컴퓨터비전을 활용한 실내 자동비행체에 관한 연구

최영, 김계영
숭실대학교 컴퓨터학부
e-mail: fjuhy0@gmail.com

A Study on Autonomous Indoor Flight using Computer Vision System

Young Choi, Kye-Young Kim
Dept of Computer Science, Soong-Sil University

요 약

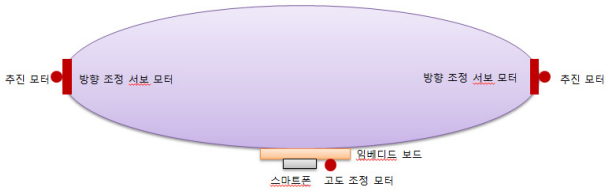
본 논문에서는 실내 환경에서 특정한 장소를 찾아갈 수 있는 자동비행체를, 컴퓨터비전과 스마트폰, 헬륨을 사용하여 구현 하는 방법을 제안한다. 마커를 이용하여 빌딩 내 각 정점을 표시하며, 이를 활용하여 자동으로 최단의 경로를 찾아서 그 경로를 따라 비행하는 알고리즘과 비행체의 구조를 보인다. 실험 결과 다양한 방면으로 적용 가능한 유의미한 결과를 얻었다.

1. 서론

실내의 특정한 장소를 이동할 수 있는 이동체는 다방면으로 연구가 진행되고 있지만, On-Board 특성상 저성능의 CPU를 탑재할 수밖에 없으므로 연산 처리에 어려움이 있다. 또한, 바퀴나 다리 형식의 이동물체는 계단과 같은 실내의 장애물들을 회피하기 어려우므로 이동에 많은 제약사항이 있다. 본 프로젝트는 이러한 문제를 해결하기 위하여 스마트폰을 이용한 헬륨 풍선 형태의 저속 자동비행체 개발을 목표로 한다. 스마트폰은 카메라를 비롯해 여러 센서를 가지고 있으므로 비행체는 상대적으로 저렴한 단가에 제작될 수 있다. 또한, 일반 비행체는 제자리 대기(Hovering)가 어렵고 추락의 위험성이 있으나 본 프로젝트에서는 비행 시 필요한 부양력을 프로펠러가 아닌 헬륨으로 충당하여 이러한 문제를 해결하고자 한다.

2. 비행체의 형태 및 크기

본 연구에서는 상하, 전후, 좌우, 회전 이동이 가능한 일반적인 형태의 비행체의 형태로써 (그림 1)과 같이 눕혀진 회전 타원체(Prolate Spheroid)의 기낭에 앞, 뒤, 하단 각각 모터를 장착하고 스마트폰을 기체 하단 중앙에 부착한 형태를 사용한다.



(그림 1) 비행체의 구조

이때 기낭의 크기는 최소한 비행체 전체를 들어 올릴 수 있을 만큼의 헬륨이 유입 가능한 크기를 가져야 한다. a가 회전타원체의 단축이고 c가 회전타원체의 장축일 때 회전타원체의 부피는 [식 1]과 같다.

$$V_s = \frac{3}{4}\pi a^2 c \qquad \text{[식 1]}$$

또한 t가 기낭 원단의 두께이고 g가 기낭 원단 재질의 비중일 때 기낭의 무게는 [식 2]와 같다.

$$M_s = tg2\pi a^2\left(1+\frac{c}{ae}sin^{-1}e\right) \qquad \text{[식 2]}$$

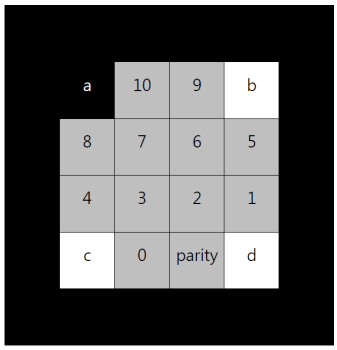
여기서 $e^2 = 1-\frac{a^2}{c^2}$

헬륨 $1m^3$ 당 부양력은 1,127g으로 알려졌으므로 기낭의 부피에 따른 부양력은 $V_c \times 1127$g이 된다. 따라서 기낭을 제외한 비행체의 나머지 무게를 M_o라고 한다면 V_s는 아래의 [식 3]을 만족하여야 한다. (단위 : m^3)

$$V_s > \frac{M_s + M_o}{1127} \qquad \text{[식 3]}$$

3. 마커의 필요성 및 인식 방법

비행체가 각 정점(현 위치, 목적지 등)을 인식할 수 있도록 하기 위해서는 정점의 위치 나타내는 표식이 필요하다. 본 연구에서는 비행체가 비행하는 빌딩의 각 정점의 바닥에 (그림 2)와 같은 형태의 마커를 미리 설치해 두고 비행체에 부착된 스마트폰의 카메라로 인식함으로써 이러한 문제를 해결하도록 한다.

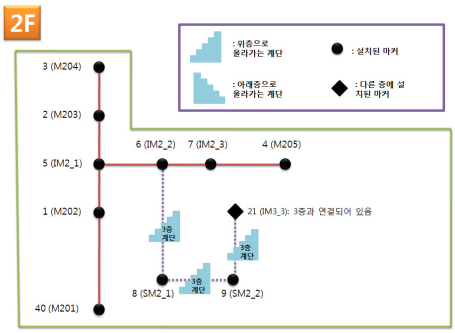


(그림 2) 마커의 예

마커의 인식은 양광웅[1]이 제안한 내용에 따른다. 마커의 테두리를 이루는 사각형들은 항상 흑색으로 표현되어 마커의 영역임을 인식하게 하며 마커의 가장 좌상단을 항상 흑색으로 표현함으로써 마커의 방향을 표현한다. (그림 2)의 중앙 영역은 bit 번호를 의미하며 이 bit 조합으로 마커의 고유한 ID를 표현한다.

4. 가상맵 및 자동 경로 탐색

비행체의 자동 비행을 위해 빌딩 내 각 인접경로는 미리 데이터베이스에 저장되어 있을 필요가 있다. 저장되는 정보는 인접경로의 이름(SRC to DEST), 목적지의 방향(degree), 경사(계단)이동 유무(상승, 하강, 유지)이다.



(그림 3) 가상맵의 개념도

저장된 데이터베이스의 정보를 가지고 위의 (그림 3)과 같은 개념의 가상맵을 생성한다. 가상맵은 그래프 자료구조로 이루어지며 경로 탐색에 가장 널리 쓰이는 Dijkstra's Algorithm[2]을 적용하여 비행체가 최적의 경로를 자동으로 찾을 수 있도록 한다.

5. 비행 알고리즘

비행체의 비행은 3가지 상태로 분류된다. 첫 번째는 대기상태로 비행체는 정점(마커) 위에서 벗어나지 않으면서 정적인 움직임을 목표로 하는 상태이다. 두 번째는 출발상태로 목적지가 정해졌을 시 비행체가 인접한 다음 마커를 향해 방향을 맞추는 상태이다. 세 번째는 비행 상태로, 비행체가 마커와 마커 사이를 실제로 비행하는 상태이다. 비행체의 상태는 (그림 4)의 의사코드를 따른다.

```
void function state_change()
{
    state = 대기상태
    while()
    {
        if(state == 대기상태)
            if(목적지 정보 수신)
                state = 출발상태
            5-1. wait_state_control()

        else if (state == 출발상태)
            5-1. start_state_control()
            if(다음 목적지로의 방향을 맞췄다면)
                state = 비행상태
                현재 azimuth값 저장

        else // 비행상태
            5-2. flight_state_control()
            if(다음 목적 마커를 찾았다면)
                if(최종 목적지라면)
                    state = 대기상태
                else
                    state = 출발상태
    }
}
```

(그림 4) 비행 알고리즘의 의사코드

출발상태에서 azimuth(방위)를 저장하는 이유는 비행상태에서는 영상 정보만으로는 현재 비행체의 방향이 최초 출발 시의 방향과 동일하게 유지되고 있는지 알 수 없기 때문이다. azimuth 값은 비행체에 장착되는 스마트폰의 오리엔테이션 센서를 이용하여 구한다.

5-1. 대기 및 출발상태 제어 알고리즘

대기상태와 출발상태는 최대한 정적인 움직임을 목표로 하기 때문에 현재 비행체의 위치뿐만이 아니라 각 방향으로의 속도, 가속도, 회전가속도 모두를 고려할 필요가 있다. 특히 본 연구에서 가정하고 있는 비행체는 헬륨을 이용한 형태이기 때문에, 비행체는 알짜힘이 0인 무중력 상태와 비슷한 상태로써 미세한 모터의 출력에도 매우 민감하게 반응하게 된다. 따라서 대기 및 출발상태 제어 알고리즘에서는 비행체를 마커에 수직인 정 위치로 이동시키는 것을 기본으로 하나 만일 비행체가 특정 방향으로의

속도 또는 회전이 너무 빠르거나 이미 정 위치로 이동 중이라면 추가적인 모터의 움직임을 최대한 제한하여야 한다. 속도 등 제어에 필요한 값들은 스마트폰의 카메라로 인식한 마커의 위치와 크기 변화로 알 수 있다. 이 전체적인 과정을 의사코드로 표현하면 다음 (그림 5)와 같다.

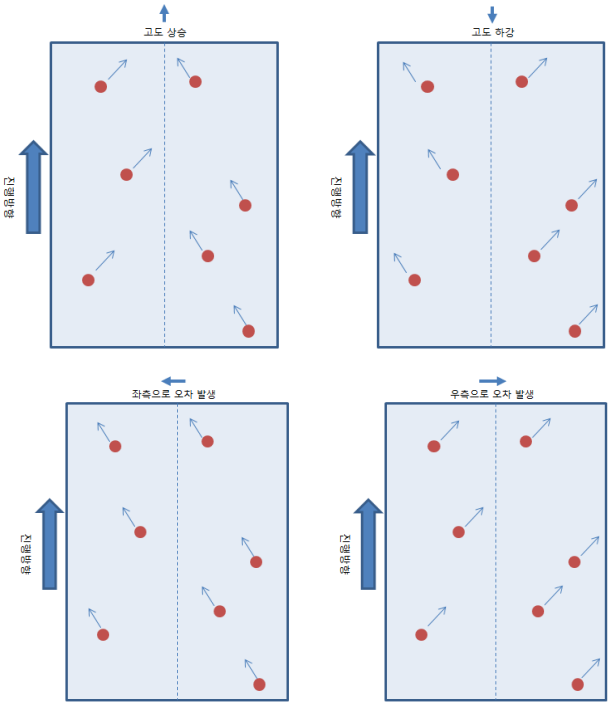
```
void function wait_state_control(다음 목적지 방향)
{
    고도 변화량 검사
    if(고도 변화량이 기준치 이하)
        현재 고도 검사
        if(기준 고도와 다르며 기준 고도로 이동 중이지 않다면)
            기준 고도로 향하도록 고도 모터 명령 값 결정
        else
            고도 모터 정지값
    else
        현재 움직임과 역방향으로 고도 모터 명령 값 결정

    위치 및 회전 변화량 검사
    if(위치 및 회전 변화량이 기준치 이하)
        현재 위치 검사
        if(기준 위치와 다르며 기준 위치로 이동 중이지 않다면)
            return 기준 위치로 향하도록 모터 명령
        else
            return 모터 정지
    else
        return 현재 움직임과 역방향으로 모터 명령

    /* start_state */
    현재 방향 검사
    if(다음 목적지 방향과 같다면)
        return 비행 상태로 전환
    else
        return 다음 목적지 방향으로 회전하도록 모터 명령
}
```

(그림 5) 대기 및 출발 상태 제어 알고리즘의 의사코드

고도의 변화량을 가장 먼저 검사하는 이유는 비행체의 형태상 고도 조절 모터가 따로 존재하므로 고도의 조정은 현재 위치 및 회전의 변화에 상관없이 독립적으로 조절할 수 있기 때문이다. 때문에 고도 모터의 명령값을 미리 결정해 놓고 위치 및 회전 변화에서 판단한 모터의 명령에 고도 모터의 명령값을 결합하는 형태가 되도록 한다.

5-2. 비행 상태 제어 알고리즘

비행 상태에서는 대기상태와 출발상태와 달리 마커를 인식할 수 없는 상태이므로 이 상태에서 비행체가 현재의 위치와 고도를 알아내려면 전혀 다른 알고리즘이 필요하다. 본 연구에서는 컴퓨터 비전 알고리즘 중 하나인 Lucas-Kanade Optical Flow[3]를 응용하여 이러한 문제를 해결한다. Optical Flow는 영상의 특정점을 추적하여 각 특징점이 영상의 프레임마다 어떻게 이동하였는지 알아내는 방법으로 이를 응용하면 비행체의 좌우 변화량과 고도 변화량을 알아낼 수 있다. 따라서 출발상태에서 비행상태로 넘어가기 직전의 비행체의 고도와 위치를 기준으로 잡고, 비행체가 비행 도중 얼마만큼 그 기준에서 벗어났는지를 측정하여 보정하는 알고리즘을 통하여 비행체가 다음 이탈하지 않고 다음 정점까지 도착하도록 한다.

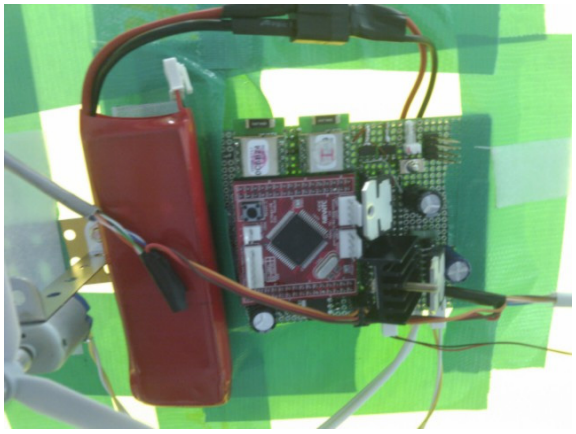


(그림 6) Optical Flow를 이용한 고도 / 좌우 오차 계산

위의 (그림 6)은 Optical Flow를 통하여 고도와 좌우 오차를 계산하는 예시를 나타낸 것이다. 좌우 오차의 경우 카메라로 인식한 모든 특징점의 좌우 변화값의 평균을 내어 측정한다. 고도 오차의 경우 영상을 좌, 우로 나누어서 좌측 특징점들의 변화값과 우측 특징점들의 변화값을 측정하면 알아낼 수 있다. 카메라는 항상 비행체의 정 중앙에 위치하므로 특징점들이 중심에서 좌우로 퍼져 나갔다는 것은 영상의 확대, 즉 고도의 하강을 의미하며 반대의 경우 영상의 축소, 즉 고도의 상승을 의미한다. 이렇게 측정한 값들을 누적시키면 비행체가 처음 출발 상태에서 얼마만큼 상하 좌우로 벗어 낫는지를 측정할 수 있다.

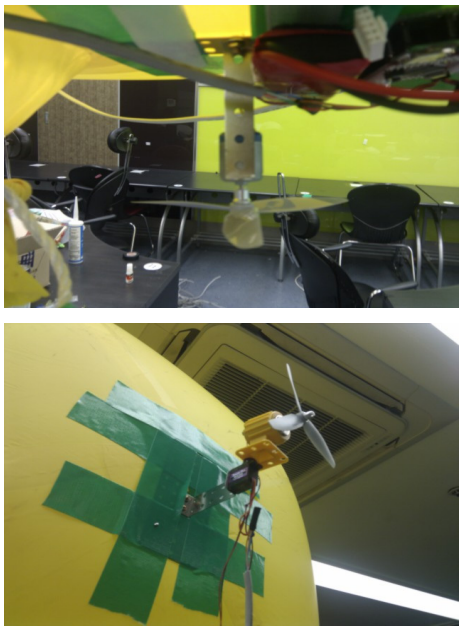
6. 임베디드 보드 및 비행체의 제작

본 연구내용의 실험을 위해 임베디드 보드를 제작하였다. 메인프로세서로 Atmega128를 사용하였고 L298모터 드라이버를 장착하였다. 스마트폰과의 통신을 위해 블루투스 모듈 ESD-200을 사용하였고 Turnigy 1000mAh 배터리를 이용하여 전원을 공급하였다.



(그림 7) 제작한 임베디드 보드

제작한 임베디드 보드의 무게는 325g으로서 스마트폰, 기낭, 기타 기체 부품의 무게를 고려하면 비행체는 최소 800g 이상의 무게를 들어 올릴 수 있는 형태로 제작되어야 한다. 2장의 연구결과를 바탕으로 하여 높이 100cm, 너비 90cm, 길이 180cm에 준하는 실험용 비행체를 제작하였다.



(그림 8) 제작 중인 실험용 비행체

7. 실험 및 분석

제작한 실험용 비행체와 본 연구의 알고리즘을 적용하여 비행 성공 여부를 실험한 결과는 아래의 <표 1>과 같다. 실험은 숭실대학교 정보과학관 2, 3층에 마커를 설치하고 가상 맵을 제작하여 20번에 걸쳐서 이루어졌다.

<표 1> 실험 결과

구 분	실험 내용	성공률
대기상태	출발상태 전까지 마커에서 벗어나지 않는지의 여부	100%
출발상태	다음 마커로의 방향을 정확히 찾아서 출발하는지의 여부	95%
비행상태	다음 마커까지 정상적으로 도착하는지의 여부	80%

실험 결과 대기상태와 출발상태의 경우 높은 성공률을 보였으나 비행 상태에서는 아직 불안정한 모습을 보였다. 빌딩 내부에 흐르는 기류를 비행체가 완벽히 제어하지 못하는 게 가장 큰 원인이었다.

8. 결론 및 향후 연구

실험 결과에 나타나듯이 아직 완벽한 성공률을 보이고 있지는 않으며 기체의 소형화, 헬륨의 유지력 보완 등 개선해야 할 부분이 남아있다. 특히 바람의 저항에 더욱 강한 형태로 비행체의 형태를 개선하고 알고리즘을 보완할 필요가 있으리라 본다. 그러나 기존의 다른 비행체처럼 자이로센서 등 고가의 센서에 의존하지 않고 영상정보와 스마트폰에 내장된 센서만으로 비행이 가능한 비행체 제작 및 알고리즘 개발은 분명 유의미한 일로써 물건 배송, 무인 방법, 엔터테인먼트 등 다양한 방면으로 활용될 수 있을 것이다.

참고문헌

[1] KITECH 양광웅 : OpenCV Marker Recognition 2.

[2] Dijkstra, E. W. (1959). "A note on two problems in connexion with graphs". Numerische Mathematik 1: 269 - 271. doi:10.1007/BF01386390.

[3] B. D. Lucas and T. Kanade (1981), An iterative image registration technique with an application to stereo vision. Proceedings of Imaging Understanding Workshop, pages 121-130