To Learn Programming through Internet of Things

Yi-Bing Lin and Min-Zheng Shieh

ABSTRACT

Learning computer programming is a slow process that may make learners depressed. Studies suggested that the students should take a problem in real world and then translated it to program code to solve it. Unfortunately, most problems in real world require complicated inputs/outputs that are not directly provided in the basic functions of the programming languages. In this article, we use Internet of Things (IoT) as an appropriate input/output mechanism for the beginners to learn programming through every-day real-world problems. Specifically, we transparently integrate an IoT development platform called IoTtalk with the programs and/or programming ideas of the new learners. Our study indicates that new learners can catch programming skills in the proposed approach and create non-trivial applications with their imagination.

I. INTRODUCTION

Learning to program takes time and persistence. In [1], the author pointed out that programming is difficult. On the other hand, the author of [2] suggested that coding is not hard to learn. Although they have different opinions on the difficulties of learning programming, both of them agree that the beginner should understand that the learning process is slow. If the courses emphasize theoretical aspects of programming and the students do not receive enough feedback for their programming home works, then the learning process is even more difficult [3].

The students face several problems in learning programming. For example, they typically lack skills in analyzing problems, which results in ineffective use of teaching strategies for problem solving and coding. These problems were identified by a survey on 1000 students [4]. The study suggested that the students should take a problem in real world and then translated it to program code to solve it. The study in [5] further suggested that lab exams should replace pen-and-paper tests in the programming courses. In lab exams with real word problems, practical work on developing programs, including debugging, may frustrate the beginners. Therefore, beginners should understand that errors and frustrations are a natural part of the learning process [6].

(1.1) On the contrary, the study in [7] shows that challenge, enjoyment, and perceived value all greatly affect students' preferences on learning approaches. If the learners feel the joy and the importance of programming, they are likely to overcome the frustrations.

To succeed in learning programming, it is important to start with a beginner-friendly high-level language, for examples, Python [2] and Processing [8]. Besides the beginner-friendly language, one of the most popular environments for teaching programming to children is Scratch. Through the interaction with the Scratch environment, the children learn basic concepts, including operators, sequences, loops, conditionals, events, and data. The advantages of Scratch and how to improve the Scratch learning environment are out of the scope of this article, which can be found in [9]. Although Scratch is good for learning programming concepts, it cannot be used to develop sustainable programs, especially for commercial purposes. Therefore, it is essential to bridge programming learning from Scratch to a high-level programming language.

Another popular topic is "learning IoT application programming". To address this issue, several tools have been used to create innovative IoT applications on embedded systems such as Arduino, MediaTek LinkIt Smart 7688 duo, ROHM IoT kit, and ESP8266 ESP-12F. The programming environments include Arduino Yun, Webduino, ArduBlock, mBlock, and Minibloq. Details of these tools are described in [10] and the references therein.

This article aims to provide an easy way to learn programming. We would like to achieve three goals that allow the beginners to

1. learn the program logic concepts of the real-world applications at the beginning of the class (through the connect-the-dot approach)
2. learn simple logic for Python (through the fill-the-blanks approach)
3. bridge Scratch programming to Python learning (if the first programming environment for children is Scratch)

The proposed mechanism to achieve the above goals is developed based on an Internet of Things (IoT) application platform called IoTtalk [11]. In this learning mechanism, students will also unknowingly learn the concept of IoT.

To make program teaching more efficient, blended learning was proposed to mix the traditional teaching

method with online learning [3]. Instead of blended learning, the costs of learning programming can be reduced by pure online learning. Indeed, online learning, in particular, collaborative learning, is widely believed to be effective [5]. Following this approach, IoTtalk is a cloud solution that allows one to learn programming online through web-browser access.

This article is organized as follows. Section II describes how a smartphone can be easily included as the inputs/outputs of IoTtalk so that a beginner can focus on learning program logic without being distracted from the tedious details of inputs/outputs. Section III shows how Arduino boards can be transparently accommodated in IoTtalk so that the beginner can focus on IoT application programming without being bothered by the details of Arduino setup. Section IV elaborates on integrating IoTtalk with various programming languages. Section V shows how to detect program configuration and type errors for the new learners automatically.

II. SMARTPHONE AS AN INPUT/OUTPUT MECHANISM

As we pointed out, a student should take a problem in real world and then translated it to program code in the programming course. Unfortunately, our experience indicates that the new learners are often confused by the input/output (I/O) mechanisms of a computer language. In particular, to create I/O for real world applications are not trivial. In this article, we propose to use a smartphone as a major mechanism for inputs/outputs of the student's programs. Clearly, smartphones are used in daily activities that are real world applications. Therefore, it makes sense to use smartphones in learning programming. Specifically, IoTtalk allows beginners to learn the I/O concepts through their smartphones transparently.

Many approaches require the installation of mobile apps in smartphones for accessing smart applications. In [12], we used smartphones to access applications without the need to install any app software. Suppose that we want to use the orientation sensor of a smartphone to control the "moon orbiting earth" animation. We first use the smartphone to scan the QR code for accessing the amination (Fig. 1 (1)), where the web browser of the smartphone automatically establishes a data connection to the animation program (The details will be given in the Appendix). Then the (α, β, γ) values of the orientation sensor (Fig 1 (2)) are transformed into the gravity of the earth and the speed of the moon (Fig. 1 (3)) to create the moon orbiting earth animation.

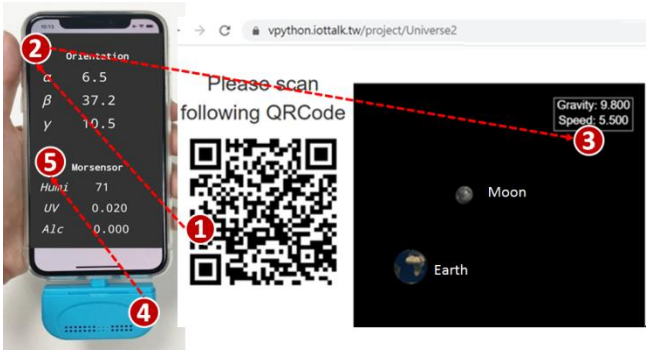

Fig. 1. Interactive moon orbiting earth animation

The sensors of smartphones have been intensively used for games and physics experiments carried out on our programming class. For example, we have developed a VR game called "A Boy Fighting Devil Party" (諸葛四郎大戰魔鬼黨), which allows several students to simultaneously shake smartphones (i.e., changing accelerations) to enhance, e.g., the sandy wind effect in the game. All audiences experiencing the game loved the effects. Note that some sensors or actuators such as flashlight and sounds are not standard, and some handset manufacturers do not allow access to these features through web browsers. For the sensors not built in the smartphones, we have developed the "MorSensor" module that can be plugged in the smartphone [12]. As illustrated in Fig. 1 (4), the blue MorSensor consists of three sensors for humidity, ultraviolet, and alcohol, and their readings are shown on the screen of the smartphone (Fig. 1 (5)).

A new learner can easily create a smartphone-controlled application through "connect-the-dot". The IoTtalk GUI provides a device model list "Model" (Fig. 2 (1)) that collects all IoT devices that can be accessed by IoTtalk. If we select an item in the list (e.g., "Smartphone" in Fig. 2 (2)), then the icon of the corresponding device (named "Remote"; see Fig. 2 (3)) is shown in the IoTtalk GUI window. To build a voice-controlled fan application, we select the smartphone as an input device (placed at the left hand side of the window) for remote voice control. We also select the fan as the output device to be controlled, and its icon is placed at the right hand side of the window (Fig. 2 (4)). To control the fan by the smartphone, we only need to drag a line from the "Remote" icon to the "Fan" icon. In Fig. 2 (3), the keyboard of the smartphone is used to control the fan. Alternatively, the microphone of the smartphone can be used for voice control. When the child shouts louder to the microphone of the smartphone, the fan will blow stronger. IoTtalk automatically maps the voice level to the speed of the fan motor through the "dynamic ranging" feature proposed in [11]. One can also write a control program to map the voice level to the fan speed. The details will be given in Section IV.

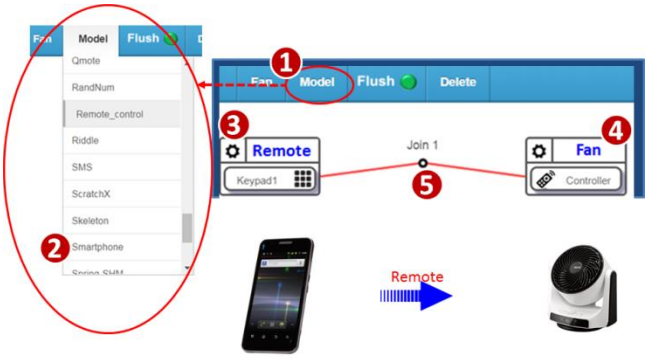

Fig. 2. Voice controlled fan application

III. ARDUINO AS AN I/O MECHANISM

Besides smartphones, we may need to connect other IoT devices to the students' programs. In Fig. 2, we assume that the fan is a device already built in IoTtalk. However, the students may need to include other IoT devices to serve as the I/O of their programs. In application development, Arduino is often used for controlling IoT devices [10]. Several Arduino-based tools provide solutions for IoT device connections on a single Arduino board. In these solutions, intensive C programming efforts are required, and interactions among multiple Arduino boards need significant work to establish. To mitigate the difficulty of using Arduino for a new learner, we proposed ArduTalk [10], an Arduino device model built in IoTtalk, which allows using a web browser to develop applications with multiple Arduino boards. Consider a security light & curtain application deployed in the front door with a glass window. During the night, the light in the front gate is turned off, and the window curtain of the front door may or may not be open. When someone passes through the front door, the PIR motion sensor will detect the movement and send a signal to activate the motor to close the curtain and turn on the light. Also, we have two switches to manually control the curtain and the light, respectively. In this application, the PIR motor sensor is connected to the input pin A0 in an Arduino board "Arduino1" (Fig. 3 (a)). The light is connected to the output pin D4 in Arduino1 (Fig. 3 (b)). The curtain motor is connected to the output pin D5 of Arduino1 (Fig. 3 (c)). The light switch is connected to the input pin A1 of the Arduino board "Arduino2" (Fig. 3 (d)), and the curtain switch is connected to the input pin A0 (Fig. 3 (e)). The student can easily implement this application through the IoTtalk GUI. Note that in Fig. 3, the input pins and the output pins of Arduino 1 are illustrated in different boards for the demonstration purpose. In reality, these pins are located on the same physical board.

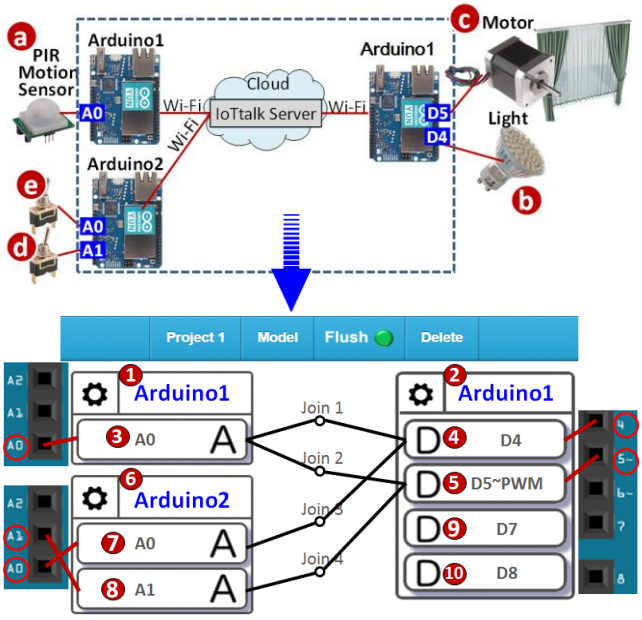

Fig. 3. Security light & curtain

In IoTtalk, we have built in the Arduino model that can be selected from the Model list of the GUI. For the security light & curtain application, we create the Arduino1 and the Arduino2 icons in the GUI. These icons are bound to the real Arduino boards by the IoTtalk server [11]. The input pin icons of Arduino 1 (Fig. 3 (1)) are automatically connected to the corresponding physical input pins of the Arduino 1 board, and the output pin icons (Fig. 3 (2)) are connected to the physical output pins. Similarly, the input pin icons of Arduino 2 (Fig. 3 (6)) are connected to those of the physical Arduino 2 board. To implement the security light & curtain application, the student connects the A0 icon (Fig. 3 (3)) to the D4 icon (Fig. 3 (4)) and the D5 icon (Fig. 3 (5)). The A0 icon (Fig. 3 (7)) is connected to the D4 icon (Fig. 3 (4)) for the light switch. The A1 icon (Fig. 3 (8)) is connected to the D5 icon (Fig. 3 (5)) for the curtain switch. IoTtalk will automatically generate the network program (Python codes) to implement the data channels for Joins 1-4. Through connecting the icons, the student learns the program logic for data flow without involving tedious Arduino installation and C programming work.

IV. INTEGRATING VARIOUS PROGRAMMING LANGUAGES WITH IOTTALK

Besides "connect-the-dot" described in Sections II and III, IoTtalk provides "fill-the-blanks" for the students to learn programming logic by writing a small segment of Python codes. Consider the example in Fig. 2. Besides voice control, we can use the temperature sensor (e.g., a temperature MorSensor attached to the smartphone in Fig. 1 (4)) to automatically control the fan. To create this feature, the student simply clicks the Join 1 circle (Fig. 2 (5)). Then the "Function Management" window in Fig. 4 (a) pops up.

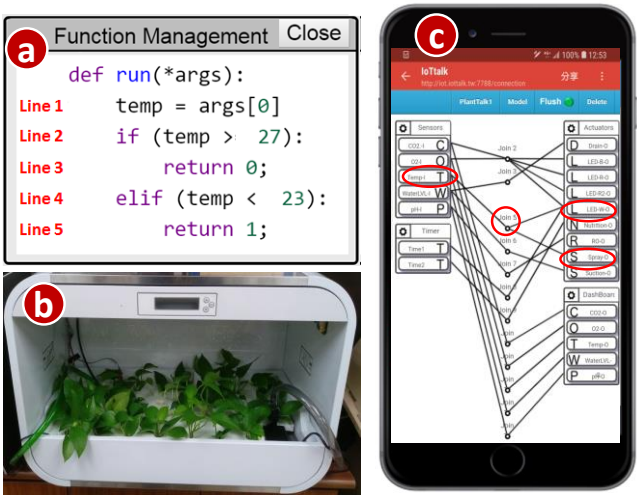

Fig. 4. Temperature control Python code

In the window, the student writes the following Python code: In Line 1, "args[0]" is the input from the smartphone (Fig. 2 (3)), which is assigned to the variable "temp". In Line 2 of Fig. 4 (a), if the room temperature is higher than 27℃, then Line 3 returns 0 (to turn on the fan). If the room temperature is lower than 23℃ in Line 4, then Line 5 returns 1 (to turn off the fan). If the room temperature ranges from 23℃ to 27℃, then the function returns "None", which does nothing. This function can be saved in IoTtalk to be reused by other applications. For example, in a smart application for a plant box (Fig. 4 (b)), the Join 5 link in the IoTtalk GUI (the red circle in Fig. 4 (c)) connects the temperature (the "T" icon) to control the fill light (the "L" icon) and the water spray (the "S" icon). The Join 5 circle can simply reuse the function in Fig. 4 (a) such that if the plant box temperature is higher than 27℃, both the light and the spray are turned on. When the plant box temperature is lower than 23℃, both the light and the spray are turned off. The only difference between the fan and the plant box applications is that in the former case, the Join output "0" means "turn on", and it means "turn off" in the latter case. In the above example, the student learns function logic and "software modularization" concept that the function can be reused by other programs.

IoTtalk can also automatically translate an animation program into an output IoT device if the animation is written in Processing, Scratch, Python, or Unity (The details will be given in the Appendix). Therefore, if a child only learns, e.g., Scratch, then she/he can use her/his smartphone to automatically generate the input to the Scratch animation program for real time interaction. In Fig. 5 (1), the student writes a "morphing animation" by using Processing (a programming language designed for electronic art and capable of interacting with Java). IoTtalk provides QR code scanning that connects this animation (Fig. 5 (2)) to a smartphone (Fig. 5 (3)). When one shakes the smartphone, the object in the animation will change its shape, its color, and its motion. The details can be found in a demo video [13]. Fig. 5 (4) illustrates the "cat catching mouse" program written in Scratch, where a smartphone (Fig. 5 (5)) transparently controls cat movement in the Scratch animation (Fig. 5 (6)). An animation video for this application can be found in [14]. Fig. 5 (7) shows a Python program for the snake pendulum physics experiment. Though "fill-the-blanks", the student fills the equation of the snake pendulum movement she/he learned from the physics class in the red area of Fig. 5 (7). Then one can shake the smartphone to change the speed and the gravity to affect the snake pendulum behavior (Fig. 5 (8)). Fig. 5 (9) is a tree animation developed by using Unity. IoTtalk automatically connects this animation to a micro weather station Fig. 5 (10). Therefore the light intensity and humidity readings of the weather station are shown in Fig. 5 (11), and the tree grows according to the weather conditions. The details can be found in a demo video [15].

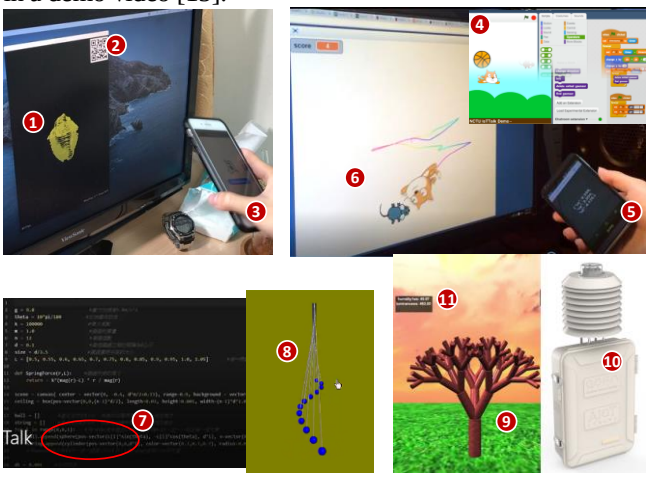

Fig. 5. Integrating IoTtalk with animations written in various programming languages: (1) Processing; (4) Scratch; (7) Python; (9) Unity

The examples in Fig. 5 are sustainable without any maintenance cost. They can be played anytime and anywhere by anyone with his/her smartphone. No extra IoT hardware is required except for the micro weather station. In fact, the weather station can be replaced by the open data provided by the local weather bureau.

V. REMINDING THE PROGRAMMING ERRORS

When the new learners create Join links, they may connect to the wrong icons. To resolve this issue, we propose BigraphTalk [16], a verification framework for IoTtalk, which utilizes the bigraph formal techniques [17] to statically guarantee that unwanted configurations do not arise. In particular, we check for invalid connections between devices. BigraphTalk provides fully automated verification and feedback without end-users ever needing to specify a bigraph. Therefore any application specifiable in IoTtalk is guaranteed, so long as verification succeeds, not to violate the given configuration constraints when deployed, with no extra cost to the user.

Consider an application where the acceleration sensor of a smartphone (Fig. 6 (1)) is used to control a curtain (Fig. 6 (2)). Through Join 1, the acceleration sensor is connected to

the Down, the Up, and the Stop switches of the curtain. This connection will result in oscillation because both the Up and the Down actions are taken. Such a mistake will drive the motor in different directions and potentially lead to hardware damage.

The above IoTtalk configuration is automatically transferred into the bigraph notation in Fig. 6 (3) and is verified by an open-source tool BigraphER [16]. The verification will indicate that the configuration is not allowed. Specifically, the link connecting the three red circles in Fig. 6 (3) represents a forbidden configuration. Open links at the top of the diagram (Fig. 6 (4)) are used to type check the Join. BigraphTalk checks all data transforms, implemented in the Joins, for type errors, e.g., passing a float to a Boolean switch. Since Python is not a statically typed language, when a beginner makes mismatched type errors, the Python interpreter will not detect such errors. On the other hand, BigraphTalk will remind the beginner to avoid such errors. Details of Bigraph (Fig. 6 (3)) are out of the scope of this article, which can be found in [16].

Execution of IoTtalk with verification may require a lot of resources in the cloud computing environment, especially when we support many students in a programming class. The computing resources of IoTtalk are investigated as follows. We host an IoTtalk cloud server of 4 virtual core at 2.1GHz with 8GB memory. With the HTTP communications, the interaction between a student and the IoTtalk server is non-session oriented, which requires a small amount of memory (1.4MB per connection), but the CPU consumption is significant. With the MQTT communications, the interaction between the student and the IoTtalk server is session oriented, which requires a large amount of memory (10.1MB per connection), but the CPU consumption is insignificant. Fig. 6 (5) shows the CPU consumptions against the number of student connections. The curves indicate that HTTP-based IoTtalk consumes 50% more CPU resource than that of MQTT-based IoTtalk. Fig. 6 (6) plots the memory usage against the two communication approaches. Clearly, the MQTT version consumes much more memory than the HTTP version due to the session nature. The CPU and the memory consumption curves indicate that we can support over 300 students to use IoTtalk simultaneously.

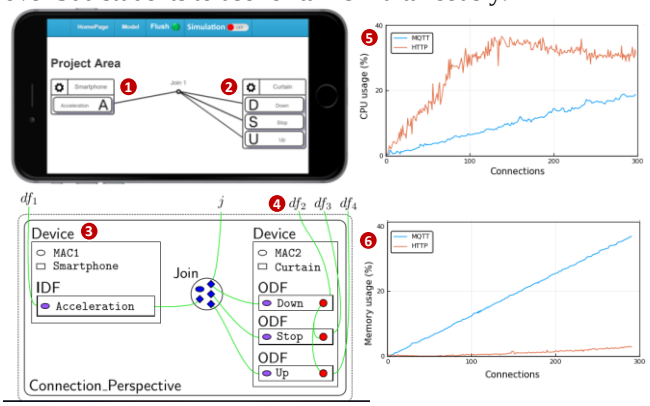

Fig. 6. Code verification and CPU/memory consumptions for program executions

VI. CONCLUSIONS

This article proposed an easy way to learn programming using IoT. Through the IoT platform called IoTtalk, we transparently integrate smartphones, Arduino boards, and other IoT devices as inputs/outputs of program development for the new learners so that they can focus on program logic for real-world applications. IoTtalk solution has the following advantages:

1. The student's computer is an installation-free client, and the programming environment in the cloud can be accessed through a web-based GUI.
2. Smartphone sensors are transparently connected to IoTtalk by scanning QR codes. There is no need to purchase extra sensor hardware for, e.g., acceleration, orientation, GPS, and so on.
3. The IoTtalk platform has installed various devices ready for use. New devices can be added through an open-source approach. They can be sustainably maintained with low costs.
4. Existing programming learning solutions can be easily integrated with IoTtalk. For example, Scratch may access devices on IoTtalk via extension blocks.

As an example of creative and interesting projects students can achieve as Makers do, two high school sophomores used what they learned from IoTtalk to design a virtual shooting machine, a darts machine, and a doll-clamping machine, in which a player could interact with these games from anywhere with any smartphone. This work was submitted to participate in the 2019 Mobileheroes Competition of Ministry of Economic Affairs (Taiwan), competing with the products of universities, start-ups, and commercial companies, and was the only high school team to make the cut after three rounds of elimination among 167 teams, and was finally ranked top 30. This example indicates that by using the IoTtalk approach, the students can quickly master the programming skills to develop real-world applications.

ACKNOWLEDGMENT

Ming-Feng Shih contributed to the original idea of VPython for physics experiments. Jiun-Yi Lin conducted experiments to generate Fig. 6 (5) and (6). This work was supported in part by the Center for Open Intelligent Connectivity from The Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education in Taiwan, and Ministry of Science and Technology 108-2221-E-009-047, Ministry of Economic Affairs 107-EC-17-A-02-S5-007.

APPENDIX. INTEGRATING IOTTALK WITH VARIOUS PROGRAMMING LANGUAGES

Many applications mentioned in this article use the browsers on smartphones and computers to retrieve JavaScript codes from the website and execute them. IoTtalk offers a JavaScript template code that turns the browser into an input

or an output device in IoTtalk. For example, a user may permit the browser to access sensors built-in his/her smartphone and make it as an input device in Fig. 1 (2) by modifying the remote control template into the JavaScript accessed via the QR code in Fig. 1 (1). The web animation in Fig. 1 (3) also utilizes the JavaScript template code to interact with IoTtalk as an output device. IoTtalk receives the data from the smartphone and delivers it to the animation with proper transformation. This approach also applies to other web-based applications, including Scratch. By default, Scratch has no direct connectivity to smartphones and other IoT devices. However, Scratch allows users to make extension functions with JavaScript. We turn Scratch into an IoTtalk device that can transmit data in both directions. Hence, the Scratch program (Fig. 5 (6)) can retrieve the acceleration data from the smartphone (Fig 5. (5)) via IoTtalk.

For applications not fit in the browser environment, IoTtalk also offers Device APIs in various languages, including Java and Python. The sensors use the push function to send data to IoTtalk, and the devices use the pull function to retrieve data from IoTtalk proactively. IoTtalk also allows devices to register callback functions to receive data from IoTtalk passively. For example, with the IoTtalk Device API for Java, we have implemented the animation shown in Fig 5 (1). Another example is the VPython snake pendulum animation in Fig 5 (8). It uses the IoTtalk Device API to connect Python to IoTtalk. As a result, via IoTtalk, the application can retrieve data from various sensors, including built-in smartphone sensors and micro weather stations.

REFERENCES

[1] C. Eygi, "How to Deal With the Difficulties of Programming," Better Programming, March 4, 2020.

[2] P. Larkin, "Is Coding Hard to Learn?" Career Karma, April 2020.

[3] M. N. Demaidi, M. Qamhieh and A. Afeefi, "Applying Blended Learning in Programming Courses," in IEEE Access, vol. 7, pp. 156824-156833, 2019, doi: 10.1109/ACCESS.2019.2949927.

[4] A. S. Hashim, R. Ahmad and M. S. Shahrul Amar, "Difficulties in Learning Structured Programming: A Case Study in UTP," 2017 7th World Engineering Education Forum (WEEF), Kuala Lumpur, 2017, pp. 210-215, doi: 10.1109/WEEF.2017.8467151.

[5] R. Juárez-Ramírez, C. X. Navarro, V. Tapia-Ibarra, R. Macías-Olvera and C. Guerra-García, "What is Programming? Putting all Together - A Set of Skills Required," 2018 6th International Conference in Software Engineering Research and Innovation (CONISOFT), San Luis Potosí, Mexico, 2018, pp. 11-20, doi: 10.1109/CONISOFT.2018.8645956.

[6] A. Gomes, W. Ke, C. Lam, M. J. Marcelino and A. Mendes, "Student motivation towards learning to program," 2018 IEEE Frontiers in Education Conference (FIE), San Jose, CA, USA, 2018, pp. 1-8, doi: 10.1109/FIE.2018.8659134.

[7] K. J. Harms, E. Balzuweit, J. Chen and C. Kelleher, "Learning programming from tutorials and code puzzles: Children's perceptions of value," 2016 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), Cambridge, 2016, pp. 59-67, doi: 10.1109/VLHCC.2016.7739665.

[8] H. Tsukamoto, Y. Takemura, H. Nagumo, I. Ikeda, A. Monden and K. Matsumoto, "Programming education for primary school children using a textual programming language," 2015 IEEE Frontiers in Education Conference (FIE), El Paso, TX, 2015, pp. 1-7, doi: 10.1109/FIE.2015.7344187.

[9] D. Pérez-Marín, R. Hijón-Neira and M. Martín-Lope, "A Methodology Proposal Based on Metaphors to Teach Programming to Children," in IEEE Revista Iberoamericana de Tecnologias del Aprendizaje, vol. 13, no. 1, pp. 46-53, Feb. 2018, doi: 10.1109/RITA.2018.2809944.

[10] Y.-W. Lin, Y.-B. Lin, M.-T. Yang and J.-H. Lin, "ArduTalk: An Arduino Network Application Development Platform Based on IoTtalk," in IEEE Systems Journal, vol. 13, no. 1, pp. 468-476, March 2019, doi: 10.1109/JSYST.2017.2773077. **(1.1)**

[11] Y.-B. Lin, Y.-W. Lin, C.-M. Huang, C.-Y. Chih, and P. Lin, "IoTtalk: A Management Platform for Reconfigurable Sensor Devices," in IEEE Internet of Things Journal, vol. 4, no. 5, pp. 1552-1562, Oct. 2017, doi: 10.1109/JIOT.2017.2682100.

[12] Y.-B. Lin, L.-K. Chen, M.-Z. Shieh, Y.-W. Lin and T.-H. Yen, "CampusTalk: IoT Devices and Their Interesting Features on Campus Applications," in IEEE Access, vol. 6, pp. 26036-26046, 2018, doi: 10.1109/ACCESS.2018.2832222.

[13] "Morphing Demo," 2020. https://youtu.be/71bnLSKTJDw

[14] "Cat-Mouse Demo," 2020. https://youtu.be/pokyllFIdYM

[15] "Tree-Growing Demo," 2020. https://youtu.be/bGb-pf3ra4s

[16] B. Archibald, M.-Z. Shieh, Y.-H. Hu, M. Sevegnani, Y.-B. Lin, "BigraphTalk: Verified Design of IoT Applications," in IEEE Internet of Things Journal, vol. 7, no. 4, pp. 2955-2967, April 2020, doi: 10.1109/JIOT.2020.2964026.

[17] R. Milner, The Space and Motion of Communicating Agents. Cambridge University Press: Cambridge, UK, 2009.

BIOGRAPHIES

Yi-Bing Lin (M'96-SM'96-F'03, liny@nctu.edu.tw) is Winbond Chair Professor of National Chiao Tung University (NCTU). He received his Bachelor's degree from National Cheng Kung University, Taiwan, in 1983, and his Ph.D. from the University of Washington, USA, in 1990. From 1990 to 1995 he was a Research Scientist with Bellcore (Telcordia). He then joined NCTU in Taiwan, where he remains. In 2010, Lin became a lifetime Chair Professor of NCTU, and in 2011, the Vice President of NCTU. During 2014 - 2016, Lin was Deputy Minister, Ministry of Science and Technology, Taiwan. Since 2016, Lin has been appointed as Vice Chancellor, University System of Taiwan (for NCTU, NTHU, NCU, and NYM). Lin is AAAS Fellow, ACM Fellow, IEEE Fellow, and IET Fellow.

Min-Zheng Shieh (mzshieh@nctu.edu.tw) received the B.S. and M.S. degrees in Computer Science and Information Engineering and the Ph.D. degree in Computer Science and Engineering, all from National Chiao Tung University, Taiwan, in 2003, 2004 and 2011, respectively. From 2012 to 2016, he served as an assistant research fellow of the Information and Communication Technology Laboratories, National Chiao Tung University. Since 2016, he has been an assistant professor of Information Technology Service Center at National Chiao Tung University.