JPNT 9(4), 405-409 (2020)
https://doi.org/10.11003/JPNT.2020.9.4.405

JPNT Journal of Positioning, Navigation, and Timing

Time Management System for Applications of UAV Network

Won-Seok Lee[1,2], Jun-Yong Jang[1,2], Hyoung-Kyu Song[1,2†]

[1]Department of Information and Communication Engineering, Sejong University, Seoul 05006, Korea
[2]Department of Information and Communication Engineering, Convergence Engineering for Intelligent Drone

ABSTRACT

This paper proposes time management system for unmanned aerial vehicle (UAV) network. The computers of the UAVs need time synchronization that time offset does not exceed the minimum interval of data samples for errorless data blending between the computers. The proposed time management system is composed of time synchronization and general management systems for UAV control. The systems communicate each other for time information and control signals. The synchronization system uses improved version of existing time offset estimation that network time protocol (NTP) uses. The time synchronization is operated when the time offset of any UAV exceeds threshold that preconfigured by the general management system. The demonstration of prototype shows stable time synchronization satisfying preconfigured threshold.

Keywords: time synchronization, UAV, NTP, oneM2M, IoT

1. INTRODUCTION

드론으로 통칭되는 무인비행체는 미래에 유통, 건설, 재난 현장 복구 및 구조 등 다양한 분야에서 유용하게 사용될 것으로 예측되고 있다 (Culver 1998, Choi et al. 2016, Alwateer et al. 2019). 임무를 위해 운용되는 무인 비행체는 일반적으로 두 대의 컴퓨터를 탑재한다. 두 컴퓨터는 비행제어컴퓨터와 임무컴퓨터라고 불리며 비행 장치와 임무 장비의 제어를 담당한다. 비행제어컴퓨터에는 비행을 위한 항법 제어 알고리즘이 구현된 소프트웨어가 실시간으로 동작하며 연결된 비행 장치들을 제어한다. 대표적인 비행제어컴퓨터의 하드웨어 플랫폼으로 Pixhawk시리즈가 있으며 비행제어 소프트웨어로는 Ardupilot과 PX4가 있다. 임무컴퓨터에는 임무 장비 제어를 위한 소프트웨어가 설치되고 전송 대역폭이 넓은 임무용 통신 장치로 사용자가 임무 장비를 제어하기 위한 기능이 구현된다. 임무 장비는 목적에 따라 다양한 장비가 사용될 수 있기 때문에 임무컴퓨터로 범용 인터페이스가 존재하는 소형 컴퓨터가 사용된다. 대표적인 임무컴퓨터용 보드로는 Raspberry Pi와 NVIDIA의 Jetson TX 보드가 있다. Raspberry Pi는 Jetson TX 보드에 비해 상대적으로 가격이 싸고 범용성이 높기 때문에 교육용으로 많이 사용되는 보드이다. Jetson TX보드는 Raspberry Pi에 비해 가격이 비싸고 요구되는 전력 사용량도 높지만 인공지능과 같이 짧은 시간에 많을 계산을 수행해야하는 임무에서 적합한 성능을 보장한다.

다수의 무인비행체가 사용되는 응용에서는 무인비행체들의 임무컴퓨터가 수집한 데이터를 관제컴퓨터에서 통합하여 분석하고 임무에 대한 의사 결정을 내리는 경우가 있다. 대표적인 응용으로는 다수의 무인비행체에서 촬영한 영상을 응용 서버에서 통합하여 하나의 확장 영상을 출력하는 응용과 평창 올림픽에서 보인 드론쇼와 같이 많은 수의 무인비행체가 정밀하게 제어되는 응용이 있다. 두 응용에서 각 임무컴퓨터는 운영체제 시간을 기준으로 데이터를 기록하고, 응용 서버에서는 데이터를 통합할 때 각 임무컴퓨터가 기록한 시간을 이용한다. 임무컴퓨터의 운영체제는 일반 컴퓨터의 운영체제와 마찬가지로 네트워크에 연결되어 있을 때 NTP를 이용하여 다른 컴퓨터들과 시간을 동기화 한다. 하지만 NTP의 경우 동기화를 통해 보장되는 시간 오차가 최소 30 ms를 넘어간다 (Mills 1991, Mills et al. 2010). 무인비행체의 비행제어컴퓨터는 실시간으로 주변 환경에 반응하기 위해 최소 10 ms 정도의 간격으로 센서의 데이터를 샘플링하여 기록한다 (Ardupilot 2018, DJI 2020). 임무컴퓨터는 비행제어컴퓨터의 데이터와 함께 임무 장비의 데이터를 처리하기 때문에 두 컴퓨터의 데이터 처리 속도 차이를 줄이는 것은 정밀한 제어가 필요

Received Nov 27, 2020 **Revised** Nov 30, 2020 **Accepted** Dec 01, 2020
†**Corresponding Author**
E-mail: songhk@sejong.ac.kr
Tel: +82-2-3408-3890 Fax: +82-2-3409-4264
Won-Seok Lee https://orcid.org/0000-0002-5317-6136
Jun-Yong Jang https://orcid.org/0000-0001-7026-3453
Hyoung-Kyu Song https://orcid.org/0000-0002-3274-4982

한 응용에서 중요한 요소이다. 비행제어컴퓨터의 데이터 처리 속도를 고려하면 NTP보다 더 작은 시간 오차를 보장할 수 있는 동기화 기술이 필요하다. 유선 네트워크의 경우 하드웨어 모듈 간에 정밀한 시간 기록 패킷 교환을 통해 수행되는 precision time protocol (PTP)를 이용하여 1ms 이하의 시간 오차로 동기화를 수행할 수 있다 (Li et al. 2019). 하지만 하드웨어 모듈의 지원을 받기 어려운 무선 네트워크 채널에서 1ms 이하의 동기화 성능을 달성하는 PTP를 구현하는 것은 불가능한 상황이다. 무인비행체 네트워크에서 적절한 수준의 시간 동기화가 수행되지 않으면 확장 영상을 출력하는 응용에서는 통합 영상에 심한 잡음이 발생할 수 있으며 정밀 제어 응용에서는 정확하게 동일한 시간에 동작해야 하는 비행 명령이 서로 다른 시간에 동작하여 출동하는 사고가 발생할 수 있다. 특히 드론쇼와 같은 정밀 제어 응용은 무인비행체 간의 거리가 일반적인 응용보다 짧기 때문에 제어의 정밀도는 다른 응용보다 높은 수준을 요구한다.

임무컴퓨터의 데이터 샘플링 간격이 10 ms이고 기준 시간으로부터 시간 오차의 최대값이 2.5 ms보다 작을 경우 시스템 간에 오차 없는 데이터 통합이 가능하다. 본 논문에서는 오차 없는 데이터 통합이 가능한 수준으로 임무컴퓨터 간 시간을 동기화하기 위한 시스템을 제안한다. NTP와 PTP 프로토콜은 차이점인 하드웨어 모듈의 지원 여부를 제외하고 시간 기록 패킷의 교환을 이용하는 유사한 방식의 시간 오차 추정을 수행한다. 본 논문은 기존의 시간 기록 패킷 교환을 통한 시간 오차 추정 방식을 무선 네트워크에서 요구하는 성능을 달성할 수 있도록 개선한다. 제안하는 시간 동기화 시스템은 범용 제어 기능을 제공하는 통합 운용 플랫폼과 연결되어 동작하도록 설계되었다. 시제품을 위한 통합 운용 플랫폼은 oneM2M 표준을 따르는 소프트웨어인 Mobius를 이용하여 구현했다 (Choi et al. 2017). 구현 시스템은 Mobius를 통해 네트워크의 시간 오차를 확인할 수 있는 조회 기능과 동기화 시스템을 제어하기 위한 기능을 제공한다.

2. 동기화 시스템의 구조

2.1 동기화 시스템 및 인터페이스 구조

제안하는 시스템은 시간 동기화 기능이 동작하는 동기화 시스템과 범용 제어 기능이 동작하는 통합 운용 플랫폼으로 구성된다. 동기화 시스템의 목적은 관제 컴퓨터의 동기화 서버와 무인비행체의 임무컴퓨터 간에 시간 동기화를 수행하는 것이다. Fig. 1은 본 논문에서 제안하는 시스템의 구조를 보인다. Fig. 1에서 좌측에는 관제컴퓨터와 임무용 무인비행체가 위치한다. 관제컴퓨터는 다수의 무인 비행체를 동시에 관제하며 시간 동기화 서버가 운용되는 컴퓨터이다. 시간 동기화 서버는 2개의 인터페이스를 통해 통합 운용 플랫폼 및 시간 동기화 클라이언트와 데이터를 주고받는다. 시간 동기화 서버는 통합 운용 플랫폼과의 인터페이스를 통해 시간 동기화를 위한 설정 값들을 전달받고 시간 동기화 클라이언트와의 인터페이스를 통해 설정 값들을 전달하여 시스템이 요구하는 설정의 시간 동기화가 동작하도록 한다. Fig. 1은 무인비행체의 구성 요소로 비행제어컴퓨터와 임무컴퓨터를

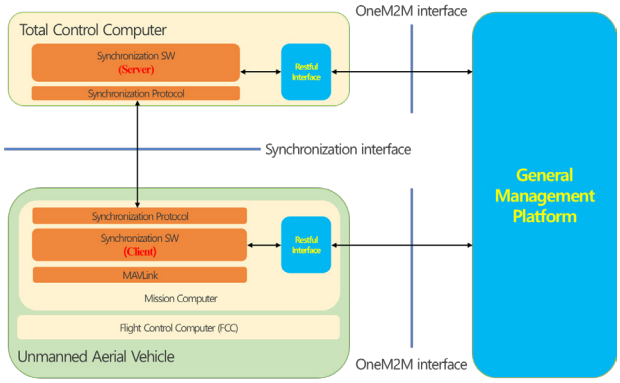


Fig. 1. Interface structure of the proposed system.

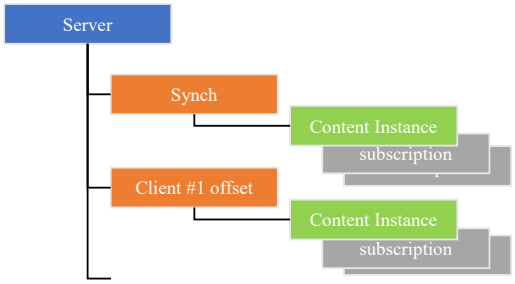


Fig. 2. Structure of stored data in the general management platform.

보인다. 임무컴퓨터에는 시간 동기화 클라이언트가 동작하며 시간 동기화 클라이언트는 3개의 인터페이스를 통해 비행제어컴퓨터, 시간 동기화 서버 그리고 통합 운용 플랫폼과 데이터를 교환한다. 관제컴퓨터와 마찬가지로 시간 동기화 서버와의 인터페이스는 시간 동기화를 위한 설정 값을 전달받고 시간 동기화를 진행하기 위해 사용된다. 통합 운용 시스템과의 인터페이스는 동기화와 관련된 무인비행체의 상태 정보를 전달하기 위해 사용된다. 대표적인 상태 정보로는 시간 동기화 서버와의 시간 오차 정보와 시간 오차가 허용 범위 내에 존재하는지를 알리는 변수가 있다. 비행제어컴퓨터와의 인터페이스를 통해서는 PX4와 Ardupilot 등의 실시간 운영체제와 시간을 동기화하기 위한 메시지를 교환한다. PX4와 Ardupilot은 MAVLink 프로토콜을 이용하여 외부와 데이터를 주고받는다. 시간 동기화 클라이언트는 비행제어컴퓨터와의 인터페이스를 위해 MAVLink 메시지 처리 기능이 구현된다.

2.2 통합 운용 플랫폼

통합 운용 플랫폼은 동기화 시스템과 같은 서로 다른 목적을 위해 운용되는 응용 시스템들을 외부에서 모니터링하고 제어하기 위한 범용 인터페이스를 제공한다. 통합 운용 플랫폼이 존재하기 때문에 각 응용 시스템은 사용자에게 제공하기 위한 모니터링 및 제어 기능을 구현할 필요가 없다. 국제 IoT 플랫폼 표준인 oneM2M은 이러한 목적을 위한 미들웨어 계층의 프로토콜을 제공한다. 본 논문에서 제안하는 시스템은 oneM2M 표준을 기반으로 하는 플랫폼을 이용한다. Fig. 2는 통합 운용 플랫폼에서 관리하는 데이터의 구조를 보인다. 통합 운용 플랫폼은 Fig. 2와 같이 트리 구조로 데이터를 관리하며 데이터를 참조하기 위한 방법으

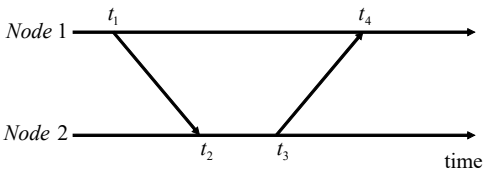


Fig. 3. Transaction of timestamp packets of NTP.

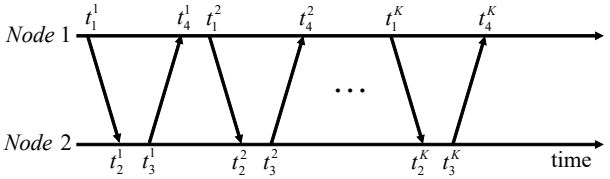


Fig. 4. Transaction of timestamp packets of proposed synchronization method.

로 URL 표현을 이용한다. 또한 내부에 구독 기능을 위한 MQTT Broker를 운용한다. 구독 기능은 특정 위치에 구독을 신청한 모든 컴퓨터에 새로운 데이터가 저장될 경우 해당 데이터에 대한 알림을 전송하는 기능이다. 통합 운용 플랫폼은 메시지 교환을 위해 HTTP와 CoAP과 같은 restful API기반의 프로토콜을 이용한다.

3. 동기화 프로토콜

본 논문에서는 관제컴퓨터와 무인비행체의 임무컴퓨터 간 시간 동기화를 위해 기존에 인터넷에서 동작하는 NTP를 개선하여 시스템에서 요구하는 동기화 성능을 구현했다. 기존의 NTP는 동기화를 위한 두 컴퓨터 사이에서 시간을 기록한 패킷을 교환하여 시간 오차를 계산하고 한 곳에서 계산한 오차를 보정하는 방식으로 동기화를 수행한다. Fig. 3은 NTP에서 시간 동기화를 위해 두 컴퓨터 사이에서 패킷을 교환하는 상황과 패킷에서 기록하는 시간을 변수로 나타낸다. 식 (1)은 NTP에서 교환한 시간 정보를 통해 시간 오차를 계산하는 방법을 보인다.

$$\hat{e} = \left\{\left(t_2 - t_1\right) - \left(t_4 - t_3\right)\right\} / 2 \tag{1}$$

$$t_2 = t_1 + e + d + n_1 \tag{2}$$

$$t_4 = t_3 - e + d + n_2 \tag{3}$$

$$\hat{e} = e + \left(n_1 + n_2\right) / 2 \tag{4}$$

식 (2)와 (3)에서 e, d, n_1, n_2는 각각 실제 시간 오차, 패킷이 전달되는데 걸리는 지연 시간, 첫 번째와 두 번째 전송에서 네트워크와 컴퓨터의 실행 환경으로 인해 발생하는 시간 잡음을 나타낸다. 시간 잡음은 Gaussian 분포를 따르는 랜덤 변수로 모델링할 수 있다 (Li et al. 2019). 첫 번째 노드에서 식 (1)을 통해 시간 오차를 측정했지만 식 (4)의 결과처럼 전송 시 발생하는 시간 잡음으로 인해 실제 오차의 값과 다른 값이 계산된다. NTP에서는 시간 잡음으로 인해 실제로는 30 ms 이상의 측정 오차가 발생하며 네트워크의 규모나 상황에 따라 해당 오차는 더 커질 수 있다. 본 논문에서는 NTP의 성능을 개선하기 위해 두 컴퓨터 간에 시간 기록 패킷을 여러 번 교환하고 시간 오차를 계산하여 시간 잡음의 영향을 줄이는 방법을 이용했다. Fig. 4는 제안 알고리즘에서 시간 기록 패킷을 교환을 교환하는 상황을 나타낸다. Fig. 4에서는 총 K번의 패킷 교환을 가정한다. 식 (5)는 첫 번째 노드에서 교환한 패킷의 정보들로 시간 오차를 계산하는 방법을 나타낸다.

$$\hat{e} = \frac{1}{2K}\sum_{k=1}^{K}\left\{\left(t_2^k - t_1^k\right) - \left(t_4^k - t_3^k\right)\right\} \tag{5}$$

$$\hat{e} = e + \frac{1}{2K}\sum_{k=1}^{K}\left(n_1^k + n_2^k\right) \tag{6}$$

시간 잡음의 통계 분포가 Gaussian 분포를 따르면 식 (6)의 두 번째 항은 K가 증가할수록 0에 수렴하게 된다. 이론적인 환경에서는 인접한 패킷 교환 시도에서 발생하는 시간 잡음이 독립성이 보장되지만 실제 환경에서는 두 패킷 교환 시도 간의 시간이 짧을수록 잡음의 상관도가 증가하는 경우를 관측할 수 있었고, 실제로 이러한 상황이 발생했을 때 패킷 교환 시도 간에 시간 간격을 늘리는 것으로 시간 잡음의 상관도를 낮춰 알고리즘의 성능을 개선하는 것이 가능했다.

여러차례 패킷을 교환하는 방법으로 NTP보다 동기화 성능을 개선할 수 있지만 네트워크에 다수의 노드가 연결되고 동시에 시간 동기화를 수행하게 되면 네트워크에 과한 트래픽을 발생시킬 수 있다. 무인비행체 네트워크의 경우 네트워크가 무선으로 구현되기 때문에 유선 네트워크에 비해 네트워크가 불안정하다. 그러므로 주기적인 동기화는 네트워크의 불안정성을 가중시킬 수 있다. 이를 위해 제안 시스템은 설정 값을 통해 시간 오차의 허용 한계를 결정하고 시간 동기화 클라이언트에서 오차 한계를 넘을 시 해당 클라이언트만 동기화 프로세스를 진행하도록 했다. 또한 네트워크의 안정성이 충분히 보장되지 않아 다수의 노드가 동시에 동기화 프로세스를 진행하는 상황이 늘어나면 서버에서 시간 동기화에 수행되는 패킷 교환의 수를 조정하는 방법을 통해 제안 기법의 문제점을 보완하는 것이 가능하다.

4. 시제품

시간 동기화 시스템을 위한 시제품은 시간 동기화 서버, 클라이언트 소프트웨어와 통합 운용 플랫폼인 Mobius를 연동하기 위한 인터페이스 소프트웨어로 구성된다. 시간 동기화를 위한 서버와 클라이언트는 컴퓨터 간 시간을 측정하고 보정하는 기능을 고려하여 동작 시간이 가장 빠른 c언어를 이용하여 구현했다. 인터페이스 소프트웨어의 경우 임무컴퓨터에 설치되는 다른 소프트웨어들이 Mobius와 연결되는 상황을 고려하여 구현의 유연성과 개발 속도가 빠른 Python을 개발 언어로 선택했다. Fig. 5는 개발한 시제품에서 클라이언트의 시간 정보가 Mobius로 전송되어 저장되는 경로를 보여준다. 구현된 시간 동기화 클라이언트는 실행파일 형태로 개발했으며 호출 시 전달하는 문자열을 통해 시간 오차 측정과 동기화 프로세스 등 구현된 기능들을 실행

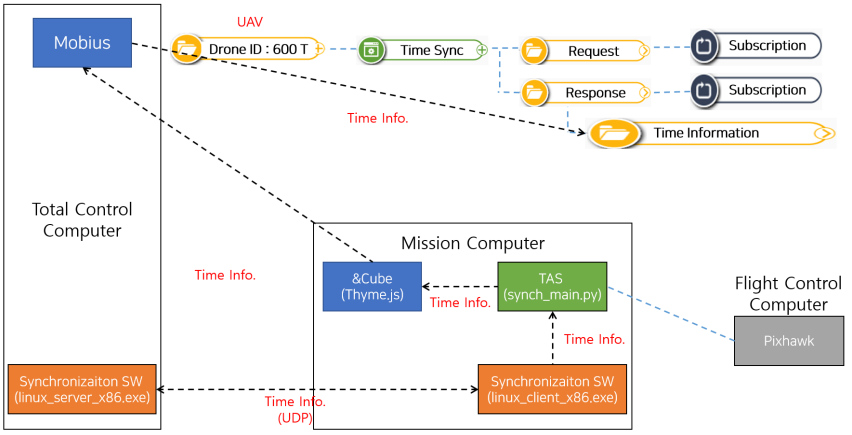


Fig. 5. Upload path of time offset data from the synchronization client to general management platform.

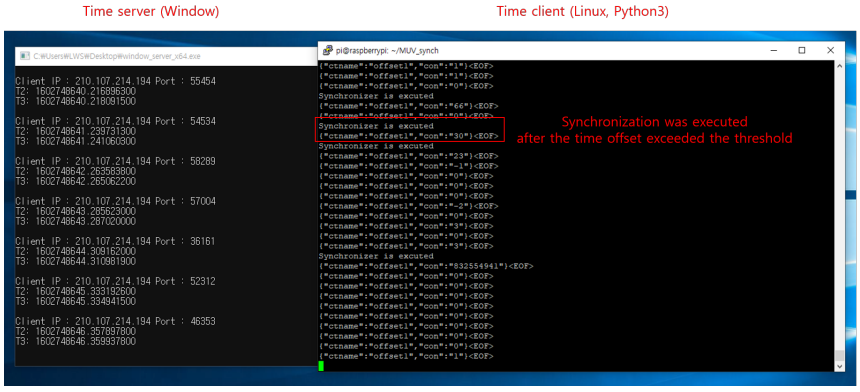


Fig. 6. Time offset measurement of prototype.

하도록 하였다. 클라이언트의 호출은 Python으로 구현된 인터페이스 소프트웨어에서 발생한다. 인터페이스 소프트웨어는 필요한 기능에 따라 시간 동기화 클라이언트를 호출하고 전달받은 정보를 oneM2M 프로토콜 메시지로 인코딩하여 Mobius로 전달한다. Fig. 5에서 &Cube는 Mobius와의 인터페이스를 구현한 클라이언트 소프트웨어이며 인터페이스 소프트웨어는 &Cube와 통신하여 Mobius와 데이터 교환이 가능하다. 또한 &Cube를 통해 Mobius의 특정 데이터에 MQTT 프로토콜의 구독 기능으로 알림 메시지를 받는 방식으로 동기화 클라이언트의 필요 기능을 호출한다.

Fig. 6은 시제품으로 구현한 서버와 클라이언트의 실행 화면을 보인다. 출력 화면은 허용 시간 오차를 5ms로 설정하여 동기화 기능을 실행한 결과이다. 왼쪽의 화면은 Window 운영체제에서 시간 동기화 서버를 동작시킨 결과이다. 서버의 화면에는 클라이언트에서 전송한 시간 기록 패킷이 도착한 시간과 서버에서 다시 시간 기록 패킷을 전송한 시간이 나타난다. 오른쪽 화면은 Raspberry Pi에서 Python을 이용하여 인터페이스 소프트웨어를 이용하여 시간 동기화 클라이언트를 호출한 결과이다. 화면에는 시간 오차 정보가 oneM2M 메시지로 인코딩 후에 Mobius로 전송되는 상황이 출력된다. 인코딩 메시지에서 con 다음에 나오는 숫자가 millisecond 단위의 시간 오차 추정 값을 의미한다. 출력 결과를 통해 시간 오차가 5 ms를 초과한 것이 확인되면 동기화 수행 메시지가 나타나는 것을 확인할 수 있다. Fig. 6의 결과를 통해 제안 기법의 추정 방식으로 계산된 시간 오차를 클라이언트에서 보정하여 시제품이 5 ms의 허용 시간 오차 이내로 시간 동기화를 수행하는 것을 확인할 수 있다.

5. 결론

본 논문은 다수의 무인비행체를 관제하는 시스템에서 관제컴퓨터를 기준으로 무인비행체 컴퓨터의 시간을 동기화하기 위한 시스템을 제안한다. 시간 동기화 시스템은 통합 운용 플랫폼과 함께 동작하며 시간 및 제어 정보를 주고받는다. 통합 운용 플랫폼은 국제 IoT 플랫폼 표준인 oneM2M 기반의 Mobius로 구현했으며 restful API를 통해 시간 동기화 시스템과 데이터를 교환한다. 시간 동기화를 위해 NTP의 시간 오차 측정 방법을 개선한 동기화 알고리즘을 제안했다. 제안 알고리즘은 시간 기록 패킷의 교환을 여러 차례 수행하여 네트워크에서 발생하는 시간 잡음이 시간 오차 측정에 미치는 영향을 개선한다. 개발한 시제품의 출력을 통해 제안 알고리즘이 설정한 허용 시간 오차를 안정적으로 유지하는 것을 확인할 수 있다.

https://doi.org/10.11003/JPNT.2020.9.4.405

ACKNOWLEDGMENTS

본 연구는 국토교통부/국토교통과학기술진흥원, 과학기술정보통신부, 산업통산자원부의 지원으로 수행되었음 (과제번호 1615011058).

AUTHOR CONTRIBUTIONS

Methodology, Jun-Yong Jang; software, Jun-Yong Jang, Won-Seok Lee; validation, Jun-Yong Jang, Won-Seok Lee and Hyoung-Kyu Song; writing—original draft preparation, Won-Seok Lee; writing—review and editing, Won-Seok Lee, Hyoung-Kyu Song; supervision, Hyoung-Kyu Song; project administration, Hyoung-Kyu Song.

CONFLICTS OF INTEREST

The authors declare no conflict of interest.

REFERENCES

Alwateer, M., Loke, S. W., & Fernando, N. 2019, Enabling Drone Services: Drone Crowdsourcing and Drone Scripting, IEEE Access, 7, 110035-110049. https://doi.org/10.1109/ACCESS.2019.2933234

Ardupilot, ROS time synchronization [Internet], cited 2018 Nov 20, available from: https://ardupilot.org/dev/docs/ros-timesync.html

Choi, C. H., Jang, H. J., Lim, S. G., Lim, H. C., Cho, S. H., et al. 2016, Automatic wireless drone charging station creating essential environment for continuous drone operation, in 2016 International Conference on Control, Automation and Information Sciences (ICCAIS), Ansan, South Korea, 27-29 Oct 2016. https://doi.org/10.1109/ICCAIS.2016.7822448

Choi, S., Sung, N., Park, J., Ahn, I., & Kim, J. 2017, Enabling drone as a service: OneM2M-based UAV/ drone management system, in 2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN), Milan, Italy, 4-7 Jul 2017. https://doi.org/10.1109/ICUFN.2017.7993739

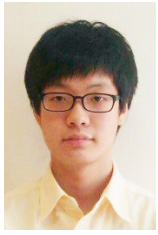
Culver, K. B. 1998, From battlefield to newsroom: Ethical implications of drone technology in journalism, Journal of mass media ethics, 29, 52-64. https://doi.org/10.1080/08900523.2013.829679

DJI, Guide of time synchronization [Internet], cited 2020 Jan 20, available from: https://developer.dji.com/onboard-sdk/documentation/guides/component-guide-time-sync.html

Mills, D., Martin, J., Burbank, J., & Kasch, W. 2010, Network time protocol version 4: Protocol and algorithms specification. https://www.hjp.at/doc/rfc/rfc5905.html

Mills, D. L. 1991, Internet time synchronization: the network time protocol, IEEE Transactions on communications, 39, 1482-1493. https://doi.org/10.1109/26.103043

Li, P., Gong, H., Peng, J., & Zhu, X. 2019, Time Synchronization of White Rabbit Network Based on Kalman Filter, in 2019 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE), Xiamen, China, 18-20 Oct 2019. https://doi.org/10.1109/EITCE47263.2019.9094978



Won-Seok Lee received the B.S. and M.S. degrees in Information and Communication Engineering, Sejong University, Seoul, Korea, in 2016 and 2018, respectively. He is currently working toward to the Ph.D. degree in the Department of information and communications engineering, Sejong University, Seoul, Korea. His research interests are in the areas of signal processing for wireless communication system and ambient RF communication system.

Jun-Yong Jang received the B.S. degree in the Department of Information & Communication Engineering, Sejong University, Seoul, South Korea, in 2020. He is currently pursuing the M.S. degree in the Department of Information & Communications Engineering, Sejong University, Seoul, South Korea. His research interests are in the area of wireless communication systems design and MIMO signal processing.

Hyoung-Kyu Song was born in ChungCheong-Bukdo, Korea on May 14 in 1967. He received B.S., M.S., and Ph.D. degrees in electronic engineering from Yonsei University, Seoul, Korea, in 1990, 1992, and 1996, respectively. From 1996 to 2000 he had been managerial engineer in Korea Electronics Technology Institute (KETI), Korea. Since 2000 he has been a professor of the Department of information and communications engineering, Sejong University, Seoul, Korea. His research interests include digital and data communications, information theory and their applications with an emphasis on mobile communications.