Smart bus system for estimating waiting time and tracking special passenger tracking

***Abstract: As one of the main public transportation modes in modern society, buses are distributed everywhere. However, there are also many problems around bus. Firstly, not knowing the specific position of buses, riders always waste a lot of time waiting at bus stops. There are also cases where people find the buses are full when they arrive and have to wait for the next one. Additionally, parents, always wonder if their young kids have boarded the bus or miss the bus. This study researches a smart bus system that can estimate waiting time based on bus position tracking and passenger load measuring. It can also track children on the bus. All buses have a GPS module attached to them to get the real-time location, and a raspberry pi which can measure the passenger load on the bus by detecting the Bluetooth transmissions from Bluetooth devices. There is also a small feature in the system where parents are able to monitor whether their children missed the bus or not. After analyzing the data on the cloud, the location of the bus, the number of passengers and empty seats on the the estimated waiting time will be fed back to riders and the transportation company.***

***Keywords: , Internet of Things(IoT), smart bus system, GPS, waiting time, Bluetooth, and real-time***

1. Introduction

A smart city is a hot tendency because it can help the government organize the city and make people’s lives more convenient by analyzing the data collected by different devices based on Internet of Things (IoT technology.edu As a part of smart city, transportation system plays an important role in building it. Smart bus is an efficient tool to collect useful data, so there is a need to improve the bus service. Nowadays, people waste a lot of time waiting for the bus. There are cases where people find the bus is full and have to wait for the next one. The main reason for these problems is that passengers don't know the exact time when the bus

that they want to ride arrives. A better smart bus system can be implemented if the real-time position of the bus, estimated waiting time, and the number of passengers are known. In addition, unlike home and school where young children can be protected by teachers and parents, the bus is a public place which is lack of security guarantees. Being worried of kids, parents wonder whether their children getting on the bus or not. It is better for a smart bus that the bus has a system that can track young kids to see whether they get on the bus or not. Furthermore, it cannot be denied that optimizing transportation is very important in the smart city. All the real-time data collected by the bus can be sent to the bus control center to help them make the transportation system more flexible.

1. Previous WORK:

In the past several years, vehicle tracking became a hot topic that helped a lot to offer the real-time information of buses. To build a vehicle tracking system, the most common methods are using GPS and Bluetooth technologies. As for the GPS module, some researchers used Arduino Microcontroller, GPS module, and GSM module to establish a bus tracking system(Lee, [Tewolde](https://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22First%20Name%22%3A%22Girma%22&searchWithin=%22Last%20Name%22%3A%22Tewolde%22&newsearch=true), and Kwon’s lecture, 2014), which achieved real-time bus tracking. Another group of researchers used Bluetooth for the bus tracking (A.Kumar, A. Mathur, A. Kumar, N. Aggarwal , 2017).

The concept of bus tracking was also applied in the area of people tracking. Different from bus tracking, people tracking needs the recognition of ID, which helps distinguish people. The RFID technology represents a breakthrough in embedded communication and can be used to identify virtually any object, including animals, clothes, and

Even human beings.**[4]** (Davis, Steve,2004) As a low consumption application, Bluetooth beacons also play an important role in people tracking.(Lee, g, Kim.J, Lee, Sw , Ko, Yw, 2017)

The number of people measured in public places has been researched for many years. There is much related work about it that has been done using different methods. A group of researchers achieved passenger trip records on public buses by using RFID to track bus cards and Bluetooth to count the number of passengers on the bus. ([V. Kostakos](https://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22First%20Name%22%3A%22Vassilis%22&searchWithin=%22Last%20Name%22%3A%22Kostakos%22&newsearch=true),[T. Camacho](https://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22First%20Name%22%3A%22Tiago%22&searchWithin=%22Last%20Name%22%3A%22Camacho%22&newsearch=true),[C. Mantero](https://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22First%20Name%22%3A%22Claudio%22&searchWithin=%22Last%20Name%22%3A%22Mantero%22&newsearch=true), 2010) It is also accurate to count passengers through a single camera fixed at an overhead position. .Perng, [T.Wang](https://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22First%20Name%22%3A%22Ting-Yen%22&searchWithin=%22Last%20Name%22%3A%22Wang%22&newsearch=true), [Y.Hsu](https://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22First%20Name%22%3A%22Ya-Wen%22&searchWithin=%22Last%20Name%22%3A%22Hsu%22&newsearch=true), [B.Wu](https://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22First%20Name%22%3A%22Bing-Fei%22&searchWithin=%22Last%20Name%22%3A%22Wu%22&newsearch=true), 2016) With the rapid development of BLE, which has low power consumption. Using Bluetooth Low Energy to measure the number of people is becoming more popular.(J. Perng, P.Mathur, 2016)

1. SPECIFICATION:

As for estimating the waiting time, the study researched two main factors that decide the waiting time of riders. One is the real-time position of the bus, and the other is the number of passengers on the bus.

A brief discussion on technologies and methods to meet the different functions of the smart bus system is given below.

1. *Bus Tracking System:*

Every smart bus has a GPS module to locate the bus. The position is one of the significant factors that is needed to estimate the waiting time for riders. Wireless communication can be defined as the transfer of information between the GPS module and the cloud without using wires or cables. Firstly, the time has a relationship with the distance between the bus and bus stop. What’s more, comparing the time between one stop and the next one to the normal time,we can know the traffic condition at that area at that time, which can offer more accurate information for the system to analyze the time when the bus will arrive. Passengers can control the time when they need to get to the bus stop based on the waiting time provided. It helps passengers who utilize this system save much more time. It also helps bus dispatch center to develop the public transportation system like adjusting the bus schedule, rebuilding the bus routine etc.

1. *Passengers Load Measuring*

With the rapid development of technology, almost everyone has a smartphone with Bluetooth, and the number of Bluetooth signals on a bus can be regarded as the number of passengers. As is known, Raspberry Pi has a Bluetooth receiver which has the function of detecting the Bluetooth transmissions from beacon devices. Installed on the bus, the Raspberry Pi collects Bluetooth signals of phones on the bus. The number of addresses equals the number of passengers on every bus, and the data is sent to the cloud and fed back to riders. So,riders can know if there are empty seats on the next coming bus. The data can also be used to estimate the waiting time. The system compares the real-time number to the load factor automatically. If the real-time number of passengers is close to the load factor, the waiting time estimated by the system is supposed to be much longer.

1. *Parental monitor application*

In addition, this smart bus system will contain an application to inform parents about their children’s current status. The aim of this application is to automatically transmit a message by text or through an app simply telling the parents if their children missed the bus or not. A bus time chart will also be sent to the parents, or it could be just a single message containing the time of the next bus. Beacon can transmit a unique identifier which helps distinguish people. This unique identifier will be used to identify people who are 12 years under or people aged over 12. This application will prevent delays simply because parents are able to react and quickly pick up the child and drive them to school in the event of missing the bus.

1. METHOD:

The concept of this project uses a GPS module attached to a WiFi node, which is ESP8266 12E in this project, to locate the position of the bus and collect real-time speed of the bus. With this information, the group calculated the arriving time of the bus on the cloud. BLE embedded in raspberry pi is regarded as a receiver to detect bluetooth signals from smart phones and ibeacon which has specific address. The Raspberry Pi has the ability to verify multiple Bluetooth devices at a time and recognize specific addresses. Therefore, multiple ibeacons approaching

The bus can be uniquely identified by the system

**[5]**.

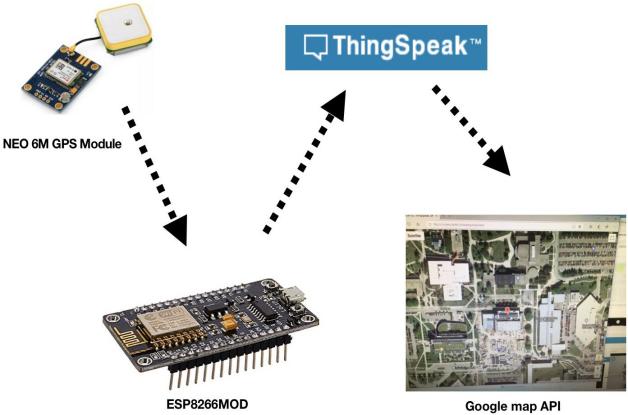


Fig.1 Architecture,

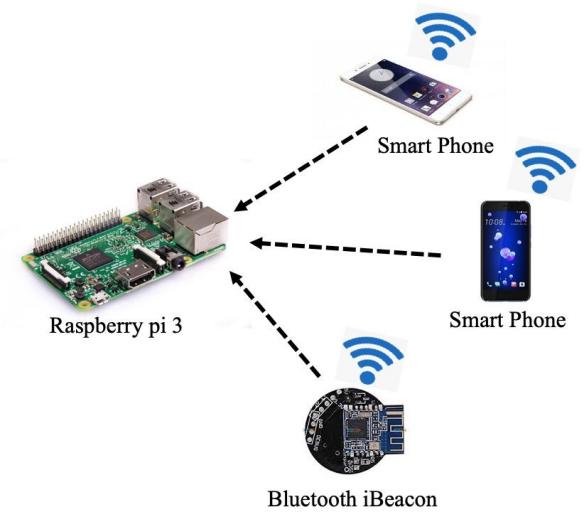


Fig.2 Architecture,

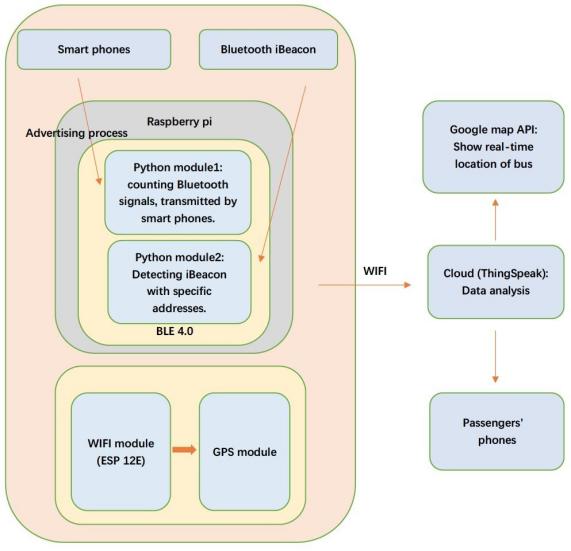


Fig.3 Block diagrams.

The various electronic equipment used in this prototype are:

1. Hardware
   1. *Raspberry Pi 3:*

The Raspberry Pi 3 is the third-generation Raspberry Pi. It replaced the Raspberry Pi 2 Model B in February 2016. It consists of an in-built Bluetooth 4.0 which keeps on looking for a Bluetooth device nearby**[6]**.

* 1. *GPS module:*

The Global Positioning System in vehicle tracking systems is commonly used to provide users with information such as the location coordinates, speed, time, and so on, anywhere on Earth [2].Our project decided to use NEO 6m as the GPS module.

* 1. *iBeacon:*

Every iBeacon has a unique address that can be used to separate it from others. The system used iBeacon as a symbol of a kid and utilized BLE 4.0 as a receiver to recognize the signal transmitted by iBeacon.

* 1. *Wi-Fi module:*

Wi-Fi module helps to send data measured by the GPS module to the cloud through Wi-Fi. The group chose ESP8266 12E as the Wi-Fi module in this project. ESP8266 is a low-cost Wi-Fi microchip with full [TCP/IP stack](https://en.wikipedia.org/wiki/TCP/IP_stack) and microcontroller capability. It can be easily coded through Arduino software.

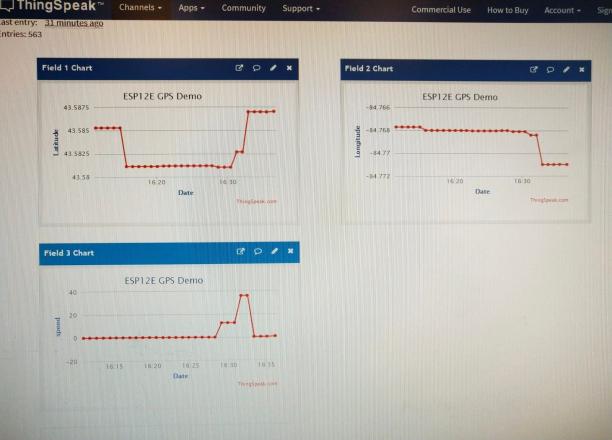
1. Software -
   1. *MQTT:*

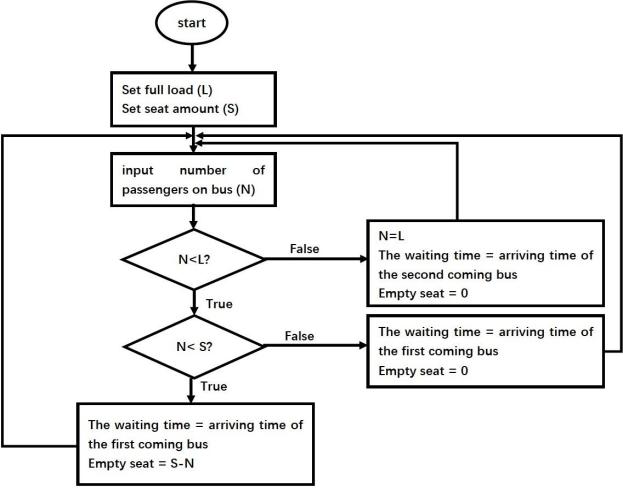
MQTT is a machine-to-machine / "Internet of Things" connectivity protocol, which is a special set of rules that specify interactions between communication.

The basic structure of MQTT is that when a message is published with a topic name to the MQTT Broker, the MQTT Broker broadcasts the message, and only the subscriber with the same topic name can receive the message.

* 1. *Cloud:*

The project used ThingSpeak cloud as a platform to analyze the data that was transferred to the cloud. The specific analysis is as follows:



Fig.4 Logic on cloud

* 1. *Google Map API:*

The Google Maps API gives developers the opportunity to overlay their own data on top of tiled map layers from Google Maps. The overlaid data is typically supplied through KML files and is displayed as interactive vector graphics drawn on the client-side[7].

1. Expriments And Results
2. Experiment 1: Global Position System

The aim of this experiment was to get the location, time, and speed of the GPS module, which will be installed on the bus in the future.

The group connected the GPS module (NEO 6m) to the Wifi module(ESP8266 12E).Firstly, we wrote the code of ESP8266 12E, which included writing API key and reading API key from Thingspeak, to send the data (longitude, latitude, speed) to the cloud. After coding for the ESP8266 12E, we created a notepad and put the Google map API’s code on it that needed a Google map API key, then translated it to HTML so that it could display the actual location on Google map.

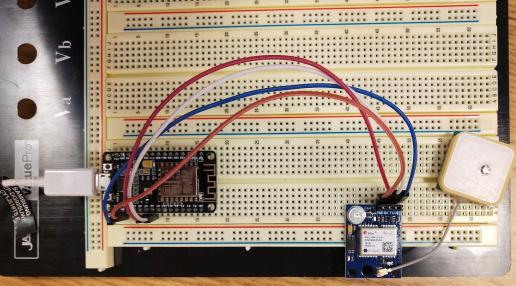


Fig.5 Hookup

Fig.6 interface of cloud



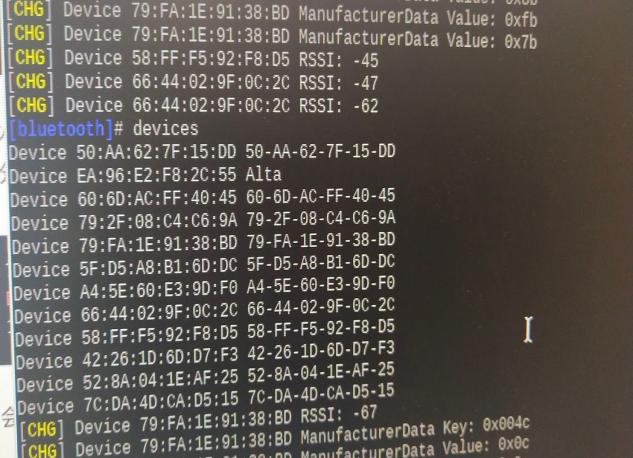
Fig.7 show location on Google map

1. Experiment 2: Bluetooth signals scanner

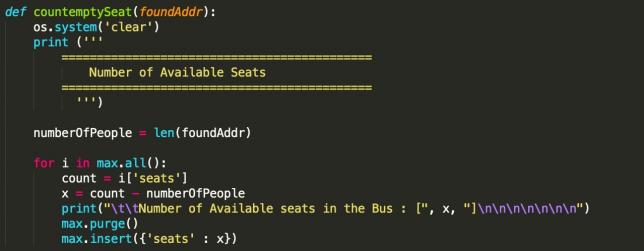
The aim was to determine how many the bluetooth signal detected by Raspberry Pi 3 nearby.

Set up BLE 4.0 on Raspberry Pi using python

language. A list of devices and their addresses showed on the screen, and it represented the number of smartphones nearby.



Python code to count the number of seats assuming the seat number is 40:



After running the code, we opened 2 Bluetooth devices, which means there were 2 passengers on the bus, and we got the number of empty seats as 38.

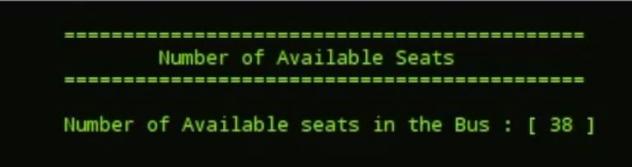
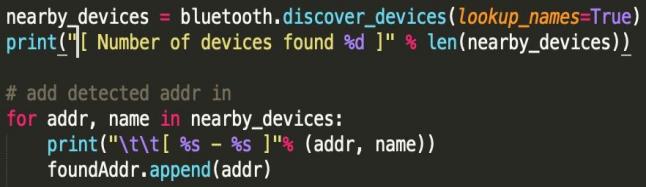


Fig.8 Results of scanned devices

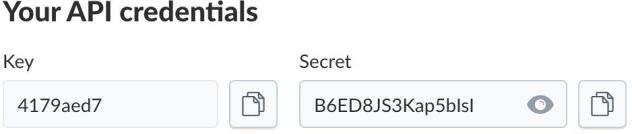
1. Experiment 3: Parental Monitor Application.

The aim of the Parental Monitor Application is to detect the Bluetooth signals from an iBeacon or phones using the built-in Bluetooth sensor in Raspberry Pi. We then used the Bluetooth package in Python to store all the detected Bluetooth devices in an array, which will be used in the compare method. Before any calculation, there are two detection stages. The first detection is the initial detection, and then the second detection is to check for any remaining devices that arrived late. After two detections, the bus countdown of ten seconds starts, then the door closes, and the calculation starts.

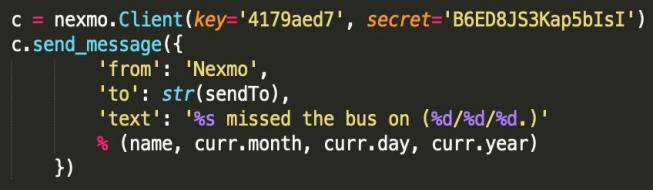
Python code for detecting devices:



The group decided to use the Nexmo API to send SMS in the event of any absence from a certain child. First, we created a Nexmo account to get our API credentials, then we installed the Nexmo module using the command:



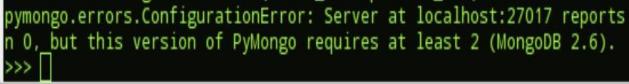
Python code for sending SMS:



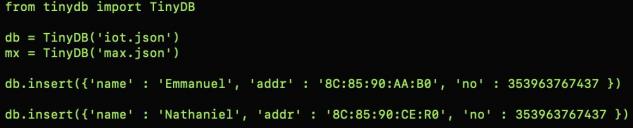
The group decided to use TinyDB to store and access data. The database will include children's names, their unique BLE address, and parents' mobile numbers.



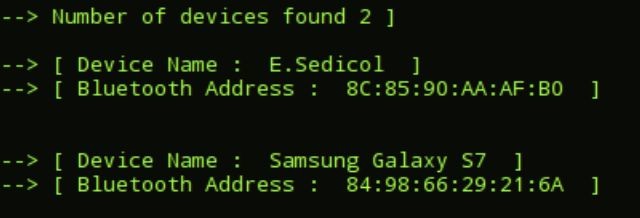
Unfortunately we aren't able to use the Mongo Database due to a version error, so we decided to use a different DB called .



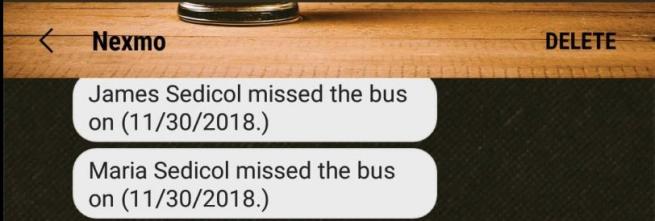
TinyDB



Our presence detection code worked, and the Raspberry Pi was able to detect the beacons by scanning for addresses and comparing the addresses found to the registered addresses.



Which means the system is able to detect the children, calculate their presence, and send alert messages.



1. Evaluation

Overall, the goals of the smart bus system are obvious.

Waiting time evaluation system success:

The system provides an estimated waiting time to people who wait for a bus.

Firstly, the system estimates the bus arrival time on the cloud based on the real-time speed as well as the distance between the bus and bus stop collected by GPS module. Furthermore, the system measures the number of Bluetooth signals successfully and have ability to do analysis on the cloud. and the system can send the estimated waiting time back after

Analyzing the data on the cloud. Lastly, the system uses Google Maps API to display real-time bus tracks. Failure:

The final result is far from the real situation:

The microcontroller cannot connect with the cloud and do analysis on it. Additionally, the Bluetooth receiver cannot detect signals successfully. Furthermore, the system fails to establish a database of traffic information to predict arriving time. And the system can not show the bus track successfully.

Parental monitor application Success:

System is able to detect and record the presence or absence of a child and also analyze it by running Python scripts. Moreover, the system sends notifications (text) to the school or parents whether the child missed the bus or not and is able to get the timetable and send it to parents.

Failure:

System is not able to detect the presence of the child and not able to send messages. Furthermore, the system cannot access the cloud, allowing device connections to allow device connections.

1. Discussion

The group used ESP12E to connect the GPS module because the NEO 6m GPS module cannot get satellite signals indoors. Therefore, the group used ESP 12E instead of connecting with Raspberry Pi directly. After coding on ESP 12E, the data collected from the GPS module can be displayed on the cloud( ThingSpeak), including longitude, latitude, and speed. Through analysis on the cloud, the waiting time can be calculated. Additionally, ThingSpeak can share data with Google Maps API by using a Google Maps API key. However, some limitations are still not solved. Firstly, the distance is straight-line distance, which is used in waiting time calculation, but in the real world, there are a variety of road conditions that may lead to waiting time estimation not being accurate. During the real distance of every degree changes when the latitude changes. This problem also affects the accuracy of waiting time estimation. As for the Bluetooth signal detection, the group successfully detected several signals. However, the range of signal is going to be considered because the system doesn’t expect a very large signal range of bluetooth receiver. The system will only scan the smartphones and iBeacons that are on the bus. Additionally, the group did bluetooth ibeacon detection, which is an important part of the parental monitoring system. Overall, the parental

monitoring system worked as expected. We were able to detect and store the detected Bluetooth addresses. Also, we were able to use TinyDB for storing purposes in the cloud. Accessing TinyDB was super simple, and we were able to query the data needed. Using the queried data, we were able to perform our compare algorithm, which worked with minimal errors. We made sure the output was user-friendly and easy to read, which we achieved using Python scripts. The last point to work on is with Twilio. We are starting to try out a few more options other than Twilio as a way of sending messages, and we are comparing each one. In the end, after our test with different APIs, we finalized on using Nexmo, which worked the best compared. As of now, we aren’t able to send anything. A timetable system which we added in the end using the Date Module in Python, which gave us the current time. Based on the current time, we decided that our bus would arrive every 30 minutes. We basically did basic operations on the current time to output the estimated time of the next bus. Our main goal was to have a predefined timetable stored in our database. We would then use that data to calculate the next bus. Overall,we are happy with this application, and even though it has a lot of room for improvements, we are proud of what we have accomplished in a short amount of time.