Accu-Pop

DESIGN DOCUMENT

Team Numer:
Dec1701

Client:
Accu-Pop L.L.C.

Faculty Advisor:
Dr. Ahmed Kamal

Team Members/Roles:
Andrew Zellar - Team Leader
Jordan Young - Communications
Stefan Kraus - Webmaster
Josh Wassenaar - Key Ideas Holder

Contact:
dec1701@iastate.edu

Team Website:
http://dec1701.sd.ece.iastate.edu

03/10/2017 - Version 1

Contents

1 Introduction

Accu-Pop is a product that will accurately and systematically capture pop-time of players and display the statistics in a simplistic presentation.

1.1 Project statement

In baseball, "pop-time" is the time it takes from the moment that a baseball hits a catcher's mitt to the moment the ball hits the glove of the second basemen. This is one of the most important ways the skill level of a catcher can be assessed by a scout. The most commonly used method for obtaining pop-time is by using a stopwatch to time the throw it. This is a very inaccurate process and renders results with an error of .2 seconds, which is a very large error when the average pop-time is around 2 seconds. Therefore, a better method for accurately recording the pop-time is needed. The goal of this project is to accurately measure the time it takes for a catcher to throw a ball down to second base, and then display that time on an app and website.

1.2 Purpose

This product will make significantly more accurate readings of pop-times of catchers, which will help baseball recruiters to better assess and compare the skill level of catchers at clinics and showcases. A more accurate reading of pop-time is necessary because recording pop-times by hand is inaccurate and can have an error of up to .2 seconds (10% of the average pop-time). This error can be the difference between an elite catcher and a sub-par catcher. It is necessary to return accurate readings so scouts and other interested parties can correctly judge catchers based on their pop-times. Given that this is phase two of this project our main purpose is to further increase the accuracy of our system to better record catcher's pop-times.

1.3 Goals

This project has the following required goals:

1. Design and implement a complete prototype, accurate to within 1/100th of a second.
2. Develop a phone application program (available on iOS and Android platforms) and a live-updating website.
3. Clear documentation of all products and code.

Stretch Goals:

1. Measurement device
    a. The device will be suitable for use in-game
    b. The device can also be used to measure the time from pitcher to catcher
    c. the cost of the device components should not exceed $70
    d. The device has a way of receiving software updates automatically when connected to a computer
2. Web, Android, and iOS application
    a. Users should be able to track pop-times over time in a graph view
    b. Users should be able to share pop times for players with other users

3. Stretch Goal: measure time from pitcher to catcher.

2 Deliverables

The following are the expected deliverables by the end of the project:

1. A completely functional prototype, accurate to within 1/100th of a second.
2. A phone application program (available on iOS and Android platforms) and a live-updating website.
3. Stretch Deliverable: measure time from pitcher to catcher.

Requirements:

1. Technical Requirements
    a. The device should measure the pop time with a reliability of 90%
    b. The device should measure the pop time with a margin of error not exceeding 5 millisecond
    c. Users should be able to view the data transmitted from the measurement devices across web and mobile platforms.
    d. Users should have a way to save recorded pop times to the server and retrieve the times from other devices
    e. The device should have a way of transmitting data via bluetooth to a central control
2. Non-technical Requirements
    a. The device should not prevent players from moving freely
    b. The cost of the components of the device should not exceed $100

3 Design

The following subsections detail the specifications and functional requirements for the design of this project.

3.1 System specifications

The system specifications have been broken up into two main parts. The first part is for non-functional and the second part is for functional.

3.1.1 Non-functional

1. The system must not interfere with normal play
2. System cost should be kept under $100 per system, preferably under $70

3.1.2 Functional

The following functional requirement have been divided into two sections: hardware and software

3.1.2.1 Hardware Functional Requirements

1. The sensor needs to be reliable enough to get an accurate reading 95% of the time
2. The hardware needs to account for latency and other factors caused by the hardware to keep the error under one hundredth of a second
3. The system must connect to a wireless network and report data to a mobile device or laptop

3.1.2.2 Software Functional Requirements

1. There must be an app for iOS and Android devices
2. There must be a website that displays the data after it is collected
3. Both the app and the website should allow for data filtering, sorting, and searching.
4. The system should allow the user to stop a pop time

3.1.3 Standards

Coding Standards:

General:

Any method above two lines should have a docstring, and those that are below two lines should probably not exist. The specific coding standards for any language used without a single prescribed or generally followed code style guide should be kept consistent within the project. Tests are encouraged, but not strictly required in favor of quick prototyping functionality. The attitude towards code quality is as follows: "Make it work, then make it pretty, then make it scale."

Python:

Python code should be, well, Pythonic. Methods and classes should be small and simple, and deliver specific functionality within the context of their module. Classes should not have gratuitous properties on them, and dependencies should be kept to a minimum.

As far as style guides, PEP-8 should be followed: https://www.python.org/dev/peps/pep-0008/.

JavaScript (JSX / React Native):

JavaScript should be as functional as possible when possible. We will be using React Native, so the exact semantics will not be identical, but in general the app should follow JSX patterns of neatness and organization.

Since the majority of the JS will be in React (JSX) code, the airbnb JSX style guide should be followed: https://github.com/airbnb/javascript/tree/master/react.

3.2 Proposed Design/Method

This project is phase 2 of an already existing project. In phase 1 the project was successful in making pop-time readings to roughly a tenth of a second, additionally they have basic functioning applications for both Android and iOS.

For the continuation of this project, we have the option to improve on the previous group's design or use a completely alternate design of our choosing. We have come up with a few options, the benefits and drawbacks of each options are considered below:

1. **Improving on the Previous Design**
    a. Benefits
        i. The project structure is known, well documented, and shown to work in at least a minimal capacity.
        ii. Hardware components are already bought and installed with the recording software
        iii. Most of the work would be a refinement of the current technology
    b. Drawbacks
        i. It may not be possible to improve the previous design to meet the client's standards
        ii. We would have to upgrade the measurement devices to use Bluetooth 5
2. **Sound Sensor(s)**
    a. Benefits
        i. Fairly cheap sensor
        ii. Strikes a good balance between software and hardware development load
        iii. Passive or almost-passive data collection, depending on where the sensor must be located in relation to the player
        iv. Sound processing is very fast, meaning the results will require relatively little processing and will therefore return quickly with a small processor load
    b. Drawbacks
        i. Would not work at all for measuring the time from pitcher to catcher
        ii. For one solution, device synchronization is required which the previous team struggled with
        iii. Errant sounds would interfere with the results, so we would either have to find a way to control for those or make a requirement that the user does not create sounds that would be picked up by the sensors
        iv. Sounds has a relatively slow propagation through air (340 m/sec) and it the devices are far enough apart there will be significant time measurement errors
3. **Pressure Sensor**
    a. Benefits
        i. Cheap sensor
        ii. A good balance between hardware and software
        iii. Likely one of the easiest signals to process, making results most likely to be accurate and return fast
    b. Drawbacks
        i. Somewhat invasive, as it would require the player to wear sensors in their gloves
        ii. requires multiple sensors or very specific catching area in order to trigger
        iii. Durability may be an issue, so the sensors would have to both have good protection and be easily replaceable
        iv. sensitivity of pressure sensors may cause false readings

3.2.1 Hardware Design

The hardware will be designed to be worn by the users without hindering their ability to catch and throw. The final solution picked after experimentation will be wearable and simple to use. The user will need to connect to a phone and put the sensor(s) in the correct location.

3.2.2 Software Design

The software will allow users to get readings from the hardware from a phone with bluetooth connectivity on a successful trial the data will be sent through the phone to a database that will be accessible through a multiplatform mobile app and website. The data will be available from all available players and will be sortable. The software will be similar regardless of which solution is picked.

3.3 Design Analysis

At this point, our team has yet to finalize our selection for which design path to follow as we have just received the hardware necessary for our project. However, we have made some definitive decisions to reject a couple of proposed designs. Specifically, we have rejected using a camera sensor and tracking system to follow the path of the baseball for cost and consistency reasons.

4 Testing/Development

4.1 Interface specifications

We are developing on two main platforms iOS and Android. We are also creating a Web interface for this project as well. The interface should provide the user with the ability to create a new instance of a player to track their pop-time. The user should also be able to view (perhaps in chart formatting) the list of players and their corresponding pop-time. The interface is expected to be very simple and easy to use and navigate.

4.2 Hardware/software

We will use Android Studio (AS) for development on the Android platform. AS simplifies testing for the software. It provides an emulator so that you can view the progress and functionality of the app you are creating. If the app were to crash during development, then AS provides a detailed report on issue. This allows us to troubleshoot the problem and the source of the issue so that we can fix. AS also has two type of unit test that we utilize: local and instrumented [1]. The local unit tests are just a smaller, automated version of the what was described above. Instrumental unit tests are very similar to local, except these test allow you to implement them on actual devices instead of an emulator which is useful for Android dependencies.

iOS development is similar. We use XCode and this development space which also offers an emulator to monitor app functionality during development. It allows us to utilize XCTest which is the iOS testing suite. XCTest can be used for acceptance and unit testing as well.

For hardware we will use the Arduino IDE. This environment will allow us to make sure that the hardware is responding in the way that it should be. Just as with AS and XCode, the Arduino IDE comes with an emulator. This will aid in correctly and efficient finding bugs as well as aiding in the correction of those bugs.

4.2 Process

Explain how each method indicated in the design section was tested. It might be a good idea to insert a flow diagram of the process.

5 Results

We have yet to run any substantive tests at this point as our hardware just arrived on 3/4/15.

6 Conclusions

Summarize the work you have done so far. Briefly re-iterate your goals. Then, re-iterate the best plan of action (or solution) to achieving your goals and indicate why this surpasses all other possible solutions tested. ← **Needs to be removed before submission, left here in case anyone wants to add to what is below**

Accu-pop is designed to be a timing system for catcher's pop-times. Our goals are to produce an accurate, consistent, and cheap product that will accurately record a catcher's pop-time. Additionally we seek to develop fully functioning Android and iOS apps, as well as a live-updating webpage for easy access to pop-time results.

As for now, the options of design are being tested to ensure that our implementation will be the most efficient and produce the most accurate results. To that end, we hope to find the most effective sensor. We also wish to improve upon the mobile app interface, have basic website functionality, and be able to have stable/reliable device communication.

7 References

List any references used in the document. These are an essential part of your review so far.

1. https://developer.android.com/training/testing/unit-testing/index.html