A SMARTWACTH ACTIVATION SYSTEM DESIGN

Sheng Shen, Jifu Zhao and Duc Phan

CS 598 Final Project, University of Illinois at Urbana-Champaign

Motivation

Recent years have witnessed the boost in the wearable device community, and wrist-worn smart watches are getting popular. As opposed to the rich set of sensors and functionality, smartwatches are always suffering from short battery life, typically less than one day. This limitation prevents one of the biggest battery drainer, the smartatch's screen, from being always on. As a result, whenever the user needs to look at the watch to get the time, a tapping on the screen is required, making this most basic functionality inconvenience.

In order to remediate this issue, we are developing an accurate activation system based on wrist raise detection by monitoring sensors of smartwatch.

Goal:

In the scope of a term projects we set a following goals:

•Assuming that the body is stationary, our first goal is to design an algorithm that effectively differentiates the motion of raising wrist to see time from other similar motions, such as lifting hand to perform other tasks

•Secondly, we want to detect the motion of raising wrist to see time from other general human activities, while the body is moving such as walking, running and etc.

Smartwatch Basics:

Sensor of Interest:

•Accelerometer: measures acceleration along X, Y and Z (includes gravity); wildly available in wearable devices; costs the least energy compared to other sensors.

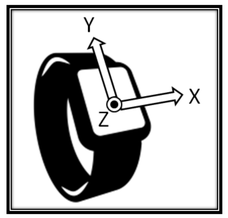


Fig 1.Convention coordinates for sensor data readings in smartwatch. Z is pointing out of watch face. The reference of the coordinate is on the smartwatch.

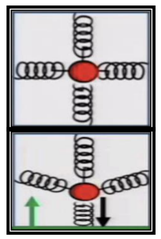


Fig 2.Conceptual understanding of accelerometer and an example reading measures gravity.

Findings

Data Collections:
We developed an Android app to record all sensor readings from smartwatch. Data used for this project is generated by three difference users.

Data is labeled in four categories: true_static, false_static where users are stationary, and true_moving, false_moving where users are moving.

Observations :

Looking at time on smartwatch could be decomposed into a sequence of actions:

1. Wrist movement into region of interest => rising edge in term of accelerometer reading for Z and Y axes
2. A short pause => gravity vector is stable; sensor reading variance decreases.

Notes: some false_movings could generate the above sequence, but the reading value at final position is different.

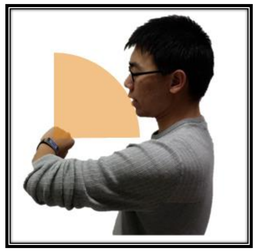
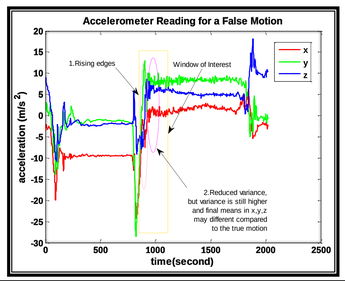


Fig 3. Region of Interest ; user hand position for looking at time.

Fig 4. A samle reading from false_moving which generates a similar sequence to true_moving; but means and variance at final position are different



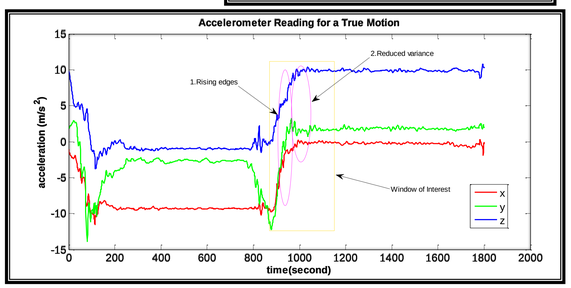


Fig 5. An Illustration of the observations from Accelerometer readings for a true_moving

Solution

Ideas:

We first need to detect whether the smartwatch is in the region of interest (the states where user can possibly look at the watch;
If the smartwatch is in the region of interest, we generate a set of features from accelerometer readings to feed into a classifier to detect whether the current state is the state in which user is looking at time.

1. Check for the following conditions satisfied:
   - Accelerometer in Y, $a_y(t) > t_1$
   - Accelerometer in Z, $a_z(t) > t_2$
   - $a_{yz}(t) = \sqrt{(a_y^2(t) + a_z^2(t))} > t_3$

   Notes: $t_1, t_2, t_3 > 0, and\ t_3 \approx gravity\ (9.8)$
2. Right after conditions in (1) are satisfied at t_0, consider a windows of length L and generate four features:
   - Features(1) = $max(a_z(t_0 : t_0 + L)) - min((a_z(t_0 : t_0 + L))$
   - Features(2) = $mean((a_z(t_0 : t_0 + L))$
   - Features(3) = $mean((a_y(t_0 : t_0 + L))$
   - Features(4) = $mean((a_x(t_0 : t_0 + L))$
3. Feed vectors of Features to training classifiers

Fig 6. Detail implementation

Results:

Three classifiers, including LDA, SVM with linear kernel and polynomial kernel (order of 3), are used to measure the performance of our algorithms

	Training Accuracy	Test Accuracy	Test False Positive	Test False Negative
LDA	0.9606	0.9477	0.1116	0
SVM with linear kernel	0.9992	0.9864	0.0178	0.0089
SVM with polynomial kernel	1	0.9818	0.0178	0.0089

Table 1. Results for static data (assume user body is stationary)

	Training Accuracy	Test Accuracy	Test False Positive	Test False Negative
LDA	0.9459	0.9288	0.1049	0.0137
SVM with linear kernel	0.9876	0.9699	0.0368	0.0172
SVM with polynomial kernel	1	0.9616	0.0368	0.0172

Table 2. Results for all data (including cases in which user's body is static or moving)

	Training Accuracy	Test Accuracy	Test False Positive	Test False Negative
LDA	0.9296	0.9783	0.0333	0
SVM with linear kernel	0.9749	0.9891	0.0167	0
SVM with polynomial kernel	1	0.9674	0.0333	0.0312

Table 3.Results when restrict training sets include data of two users, and the test set data belongs to remaining user.

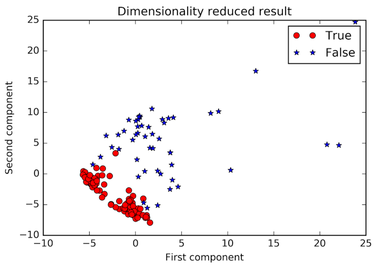


Fig 7. Data visualization after performing PCA

Conclusions

•Our algorithm can achieve 98% accuracy in test set.

•Visualization of the data and the results from difference classifiers suggest that linear classifiers, such as LDA or SVM with linear kernel, are sufficient.

Future Work

•Test on more users and report accuracy.

•Report energy usage (e.g. with different sampling rates)

•User feedback to evaluate latency

ILLINOIS