TimeCube

HSLU Informatik Department, Switzerland

Group 4 (P. Bucher, P. Kiser, B. Kuhn, A. Ruckstuhl)

May 29, 2020

Group Members

Patrick Bucher
`patrick.bucher@stud.hslu.ch`
Pascal Kiser
`pascal.kiser@stud.hslu.ch`
Benno Kuhn
`benno.kuhn@stud.hslu.ch`
André Ruckstuhl
`andre.ruckstuhl@stud.hslu.ch`

Contents

1 Abstract

TimeCube is a smart IoT device that allows for easy time tracking. Time tracking devices are often cumbersome to use and require a focus switch on the computer screen. The TimeCube is a physical device that makes it obvious which activity is tracked at any given time, which is indicated on the cube's side pointing upwards. The tracked activity can be changed by turning the cube to another side. The prototypes are made of plywood. A prototype contains an accelerometer to detect the cube's position, an A/D converter to convert the signal, and a Raspberry Pi Zero W that forwards the cube's position to a gateway server. From there, the data is forwarded to an InfluxDB instance running in the cloud. The activities to be tracked can be configured in a web interface and are stored in a Redis key-value store. The web interface, which is implemented in Node.js, Express.js, and D3.js, offers tabular and graphical reporting facilities.

2 Introduction

Keeping track of how much time is spent on a particular project is a problem that everyone has faced numerous times at work, at school or while tinkering on different side projects. Time tracking tools are often cumbersome and usually require the use of a web browser or smartphone app. This can be distracting or inconvenient, especially if you don't want to start up your window manager. Or maybe you are even sitting at your soldering station, and don't want to access a computer just to start working on another one of your IoT projects.

In this case, a real-world, physical device you can put on your desk and even carry around with you would be the preferable solution. Our motivation is to help hard-working people to keep track of the time they have invested without distracting them from their work. You can use the TimeCube to quickly and easily track your activities without the need for a computer and access all relevant data and configurations later via a convenient web interface.

The TimeCube is an IoT device that keeps track of which side of the cube is currently up using an accelerometer, and logs this value in a cloud-based InfluxDB instance. To change the currently tracked project or activity, one only needs to rotate the cube. This eliminates the need to switch to another application and is far less distracting to the user. The user is also not required to pick up the smartphone, which could initiate a whole cascade of distractions.

Using the TimeCube web interface, the six sides of the cube can be mapped to different activities, and the data collected by the TimeCube can be displayed in a convenient manner. The physical sides of the cube could be marked using colors or stickers, to help the user to distinguish the different activities currently configured for the TimeCube.

Within the scope of this project, there were to two different prototypes of the TimeCube device built. The detection of the side pointing upward was developed. Both a gateway server and a persistent storage in the cloud were provided. A web application was built in multiple iterations, which can be used for configuration and reporting.

3 Related Works

While time tracking for athletes competing in sport events is a well known and well understood problem (Pers, Vuckovic, Kovacic, & Dezman, 2001), we thought about finding a solution to provide time tracking for other common tasks using an IoT device.

During the research and design process, we figured out that there are already some time tracking systems that are based on physical devices on the market. So time tracking based on an IoT device is not a new idea. The different projects, which are quickly introduced hereby, are all based on a similar approach. Their implementations, however, are slightly different:

TimeFlip is a similar project (timeflip.io, 2016) based on a cube-like device. The company offers different options, with six, eight and twelve sided cubes.

Timeular (timeular.com, 2018) is also a cube-like tracking device, but uses eight blank sides, which can be labeled individually using a felt-tip pen.

Tiller (gettiller.com, 2016) uses a different hardware design. Instead of a cube, it is based on swiping wheel. There is no visible physical information concerning the current activity, which requires a software to be displayed when changing activities. One of the goals of the TimeCube project is to avoid such context switches.

4 System Design and Implementation

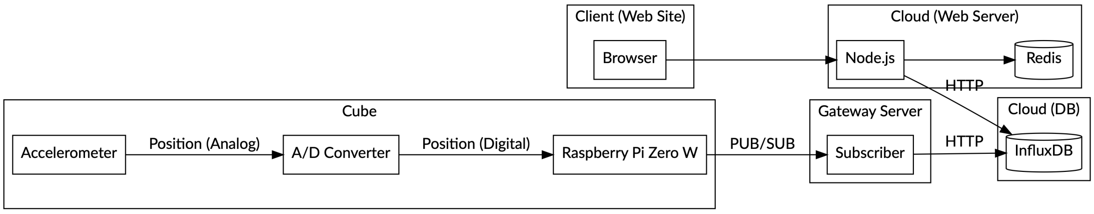


Figure 1: The overall system architecture of the TimeCube system.

4.1 System Overview

The TimeCube is a distributed system. The actual cube is a wooden construction containing a Raspberry Pi Zero W, an accelerometer, and an A/D converter. This part of the setup is responsible for figuring out which side of the cube is facing the top, and to provide a facility to publish that information to the outside world.

A gateway server is used to gather that information from one or multiple cubes, which can be distinguished by an identifier published with the side indication. The data is collected and forwarded in order to be stored in a database. The gateway's role is to connect the local network to the world wide web.

The actual database – an InfluxDB time series database – is located in the cloud. The data from one or many gateways, which could be connected to one or multiple cubes, is collected in a single database. The devices are distinguished by a UUID, which is reported alongside the cube's top-facing side.

A web site for configuration (the cube's side's meaning) and reporting (tabular and visualized as a bar plot) is provided on a web server. Conceptually, the web and gateway server are two distinct entities, but can be run on the same physical server.

A detailed system architecture is shown on figure 1, page 4.

4.2 Software Components

The `timecube` component figures out the top-facing side of the cube and publishes that information, alongside a UUID to identify the device, to the gateway server. The `sub_influx` component subscribes to the data streams of multiple TimeCube devices, and forwards the data points to an InfluxDB instance running in the cloud.

A web application is used for reporting. It consists of two components: First, the `web_backend` component deals with persistent data. It offers HTTP endpoints for storing and receiving configuration data (the value of the cube's six sides) and transaction data (the activities reported by the individual devices). Second, the `web_frontend` component offers a web interface to configure the cube's six sides individually, and create reports for a specific time frame. The activities are reported in a table (list of activities with start and end date, duration, and the name of the respective activity in respect to the configuration) and as a bar chart, which reports the total duration of the six configured activities.

The `timecube` and `sub_influx` components are implemented in Python. The `Adafruit_ADS1x15` library is used to interpret the signals from the A/D converter. A ZeroMQ publish/subscribe socket (Hintjens, 2013, p. 245-247) is used for the communication between the TimeCube and the gateway server.

The `web_backend` and `web_frontend` components are both implemented in JavaScript. Node.js with the Express.js framework is used on the backend. The frontend uses plain JavaScript and D3.js for the plot. The web application can be built and run using Docker Compose.[1]

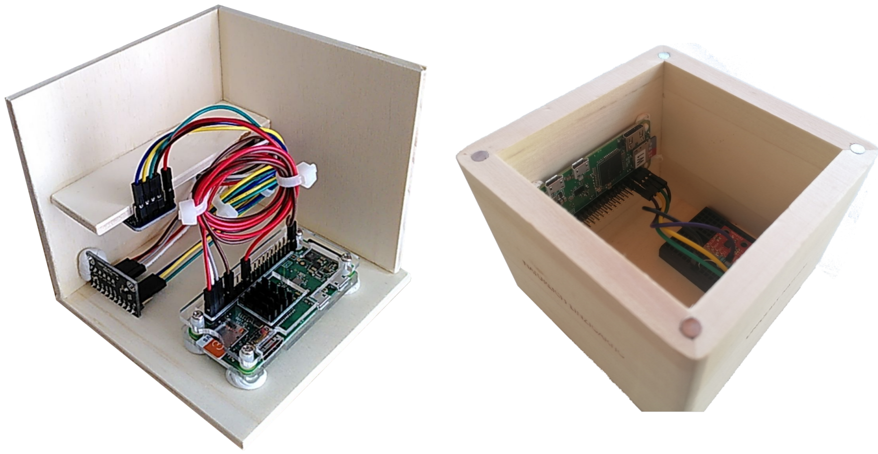
4.3 Hardware Components

The analog signal of the accelerometer is converted to a digital signal using an A/D converter. An I²C (Inter-Integrated Circuit) connection is used between the A/D converter and the Raspberry Pi Zero W. These components are wired together as follows:

1. Raspi (3V3) → Accelerometer (Vin)
2. Raspi (GND) ← Accelerometer (GND)
3. Accelerometer (Xout) → A/D Converter (A0)
4. Accelerometer (Yout) → A/D Converter (A1)
5. Accelerometer (Zout) → A/D Converter (A2)
6. Raspi (3V3) → A/D Converter (VDD)
7. Raspi (GND) ← Accelerometer (GND)
8. Raspi (SCL) ↔ A/D Converter (SCL)
9. Raspi (SCA) ↔ A/D Converter (SCA)

The physical wiring of the two prototypes is shown on figure 2a, page 6 (Prototype I) and figure 2b, page 6 (Prototype II).

[1] See the `README.md` file in the root directory of the repository for further instructions.



(a) Prototype I (b) Prototype II

Figure 2: The wooden TimeCube prototypes, containing a Raspberry Pi Zero W, an accelerometer, and an A/D converter, wired together.

5 Evaluation, Experiments, Results, Discussion

The outputs of the accelerometer first needed to be understood in order to map the accelerometer's outputs (three voltage levels, one for every axis) to the respective cube's sides. Since installing the accelerometer upside down in the first prototype turned out to be the best solution for the physical constraints, the cube's sides were only labeled after its construction was finished.

Several tests were conducted to find the best frequency for uploading the data to the cloud database, so that the results are displayed without too much delay. At the same time, the sampling rate has been reduced as much as possible to avoid unreasonable amounts of data being sent to and stored in the database. A lower sampling rate also reduced the number of invalid data points during the rotation of the cube.

After finishing the hardware prototypes, the behavior of the sensors during movement or while being placed in a way, so that no side was clearly facing up, was tested. The results of these tests were used to optimize the interpretation of the sensor data by dismissing faulty values during rotation by filtering them out.

The performance of the reporting could be further improved to allow for larger timespans to be queried and displayed at once, or to reduce the size of the database. Currently the data for a single day can be queried and displayed without performance issues, which was deemed sufficient for most use cases of a prototype.

6 Application(s)

There are two main applications of the TimeCube, both usable from the web interface:

Configuration The user can configure the meaning of the different sides of the cube, or use the predefined values. The configured values are stored in the web application and specific to the individual TimeCube.

Reporting Reports in the form of tables and bar graphs can be generated for each connected TimeCube. Using the web interface, the user can choose which time period to view.

Time tracking is not only limited for work-related projects. Tracking the amount of time one is spending on personal projects or other pastimes is a possible foundation for self development. If somebody would like to spend more time reading or exercising, but less time watching TV or surfing the web, tracking those activities would be a good start.

7 Conclusion

The prototype is working, and time can be tracked and visualized as intended. Every side of the cube represents a project, which can be defined over a web interface. Although more optimization is certainly possible to allow data collection and reporting over an extended period of time, like, for example by aggregating the data before storing it, or reducing the frequency of the measurements being sent to the cloud database.

Making the frequency of the data collection configurable would allow for additional use cases, where more accurate time tracking is required. In that manner, the TimeCube could also be used as a stopwatch.

For a commercial use, there should be an authentication mechanism, both per device for time tracking, and for connecting to the web interface. Data could be filtered/grouped for certain users, which would allow for reporting of multiple people or entire teams.

Currently, each cube is identified by a UUID, and each cube is meant to be used by a different person. Additional changes could be made to allow sharing a cube between different people, or to allow the use of multiple cubes by the same person. To increase the ease of use, the different sides of the cube should be made clearly identifiable as belonging to a specific project. This could be done using labels or stickers, as it has been done for the prototypes, or better by adding displays to the sides in order to show the currently configured activities, which would greatly improve the user experience.

8 Contributions, Acknowledgments

Name	Role
Patrick Bucher	Prototype I, Gateway, Web Application, Report
Pascal Kiser	InfluxDB, Gateway, Report
Benno Kuhn	Prototype II, Evaluation, WebApp
André Ruckstuhl	Prototype II, Evaluation, WebApp, Report

9 Major Milestones & Deliverables

9.1 Team and Roles

Project Work Packages	Owner
Build First Prototype	Patrick Bucher
Build Second Prototype	Benno Kuhn
Build Second Prototype	André Ruckstuhl
Publish Orientation Data	Patrick Bucher
Evaluate Node Red and Azure	André Ruckstuhl
Setup InfluxDB	Pascal Kiser
Setup InfluxDBCloud2 Tests, JS Web Application	André Ruckstuhl
Interface Evaluation InfluxDBCloud2 to NodeRed	Benno Kuhn
NodeJS Integration in JS Web Application	Benno Kuhn
Implemented Gateway	Pascal Kiser
Develop Web Application	Patrick Bucher
Design the Logo	Pascal Kiser

9.2 Project Planning, Timelines, Milestones & Deliverables

A project meeting was conducted every week after the AIOT class. In that meeting, the activities of the week passed were discussed. Had there been any issues, possible solutions were discussed and assigned to the individual team members, and reviewed in the meeting the following week.

There was no top-level planning conducted at the beginning of the project. The project idea was discussed in the first couple of weeks. The tinkering process with the sensors was also ongoing through that early period of the project.

The two prototypes have been built independently at home, because lessons were no longer held at HSLU in Rotkreuz. The collaboration on the physical devices was thus restricted.

In order to allow the colleagues to work on the software parts of the project (gateway server, persistence, web application), the output of the prototype was simulated using a script. This allowed all team members to fill the data base with demo data. This data was further used to create the queries and web application prototypes.

Unfortunately, Node Red didn't offer a facility to connect to InfluxDB Cloud2.0 instances. The approach of the web application had to be changed. The next idea was to build a single-page JavaScript application that runs entirely in the browser (no backend).

This approach had to be given up quickly, because the facilities for storing persistent data in a browser, which is needed to configure the cube's sides, are very limited, and do not work when using multiple browsers for the same device. Instead, a small web application based on Node.js and Express.js was built, which stores the configuraton in a Redis key-value store.

This web backend also aggregates the data for reporting, which then is displayed by the frontend both as a table and as a bar plot.

At that point, the project was rounded off with a nice logo, and, of course, by finishing the report (this document).

References

gettiller.com. (2016). *How it works.* Retrieved from `https://www.gettiller.com/how-it-works` (Accessed on 2020-05-21)

Hintjens, P. (2013). *ZeroMQ: Messaging for Many Applications*. O'Reilly.

Pers, J., Vuckovic, G., Kovacic, S., & Dezman, B. (2001). A low-cost real-time tracker of live sport events. In *Ispa 2001. proceedings of the 2nd international symposium on image and signal processing and analysis. in conjunction with 23rd international conference on information technology interfaces (ieee cat.* (p. 362-365).

timeflip.io. (2016). *What is TimeFlip?* Retrieved from `https://timeflip.io/en/device` (Accessed on 2020-04-27)

timeular.com. (2018). *It's never been easier to see where your time goes.* Retrieved from `https://timeular.com/product/tracker/` (Accessed on 2020-05-21)