안드로이드 기반 실시간 원격 쿼드로터 퓨전제어기법

양성민*, 오홍식*, 강성민*, 이현*
*선문대학교 컴퓨터공학과
e-mail: ysm901124@gmail.com dhghdtlr@hanmail.net,
seokmin.kang@gmail.com, mahyun91@sunmoon.ac.kr

Android based real-time remote Quadrotor fusion control method

Sung-Min Yang*, Hong-Sik Oh*, Seok-Min Kang*, Hyun Lee*
*Dept of Computer Science and Engineering, Sun Moon University

요 약

최근 몇 년간 무인항공기(UAV) 시장이 점차 커지면서 군뿐만 아니라 민간, 상업적으로도 무인항공기의 수요가 증가하고 있다. 이에 무인항공기의 한 종류인 에어드론(AR.Drone)을 활용한 실시간 원격 쿼드로터(Quadrotor) 퓨전제어기법을 제안하였다. 특히, 본 논문에서는 아이폰(i-phone) 기반의 제어기법이 아닌, 안드로이드(Android) 기반의 퓨전제어기법을 통하여 에어드론을 실시간으로 원격조정가능하게 했는데, 이는 아이폰 App 개발 시 제공되는 API와 PC 기반의 쿼드로터 제어기법을 퓨전하는 방식으로 쿼드로터의 비행제어와 영상처리를 분리시켜 기존의 방식보다 영상처리 속도를 향상시키는 방식이다. 그리고 제안된 퓨전제어기법의 우수성을 보여주기 위해, 기존의 방식들과 영상처리 속도를 비교분석하였다.

1. 서론

최근 몇 년간 무인항공기 시장이 점차 커지면서 군뿐만 아니라 민간, 상업적으로도 무인항공기의 수요가 증가하고 있다[1]. 이는 사람이 직접 접근할 수 없는 재난 발생지역, 환경오염지역, 화생방오염지역 등에 Hexarotor, Quadrotor, Micro-UAV 등과 같은 무인항공기를 투입하여 실시간 원격으로 재난피해 정도를 관찰하고 인명구조 작업, 생존자 위치파악 등에 효과적인 시스템으로 활용할 수 있다. 실험을 위한 테스트 기반 쿼드로터 형태의 무인항공기로는 와이파이 기반 무선통신 및 리눅스 기반의 운영체재가 탑재된 에어드론[2]이 활용되고 있다. 특히, 에어드론은 몸체가 작고 가벼운 장난감 용도로 출시되었으나, 기본적으로 이·착륙, 비상착륙, 전후이동, 상하이동, 좌우이동, 회전 등의 기능을 갖고 있다. 또한 전방과 밑쪽에 카메라가 달려 있으며, 바닥에 달린 초음파 센서로 바닥과의 거리를 알고 수직 이동을 할 때, 속도 및 거리를 조절할 수 있다. 하지만, 현재 이용되는 에어드론의 경우, 문제점으로 아이폰 App을 위한 API만 제공하고 있고, 안드로이드 기반의 App을 적용하는 경우에는 개발어려움이 있다. 기존의 자바로 개발된 제어기법도 있으나 미완성이거나 PC 전용으로만 동작하는 한계점을 가지고 있다[3]. 결국 기존의 제어기법들은 영상의 깨짐이나 지연이 빈번히 발생하였다[4].

본 논문에서는 이런 문제점을 해결하기 위해 안드로이드 기반의 퓨전제어기법을 개발하여 쿼드로터테스트 기반인 에어드론에 적용하여 보고자 한다. 안드로이드 기반의 퓨전제어기법이란 크게 3가지의 특징을 가지고 있는데, 첫째, 자바로 개발된 소스코드를 안드로이드 개발환경에 맞게 변환시키고, 둘째, 쿼드로터의 비행제어와 영상처리를 분리시키며, 마지막으로 아이폰 App 개발 시, 제공되는 JNI를 이용하여 영상처리 속도를 향상시키는 방식이다. 이는 실시간 원격으로 쿼드로터를 제어하는데 있어 큰 장점을 갖는다. 또한 퓨전제어기법과 기존의 제어기법들을 비교분석하여 제안된 퓨전제어기법의 우수성을 보여주고자 한다.

본 논문은 다음과 같이 구성된다. 2장에서는 관련연구를 소개하고, 3장에서는 제안된 퓨전제어기법을 설명하며, 4장에서는 제안된 퓨전제어기법의 비교우위를 보여주기 위하여 기존의 제어기법들과 비교분석한다. 그리고 마지막 5장에서 결론을 맺는다.

2. 관련연구

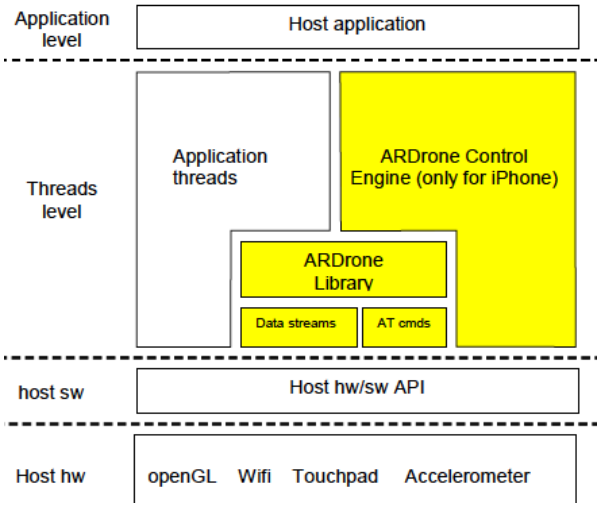
현재 국내외적으로 무인항공기에 관한 많은 이목이 집중되고 있다. 먼저 미국의 경우, 브라운 대학 및 여러 연구소에서 쿼드로터를 위한 로봇운영체제(Robot Operating System) 드라이버를 개발하고 있다.[5],[6] 캐나다의 캘거리 대학의 경우에는 손동작 움직임을 이용한 쿼드로터 조종기술을 개발하고 있다[7]. 하지만 [7]의 경우 현재까지 아직 큰 손동작만을 인식하여 개선의 여지가 필요하다. 또

한 각종 다른 형태의 무인항공기에 비전(Vision)을 활용한 장애물 회피, 자율주행 등 다양한 주제로 연구가 진행되고 있다[9], [10].

특히 무인항공기의 한 종류인 에어드론을 활용한 실시간 원격 쿼드로터 제어기술도 개발되고 있는데, parrot사에서 제공하는 OpenApi[8]의 경우에는 실행 할 때마다 제대로 동작되지 않는 단점이 있다. JavaDrone[3]의 경우에는 100% 자바로 개발된 제어 프로그램으로 PC 기반에서 작동되는 단점을 가지고 있고 영상처리에 있어서 지연 현상이 발생하여 코드최적화 개선이 필요하고, 안드로이드에서 사용할 수 없는 클래스들이 존재하여 실제 변환에 문제점을 가지고 있다. FusionDrone[4]의 경우에는 안드로이드기반으로 동작되지만, 상대적으로 느린 영상처리와 기본적인 이착륙(take-off, landing) 기능만 처리하여 실시간 쿼드로터의 동작제어 처리를 못하는 한계점이 있다. 따라서 본 논문에서는 이러한 관련 연구를 바탕으로 기존의 관련 연구들에서 개선하지 못한 영상처리속도 부분을 본 논문에서 제안하는 안드로이드 기반의 퓨전제어기법을 통해 처리하고자 하였다.

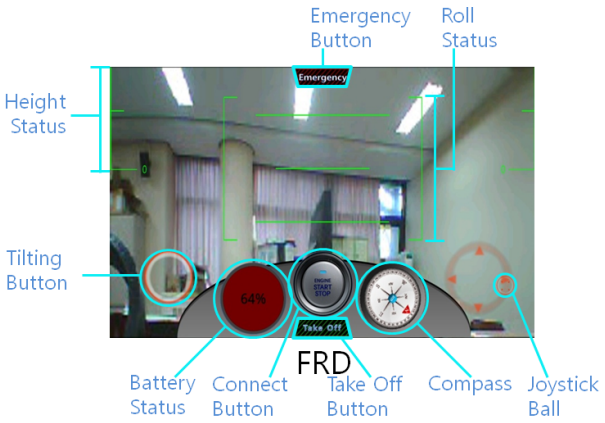
3. 안드로이드 기반의 퓨전제어기법

퓨전제어기법이란 아이폰 기반으로 제공된 JNI와 PC에서 동작하는 자바 기반 JavaDrone을 활용하여 개발한 실시간 원격 쿼드로터 제어기법으로 안드로이드 기반의 디바이스에서 사용하도록 하였는데, 그림1의 Threads 단계의 에어드론 제어 엔진 (only for I-phone), 에어드론 라이브러리, 데이터 스트림, AT cmds 등을 안드로이드 기반으로 수정하였다.



(그림 1) AR.Drone의 시스템 구조 및 수정코드부분

예를 들어, 에어드론을 제어하기 위해 기존제어방법으로는 키보드를 통한 이벤트 처리 방식이었는데, 제안된 퓨전제어방식에서는 스마트폰의 터치 기능과 떨림(Tilting) 기능을 이용하여 에어드론의 비행을 제어하도록 그림2와 같이 개발하였다. 특히 기존의 JavaDrone에서의 자바 소스코드를 안드로이드 기반에 맞게 변환시켜 UI를 구성하였다.



(그림 2) Android 기반에 맞게 변환시킨 UI의 모습

기존의 소스코드는 하나의 클래스 내에서 에어드론 제어부분, 영상출력처리부분, 정보수신부분까지 모든 기능이 activity 하나에서 처리되고 있다. 이는 프로그램의 자원의 비효율적인 관리를 초래하고 영상처리속도의 느려지는 효과로 나타나고 있다. 따라서 본 논문에서는 그림 3과 같이 소스코드를 수정 변환함으로써 영상처리부분과 비행제어부분을 분리함으로써 다른 부분으로 접근을 최소화시켜 프로그램 및 자원의 효과적 관리를 진행하고자 하였다. 하지만 실제 영상처리부분과 비행제어부분을 다른 아이피로 설정하여 각각 독립적으로 에어드론에 연결하여 실행하여 보았으나, 이와 같은 방법은 다른 문제점을 발생시켜, 본 연구는 안드로이드 activity 부분에서 실질적으로 에어드론과의 연결을 만들고 카메라 영상을 받을 포트를 독립적으로 열어주어 통신하도록 처리하였다.

```
public void connect() throws IOException
{
    ....
    video_reader  = new VideoReader(this,drone_addr,VIDEO_PORT);
    Video_reader_thread  = new Thread(video_reader,  "Video Reader");
    Video_reader_thread.start();
}

DEFINE_THREAD_ROUTINE(video_stage,  data)
{
    ....
    COM_CONFIG_SOCKET_VIDEO(&my_icc.socket, VP_COM_CLIENT,
                            VIDEO_PORT, WIFI_ARDRONE_IP);
}
```

(그림 3) 영상처리부분과 비행제어부분을 분리시킨 소스코드

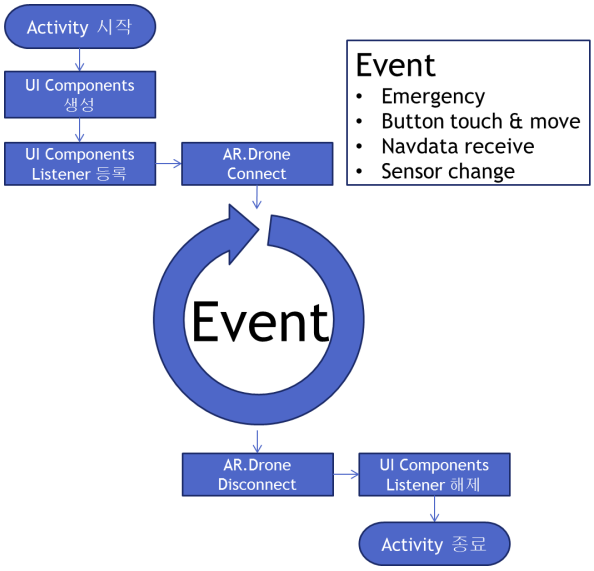
JavaDrone[3]에서는 영상처리 속도를 향상시키기 위하여 코덱을 사용하였으나, C언어와의 기본적인 속도차이를

개선하지 못하였다. 특히 안드로이드 기반의 제어기법을 적용한 FusionDrone[4]은 영상의 지연이 빈번히 발생하여 실시간 제어로 보기 어려울 정도이었다. 이런 문제점을 해결하기 위하여 본 논문에서는 JNI를 사용하여 그림 4와 같이 보다 native한 소스 코딩을 통해 영상처리 속도(예를 들어, 초당 프레임 처리속도)를 향상시켰다. 그리고 이런 퓨전제어기법을 기반으로 그림5와 같이 에어드론의 동작 순서를 진행하도록 하였다. 안드로이드 App이 실행되면 activity가 활성화되고, 각각의 컴포넌트들이 생성 및 초기화, 이벤트 Listener들의 연결 과정을 거친다. 에어드론과의 연결을 통해 송신/수신 체크를 하고 에어드론로부터 영상, 내비게이션 데이터 들을 수신 받아 통신 상태를 점검한다. 위와 같은 동작이 완료되면 각각의 컴포넌트는 이벤트에 의해 에어드론의 비행제어 또는 영상표현 등을 수행하게 된다. 에어드론의 제어가 종료되면, 연결을 끊고 각각의 컴포넌트 자원을 해제 시킨 후 activity를 종료한다.

```
public void videoFrameReceived( . . . . .)
{
  BufferImage im = new BufferedImage(w,h,BfferedImage.TYPE_INT_RGB);
  im.setRGB(startX,startY,w,h,rgbArray,offset,scansize);
  image.set(im);
  repaint();
}

void video_render(long  tick, int width, int height)
{
  . . . .
  glTexSubImage2D(GL_TEXTURE_2D,  0, 0, 0, VIDEO_WIDTH,
  VIDEO_HEIGHT,GL_RGB,  GL_UNSIGNED_SHORT_5_6_5,  config->data );
  . . . .
}
```

(그림 4) JNI를 이용한 Native한 소스코드 부분



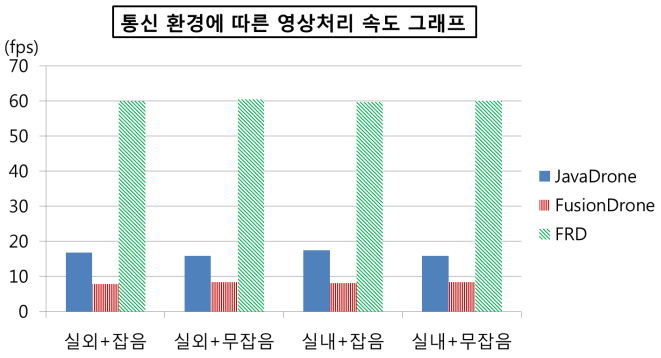
(그림 5) 퓨전제어기법의 동작순서

4. 실험 및 비교분석

4.1. 실험조건

일반적으로 에어드론에 달린 동작센서로 에어드론의 pitch, roll, yaw, 즉, 기울기와 방향을 알 수 있다. 하지만, 에어드론 시스템 H/W 자체에서의 영상처리속도 (frame per second: fps)는 전방 카메라 및 밑쪽 카메라에 모두 고정되어 있도록 설계되어 S/W 상에서 화면을 나타내는 메소드나 함수 부분을 많이 호출한다고 해서 fps가 일정 상수 이상 올라가지 않는 문제점을 가지고 있다. 또한 주변의 무선기기에 의해 손실되는 영상데이터가 있는지 알기 위해서는에어드론의 라이브러리 소스를 수정해야 하는데, 그 부분은 비공개 소스라서 직접적인 에어드론의 라이브러리를 변경하지 못하고, 대안방법으로 자바나 안드로이드 소스에서 화면을 출력하는 부분에 대해 타임스탬프를 잡아서 영상처리속도를 측정해보았다.

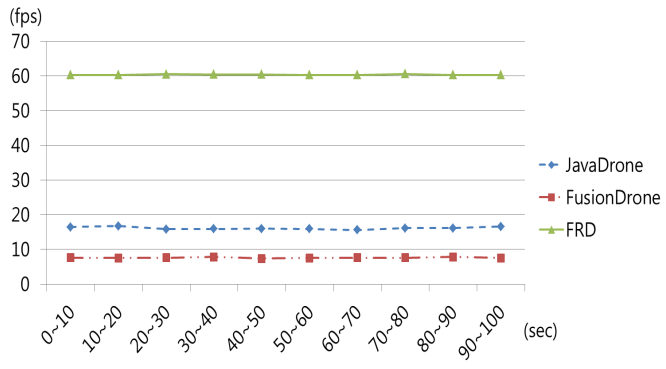
하지만, 영상처리속도가 JavaDrone은 평균 16의 값, FusionDrone은 평균 8의 값, FRD (제안된 퓨전제어기법)는 평균 60의 값으로 모든 상황에서 비슷한 값을 보여주고 있다 (그림 6 참조). 따라서 본 논문에서는 실험상태 조건으로 실내 환경에서 무선통신기기의 영향을 받지 않는 상태라고 가정을 하고, 기존의 제어방법과 퓨전제어기법의 영상처리속도를 비교 분석하였다.



(그림 6) 통신환경에 따른 영상처리속도 비교그래프

4.2. 비교분석

본 논문에서는 실험조건에 따라, 실내에서 무선통신기기의 영향을 받지 않는 상태에서 퓨전제어기법을 통한 영상처리속도와 다른 제어기법들과의 영상처리속도를 비교하였고 그림 7과 같은 결과가 나왔다. 먼저 PC 기반의 자바로 만들어진 JavaDrone은 평균 16 fps로 영상을 보여준다. PC기반 안드로이드 기반으로 바꾼 FusionDrone은 평균 10 fps 미만으로 영상의 속도가 현저하게 떨어진 것을 볼 수 있다. 반면에 안드로이드 기반에서 제안된 퓨전제어기법을 사용한 FRD에서는 영상처리속도(fps)가 기존의 안드로이드 기반 어플리케이션인 FusionDrone보다는 물론 PC기반의 JavaDrone보다도 훨씬 높아진 것을 보여주고 있다. 이를 통해 본 논문에서 제안한 퓨전제어기법이 기존의 제어기법보다 성능이 우수함을 알 수 있다.



(그림 7) 실내+무선통신기기의 영향을 받지 않는 상태에서의 영상처리속도 비교그래프

5. 결론

지금까지 본 논문에서는 안드로이드 기반의 스마트기기를 이용하여 쿼드로터 형태인 에어드론을 실시간 원격제어하는 퓨전제어기법을 제시하였다. 이는 아이폰 App 개발 시 제공되는 API와 안드로이드로 구성된 PC 기반의 쿼드로터 제어기법을 퓨전하는 방식으로 쿼드로터의 비행제어와 영상처리를 분리시켜 기존의 방식보다 영상처리 속도를 향상시키는 방식이다. 실제 쿼드로터 형태의 에어드론을 이용하여 테스트를 해본 결과, 기존의 널리 사용되는 제어기법에 비교하여 제안된 퓨전제어기법이 보다 나은 영상처리 속도를 보여주었다. 이와 같은 결과를 바탕으로 제안된 퓨전제어기법을 이용하여 에어드론 뿐만 아니라 다른 기종의 쿼드로터, 무인항공기를 원격제어 할 수 있을 것으로 예상한다.

계속적인 연구방향으로는 주변 환경변수 변화에 따른 여러 가지 결과를 비교하고자한다. 특히 원격제어 범위를 벗어난 경우를 대비하여, 특별한 조작 없이 기기가 스스로 장애물을 회피하고 임무를 수행할 수 있도록 자율주행기술을 에어드론에 적용하고, 여러 에어드론들이 서로 협력업무를 수행할 수 있도록 자율군집제어 기능을 개발하고자 한다.

참고문헌

[1] Roger S. Pressman, "Software Engineering A Practitioners' Approach", 6th Edition. McGraw Hill, New York, 2005

[2] "Parrot AR.Drone" [Online], the resource available at http://ardrone.parrot.com

[3] "JavaDrone" [Online], the resource available at http://code.google.com/p/javadrone/2011

[4] "FusionDrone" [Online], the resource available at https://projects.ardrone.org

[5] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheller, and A. Ng, "ROS: an open-source Robot Operating System", IEEE Intl. Conf. on Robotics and Automation (ICRA) workshop on open source software, May. 12-17, Kobe, Japan, 2009

[6] "Robot Operating System Documentation" [Online], the resource available at http://www.ros.org/wiki

[7] W. S. Ng and E. Sharlin, "Collocated interaction with flying robots," IEEE Intl. Conf. on RO-MAN, pp. 143-149, Jul. 31- Aug. 3, Atlanta, GA, 2011

[8] S. Piskorski and N. Brulez, "AR.Drone Developer Guide," Revision SDK 1.6, Feb. 24, 2011

[9] N. Dijkshoorn, and A. Visser, "Integrating sensor and motion models to localize an autonomous ar.drone," Intl. Journal of Micro Air Vehicles, vol. 3, no. 4, pp. 183-200, Nov. 2011

[10] C. Bills, "Autonomous MAV flight in indoor environments using single image perspective cues," IEEE Intl. Conf. on Robotics and Automation (ICRA), pp. 5776-5783, May 9-13, Shanghai, China, 2011