Simulation Modelling Practice and Theory 101 (2020) 102015

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

# A fog computing model for implementing motion guide to visually impaired

Jinhui Zhu[a], Jie Hu[a], Mei Zhang[*,b], Yinong Chen[c], Sheng Bi[d]

[a] School of Software Engineering, South China University of Technology, Guangzhou, 510006, China
[b] School of Automation Science and Control Engineering, South China University of Technology, Guangzhou, 510640, China
[c] School of Computing, Informatics and Decision System Engineering Arizona State University, Tempe, AZ 85287, USA
[d] School of Computer Science and Engineering, South China University of Technology, Guangzhou, 510006, China

ARTICLE INFO

*Keywords:*
Guide dog robot
Visually impaired
Fog computing
Neural compute stick
Embedded system

ABSTRACT

A guide dog robot system for visually impaired often needs to process many kinds of information, such as image, voice and other sensor information. Information processing methods based on deep neural network can achieve better results. However, it requires expensive computing and communication resources to meet the real-time requirement. Fog computing has emerged as a promising solution for applications that are data-intensive and delay-sensitive. We propose a fog computing framework named PEN (Phone + Embedded board + Neural compute stick) for the guide dog robot system. The robot's functions in PEN are wrapped as services and deployed on the appropriate devices. Services are combined as an application in a visual programming language environment. Neural compute stick accelerates image processing speed at low power consumption. A simulation environment and a prototype are built on the framework. The simulated guide dog system is developed for operating in a miniature environment, including a small robot dog, a small wheelchair, model cars, traffic lights, and traffic blockage. The prototype is a full-sized portable guide system that can be used by a visually impaired person in a real environment. Simulation and experiments show that the framework can meet the functional and performance requirements for implementing the guide systems for visually impaired.

## 1. Introduction

Guide dogs or service dogs are assistance dogs trained to help blind and visual impaired people to move in complex road situations, including avoiding obstacle and observing traffic signs. It requires a long period of training time to turn out a qualified guided dog. The demand is higher than the available guide dogs, and the cost is high, which makes such guide dogs not affordable for many visually impaired people. Moreover, dogs are red-green colorblind and are not capable of interpreting traffic lights. They cannot interact with people through natural language.

As the development of robotics in the last decades, guide dog robots with vision and speech can overcome the above shortfalls. The complex information processing and real-time control tasks of guide dog robots, especially vision processing, require a large amount of computing resources to achieve real-time performance. Wei and Li [1] used a laptop on their robot to process the video streams. However, it is inconvenient for using laptop for some small mobile robots or wearable devices. Cloud computing has abundant computing resources and can provide various flexible and customizable services for mobile applications. Elmannai and

* Corresponding author.
*E-mail address:* zhangmei@scut.edu.cn (M. Zhang).

https://doi.org/10.1016/j.simpat.2019.102015
Received 29 June 2019; Received in revised form 25 September 2019; Accepted 16 October 2019
Available online 17 October 2019

Elleithy in [2] used an embedded system, whose image processing is on a remote server. However, the real-time performance of mobile application services is constrained by the network distance and bandwidth between mobile devices and the remote cloud computing remote servers, which will bring delay and reduce the quality of service. To solve this problem, Cisco implemented a supplement to the traditional cloud computing model, which is the fog computing model. Fog computing services are located on devices near end users, such as set-top boxes, which can eliminate unnecessary network jumps and greatly improve the real-time performance of mobile applications. Since the fog computing server near the terminal device handles a large amount of traffic from the terminal, the network congestion in the backbone network is alleviated to a certain extent. Like cloud computing, fog computing also provides computing, storage, and network services [3]. Nasir et al. proposed in [4] a framework for distributed summarization of surveillance videos over the fog network by clustersof Raspberry Pi. Using multiple Raspberry Pi devices, Hsu et al. [5] presented an agriculture platform for cloud and fog computing, which could be used for agricultural monitoring automation, pest management image analysis and monitoring.

In recent years, deep neural networks have been well utilized in computer vision and robotics [6]. Depth neural network can be well used in motion guidance system. However, running the neural network model requires a lot of computation power, which is a huge overhead for the device with limited resources.

This paper is based on our previous research [7], in which we performed some exploratory and simulation experiments on guide dog robots.We assumed that the visual impaired person is also motion impaired and thus require an wheelchair that can follow the robot dog. In this paper, we introduced a new fog computing framework for the guide dog robot system for speeding up the computation. Furthermore, a portable device is also included that allows a vision impaired person but not motion impaired person to wear the device to obtain the vision support. The new framework is named PEN (Phone + Embedded board + Neural compute stick). We use service-oriented architecture [8] and Robot as a Service (RaaS) concepts [9,10], where both software and hardware modules communicate with each through service interface and communication standards. The orchestration among the services is implemented using a visual programming language.

As shown in Fig. 1, we use a phone (P) as the decision-making center of the robot, embedded single board computer (E) as the control and computing node, Neural compute stick (N) as the computing accelerator. Cloud provides certain complex services that are not available at the fog. We use this model to construct the guide robots including wheelchair robot, dog robot and portable robot. These robots will be described in detail in subsequent chapters. The robots can help vision-impaired persons in different application scenarios, such as safe crossing the road, avoiding obstacles, assisting in taking buses, etc.

The contributions of this research are as follows:

(1) We propose a novel fog computing model named PEN, which is built on a smartphone, low-cost low-powered embedded board and neural compute stick. These three devices complement with each other. To the best of our knowledge, no such solutions have been addressed in the existing studies.
(2) Based on the model, we develop two prototypes for implementing motion guide for visually impaired. The fog computing framework can meet the functional and performance requirements of the guide system.
(3) We apply visual programming language ASU VIPLE [11] and MIT APP Inventor [12] to integrate multiple decentralized functional services with fog computing and cloud computing.

The remainder of this paper is organized as follows. Section 2 presents the system requirement, which shows all the functions of a

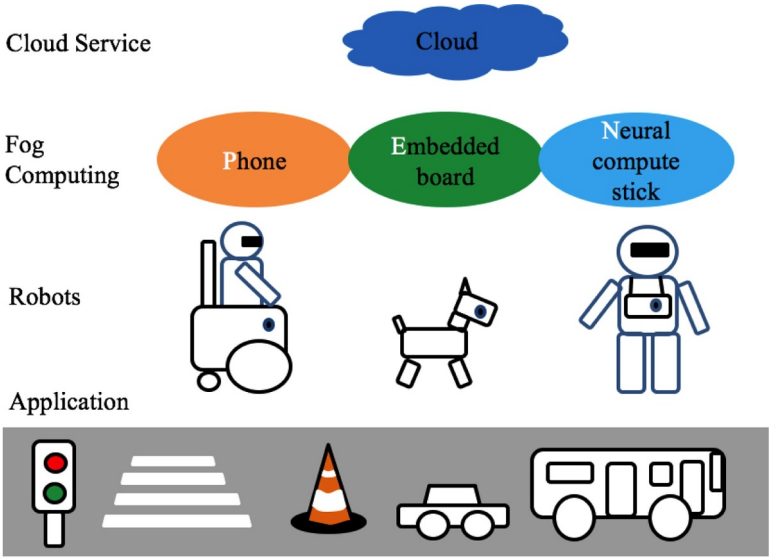



**Fig. 1.** Guide robot systems for visually impaired.

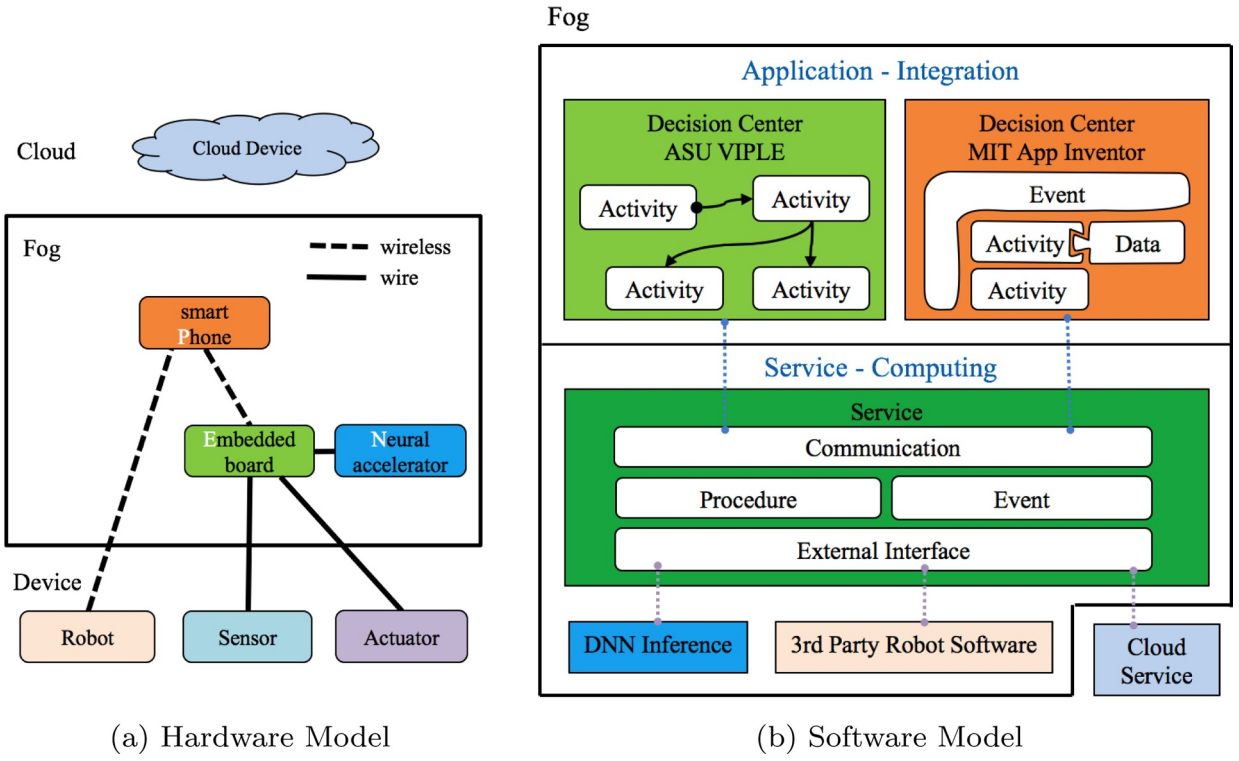



**Fig. 2.** PEN Model.

guide robot system. Section 3 introduces our PEN model and its components. In Section 4, we design a simulated guide system, as well as a portable guide system, based on our PEN framework. In Section 5, we present the experiments and results. Section 6 gives a brief review of the related work. Section 7 concludes the paper.

## 2. Models of the proposed system

We define our guide robot system to have the following basic functions.

(1) Visual function: The dog robot can visually observe the surrounding environment, identify obstacles, and recognize traffic signals and moving vehicles when crossing the road.
(2) Natural language processing function (understanding and speaking): The dog robot can listen to the voice in natural language from the visually impaired person, understand the person's intentions, and respond with the voice.
(3) Decision function: The robot can integrate various environmental information and the visually impaired person's intentions to make correct decisions. For example, if the visually impaired person commands the advancement, and while the robot sees that a vehicle is passing, the robot's decision is not to execute the forward command from the person. This is the intelligent disobedience that an intelligent guide dog should have.
(4) Motion function (optional): The robot has the ability to move autonomously. The visually impaired person with limited mobility can take a self-propelled wheelchair.

Based on these basic functions and requirements, we develop our solutions. Fig. 2 shows the hardware and software models of our PEN framework. The bottom layer of the hardware framework in Fig. 2a is the device layer, which contains hardware devices such as robots, sensors, and actuators. The middle layer is the fog layer, which consists of a smartphone, an embedded board and a neural compute stick. These three types of devices form a fog computing platform. One of the core functions of the smartphones is to provide Internet connectivity. It can also provide certain computing resources as a decision-making node for the guide system. In addition, the phone's sensors (such as GPS) can also be utilized. However, the smartphone cannot directly connect to some sensors and actuators commonly used in robots, such as ultrasonic sensors, servo motors, and so on. Therefore, the system requires an additional embedded board. The embedded board must have high computational performance and low power consumption, which is responsible for sensor information processing and actuator control. The guide system uses a deep neural network algorithm for object detection. The current algorithm is too slow to execute on the embedded CPU or smartphone. It consumes a lot of power. As a result, the system adds a neural network accelerator that speeds up deep network reasoning with lower power consumption.

Fig. 2 b shows the software framework of the PEN model, which is divided into two layers. The upper layer is the application

layer, and the lower is the service layer. The responsibility of the service layer is computing and control, including image processing, speech processing, motion control, obstacle detection, and so on. The responsibility of the application layer is to integrate various services for implementing the desired business logic.

A service component typically consists of four modules: event, procedure (event handler), communication, and external interface. The communication module provides basic network communication capabilities. When running of the service, the event module generates certain events (notifications) and pushes them to the subscriber through the communication module. For example, a button service can generate events such as press, release, and so on. The procedure module is for remote procedure call. For example, the servo service provides a function of turning to a specified angle. The external interface is use for reading or writing data from I/O devices or 3rd party robots. We design the appropriate external interface according to the type of device. One approach is to access the device through the underlying hardware interface library. For example, the display module outputs a string to the LED matrix via the SPI library. Another approach is through the SDK provided by the device manufacturer. For example, the vision module accesses the DNN (Deep Neural Network) interface engine through its SDK. The speech service receives messages by Amazon IOT SDK.

The event-driven and workflow model is more suitable for expressing the application logic of the guide dog system. Therefore, we develop decision nodes using a visual programming language. We can choose different visual programming tools, such as ASU VIPLE [11] in Windows and MIT App Inventor [12] in Android system. Visual programming makes it easier to build the workflow-based decision node, which processes events and call remote procedures from various services.

Motion guidance requires the system to have the ability to process information quickly. In order to achieve a high recognition rate, we use deep network, which causes huge computational cost. We then use the Intel neural compute stick for accelerating deep network, which also reduces the power consumption of the system. Our system has many functional modules. Visual programming tools can make system integration easier. Some data, such as voice data, need not be processed in real time, which can consume a lot of system resources. We use mobile phone network to transfer these data to the cloud for processing, which can save local system resources. Mobile phones have hardware resources such as GPS, inertial sensors, and many APIs such as map navigation can be used. Using these hardware and software resources can provide a larger expansion space for the system. In conclusion, the PEN framework combined with visual integrated environment can make our system easier to handle these components.

## 3. Implementation of proposed model

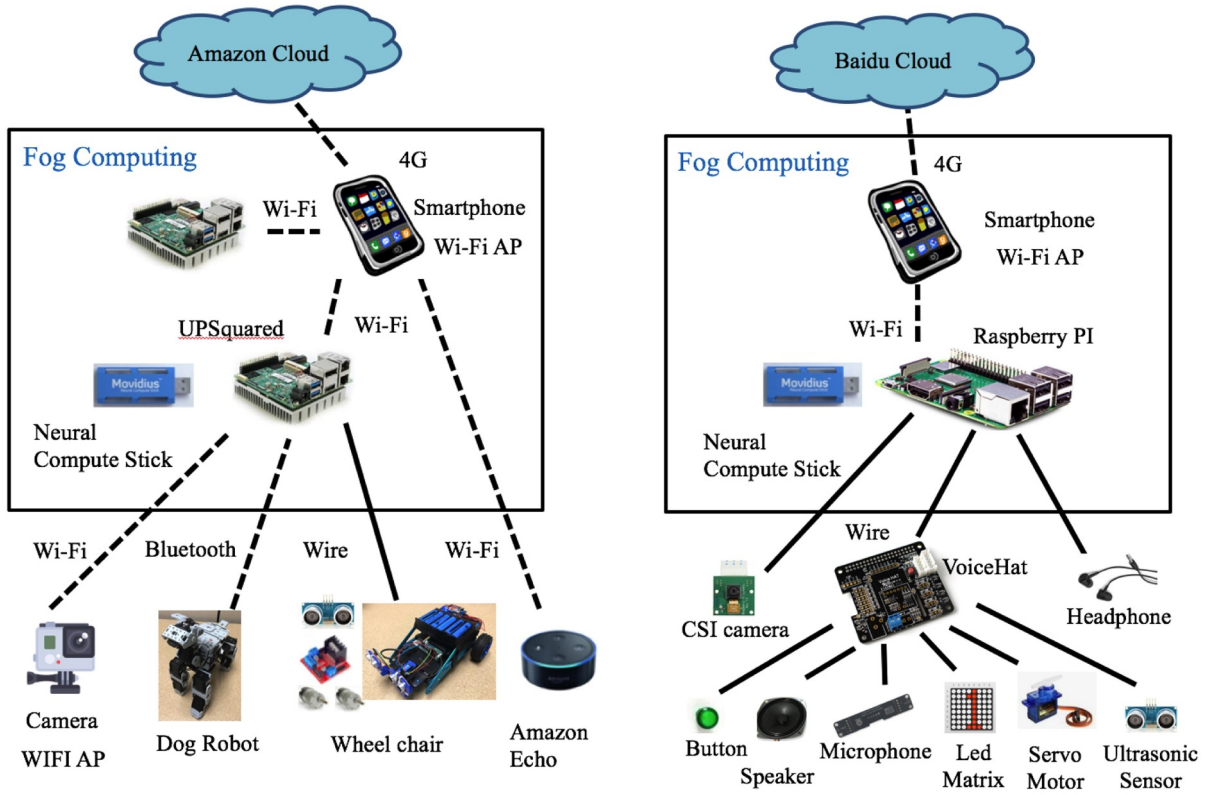
Based on the proposed model, this paper implemented a simulation environment and a prototype of guide system for visual impaired shown in Fig. 3. Fig. 3a shown a simulation robots including a small dog robot and a small wheelchair. The guide dog robot is followed by the wheelchair robot, which is for carrying a visually impaired person, assuming the person does not have adequate mobility. The robot in Fig. 3b is a portable device, which is a handheld device for the visually impaired person, assuming the person has adequate mobility. The prototype is implemented in the real environment.

### 3.1. Hardware architecture

As shown in Fig. 4, the overall hardware architectures of the two systems. Both consist of three layers. The middle level is the resource layer for fog computing. What they have in common is to use a smartphone for communication and a neural compute stick for enhanced image processing. The main difference of the two guide systems is the embedded board. The first guide system uses two Intel UPSquared boards, while the second system uses a Raspberry Pi. The lower level is the device layer consists of sensors and actuators. In Fig. 4a, the device layer consists of four relatively independent parts, including a camera, a dog robot, a wheelchair robot, and a voice device. The camera is mounted on the head of the robot dog. The wheelchair carries the voice device, and the wheelchair robot is made up of a variety of discrete components. The other three devices are off-the-shelf products. The wheelchair robot use UPSquared board as its main controller. The dog robot has its own controller. In Fig. 4b, in addition to the camera and headphones directly connected in the board's interface, the other sensors and actuators are connected to the Raspberry Pi via Google VoiceHat accessory. The upper layer is the cloud device from Amazon or Baidu. The Amazon Echo in the first system depends on the

(a) Guide Dog and Wheelchair Robot (b) Portable Guide Robot

**Fig. 3.** Guide systems for visually impaired.




(a) Simulation Robots based on UPSquared (b) Portable Robot based on Raspberry Pi

**Fig. 4.** Hardware structure of guide systems.

Amazon cloud services. The second system uses Baidu brain as its voice services.

### *3.2. Software architecture and algorithm*

According to the software model of PEN, the implementation details of the guide system are shown in Fig. 5. The software system consists of a decision center, local services and cloud services. Time-critical core tasks, including image processing and real-time control, are performed by local services. The local services are distributed among fog computing devices. Object detection based on deep neural network is performed on the neural compute stick, while other image processing methods are running on the embedded processor. The decision center is in charge of integrating all services. By combining different services, applications for different scenarios are implemented. In the simulation framework, we use ASU VIPLE as programming environment, which is running on Windows 10 OS. The prototype is an application running in a smartphone running Android OS, and built using MIT App Inventor.

#### *3.2.1. Vision*

The image capture module receives live stream from the camera. UPSquared board connects GoPro camera through H264 and UDP protocol. Raspberry Pi captures images directly through the camera serial interface from CSI camera. Then the images are dispatched to different image processing modules for detecting different kinds of objects. We use a deep neural network for complex object detection. In addition, we use recognition methods based on manual selection features to detect traffic light and obstacle cone. Finally, the fused vision information is provided as a vision service.

(1) *Object Detection.*In this paper, we use a single deep neural network named SSD (Single Shot MultiBox Detector) [13] to detect objects such as cars, cones, bottles, and so on. SSD is a fast single-shot object detector for multiple categories. A key feature of the neural network model is the use of multi-scale convolutional bounding box outputs attached to multiple feature maps at the top of the network. The model is trained on a power PC/GPU, and then exported to the neural compute stick. The image from the camera is inferred on the stick, which is faster than on the embedded CPU. Fig. 6a and b demonstrate the object detection.
(2) *Traffic Cone Detection.*First, the red regions are extracted by color segmentation in the same way as traffic light detection. Second, the edges are detected by canny algorithm and approximate into a few polygonal contours. Third, the contours that are convex hulls are selected. Finally, according to the shape of a traffic cone on the ground, a contour is recognized as a cone where the

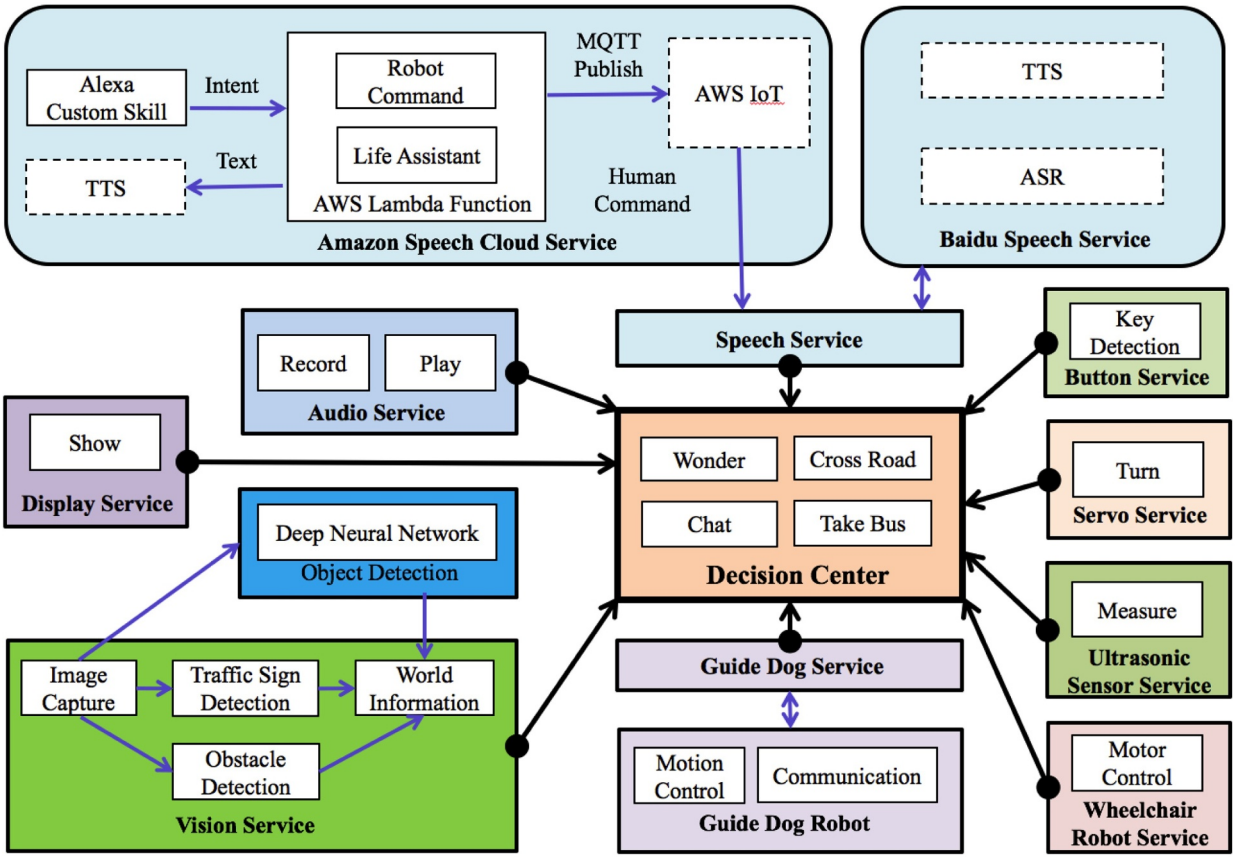
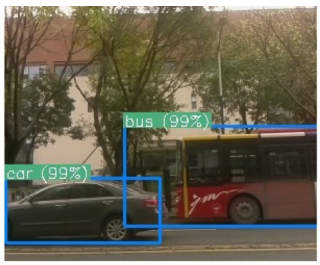
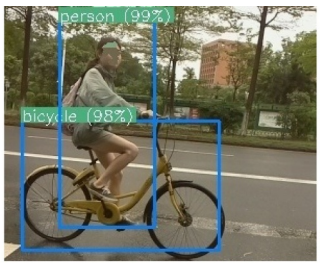


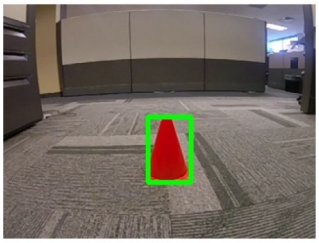

**Fig. 5.** Software framework of guide robot system.




(a) Car and Bus

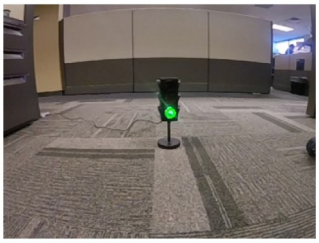



(b) Person and Bicycle



(c) Traffic Cone

(d) Red traffic light

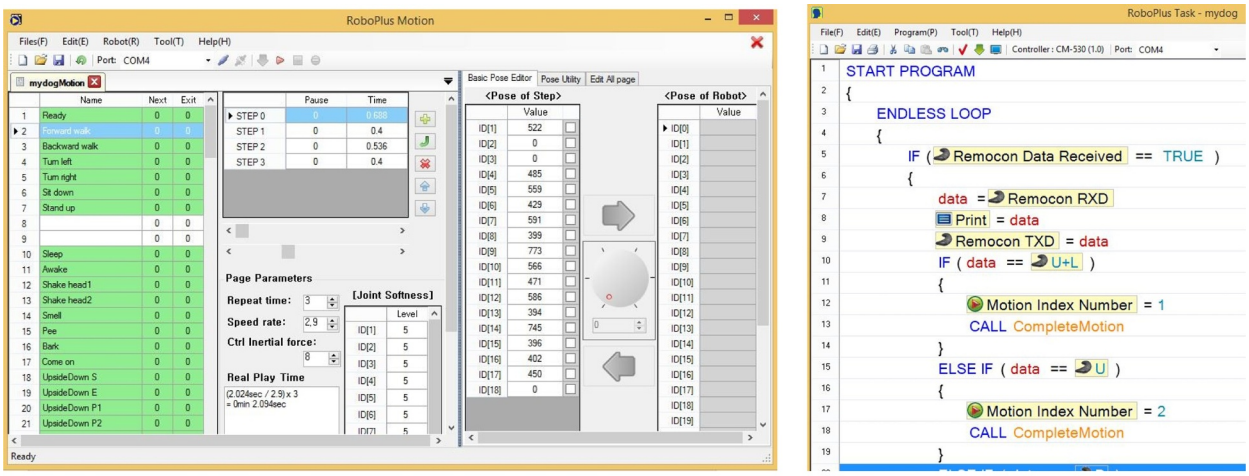


(e) Green traffic light

**Fig. 6.** Image processing demo.

upper points on the contour must be within the width of the lower points. A cone is detected in Fig. 6c.

(3) *Traffic Light detection.*The image is converted to HSI color space. After color segmentation with user defined threshold and smoothing by a median filter, Circle Hough Transform is used to detect circular object. The circle with appropriate radius would




(a) Motion Programming (b) Task Programming

**Fig. 7.** Dog robot programming.

be recognized as traffic light. Fig. 6d and e show the recognition of the red and green traffic light.

### 3.2.2. Speech and audio

The simulated framework uses Amazon Echo Dot as its voice device, which depends on Amazon speech cloud service. To use vocal commands, we created a custom skill set in the Alexa Skill Set for the Amazon Echo Dot. The invocation opens up the program that contains the robot commands, and the user may speak in a natural manner to the guide dog robot in order for it to perform the commands that the user wants it to execute. AWS Lambda, in which Node.js 6.10 is running, executes the commands defined in the Alexa Skill Set, and it sends them to the AWS IoT, where the other components of the robot are connected.

Unlike the simulated framework, the prototype uses the audio service to record the sound. The audio is saved as a wav file, and it is sent it to Baidu cloud for speech recognition. The result is presented as JSON format. Similarly, when the robot needs to talk, it sends the text to the Baidu cloud for speech synthesis. The returned sound file will be saved on the robot for later use. Other speech services can also be used instead of Baidu cloud, such as Google speech cloud service.

### 3.2.3. Dog robot

The program running on CM530 controller of the dog includes motion code and task code. The dogs actions are edited and saved through RoboPlus Motion, as shown in Fig. 7a. The saved motion file can be executed by Task Code, which is a set of motions to perform certain actions. The robot moves according to the Task Codes. RoboPlus Task shown in Fig. 7b is a piece of program to make writing these task codes easier.

### 3.2.4. Wheelchair robot

The real-time control program running on UPSquared board has three threads. The main thread is in charge of network connection. Sensor thread reads ultrasonic sensor values, and then it sends them to the decision center. Motion thread receives command from decision center, and then it controls two motors to move the robot. All data and commands are formatted as in JSON format. The interface to IO on UPSquared board is based on Intel mraa library.

### 3.2.5. Ultrasonic sensor

The ultrasonic sensor is to obtain the distance of obstacles. Firstly, the service produces a pulse of at least 10 microseconds to the trigger pin. Secondly, it monitors the pulse of the echo pin. Finally, the width of the received pulse is used to calculate the distance to the reflected object.

### 3.2.6. Servo motor

The servo motor service is used to control a servo motor. The service offers two modes of motion. One is the fixed point mode, which is rotated to the specified angle. The other one is to swing back and forth directly at two angles.

### 3.2.7. Button

In order to facilitate the operation of the device by a visually impaired person, we used one button only. In this case, a variety of key event detections are provided as much as possible to meet a variety of application needs. A state machine is designed to detect several events such as press, hold, release, and double click.

#### 3.2.8. Display

The display service shows a string or an icon on the dot LED matrix. The implementation of the service is built on the SPI protocol.

#### 3.2.9. Decision center

The decision center is the brain of the system. The center collects information from vision, speech and other sensors, and it sends commands to robots or actuators. Visual programming tool makes the integration easier.

In the simulated environment, we build the application by ASU VIPLE [11]. It is used for integrating several modules of our system together.

Fig. 8 demonstrates VIPLE code that implements the function of crossing a road and avoiding cars. Robot/IoT controller in Fig. 8a is the proxy to connect to other modules. There are four parallel IoT/Robot controllers to connect to four types of services (vision, voice, dog and wheelchair). Fig. 8b shows the process of information update. The messages from the vision service are parsed then used for updating the status variable. The complex processing functions, such as JSON_To_Object, are wrapped as VIPLE Code Activity written in C#. Similarly, the messages from wheelchair update the distance variables. In Fig. 8c, the voice command from speech service will trigger the responding motion if there is no car or truck in front of the dog robot. The four custom activities generate the four motion commands for the dog robot and wheelchair robot. Otherwise, the robots will not move and give a voice prompt to the person one the wheelchair.

In the second prototype, the decision center is deployed on an Android smartphone. For better mobility, we use the MIT APP Inventor2[12] to develop applications instead of using VIPLE. In order to call the services on Raspberry Pi, seven corresponding App Inventor extensions are developed. Each extension defines certain events and procedures. This approach makes it easy for us to build applications of guide system by combined various services.

The program in Fig. 9 helps the visual impaired to ride a bus. The program uses event handler blocks in App Inventor 2 to specify how the program should respond to certain events. When the main screen is initializing, the application tries to connect the seven services together, which own different ports. When the visually impaired person holds the button, the audio service starts to record the voice. When the button is released, the audio service stops recording and start speech service to recognize the voice. When the speech service returns the recognition result, a state machine is designed to handle different voice commands. The user can start or stop the detection services, or set the route number of bus on the LED matrix. When the vision service detects a bus object with a great confidence, the ultrasonic sensor service starts to measure the distance. If the distance is small, the audio service will play a voice prompt.

## 4. Simulation and physical experiments

Based on the two implementations, the simulated environment and the prototype, we conducted various experiments. We will show three applications here that verify the functionalities of the system. In addition, three experiments are designed and conducted to test time performance.

### 4.1. Applications

#### 4.1.1. Avoid obstacle in the simulated environment

The guide dog robot and the wheelchair can move around the obstacle, when the vision module recognizes the obstacle in front. The motion sequence of the robots is shown in Fig. 10.

#### 4.1.2. Cross road in the simulated environment

The visually impaired person says “go forward” to cross the road. If the dog robot detects that there is a vehicle in front, the dog prompts the user by voice and wait, as shown in Fig. 11. Similarly, the dog will not follow the user’s instructions when it sees a red traffic light, as shown in Fig. 12.

#### 4.1.3. Bus ride experiment in the physical environment

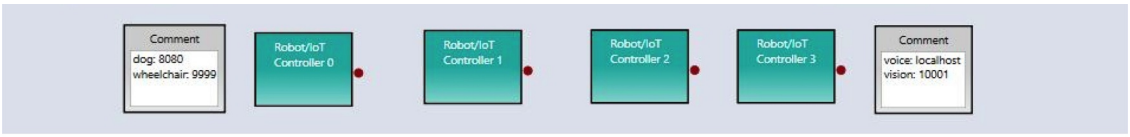
In Fig. 13, the visually impaired person is waiting for a bus. The person wears a portable robot and wants to ride bus with route number 201. The person can press the button to active voice function, and say a commands to the robot. The robot then performs certain actions based on the command. When a bus is detected, the robot will play voice prompt. Fig. 14 shows a detection sequence of a bus. The visually impaired person states the bus route number on the LED matrix (Fig. 13b) through voice interaction. This display is not for the visually impaired person, but for others. When a visually impaired person is about to get on the wrong bus, the other people or the bus driver can remind him/her.

### 4.2. Time performance and resource usage

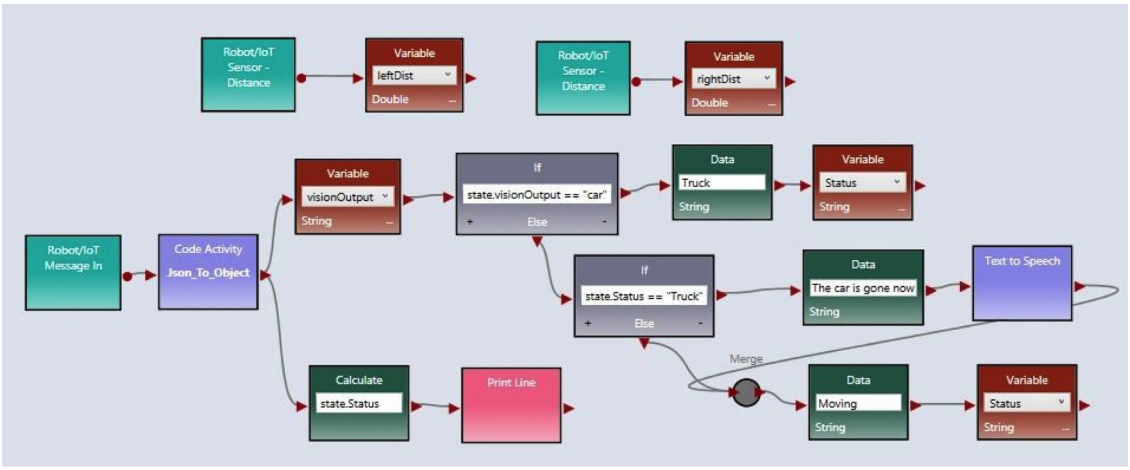
Two simulation experiments are performed on the simulated environment, which uses UPSquared board and GoPro camera. The physical experiment uses Raspberry Pi 3B and CSI camera in the prototype.

#### 4.2.1. UPSquared board and GoPro camera

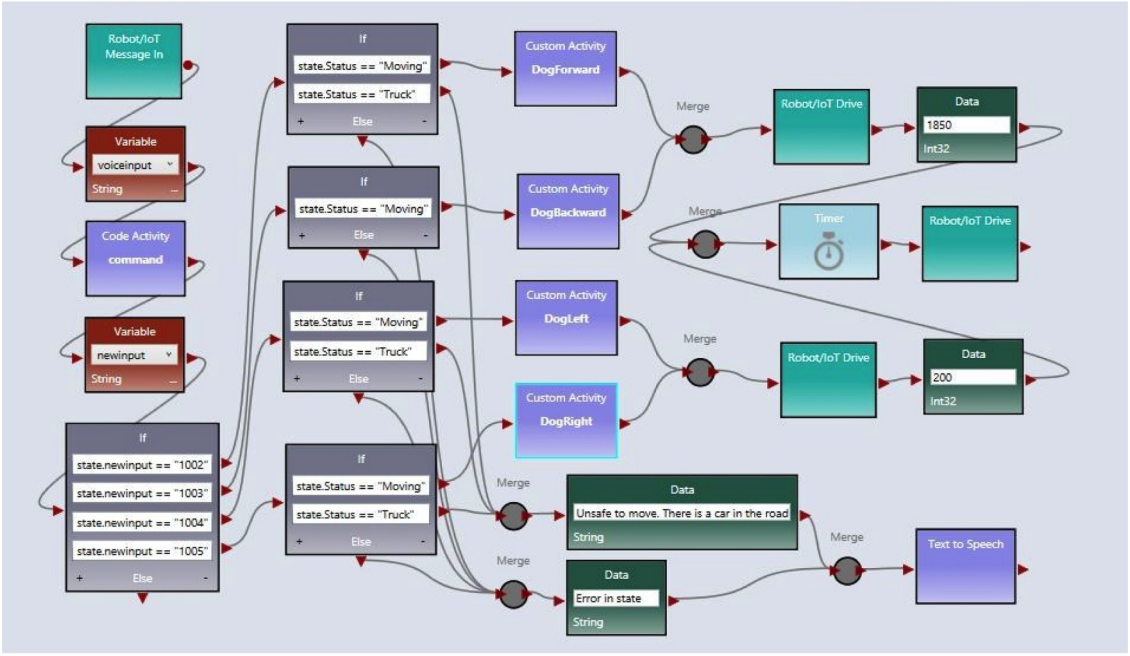
The first experiment was to compare the time performance of the system processing images with different hardware




(a) Service Connection




(b) Information Update




(c) Make Decision and Motion

**Fig. 8.** VIPLE sample code for cross road.

configurations. On the one hand, we compare the UPSquared board with a laptop to reflect the performance of the embedded board. On the other hand, we compare the DNN inference performance of CPU and VPU (Vision Process Unit).

Intel UPSquared board in the project has Intel(R) Celeron(R) CPU N3350 @ 1.10GHz, 2GB LPDDR4, 32GB eMMC, a 40-pin GP-bus and 3 USB 3.0 ports. The laptop is Lenovo ThinkPad x220, which has Intel(R) Core(TM) i5-2410M CPU @ 2.30GHz and 4GB memory.

The system uses GoPro HERO4 Session camera, which has built-in battery and Wi-Fi. The video stream from the camera is 25 fps

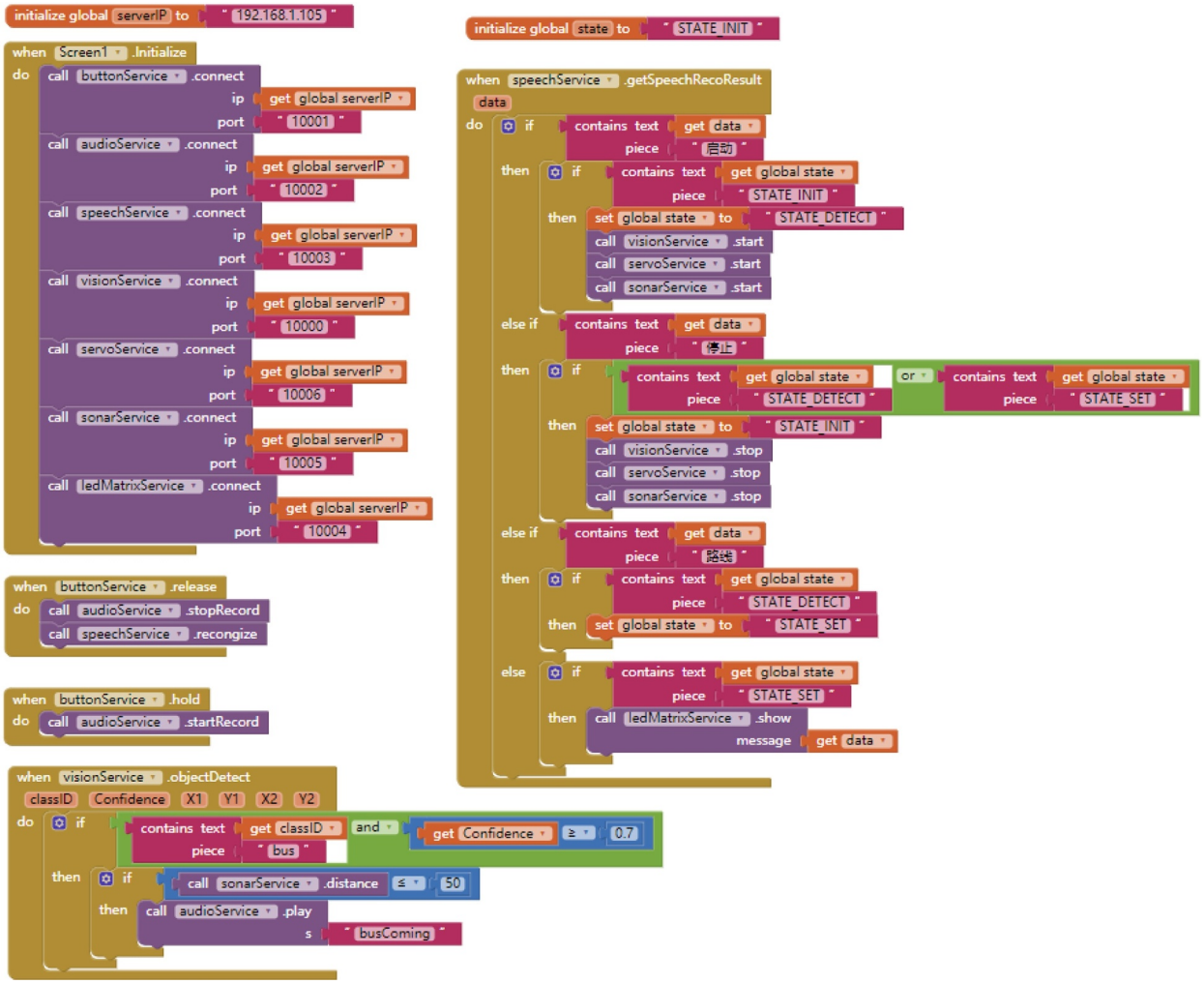



**Fig. 9.** App inventor sample code for bus ride.

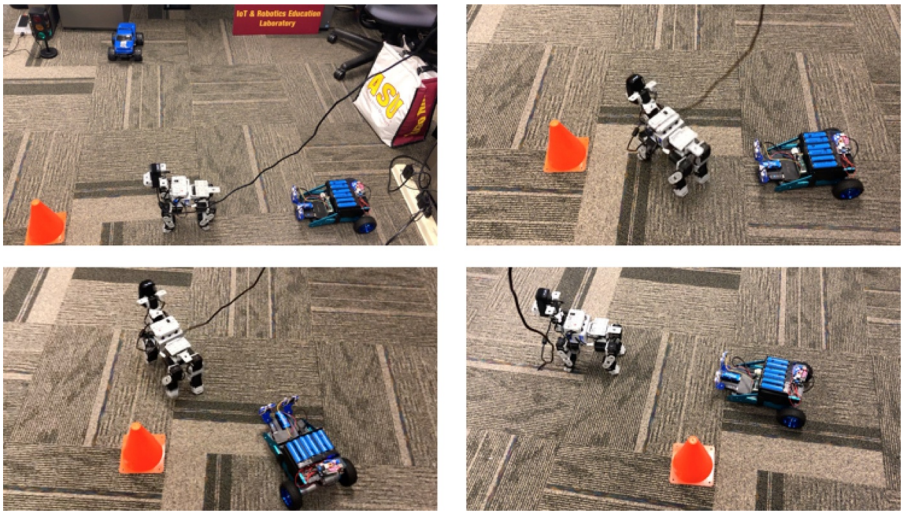


**Fig. 10.** Avoid obstacle.

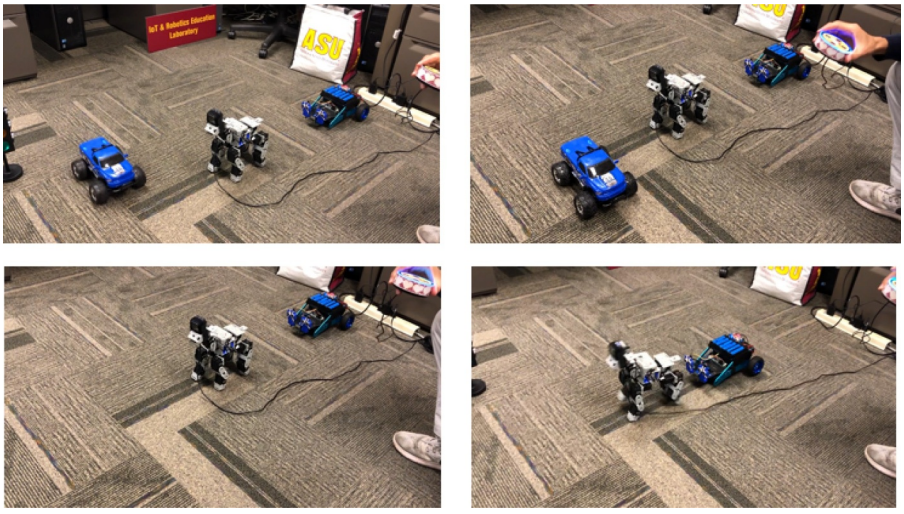


**Fig. 11.** Cross road meeting a car.

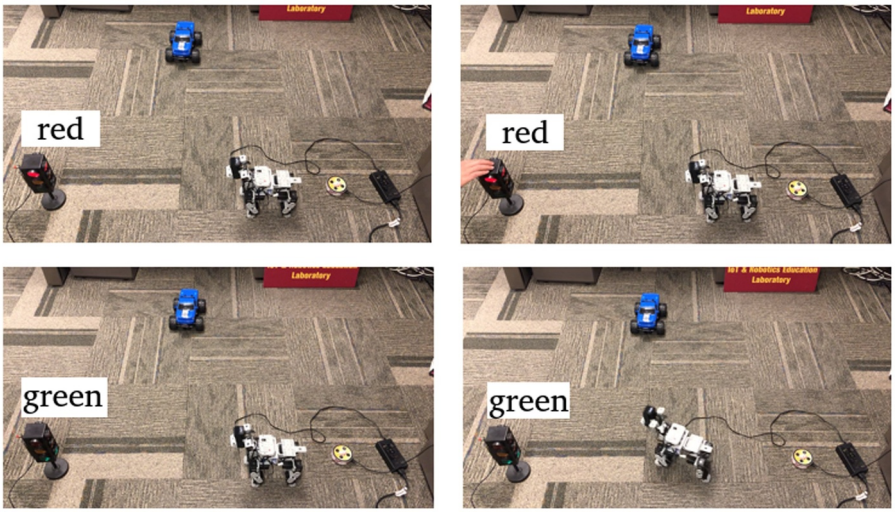
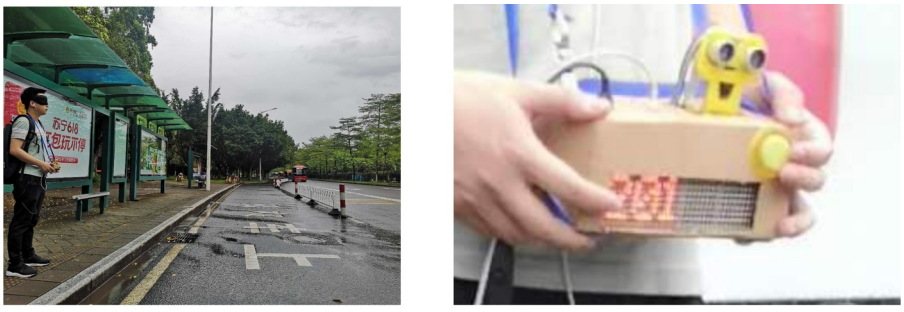



**Fig. 12.** Cross road meeting red traffic light.



(a) Waiting for a bus (b) Show Bus Route Number 201

**Fig. 13.** Application of bus ride.

(frames per second). In this experiment, the sample rate is set as 1/25, which means that the vision module only process 1/25 frames. The measured data is shown in Table 1. The time in the table is the time of image processing, including the time to access the neural compute stick. It does not include the time to read the image and write the log.


**Fig. 14.** Bus detection.

**Table 1**
Performance and usage of processing images from GoPro camera.

| Case | Computer | DNN inference device | Time (ms) | CPU (%) | Mem (%) |
|---|---|---|---|---|---|
| 1 | UPSquared | VPU | 120.0 | 13.6 | 8.6 |
| 2 | Laptop | VPU | 141.1 | 15.5 | 4.4 |
| 3 | Laptop | CPU | 371.7 | 44.7 | 9.2 |
| 4 | UPSquared | CPU | 894.5 | 99.2 | 18.6 |

Fig. 15 a and b show that using VPU is faster than using CPU, and CPU time consumption is lower. When using VPU for deep network inference, the laptop is slower than the embedded Up Square board. The reason is the USB port speed. The USB port of laptop USB 2.0, whereas the embedded board is USB 3.0. When the image is processed on CPU without VPU, the speed is slower and the CPU occupancy rate is higher. As shown in Fig. 15c, the usage rate is stable in all cases, and the use of VPU consumes less memory.

The second experiment is to test the effect of sampling rate parameter on time performance. This experiment is tested on embedded board only with neural compute stick. The measured data is shown in Table 2. The average processing time for each frame is about 120ms, thus the fastest processing speed is about 8fps and the corresponding sample rate should be 1/3. Figs. 16 and 17 show the performance and usage with sample rates 1/3, 1/10, 1/20 and 1/25. When the system runs at the highest speed, the occupancy rates of CPU and memory are acceptable.

#### *4.2.2. Raspberry Pi and CSI camera in the physical environment*

The third experiment was performed on the prototype based on Raspberry Pi3 and CSI camera. Similar to the first experiment, we compares the time performance with different hardware configurations. If the images from the CSI camera are transmitted to the laptop via the network, the result is the same as the first experiment. Therefore, this experiment only performed a combined test of Raspberry Pi and the neural compute stick. The Raspberry Pi 3B has a Quad Core 1.2GHz Broadcom BCM2837 64bit CPU, 1GB RAM, a 40-pin GP-bus and 4 usb2.0 port. The test results are shown in Table 3 and Fig. 18. Fig. 18b shows that using VPU is faster than using CPU in object detection task. Fig. 18c and d show that using VPU will greatly reduce the CPU and memory occupancy rate. Comparing with the above two experiments, the experiment can draw the same conclusion that using VPU is better than using CPU only.

In order to observe the efficiency of processing images locally and remotely, we did the fourth experiment. Our experimental data are 100 sRGB images of 300 * 300 pixels taken by raspberry cameras at bus stops. Our cloud server has Intel(R) Core(TM) i7-7800XCPU @3.50GHz and 16GB memory. We use WIFI network with the maximum assess speed of 1.03MB/s to transmit data over the LAN. We deployed a web service to process images on the cloud service, which processes the images uploaded from Raspberry Pi and returns the processing results to Raspberry Pi. In the experiment, we have three time quantities, Tp, Tn and T. Tp is the time spent actually processing images, Tn is the network overhead, and T is the total time for processing images. We take the average values of Tp, Tn and T, which are $\overline{Tp}$, $\overline{Tn}$ and $\overline{T}$. We also used the Raspberry Pi with neural computing stick to process these images locally, and then recorded the processing time.

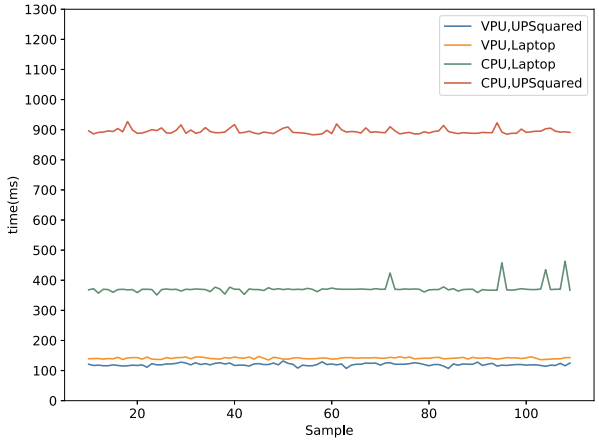
As the Table 4 shows, it takes 137.01 milliseconds to process a picture on the Raspberry Pi. The average time to process uploaded pictures by the server is 429.22 milliseconds, the average time consumed by the network is 912.99 milliseconds,and the average time to process one picture is 1342.21 milliseconds.

There are many other background service programs running on the server, the server can not use all CPUs for image processing and there's a slight fluctuation in Tp. From the Fig. 19, we can observe that the status of the network is uncertain. We can conclude that image processing in fog method is better than that of in a remote cloud server for our system.

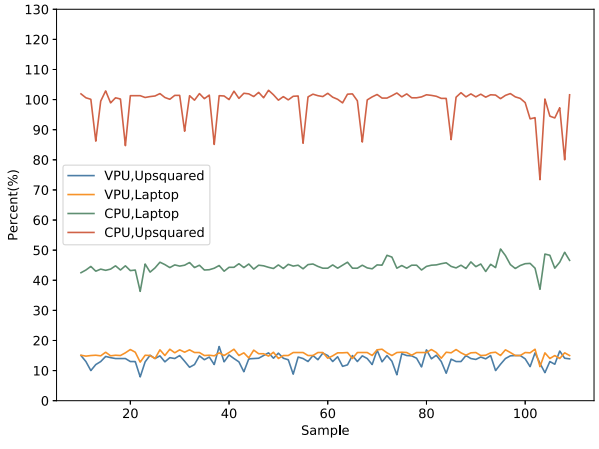
## 5. Related work

The earliest research on guide dog robot was the MELDOG project in 1985. The study in [14] developed a guide dog robot to lead and recognize a visually-handicapped person using a laser range finder sensor. The research in [15] built an outdoor navigation system using GPS and tactile-foot feedback, which were worn on a blind pedestrian.

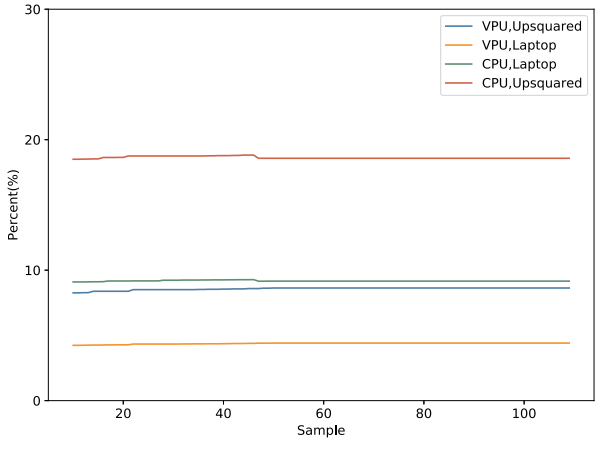
More guide dog projects [1,2,6,16–18] used cameras to perceive the world. The study in [19] developed a blind interaction handle using pressure sensors combined with Arduino. Users could control and communicate with the robot through natural tactile




(a) Time




(b) CPU usage



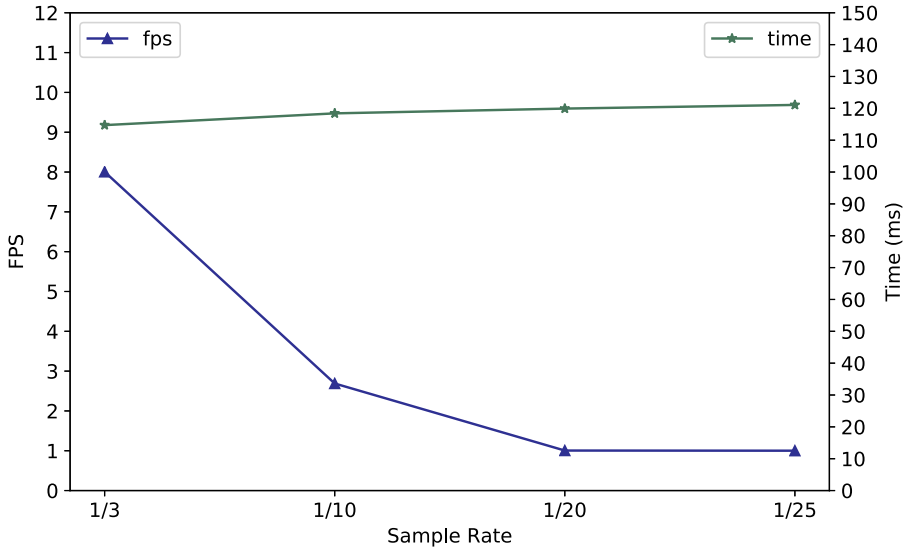

(c) Memory usage

**Fig. 15.** Performance and usage of processing images from GoPro camera.

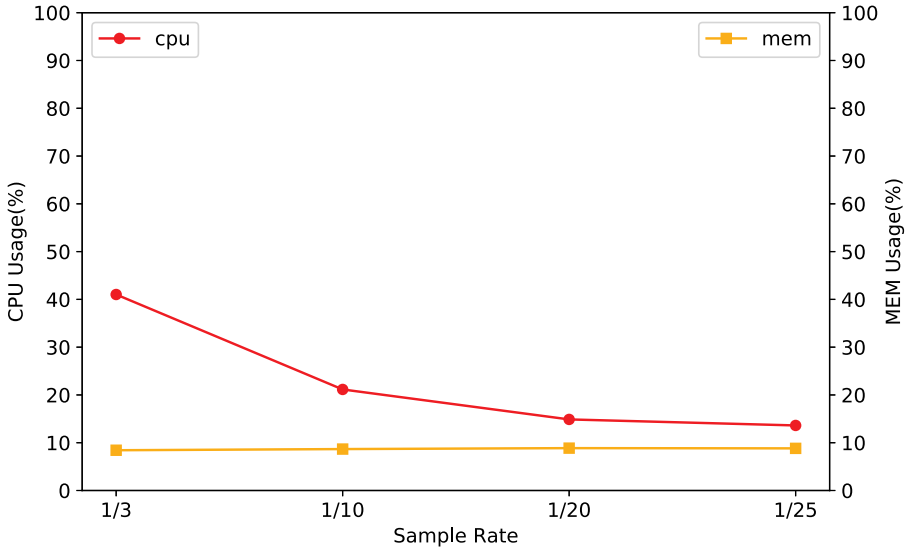
interaction. [20] provided a framework for context-aware navigation services for vision impaired people. This system was a human-in-the-loop cyber-physical system that interpreted ubiquitous semantic entities by interacting with the physical world and the cyber domain. The smartphone streams video to the laptop on the robot to process [1,16]. In contrast, the images from smartphone are sent to cloud service to process [17]. In [2], the system used an embedded system with two cameras, among other sensors. However, image processing is on a remote server, which took a longer time than an edge computing server. The study in [18] used GPU to

**Table 2**
Performance and usage with different sample rate.

| Case | Sample rate | FPS | Time (ms) | CPU (%) | Mem (%) |
|---|---|---|---|---|---|
| 1 | 1/3 | 8.0 | 114.8 | 41.2 | 8.7 |
| 2 | 1/10 | 2.7 | 119.6 | 21.4 | 8.7 |
| 3 | 1/20 | 1.4 | 120.2 | 18.3 | 8.8 |
| 4 | 1/25 | 1.3 | 120.3 | 15.5 | 8.8 |




(a) Time Performance




(b) Usage

**Fig. 16.** Performance and usage with different sample rates.

handles the video processing, mapping, and localization.

A convolutional neural network was trained to recognize the activities taking place around the camera mounted on the real guide dog and to provide feedback to the visual impaired human user [6]. However, deep models are often with a huge number of parameters and the model size is very large. As a result, deep neural network is difficult to be applied on hardware devices with limited storage and computation resources. To address this problem, model compression is an effective approach, such as channel

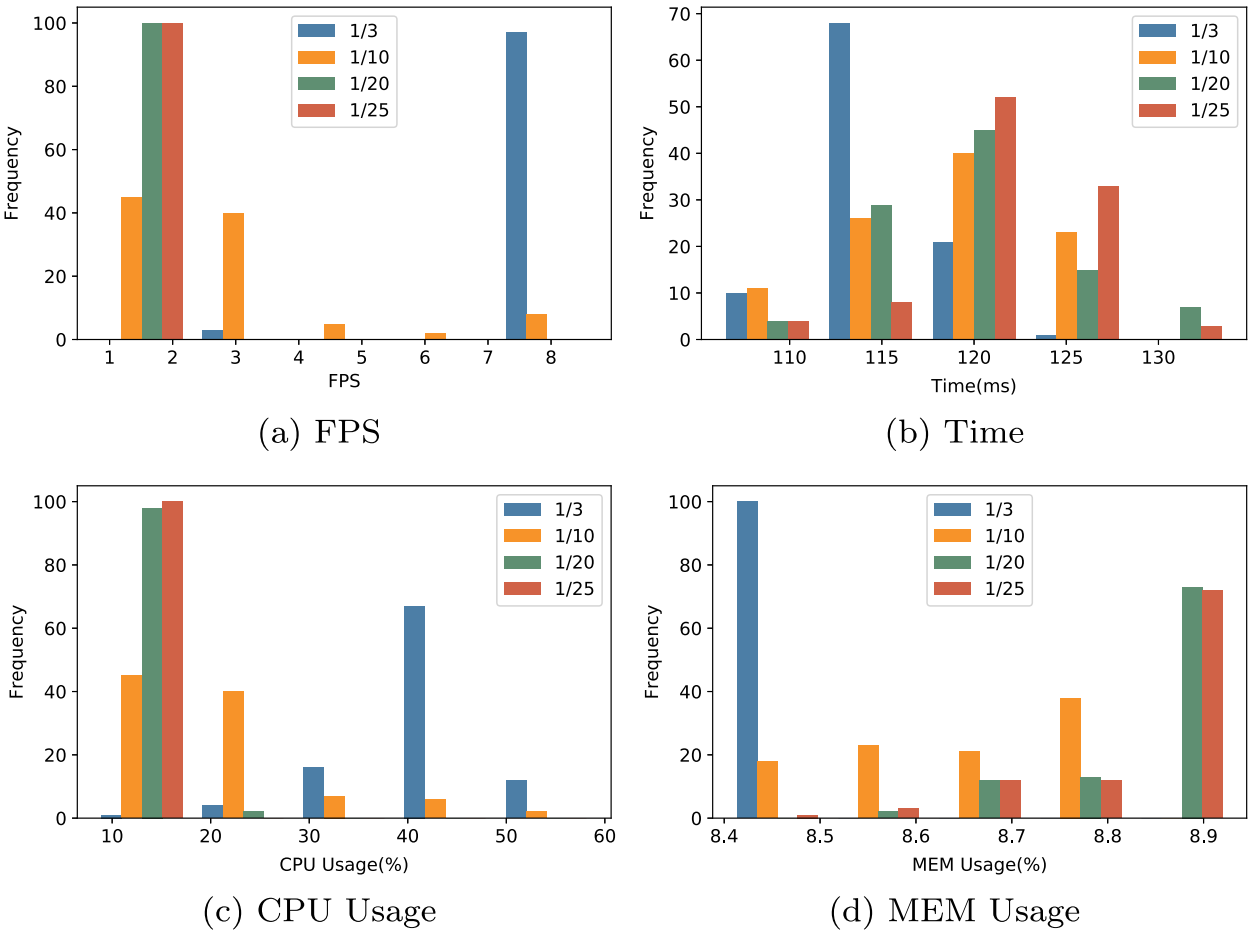



**Fig. 17.** Histogram of performance and usage with different sample rates.

**Table 3**
Performance and usage of processing images from CSI camera.

| Case | DNN inference device | FPS | Time (ms) | CPU (%) | Mem (%) |
|---|---|---|---|---|---|
| 1 | VPU | 3.75 | 138.58 | 37.06 | 12.31 |
| 2 | CPU | 0.15 | 6600.24 | 98.85 | 45.59 |

pruning [21]. Even so, the application of the compression model on embedded devices is not quickly enough. In addition, neural network reasoning using CPU consumes a lot of power. The emergence of neural network accelerators provides another possible solution. In order to perform real-time object detection on Raspberry Pi, the study in [22] used an Intel neural computing stick to run SSD model [13]. The first generation of our guide dog systems [7] also used a combination of an embedded board and a neural compute stick.

Fog computing and related edge computing, have emerged as a system model for reducing network congestion and latency of edge devices and applications, offloading some of the data originally located in the cloud to edge computers, and thus improving real-time performance. The combination of every resource (Cloud, Clusters, low cost devices and Smartphones) available can result in a significant decrease in the execution time and expenses [23]. The studies in [24] proposed a hybrid approach for scheduling real-time IoT workflows in fog and cloud environments in a three-tiered architecture, which attempted to schedule computationally demanding tasks with low communication requirements in the cloud and communication intensive tasks with low computational demands in the fog.

Currently, there is no common paradigm for fog computing. However, this idea has led to many domain-specific computing architectures. Nasir et al. [4] proposed a framework for distributed summarization of surveillance videos over the fog network. The network itself was composed of multiple regions combined as clusters of Raspberry Pi. Hsu et al. [5] presented a creative IoT agriculture platform for cloud fog computing. The use of multiple Raspberry Pi devices created a simulated decentralized environment. The platform allowed farmland with limited network information resources to be integrated and automated, including

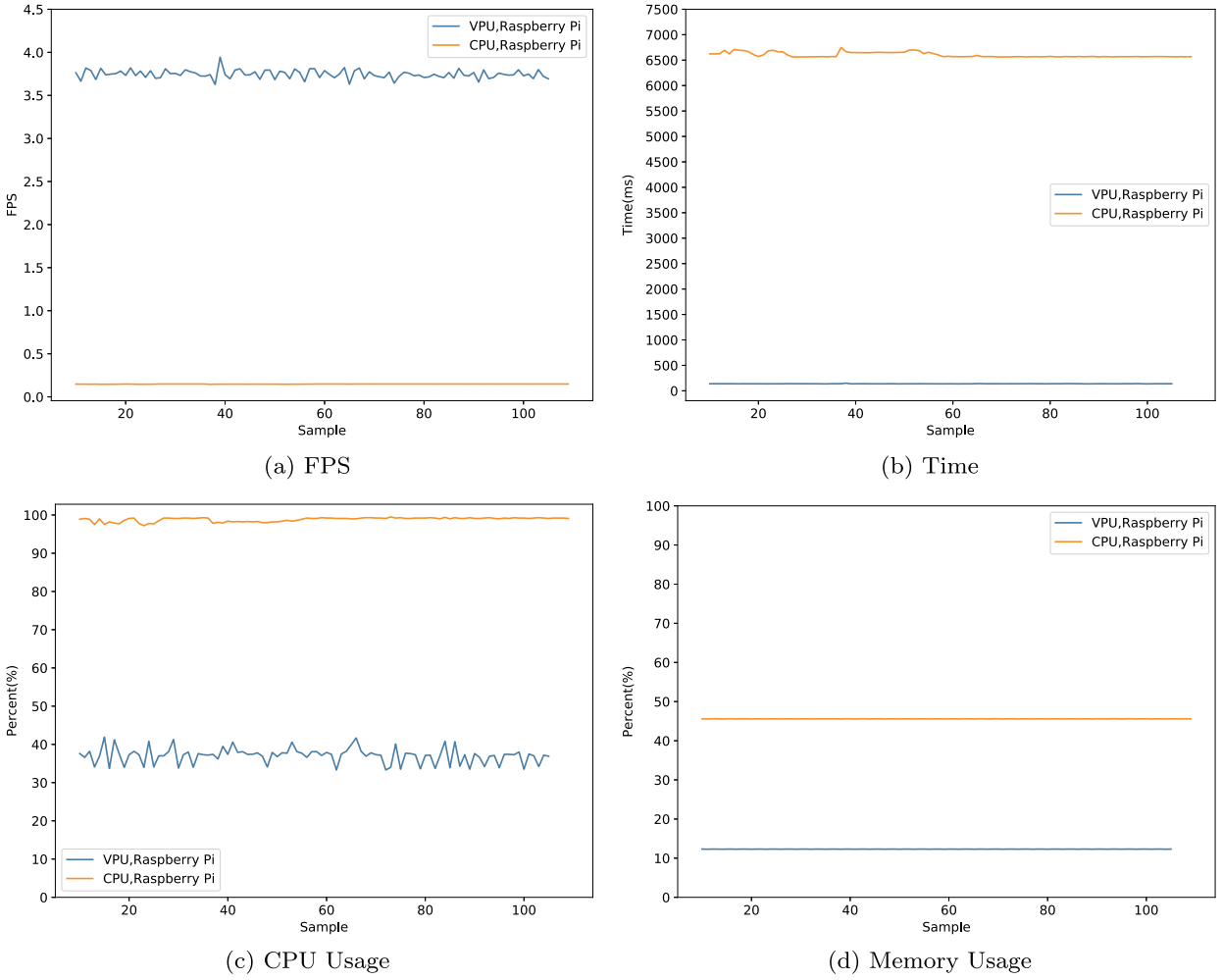



**Fig. 18.** Performance and usage of processing images from CSI camera.

**Table 4**
Time of image processing in edge and in server.

| Computing mode | $\overline{Tp}$ (ms) | $\overline{Tn}$ (ms) | $\overline{T}$ (ms) |
|---|---|---|---|
| Fog | 137.01 | 0 | 137.01 |
| Cloud | 429.22 | 912.99 | 1342.21 |

agricultural monitoring automation, pest management image analysis and monitoring. In this paper, we present a novel fog compute model PEN of guide dog robot for visually impaired. We innovatively combine smartphones, embedded boards and neural network accelerators to build the simulated and physical platforms.

Combining the Service Oriented Architecture (SOA) [8] with the fog computing architecture is still an open challenge. Alturki et al. [25] decomposes services to create linked-microservices which run on multiple nodes but closely linked to their linked-partners. Shi et al. [26] uses Internet of Things protocol CoAP to link mobile device clouds with fog computing. Smart devices, sensors and resources are exposed to the mobile cloud in a unified way as RESTful services. Alturki et al. [27] developed an open source visual programming platform Distributed Node-RED (DNR) to build fog computing applications, which could be decomposed and deployed to a geographically distributed infrastructure. The model of this paper uses service-oriented architecture [8] and Robot as a Service (RaaS) concepts [9,10]. Both software and hardware modules communicate with each other through service interface. The orchestration among the services is implemented using a visual programming language environment [8].

## 6. Conclusion

In this paper, we presented a fog computing model PEN and developed a simulated guide robot environment and a physical

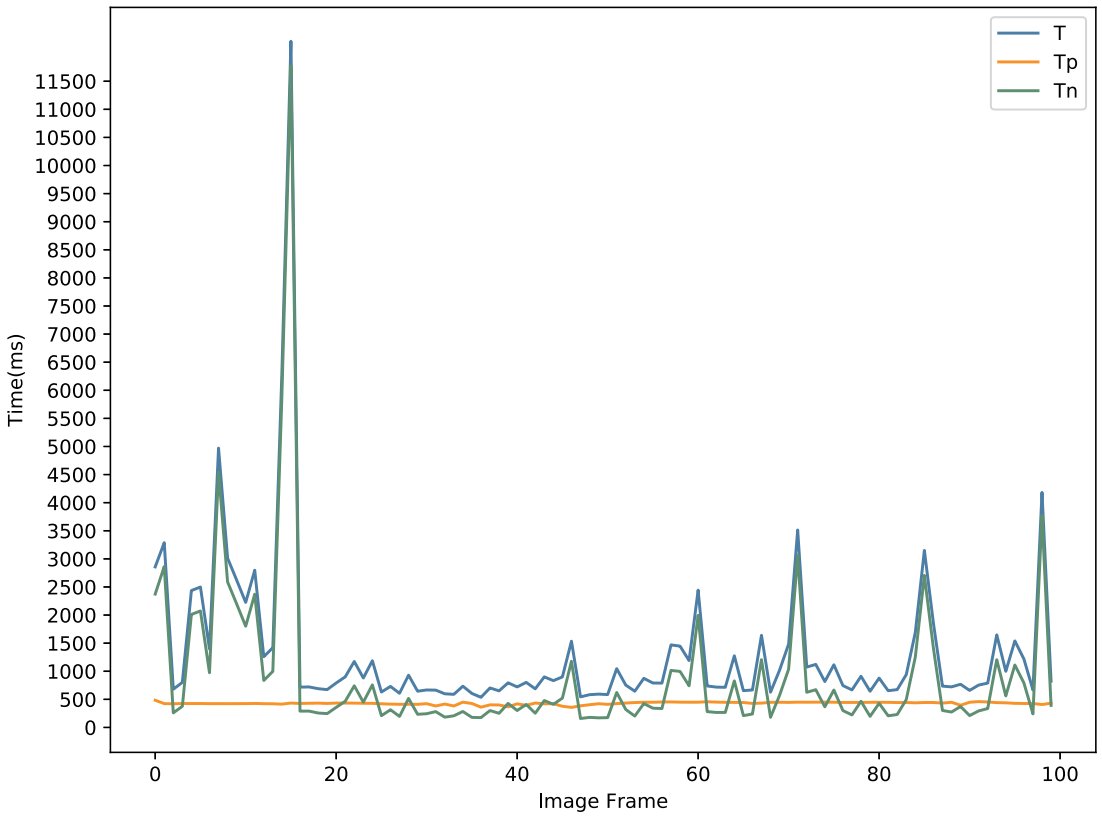



**Fig. 19.** Time of image processing by cloud server.

prototype for visually impaired persons. Phone, embedded board and neural compute stick complement cooperate with each other to form an excellent fog computing platform that met the needs of video and audio processing with real-time control. A visual programming environment effectively integrated multiple decentralized services and devices. In our future work, more resources on smartphone will be considered and utilized, and the algorithms for image and vision processing needs to be further improved.

## Acknowledgment

This work is partly supported by the Science and Technology Planning Project of Guangdong Province, China, China (Grant nos. 2018B010108002, 2017B090910005, 2017A030310163, 2017A010101031), Science and Technology Planning Project of Guangzhou, China (Grant no. 201707010437).

## References

[1] Y. Wei, M. Lee, A guide-dog robot system research for the visually impaired, 2014 IEEE International Conference on Industrial Technology (ICIT), IEEE, 2014, pp. 800–805.
[2] W.M. Elmannai, K.M. Elleithy, A highly accurate and reliable data fusion framework for guiding the visually impaired, IEEE Access 6 (2018) 33029–33054.
[3] L.M. Vaquero, L. Rodero-Merino, Finding your way in the fog: towards a comprehensive definition of fog computing, ACM SIGCOMM Comput. Commun. Rev. 44 (5) (2014) 27–32.
[4] M. Nasir, K. Muhammad, J. Lloret, A.K. Sangaiah, M. Sajjad, Fog computing enabled cost-effective distributed summarization of surveillance videos for smart cities, J. Parallel Distrib. Comput. 126 (2019) 161–170.
[5] T.-C. Hsu, H. Yang, Y.-C. Chung, C.-H. Hsu, A creative IoT agriculture platform for cloud fog computing, Sustain. Comput. (2018).
[6] J. Monteiro, J.P. Aires, R. Granada, R.C. Barros, F. Meneguzzi, Virtual guide dog: An application to support visually-impaired people through deep convolutional neural networks, 2017 International Joint Conference on Neural Networks (IJCNN), IEEE, 2017, pp. 2267–2274.
[7] J. Zhu, Y. Chen, M. Zhang, Q. Chen, Y. Guo, H. Min, Z. Chen, An edge computing platform of guide-dog robot for visually impaired, IEEE International Symposium on Autonomous Decentralized Systems, IEEE, 2019, pp. 23–29.
[8] Y. Chen, Service-Oriented Computing and System Integration: Software, IoT, Big Data, and AI as Services, Kendall Hunt Publishing, 2018.
[9] Y. Chen, H. Hu, Internet of intelligent things and robot as a service, Simul. Model. Pract. Theory 34 (2013) 159–171.
[10] Y. Chen, Z. Du, M. Garcia-Acosta, Robot as a service in cloud computing, 2010 Fifth IEEE International Symposium on Service Oriented System Engineering (2010) 151–158.
[11] ASU VIPLE, (Available at: http://neptune.fulton.ad.asu.edu/VIPLE).
[12] MIT APP Inventor, (Available at: https://appinventor.mit.edu).
[13] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A.C. Berg, Ssd: Single shot multibox detector, European conference on computer vision, Springer, 2016, pp. 21–37.
[14] S. Saegusa, Y. Yasuda, Y. Uratani, E. Tanaka, T. Makino, J.-Y. Chang, Development of a guide-dog robot: leading and recognizing a visually-handicapped person

using a lrf, J. Adv. Mech. Des. Syst. Manuf. 4 (2010) 194–205.
[15] R. Velázquez, E. Pissaloux, P. Rodrigo, M. Carrasco, N. Giannoccaro, A. Lay-Ekuakille, An outdoor navigation system for blind pedestrians using GPS and tactile-foot feedback, Appl. Sci. 8 (4) (2018) 578.
[16] Y. Wei, X. Kou, M. Lee, Development of a guide-dog robot system for the visually impaired by using fuzzy logic based human-robot interaction approach, 2013 13th International Conference on Control, Automation and Systems (ICCAS 2013), IEEE, 2013, pp. 136–141.
[17] J. Jakob, J. Tick, Concept for transfer of driver assistance algorithms for blind and visually impaired people, 2017 IEEE 15th International Symposium on Applied Machine Intelligence and Informatics (SAMI), IEEE, 2017, pp. 000241–000246.
[18] G. Chaudhari, A. Deshpande, Robotic assistant for visually impaired using sensor fusion, 2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), IEEE, 2017, pp. 1–4.
[19] L. Zeng, B. Einert, A. Pitkin, G. Weber, Hapticrein: design and development of an interactive haptic rein for a guidance robot, International Conference on Computers Helping People with Special Needs, Springer, 2018, pp. 94–101.
[20] J. Xiao, S.L. Joseph, X. Zhang, B. Li, X. Li, J. Zhang, An assistive navigation framework for the visually impaired, IEEE Trans. Human-Mach. Syst. 45 (5) (2015) 635–640.
[21] Z. Zhuang, M. Tan, B. Zhuang, J. Liu, Y. Guo, Q. Wu, J. Huang, J. Zhu, Discrimination-aware channel pruning for deep neural networks, Proceedings of the 32Nd International Conference on Neural Information Processing Systems, NeurIPS'18, Curran Associates Inc., 2018, pp. 883–894.
[22] N.A. Othman, I. AYDIN, A new deep learning application based on movidius NCSfor embedded object detection and recognition, 2018 2nd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), IEEE, 2018, pp. 1–5.
[23] D. Tychalas, H. Karatza, Simulation and performance evaluation of a fog system, Third International Conference on Fog and Mobile Edge Computing, IEEE, 2018, pp. 26–33.
[24] G. Stavrinides, H. Karatza, A hybrid approach to scheduling real-time IoTworkflows in fog and cloud environments, Multimed. Tools Appl. 78 (17) (2019) 24639–24655.
[25] B. Alturki, S. Reiff-Marganiec, C. Perera, S. De, Exploring the effectiveness of service decomposition in fog computing architecture for the internet of things, IEEE Trans. Sustain. Comput. (2019).
[26] H. Shi, N. Chen, R. Deters, Combining mobile and fog computing: using coap to link mobile device clouds with fog computing, 2015 IEEE International Conference on Data Science and Data Intensive Systems, IEEE, 2015, pp. 564–571.
[27] N.K. Giang, R. Lea, V.C. Leung, Developing applications in large scale, dynamic fog computing: a case study, Software (2019).