u_1 + h_{32}v_1 + 1} \tag{3}$$

$$y_1 = \frac{x_{r2}}{x_{r3}} = \frac{h_{21}u_1 + h_{22}v_1 + h_{23}}{h_{31}u_1 + h_{32}v_1 + 1} \tag{4}$$

(x_1, y_1)은 다음 영상에서의 (x', y')좌표이다. 식 (3), (4)의 식을 전개하면 식 (5), (6)이 된다.

$$\begin{aligned} x_1(h_{31}u_1 + h_{32}v_1 + 1) &= h_{11}u_1 + h_{12}v_1 + h_{13} \\ u_1h_{11} + v_1h_{12} + h_{13} - u_1x_1h_{31} - v_1x_1h_{32} &= x_1 \end{aligned} \tag{5}$$

$$\begin{aligned} y_1(h_{31}u_1 + h_{32}v_1 + 1) &= h_{21}u_1 + h_{22}v_1 + h_{23} \\ u_1h_{21} + v_1h_{22} + h_{23} - u_1v_1h_{31} - v_1y_1h_{32} &= y_1 \end{aligned} \tag{6}$$

식 (5), (6)을 정리 하면 식 (7), (8)과 같이 된다.

$$v_ih_{11} + v_ih_{12} + h_{13} - u_ix_ih_{31} - v_ix_ih_{32} = x_i \tag{7}$$

$$u_ih_{21} + v_ih_{22} + h_{23} - u_iy_ih_{31} - v_iy_ih_{32} = y_i \tag{8}$$

(단, i = 1, 2, 3, 4로 대응점의 인덱스를 의미)

식 (7), (8)을 이용하여 H 행렬을 구하는 연립방정식의 행렬식 (9)를 세울 수 있다. 이 8×8행렬의 역행렬을 양변에 곱하면 H 행렬을 구할 수 있다. 위와 같은 일련의 과정을 RANSAC 알고리즘의 Step 2 Homography 행렬을 예측하는 과정에서 수행하게 된다.

$$\begin{bmatrix} u_1 v_1 1 0 0 0 -u_1x_1 - v_1x_1 \\ 0 0 0 u_1 v_1 1 -u_1y_1 - v_1y_1 \\ u_2 v_2 1 0 0 0 -u_2x_2 - v_2x_2 \\ 0 0 0 u_2 v_2 1 -u_2y_2 - v_2y_2 \\ u_3 v_3 1 0 0 0 -u_3x_3 - v_3x_3 \\ 0 0 0 u_3 v_3 1 -u_3y_3 - v_3y_3 \\ u_4 v_4 1 0 0 0 -u_4x_4 - v_4x_4 \\ 0 0 0 u_4 v_4 1 -u_4y_4 - v_4y_4 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \\ y_3 \\ x_4 \\ y_4 \end{bmatrix} \tag{9}$$

2.3 Consensus Set 획득 및 Homography 행렬최종결정

전 단계에서 구한 H 행렬을 이용하여 모든 대응점 후보를 식 (1)에 적용하여 x'점을 구한다. 이 x'의 점과 옵티컬플로우로 추적된 점과의 거리를 구하여 일정 기준보다 클 경우, Outlier로 여기고 이를 Consensus Set에 포함시키지 않는다. 이러한 과정을 반복함으로써 노이즈를 제거하고, Consensus Set을 구하게 된다.

Outlier들을 제거한 후 Consensus Set을 이용하여 예측한 H 행렬의 정확도가 95% 이상일 경우 이 행렬을 영상보정 단계인 다음 단계에서 사용하게 된다. 그림 5는 Outlier들을 제거하기 전과 제거한 후 선을 그린 결과이다.

3. Motion Compensation

3.1 Planar Projective Transformation

전 단계에서 구한 최종 Homography 행렬을 식 (1)에 이용하여 이전 영상에 움직임 정도를 보정하게 되는데, 이를 Planar Projective Transformation이라 한다. 식 (10)과 같이 보정 할 경우, 이전 영상과 다음 영상은 카메라가 마치 움직이지 않은 것처럼 효과를 발휘하게 된다. 그 결과로 식 (11)과 같이 두 영상을 뺐을 때 움직이는 물체만 픽셀값이 존재하고 나머지는 '0' 값, 검은 배경이 됨을 예상할 수 있다.

$$I_c(t) = H * I(t-1) \tag{10}$$

그림 5. (a) RANSAC 알고리즘을 적용하기 전, (b) RANSAC 알고리즘을 적용한 후 비교. (b)에는 (a)에서 보이는 outlier들이 제거된 모습을 확인할 수 있다.

Fig. 5. (a) OpticalFlow without RANSAC, (b) OpticalFlow with RANSAC.

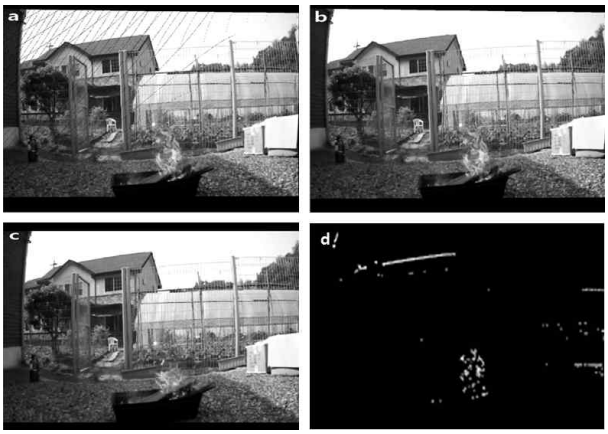

그림 6. (a) Homography로 보정된 영상 $Ic(t)$, (b) Inverse Homography로 보정된 영상, (c)다음 영상 $I(t)$, (d) 차영상 $I_{diff}(t)$.

Fig. 6. (a) Image compensated with Homography $I_c(t)$, (b) Image compensated with Inverse Homography, (c) next image $I(t)$, (d) Difference image $I_{diff}(t)$.

$$I_{diff}(t) = I(t) - I_c(t) \tag{11}$$

3.2 Inverse Homography

그런데 그림 6(a) 같이, Homography를 이용하여 보정하였을 때 이미지에 구멍들이 생기는 문제가 발생한다. 이는 RANSAC Algorithm 단계에서 Homography 행렬을 확정 지을 때 95%의 정확도로 결정하지만 5%의 오차율로 인하여 발생하는 문제이다. 이 문제는 Inverse Homography를 이용하여 해결할 수 있는데, 그림 6(b)에 그 결과가 나타나 있다.

3.3 보정 후 차영상 및 후처리

이전 영상을 보정 한 후 다음 영상과의 차를 계산하였을 때 그림 6(d)와 같이, 화재의 픽셀들이 검출됨을 확인할 수 있었다. 그러나 화재가 아닌 5%의 보정 오차율로 인하여 생긴 몇몇 움직이지 않은 특징점 들까지 움직이는 물체로 인식되는 문제가 발생한다. 이러한 문제를 해결하기 위해 후처리 과정을 거친다. 가장 먼저 가우시안 스무딩을 통하여 노이즈를 제거한다. 다음으로 밝기값이 낮은 픽셀들은 화재가 아닌 것으로 판단하여 문턱치 처리를 이용하여 제거하고 모폴로지 처리를 위해 이진 영상을 만든다. 문턱치 처리 결과로 생긴 이진영상을 침식(Erosion), 열기(Opening), 팽창(Dilation)순으로 모폴로지 처리를 하여 화재가 아닌 점들은 제거하고, 화재에 해당하는 픽셀만 부각되도록 만들었다.

IV. 실험 및 결과 고찰

1. 실험환경 및 제약사항

본 연구가 제안한 알고리즘은 쿼드로터가 화재 색상을 감지한 후 Hovering 비행모드로 전환하였을 때 수행된다. 화재를 감지하는 데 필요한 최소 화소 수를 실험적으로 측정했을 때 최소 15 화소값 이상이어야 화재를 감지할 수 있음을 확인하였다. 실험에 사용되는 화통의 규격은 가로 1.5m, 세로 0.5m이며, 화재의 높이는 약 0.4m이다. 이러한 화통과 화재의 규격을 고려하였을 때 최소 화소값을 안정적으로 감지하려면 10m 이내에서 실험을 해야 한다. 물론, 화통의 규격과 상관없이 화재의 규격이 더 컸더라면 더 먼 거리에서도 최소 화소값을 얻을 수 있었겠지만 화재실험의 위험성과 공간의 제약성을 고려하였을 때 화재의 규모를 제한할 수 밖에 없는 실험의 제약성을 갖고 있다. 결과적으로, 화통과 기체와의 거리를 5.0m, 7.0m로 실험하며 쿼드로터에서 직접 촬영한 DB를 획득하였다. 실험에 사용한 쿼드로터는 AR.DRone 2.0기종을 사용하였고, 이 쿼드로터에 탑재된 카메라는 초당 24프레임, 640*480해상도를 갖는다. 이 알고리즘을 적용한 PC는 Intel Core i3 1.3Ghz CPU를 사용하며, Visual Studio 2010과 OpenCV 2.8v을 사용하였다.

(a) 5m distance.

(b) 7m distance.

그림 7. 색상 검출 확률, 움직임 검출 확률, 최종 화재 검출 확률을 포함한 실험 결과들의 예.

Fig. 7. (a) Exaples of the experiment results including result images, color probabilities, motion probabilities and flame probabilities from left to right, respectively.

2. 화재 감지 실험

기존의 화재 감지시스템에서 사용하는 성능평가 기준은 화재를 감지하는 데 걸리는 시간이며, 평균 10초 이내로 화재를 검출하는 것을 목표로 한다. 따라서 본 연구는 화재를 감지 하는 데 걸리는 시간(Alraming Time)과 Ego-motion 보정기법을 적용한 개선 알고리즘과 그렇지 않은 기존의 알고리즘을 각각 쿼드로터 시스템에 적용하였을 때 화재검출 여부를 성능평가기준으로 정하였다. 본 연구가 개선한 기존의 화재 감지 알고리즘[4]은 고정형 CCD 카메라를 이용한 화재 감지시스템에 적용되는 알고리즘이다.

3. 실험 결과 고찰

표 1, 2는 각각 거리가 5m, 7m에서 Hovering 비행모드 중인 쿼드로터에서 촬영된 DB들을 이용하여 화재 감지 알고리즘을 적용한 결과이다. 그림 7은 알고리즘의 구현 결과 영상이다. 먼저 각 픽셀의 컬러 확률값을 구한 후 이를 모션 확률값을 구한 결과와 곱한다. 그 곱한 결과를 가지고 Randomness Test를 거쳐 화재 판정 단계를 결정한다. 화재 판정 단계는 확률값에 따라 FIre, Alert, Watch, Not_Fire 4단계로 구성된다. 화재로 검출된 모든 DB는 모든 화재 판정 단계를 거쳐 최종적으로 Fire로 판정되어 화면에 붉은 글씨로 Fire 문자가 출력된다. 그림 7(a), (b)의 첫 번째 화면에서는 이를 보여준다. 표 3는 검출률과 Alarming Time의 평균을 나타낸다. 실험 결과, 기존의 알고리즘은 화재검출이 되지 않는다. 그 이유는 알고리즘 자체가 정지카메라를 이용한 화재 감지시스템을 위한 알고리즘이여서 비행하는 쿼드로터에서 촬영된 영상으로는 화재검출이 어렵기 때문이다. 한편, 개선된 알고리즘의 평균 화재 검출률은 약 83%이고, 평균 Alarming Time은 약 8.5초로 측정되었다.

표 1. 5m 거리 DB 화재 감지결과.

Table 1. The result of fire detection on 5m distance.

거리 (m)	DB	검출 여부		Alarming Time (sec)
		기존	개선	
5	01	x	o	7.728
	02	x	o	4.379
	03	x	o	8.698
	04	x	o	9.735
	05	x	o	4.490
	06	x	o	4.566
	07	x	o	5.566
	08	x	o	8.663
	09	x	o	4.312
	10	x	o	4.221
	11	x	o	5.231
	12	x	o	4.223
	13	x	o	6.221

표 2. 7m 거리 DB 화재 감지결과.

Table 2. The result of fire detection on 7m distance.

거리 (m)	DB	검출 여부		Alarming Time (sec)
		기존	개선	
7	14	x	x	-
	15	x	x	-
	16	x	x	-
	17	x	o	10.334
	18	x	o	8.135
	19	x	o	9.112
	20	x	o	14.231
	21	x	o	7.123
	22	x	x	-
	23	x	o	18.254
	24	x	o	10.261

표 3. 화재 감지 평균 검출률과 평균 Alarming Time.

Table 3. Average Detection Rate & Averaging Alarming Time.

거리 (m)	검출률(%)		Alraming Time (sec)
	기존	개선	
5	0	100.00	6.003
7	0	63.64	11.064
평균	0	83.33	8.533

V. 결론

본 논문은 기존의 정지영상카메라를 이용한 화재 감지시스템의 한계를 극복하는 수단으로서 쿼드로터를 이용하는 화재 감지시스템에 적용되는 알고리즘을 제안한다. 기존의 정지영상 카메라를 이용한 화재 감지시스템은 차영상기법을 이용하여 화재를 감지하지만, 이는 움직이는 카메라를 이용한 화재 감지시스템에 적용할 수가 없다. 이를 해결하기 위한 방법으로 Ego-motion 보정기법을 움직임 검출 단계(Motion Detection)에 적용하여 카메라가 이동하여도 차영상기법이 적용될 수 있도록 카메라의 움직임을 보정하였다. 쿼드로터가 호버링 비행모드일 때 촬영한 DB를 이용하여 제안한 화재 감지 알고리즘으로 실험하였을 때 화재검출율은 약 83%이고, 화재 감지시간(Alraming Time)은 약 8.53초로 성능이 검증되었다. 그러나 쿼드로터의 실시간성이 떨어져 와이파이통신으로 영상 정보를 주고받는 데 딜레이가 발생한다. 이를 해결하기 위해 쿼드로터 자체에서 영상처리를 수행하도록 임베디드시스템을 구축함으로써 성능을 향상시키는 것이 향후 과제이다.

REFERENCES

[1] A. Zhao, L. Wang, and C. H. Yao, “Research on electronic-nose application based on wireless sensor networks,” *International Symposium on Instrumentation Science and Technology*, 2006.

[2] D. Y. Yun and S. H. Kim, “A design of fire monitoring system based on unmaned helicopter and sensor network,” *Journal of Korean Institute of Intelligent Systems*, vol. 20, no. 4, pp. 516-521, 2010,

[3] K. H. Cheong et al., “Automatic fire detection system using CCD camera and Bayesian network,” *Image Processing: Machine Vision Applications Book Series: Proceedings of SPIEIS&T Electronic Imaging*, vol. 6813, pp. S8130-S8130, 2008.

[4] D. Wang, X. Cui, E. Park, C. Jin, and H. Kim, “Adaptive flame detection using randomness testing and

robust features," *Fire Safety Journal*, vol 55, pp. 116-125, Jan. 2013.

[5] T. Celik, "Fast and efficient method for fire detection using image processing," vol. 32, no. 6, pp. 881-890, Dec. 2010.

[6] B. Jung and G. S. Sukhatme, "Real-time motion tracking from a mobile robot," *International Journal of Social Robotics*, vol. 2, no. 1, pp. 63-78, Mar. 2010.

[7] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd Ed, CAMBRIDGE, Cambridge, 2003.

[8] J. Shi and C. Tomasi, "Good features to track," *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, Seattle, Washington, pp. 593-600, Jun. 1994.

[9] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," *Proc. of the 1981 DARPA Imaging Understanding Workshop*, pp. 121-130, 1981.

[10] G. R. Bradski and A. Kaehler, *Learning OpenCV*, O'Reilly, 2008.

[11] M. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting applications to image analysis and automated cartography," *Proc. Image Understanding Workshop*, pp. 71-88, Apr. 1980.

이 영 완

2007년~현재 인하대학교 재학 중. 관심분야는 영상처리, 머신 비젼, 컴퓨터 비젼.

김 진 황

2009년~현재 인하대학교 재학 중. 관심분야는 영상처리, 로봇 비젼, 머신 비젼, 로봇 제어.

오 정 주

2012년~현재 인하대학교 재학 중. 관심분야는 영상처리, 컴퓨터 비젼.

김 학 일

1983년 서울대학교 제어계측 공학과 졸업(공학사). 1985년 Purdue Univ. 전기/컴퓨터 공학과 석사. 1990년 Purdue Univ. 전기/컴퓨터 공학과 박사. 1990년~현재 인하대학교 공과대학 교수. 관심분야는 바이오인식, 머신비젼, 지능형 영상감시.