Sung *et al. Hum. Cent. Comput. Inf. Sci.* (2015) 5:35
DOI 10.1186/s13673-015-0051-0

Human-centric Computing and Information Sciences
a SpringerOpen Journal

RESEARCH Open Access

Graph-based motor primitive generation framework

UAV motor primitives by demonstration-based learning

Yunsick Sung[1], Jeonghoon Kwak[2] and Jong Hyuk Park[3*]

*Correspondence:
jhpark1@seoultech.ac.kr
[3] Department of Computer Science and Engineering, Seoul National University of Science and Technology, Seoul 139-743, South Korea
Full list of author information is available at the end of the article

Abstract

Unmanned aerial vehicles (UAVs) have many potential applications, such as delivery, leisure, and surveillance. To enable these applications, making the UAVs fly autonomously is the key issue, and requires defining UAV motor primitives. Diverse attempts have been made to automatically generate motor primitives for UAVs and robots. However, given that UAVs usually do not fly as expected because of external environmental factors, a novel approach for UAVs needs to be designed. This paper proposes a demonstration-based method that generates a graph-based motor primitive. In the experiment, an AR.Drone 2.0 was utilized. By controlling the AR.Drone 2.0, four motor primitives are generated and combined as a graph-based motor primitive. The generated motor primitives can be performed by a planner or a learner, such as a hierarchical task network or Q-learning. By defining the executable conditions of the motor primitives based on measured properties, the movements of the graph-based motor primitive can be chosen depending on changes in the indoor environment.

Background

Nowadays, unmanned aerial vehicles (UAVs) have diverse types of goals, as their performance and techniques have improved. Because making UAVs fly autonomously requires motor primitives, defining and generating motor primitives is the key approach to autonomous UAVs. Especially, the framework design of UAVs are further researched [1, 2].

Usually, motor primitives are defined and generated linearly [3, 4]. Therefore, while executing motor primitives, changes in the environment are not considered, which can prevent a UAV from achieving its goal. For example, motor primitives are defined by dividing the movements learned through demonstration-based learning, and are then executed by a planner [4].

One motor-primitive generation approach is based on a graph [5–8]. Graph-based motor primitives have flexibility in dynamic environments because they consider the state of the environment while executing the motor primitives. A method that generates UAV motor primitives using demonstration-based learning [9] is suggested. The process of generating motor primitives is divided into three stages: Operation collection stage, time adjustment stage, and motor-primitive generation stage. The motor primitives,

based on measured yaw, pitches and rolls, are divided into sub-motor primitives considering pinpoints, where pinpoints are predefined spots where the UAV should arrive.

When UAVs fly, they are affected by diverse types of external factors, such as wind, building structures, etc. Therefore, motor primitives are not usually performed as well as expected, which makes it difficult to execute multiple motor primitives consecutively. For example, two motor primitives are planned to be executed, one right after the other. If the first executed motor primitive is not performed as defined, the next motor primitive cannot be executed, because the preconditions have changed. Therefore, devising a novel approach to improve graph-based motor primitives is required.

This paper proposes a method that generates graph-based UAV motor primitives using demonstration-based learning. By connecting multiple movements and executing one of the connected movements, UAVs can cope with changing environments. The proposed method can be applied not only to UAVs, but also to robots and virtual characters.

This paper proceeds as follows. "Related work" introduces the approaches used to make UAVs fly autonomously. "Graph-based motor primitive generation process" proposes a graph-based motor-primitive generation framework. "Experiments" shows the performance of the generated graph-based motor primitives. "Conclusion" concludes the paper.

Related work

In the beginning, UAVs were designed for military, but recently low-cost UAVs have been developed and sold to consumers. For example, even though the AR.Drone 2.0 contains two cameras and an infrared sensor, it is sold at low cost [10]. In addition, communication protocols are also defined for developers to control the AR.Drone 2.0 by programming. The AR.Drone 2.0 provides algorithms for inertial sensor correction and altitude estimation to maintain an accurate and robust state. An algorithm to display captured images is also contained. Finally, the AR.Drone 2.0 is connected and controlled through Wi-Fi. Diverse types of control modes based on finite-state machines are utilized.

A variety of studies has been done on controlling a UAV automatically. For example, markers attached to walls are recognized by cameras on the UAV, and then the UAV is controlled autonomously by utilizing the recognized markers [11]. For autonomous flight, direction control and altitude control are required. By recognizing the markers by UAV cameras, the distances between markers and UAVs are measured. When the UAV comes within range of one of the markers, it responds by performing the signal corresponding to the marker.

Other research solves UAV path-finding problems: one conducts useless lines when vanishing points are revealed, and the other recognizes uncrossed lines as crossed lines [12]. A vanishing point is defined as a point where the path-propagation direction is and where the outer lines are crossed. An algorithm to improve the difference when vanishing points are calculated was proposed. UAVs fly autonomously based on the algorithm.

In detail, all captured images are converted to black/white images to detect the border values of the captured images. By utilizing Canny Edge Detection, the border values are obtained and, by utilizing a Probabilistic Hough Transform, straight lines are detected. Vanishing points are found by excluding the errors of the uncrossed straight lines when

calculating vanishing points. The improved algorithm was applied to the AR.Drone in an indoor environment and allowed the AR.Drone to fly autonomously.

In addition, research on controlling multiple UAVs for indoor-based group flight has been conducted [13]. Controlling multiple UAVs concurrently not only requires a ground station that controls the UAVs, but also a motion-capture data server that obtains the indoor locations of the UAVs. The ground station controls the UAVs by utilizing a qualitative control algorithm. Motion-capture devices take pictures of the markers attached to the UAVs and then the motion-capture data server recognizes their locations based on the taken pictures.

The shortest path is set to make UAVs fly autonomously in an indoor environment [14]. Given that GPS (global positioning system) signals cannot be received in an indoor environment, an algorithm to generate 3D maps utilizing a depth camera is proposed. The UAV's current location is recognized based on the 2D images captured by UAV cameras and the generated 3D map of the UAV. An algorithm to reduce the complexity of 3D maps is proposed to calculate the UAV path for avoiding obstacles.

UAV locations are estimated based on crowd-sourced signals [15]. A local Wi-Fi fingerprint is collected and automatically updated through multiple users' smart phones. The UAV locations are calculated utilizing the strength of the measured UAV Wi-Fi and the updated Wi-Fi fingerprint. In addition, the UAV's path is estimated by utilizing the sensor values of accelerators and directions.

Traditional control research of autonomous UAVs focuses on recognizing the current location as described above. However, a novel approach is required, not to control UAVs according to a predefined path, but to make UAVs fly autonomously, depending on changes in the dynamic environment. This paper presents a method that applies and improves a motor-primitive generation technique for humanoid robots.

Graph-based motor primitive generation process

UAV motor-primitive expression

In this paper, motor primitives are defined as follows. In Eq. (1), the ith observed and measured motor primitive m_i is defined in terms of its consecutive movements with execution time t_i'. We assume that all motor primitives have the same number of movements. Therefore, p is the total number of measured motor primitives, and q is the number of movements in a motor primitive.

$$m_i = \{m_{i,1}, m_{i,2}, \ldots, m_{i,j}, \ldots, m_{i,q}, \ t_i'\}, \tag{1}$$

where the jth movement of the ith motor primitive is $m_{i,j}$. The movement $m_{i,j}$ is defined as shown in Eq. (2). Each movement is expressed by x properties and y commands. $p_{i,j,x}$ and $c_{i,j,y}$ is the xth property and yth command. A property can be not only UAV values, but also any external values. The motor primitive $m_{i,j}$ is executed by considering the properties and then sending commands at execution time t'_i.

$$m_{i,j} = \left[p_{i,j,1}, p_{i,j,2}, \ldots, p_{i,j,x}, c_{i,j,1}, c_{i,j,2}, \ldots, c_{i,j,y}\right] \tag{2}$$

The movement properties are utilized as the conditions required to send movement commands. When the difference between the movement properties and the UAV

properties is smaller than α, the command is sent by the UAV. A graph-based motor primitive G is defined as shown in Eq. (3).

$$G = \{J, M\}, \tag{3}$$

where J is the joint-ordered set and M is the movement-ordered set. j_i is the joint of the ith movements and an element of joint-ordered set J. The movement-ordered set M contains the sets of movements. Each set of movements contains movements that occur at the same time.

UAV motor-primitive generation

The proposed method generates motor primitives in three stages: Operation-collection stage, time-adjustment stage, and motor-primitive generation stage. In the operation-collection stage, an operator uses a controller to make the UAV fly along a predefined path. Given that motor primitives are generated through demonstration-based learning, the UAV should fly the same path repeatedly. Therefore, the path should be defined in advance.

The operator usually guides the UAV on the same path three to five times. Flying fewer than three times cannot make motor primitives with the flexibility needed for a dynamic environment; more than five times results in complicated motor primitives. While the UAV is controlled, it measures and records its properties (such as pitch, roll, and yaw) and its command signals. Figure 1 shows the measured motor primitives. At this time, the properties of the UAV are measured and the commands of the UAV are obtained. The time $t_{i,j}$ is the measured time of $m_{i,j}$.

Then, the movements and motor primitives of the UAV are defined, based on the properties and the commands. When a UAV faces an intermediate pinpoint to adjust its location, a new movement is defined and added to the motor primitive. The order of the visited pinpoints of motor primitives should be the same during the demonstration-based learning. A single motor primitive is generated based on a single flight.

During the time-adjustment stage, the movement times of all the motor primitives are adjusted. In the proposed method, the executed movements need not depend on a certain motor primitive. For example, movement $m_{2,2}$ in motor primitive m_2 can be

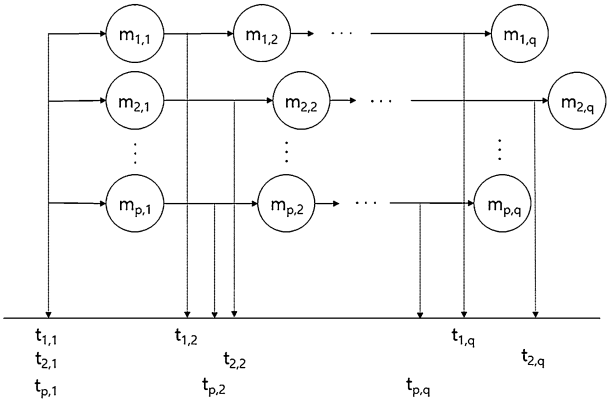

Fig. 1 Measured linear motor primitives

executed after executing $m_{1,1}$ in m_1, if movement $m_{2,2}$ is more appropriate than movement $m_{1,1}$, considering the current properties of the UAV. To execute the movements as above, the ordered movements should have the same time, as shown in Fig. 2.

t_1' is assigned by t_1, which is 0. From t_2', the time of movements is calculated by Eq. (4).

$$\mathrm{t}'_{\mathrm{j}} = \mathrm{t}'_{\mathrm{j}-1} + \mathrm{MAX}\left(\mathrm{d}_{1,\mathrm{j}}, \mathrm{d}_{2,\mathrm{j}}, \ldots, \mathrm{d}_{\mathrm{p,j}}\right), \tag{4}$$

where $d_{i,j}$ is the duration of $m_{i,j}$.

The motor-primitive generation stage merges all motor primitives into a single graph-based motor primitive by connecting the movements in order, as shown in Fig. 3. The graph-based motor primitive of Fig. 3 is defined as $\{\{j_1, j_2, \ldots, j_j\}, \{\{m_{1,1}, m_{2,1}, \ldots, m_{p,i}\}, \{m_{1,2}, m_{2,2}, \ldots, m_{p,2}\}, \ldots, \{m_{1,j}, m_{2,j}, \ldots, m_{p,j}\}\}\}$.

UAV motor-primitive execution

If a graph-based motor primitive G is executed, all joins in the joint-ordered set J are selected in order. If the ith join j_i is selected, then one of the corresponding linked movements is selected, by considering the movement properties, and executed. Given that the

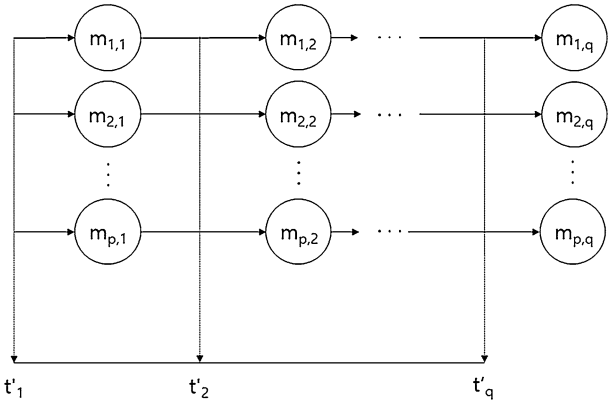

Fig. 2 Adjusted linear motor primitive structure

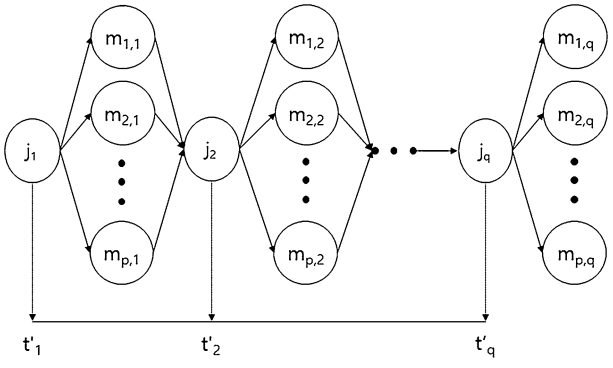

Fig. 3 Generated graph-based motor-primitive structure

UAV selects its movements based on the changed properties, it can consider changes in the environment.

Experiments

In the experiments, AR.Drone 2.0 was utilized as a UAV, as shown in Fig. 4. In the indoor environment, the AR.Drone 2.0 flew four times according to the predefined path—the black rectangle in Fig. 4—and collected the control signals and properties. The AR.Drone adjusts its position when it faces the vehicle number plates, which are defined as pinpoints.

The motor primitives of AR.Drone 2.0 were measured based on the control signals of the subject. A single movement command consists of a yaw, pitch, or roll, based on the motor power. The properties of a movement are yaw, pitch, roll, and altitude. Therefore, commands are sent by considering the yaw, pitch, roll, and altitude of the AR.Drone 2.0. When the AR.Drone 2.0 reaches each pinpoint, one movement is defined and added to the motor primitive. Figure 5 shows the yaw, pitch, and roll of the first measured motor primitive.

During the time-adjustment stage, the movement times are adjusted. Table 1 shows all the times of the measured movements.

Therefore, the times from t_1' to t_5' were set to 0, 22,048, 43,333, 70,726, and 89,064 during the time-adjustment stage. After adjusting the time of the first measured motor primitive, Fig. 6 shows the first adjusted motor primitive.

Figure 7 shows the result of the graph-based motor primitive created by the proposed method, based on four motor primitives where each motor primitive has five movements.

Conclusion

This paper proposed a novel motor-primitive generation framework for UAVs. The measured movements were integrated into a single graph-based motor primitive, which makes it possible to execute the movements of the graph-based motor primitive consecutively, considering changes in the environment.

Fig. 4 Screen shot of the indoor environment

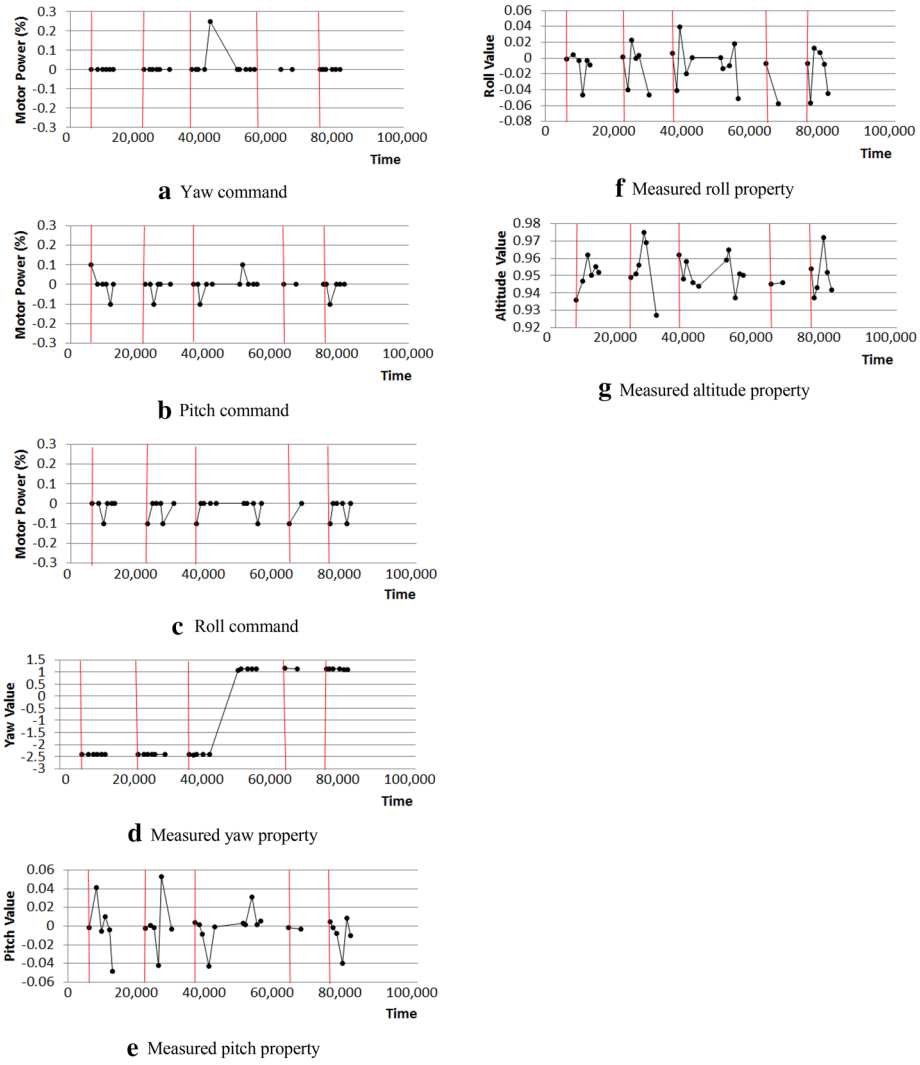

Fig. 5 First measured motor primitive. (**a**) Yaw command, (**b**) Pitch command, (**c**) Roll command, (**d**) Measured yaw property, (**e**) Measured pitch property, (**f**) Measured roll property, (**g**) Measured altitude property

Table 1 All measured movement times

	m_1	m_2	m_3	m_4
t_1	6079	6417	4577	5039
t_2	22,048	17,093	21,117	20,050
t_3	36,325	34,425	40,126	41,335
t_4	63,118	59,023	67,519	68,519
t_5	74,894	76,444	79,285	86,857

Motor primitives are very important, given that the UAVs fly by selecting and performing motor primitives repeatedly. Motor primitives decide the quality of the UAVs. Therefore, the way to define motor primitives is very important. Traditionally, motor primitives are defined by setting the values of motor primitives linearly, which means

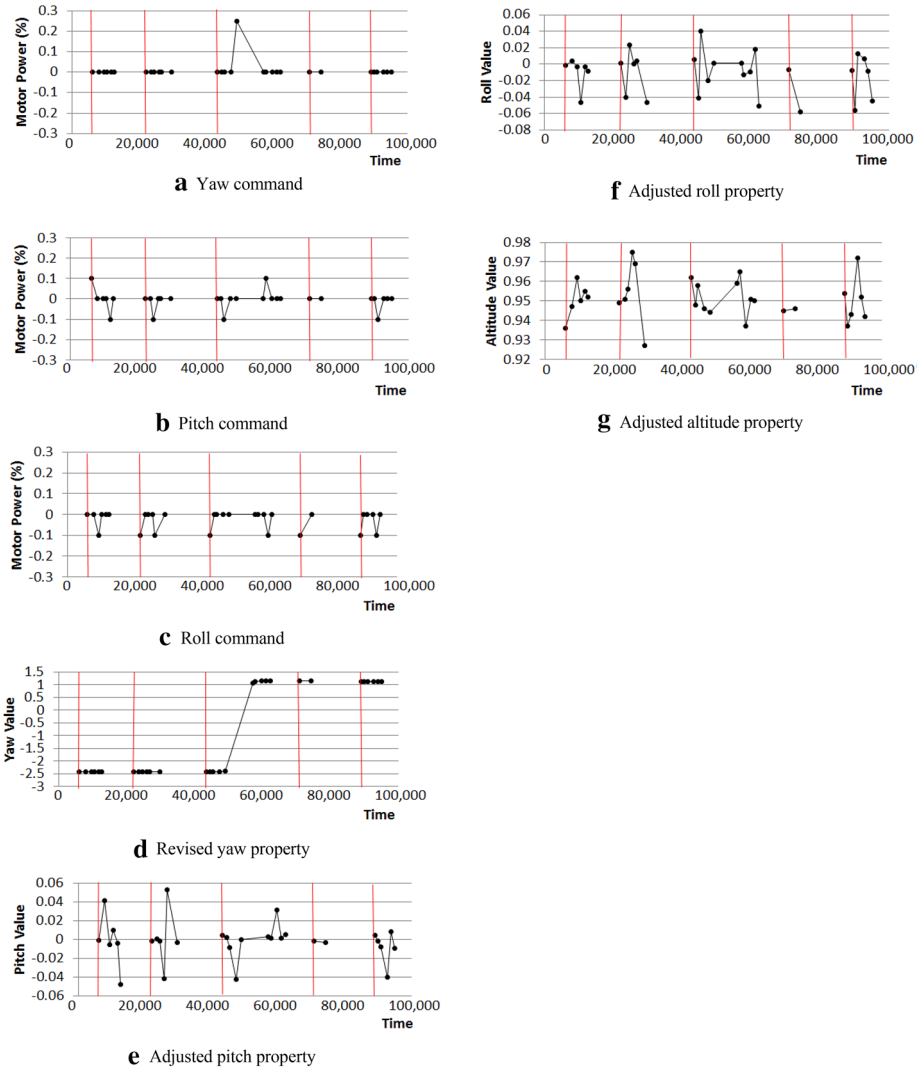

Fig. 6 First adjusted motor primitive. (**a**) Yaw command, (**b**) Pitch command, (**c**) Roll command, (**d**) Revised yaw property, (**e**) Adjusted pitch property, (**f**) Adjusted roll property, (**g**) Adjusted altitude property

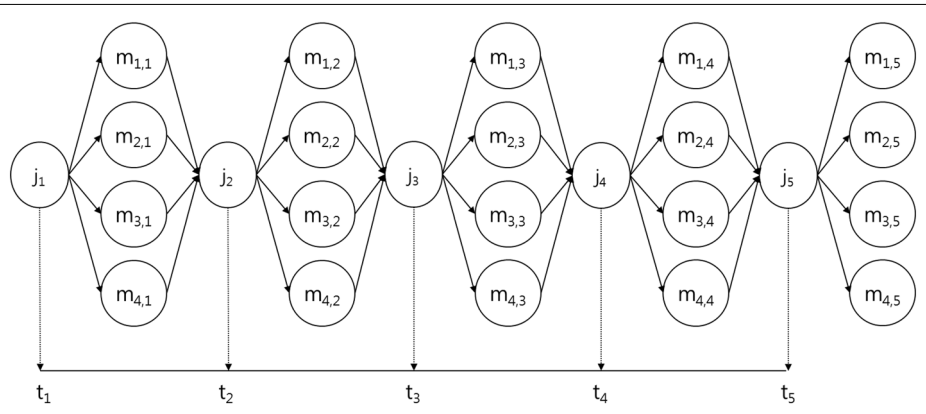

Fig. 7 Generated graph-based motor primitive

that UAVs only fly according to the predefined path. However, because of dynamic environments, it is very hard to perform selected motor primitives completely. The motor primitives that reflect dynamic environments are demanded. In this paper, motor primitives are expressed by graph, which means that graph-based motor primitives are performed completely more than linear motor primitives. Therefore, graph-based motor primitives are proper to dynamic environments. In the future, the way to perform graph-based motor primitives needs to be handled.

Authors' contributions
All authors are participated in the experiment and the writing of this paper. All authors read and approved the final manuscript.

Author details
[1] Faculty of Computer Engineering, Keimyung University, Daegu 704-701, South Korea. [2] Department of Computer Engineering, Keimyung University, Daegu 704-701, South Korea. [3] Department of Computer Science and Engineering, Seoul National University of Science and Technology, Seoul 139-743, South Korea.

Acknowledgements
This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT & Future Planning (NRF-2014R1A1A1005955).

Competing interests
The authors declare that they have no competing interests.

Received: 23 July 2015 Accepted: 26 October 2015
Published online: 15 December 2015

References

1. Sung Y, Kwak J, Yang D, Park Y (2015) Ground station design for the control of multi heterogeneous UAVs. In: Korean multimedia society spring conference, Andong, May 2015, vol 18, no 1, pp 828–829
2. Sung Y, Kwak J (2015) Tangible control interface design for drones of fire fighting and disaster prevention. In: KIPS fall conference, Jeju Island, October 2015, vol 22, no, 2, pp 1844–1845
3. Calinon S, Guenter F, and Billard A (2007) On learning, representing, and generalizing a task in a humanoid robot. In: SMC'07: proceedings of IEEE transactions on systems, man, and cybernetics, vol 37, no 2, pp 286–298
4. Koenig N, Matarić MJ (2006) Behavior-based segmentation of demonstrated task. In: ICDL: Proceedings of international conference on development and learning, Bloomington, May 2006
5. Nicolescu MN, Matarić MJ (2001) Experience-based representation construction: learning from human and robot teacher. In: IROS'01: Proceedings of IEEE/RSJ international conference on intelligent robots and systems, Wailea, October 2001, vol 2, pp 740–745
6. Nicolescu MN, Matarić MJ (2001) Learning and interacting in human-robot domains. In: SMC'01: Proceedings of IEEE transactions on systems, man, cybernetics, vol 31, no 5, pp 419–430
7. Nicolescu MN, Matarić MJ (2002) A hierarchical architecture for behavior-based robots. In: AAMAS'02: Proceedings of the first international joint conference on autonomous agents and multi-agent systems, July 2002, pp 227–233
8. Nicolescu MN, Matarić MJ (2003) Natural methods for robot task learning: instructive demonstrations, generalization and practice. In: AAMAS'03: Proceedings of the second international joint conference on autonomous agents and multi-agent systems, Melbourne, July 2003, pp 241–248
9. Sung Y, Kwak J, Park JH (2015) Graph-based motor primitive generation method of UAVs based on demonstration-based learning. In: The 9th international conference on multimedia and ubiquitous engineering, Hanoi, Vietnam, 18–20 May 2015
10. Bristeau JP, Callou F, Vissière D, Petit N (2011) The navigation and control technology inside the AR.Drone micro. In: UAV IFAC'11: Proceeding of international federation of automatic control, Milano, Italy, 28 August–2 September 2011, vol 18, no 1, pp 1477–1484
11. Nitschke C, Minami Y, Hiromoto M, Ohshima H, Sato T (2014) A quadrocopter automatic control contest as an example of interdisciplinary design education ICCAS'14. In: Proceeding of 14th international conference on control, automation and systems, KINTEX, Gyeonggi-do, Korea, 2014, pp 678–685
12. Son BR, Kang SM, Lee H, Lee DH (2013) A real time quadrotor autonomous navigation and remote control method. IEMEK J Embed Syst Appl 8(4):205–212
13. Cho DH, Moon ST, Rew DY (2014) Development of AR.Drone's controller for indoor swarm flight. KSAS Korean Soc Aeronaut Space Sci 13(1):153–165
14. Moo ST, Ha SH, Eom W, Kim WK (2014) 3D map generation system for indoor autonomous navigation. KSAS Korean Soc Aeronaut Space Sci 11(2):140–148
15. Kim JM, Choi HW, Eom DS (2014) The location estimation of UAV in indoor environments using a crowd-sourced fingerprint mapping. Korean Inst Commun Inf Sci 54(1):914–915