Computational Thinking Curriculum for Unmanned Aerial Systems

Shiqi Zhang and Christopher Stewart, The Ohio State University

***Abstract*—Unmanned aerial systems (UAS) can explore common, vast and unsafe places at low cost. They could transform multiple sectors from photography to farming to city planning. However, the software underlying UAS is complex and requires multiple distinct programming skills, e.g., AI, machine learning and flight control. Few programmers encompass these skills, hampering software development and dampening the impact of UAS. We contend that early exposure to UAS software could help align workforce skills. However, early exposure requires curriculum that (1) captures the breadth of UAS software, (2) supports multiple levels of depth for diverse programming backgrounds and (3) fits within resource and institutional challenges. We propose a computational thinking framework. In our approach, lessons fit within 20-30 minute instructional blocks, making them usable in short workshop and extended classroom settings. UAV topics and computational thinking depth link lessons. Teachers can trade breadth for in-depth coding and vice versa. In early work, we presented an autonomous UAS to middle school students. Our 1 hour workshop focused on breadth and was received well.**

I. Introduction

Unmanned aerial systems (UAS) are small aircraft equipped with compute hardware and sensors. Software pilots the aircraft. Often, humans use smart phones or remote controls to send real time commands to aircraft directly. However, increasingly, the UAS itself decides where to fly and when to use sensors—i.e., systems are increasingly autonomous. Since UAS do not carry humans, they can explore dangerous areas. For example, Djedi robot, built by University of Leeds, was used to explore the pyramids in 2012 [1].

UAS can also fly more frequently than human piloted aircraft. Neither human nor equipment exhaustion are major risk factors. This enables UAS to cover large areas with repeated flights. In agriculture, UAS provide detailed field maps for yield modeling. UAS also do not incur labor costs associated with traditional flight. They can be flown in common places where marginal profit is low.

UAS have many unrealized applications. Each of the above use cases can support deeper implementations. Agriculture applications can scan fields for multiple problems and address small issues autonomously. Everyday selfie apps could sense environments, recommend photo opportunities and support multiple users and swarms. However, these applications require programming autonomous systems. Autonomous systems combine programming skills normally taught in isolation. AI governs how the aircraft explores its surroundings, deciding where to fly and when to land. Machine learning interprets sensed data. Adaptive and autonomic systems adapt energy use between components to lengthen flight time. Flight controllers manage the physics of aerial maneuvers. Few programmers encompass these skills and appreciate their effects on each other. This inflates software development teams, delays products and slows the realization of UAS applications.

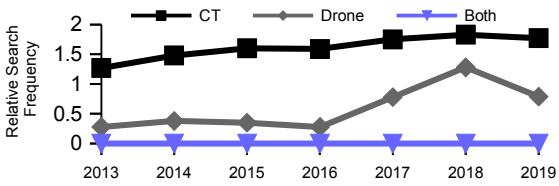

Fig. 1. Related Keywords Search Frequency for past 7 years

Figure 1 plots keyword search frequency on Google Trends. The term *drone* is searched 5X more often than 7 years ago. Students would appreciate exposure to this emerging technology [5]. However, UAS present challenges for early exposure. First, UAS software includes many concepts. Outreach events commonly used to expose students to emerging technology are short. UAS curriculum must introduce concepts efficiently. Second, programming experience varies across students attending outreach events. Curriculum must be flexible to reach a wide range of schools and students. Figure 1 also reports the growth of computational thinking. This pedagogy is widely used to introduce computer science concepts.

This paper presents a framework for outreach events targeting UAS software. Our approach employs computational thinking pedagogy to simplify UAS software teaching. We provide distinct lessons for abstraction and generalization, decomposition, and algorithms and debugging. The pedagogy provides increasingly complex material for beginners and experts. In addition, we divvy UAS software by function, including: AI planning, data collection and analysis and flight control. Our curriculum

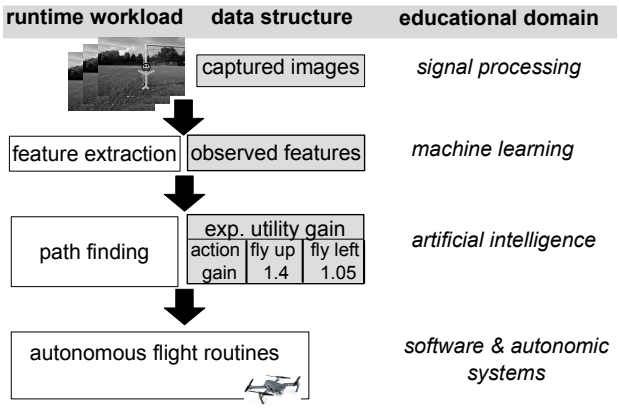

Fig. 2. Fully Autonomous UAS Runtime

concept	definition	UAS example
abstraction	setting an object's interface	*fly 1 ft. up, down, left & right*
generalization	solve multiple problems using one abstraction	*take self portraits of multiple users*
decomposition	break complex problems into smaller, solvable parts	*flight area grid; utility*
algorithms	step-by-step actions to solve a class of problems	*reinforcement learning*
coding	use programming languages to realize algorithms	*use APIs for flying & ML*
debugging	find and fix errors	*tweak training data; A* params*

TABLE I
COMPUTATIONAL THINKING PEDAGOGY

comprises lessons for each combination of computational thinking and software function topic. The lessons can be combined to align with outreach event lengths.

In early work, we presented a workshop based on our curriculum to middle school students. Our 1 hour workshop focused on breadth and was received well.

In this paper, we present work in progress toward a comprehensive curriculum for UAS software. Our approach is holistic and flexible and we have early positive results. The remainder of this paper is organized as follows: Section II provides background on UAS software, focusing on emerging fully autonomous software. Section III presents a pedagogical framework to expose early learners to UAS software. Section IV provides early results with outreach event targeted at middle school girls. Section V discusses future work and draws conclusions.

II. AUTONOMOUS UNMANNED AERIAL SYSTEMS

UAS can perform well with human defined actions. It can perform taking off, landing, hovering, and taking pictures, etc. With the popularity of AI, less or even none human intervention are increasingly realistic. *Fully autonomous aerial systems (FAAS)* allow end users to set mission goals and constraints. Then, software pilots aircraft—humans do not to control flight [13], [12]. Our curriculum attempts to present FAAS through outreach workshops.

Figure 2 shows FAAS runtime workload. First of all, user data need to be taken. For purpose of the simplicity, pictures are considered as main data inputs in this paper. The features are extracted from pictures. Then FAAS relies on image processing to compute path finding. AI model converts images into multidimensional points. Then autonomous software computes the next waypoint. To choose the next waypoint, autonomous software computes expected utility gain for each possible reaction. Finally, FAAS decides where or when to stop the mission [6], [2]. We should note that there is an intrinsic challenge of balancing compute needs and model accuracy, both in image processing and and reinforcement learning [7], [8].

This type of design requires signal processing, machine learning, image analysis, feature extraction and also AI path planning algorithms. Based on this runtime work flow, we have designed several self-flying systems. One of them is called password game. This software is able to recognize people's face, and react to pictures with people's face by shaking the Gimbal. And after the certain sequence of feature images taken, the drone will land. More details about the game are presented in the early result section.

III. CURRICULUM DESIGN

We design a framework to expose early learners to UAS software. The frame is designed based on computational thinking studying pedagogy and UAS software complexity. Each lesson fits within 20-30 mins instructional block to provide flexibility for workshop and class setting. Therefore, this framework can be used to develop various UAS workshop based on either the complexity of the topic or the best practice of computational thinking. As we can see in the Figure 3, the y-axis is divided into three sections based on the functional complexity of UAS software, in other words, the breath of the topic involved in UAS software. Flight and Energy management is the basic user implementation, including flying control based on path and energy consumption. Data and Machine Learning covers topics including features extracting and training data. AI planning and space exploration focuses on FAAS implementation. The x-axis is designed by different depth of studying. To make the depth studying process more efficient, we introduce computation thinking studying process into our curriculum. As we can see from the Table I, the computational thinking studying process is divided into 6 different phrases [3]. The first two phrases abstraction and generalization focus on finding out the general ques-

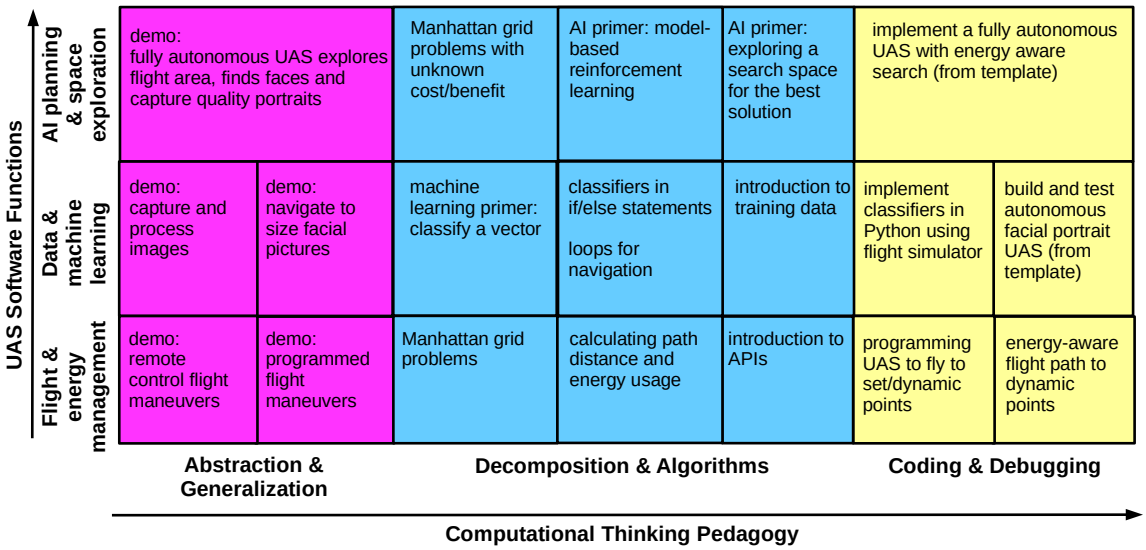

Fig. 3. Computational Thinking Curriculum for Unmanned Aerial Systems

tion pattern. The decomposition and algorithm phrase focuses on how to break down the complex topic into pieces to find out the solution to each topic and find out in general skeleton solution to solve the problem. The coding and debugging phrase mainly teaches learners how to turn algorithm into practical running code.

Abstraction and generalization phrases, our curriculum is designed to have demos to expose learners about what final results are. During the second phrase, decomposition and algorithm, we designed to have specific topics/knowledge studying to prepare learners to be ready for the final coding and debugging. This framework generally is designed followed by computational thinking studying with one extra step, debugging.

If the workshop focuses on exploring breadth of the topic, the workshop can be designed along y-axis. After completing all topics on the same depth, learners are expected to get fully experience of the topic. For example, if learners/lecturers want to focus on exploring how board UAS topic will be, the workshop design can choose the first phrase on the x-axis, Abstraction & Generalization, and go along with y-axis. This type workshop will provide student fully experience about how the UAS can be performed from simply flight control to FAAS. Teaching the in-depth workshop, going along with x-axis, gives the whole picture of designing process for students. For example, lecturers can present a workshop to let people expose to Flight & Energy Management from the demo presenting to in-depth coding, by completing all Flight and Energy Management topic studying, by going along with x-axis with the same complexity. At the end of the workshop, learners are

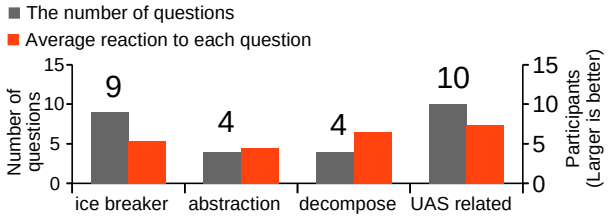

Fig. 4. Number of questions v.s. Average Participants for each question

expected to have fully knowledge about how the demo has been created.

By completing the whole curriculum sections in the coordination, the learners are expected to have fully experience about UAS software, from flying control to FAAS creation.

IV. Early Results

We created a 45-minute UAS-related breadth focused workshop, at a weekend camp called Buck-i-Code, held by Association of Computing Machinery Womens Chapter. During the workshop, we used two cameras to record the reaction of students, including the number of hand raising and responding time for each question. We divided the whole classroom into four sections, Front Right(A), Back Right(B), Front Left(C), Back Left(D) to collect data.

The lecture had four parts, including ice breaking exercise, remote control game, selfie-drone project introduction, and UAS demo – the password game. Twenty-seven questions were presented, nine of them were ice breaker,

four of them were about abstraction, four of them were about decompose, and the rest ten question were all UAS related. All questions were evenly presented out during the workshop. There were 36 people participants in the workshop.

The password game, which was implemented by UAS software and performed on the drone, DJI Mavic, aiming at giving students Flight and Energy Management related demo. A password drone was able to perform two main tasks. First, the drone could recognize if there was at least one person's face in the picture. Secondly, the drone could able to tell if pictures with faces were taken in the defined sequence. If the faces showed up in the certain sequence, the drone could land, or it would hover. During the workshop, we set up the binary password as 1001 manually where 1 stood for there was at least one humans face in the picture, and 0 stood for there was no human face in the picture. Drone took up to four pictures to check if this four-digit password, digit by digit, was correct or not. After we sent out the command to start the drone, the drone took off to four feet high and take the first picture. Based on the example we gave out, if the first picture contained at least one humans face, the Gimbal would shake twice and prepared to take the second picture, or if there was no face at all, the Gimbal would shake four times and hover, and stopped taking picture to indicating the picture was not expected. Since the password was binary, if the password 1 was not correct, 0 must be correct. This process would easily walk student through how the computer checked the password.

From result in Figure IV, we could see the UAS related questions was the most responsive, and each question had 7.4 people responded. This was the highest responding number of the topic. Also the most responsive question, having 27 people out of 36 raising their hands to react, was an UAS related question.

V. Future Work

We plan to continue to explore this curriculum. We would like to build open source repositories and augment established computer science educational approaches [4], [11]. Computational thinking pedagogy allows students to explore UAV and AUAV at multiple levels. We would also like to extend our curriculum to consider the additional layer of resource management and introduce operating system and cloud provisioning for this emerging workload [16], [14], [15], [17], [9], [10]. From an engineering perspective, our three immediate tasks include the following. Since each instructional block stands for 20-30 min lecture, we plan to prepare slides sets of all instructional blocks across the coordination. To prepare for in-depth coding part, we plan to complete the autonomous UAS related API. Considered about the length of the workshop, we plan to provide code skeleton of each in-depth coding exercises to avoid students starting from scratch during the workshop, to improving the quality of the workshop. We hope our curriculum will help students at all ages be able to expose to studying about UAS within limited amount of time, and trigger their research interest of related topic in their future studying.

References

[1] Abigail Beall. Meet the robots going places humans cannot reach. https://www.t3.com/features/meet-the-robots-going-places-humans-cannot-reach, 2017.

[2] J. Boubin, J. Chumley, C. Stewart, and S. Khanal. Autonomic computing challenges in fully autonomous precision agriculture. In *IEEE International Conference on Autonomic Computing*, 2019.

[3] Charoula Angeli, Joke Voogt, Andrew Fluck, Mary Webb, Margaret Cox, Joyce Malyn-Smith, Jason Zagami. A k-6 computational thinking curriculum framework: Implications for teacher knowledge. *Journal Of Educational Technology & Society*, 2016.

[4] P. Derosa, K. Shen, C. Stewart, and J. Pearson. Realism and simplicity: Disk simulation for instructional os performance evaluation. In *ACM SIGCSE Technical Symposium*, 2006.

[5] Google Trends. https://trends.google.com/trends, 2019.

[6] Jayson Boubin, Naveen Tukmur Ramesh Babu, Christopher Stewart, Shiqi Zhang, John Chumley. Managing edge resources for fully autonomous aerial systems. In *ACM Symposium on Edge Computing (SEC)*, 2019.

[7] J. Kelley, C. Stewart, N. Morris, D. Tiwari, Y. He, and S. Elnikety. Measuring and managing answer quality for online data-intensive services. In *IEEE ICAC*, 2015.

[8] J. Kelley, C. Stewart, N. Morris, D. Tiwari, Y. He, and S. Elnikety. Obtaining and managing answer quality for online data-intensive services. In *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, 2017.

[9] N. Morris, S. M. Renganathan, C. Stewart, R. Birke, and L. Chen. Sprint ability: How well does your software exploit bursts in processing capacity? In *International Conference on Autonomic Computing*, 2016.

[10] N. Morris, C. Stewart, L. Chen, R. Birke, and J. Kelley:. Model-driven computational sprinting. In *ACM European Conference on Computer Systems*, 2018.

[11] S. Renganathan, C. Stewart, A. Perez, R. Rao, and B. Braaten. Preliminary results on an interactive learning tool for early algebra education. In *IEEE Frontiers in Education*, 2017.

[12] J. L. Sanchez-Lopez, R. A. S. Fernández, H. Bavle, C. Sampedro, M. Molina, J. Pestana, and P. Campoy. Aerostack: An architecture and open-source software framework for aerial robotics. In *International Conference on Unmanned Aircraft Systems*, 2016.

[13] J. L. Sanchez-Lopez, M. Molina, H. Bavle, C. Sampedro, R. A. S. Fernández, and P. Campoy. A multi-layered component-based approach for the development of aerial robotic systems: the aerostack framework. *Journal of Intelligent & Robotic Systems*, 88, 2017.

[14] C. Stewart, A. Chakrabarti, and R. Griffith. Zoolander: Efficiently meeting very strict, low-latency slos. In *IEEE International Conference on Autonomic Computing*, 2013.

[15] C. Stewart, T. Kelly, and A. Zhang. Exploiting nonstationarity for performance prediction. In *European Conference on Computer Systems*, 2007.

[16] C. Stewart and K. Shen. Performance modeling and system management for multi-component online services. In *Symposium on Networked Systems Design and Implementation*, 2005.

[17] Z. Xu, C. Stewart, N. Deng, and X. Wang. Blending on-demand and spot instances to lower costs for in-memory storage. In *IEEE INFOCOM*, 2016.