International Journal of Engineering Research in Computer Science and Engineering (IJERCSE) Vol 3, Issue 6, June 2016

User Defined Hand Gesture Recognition for Android Smart Phones Using Accelerometer

[1] Radhaprasad D. Borkar, [2] Dr. J. A. Laxminarayana
[1], [2] Dept. of Computer Engineering, Goa College of Engineering
[1] rdborkar4engg@gmail.com, [2] jal@gec.ac.in

Abstract: **User experience has come up as an equally important aspect as performance in consumer electronics especially in mobile phones. A lot of research is being made in order to simplify human machine interaction so that the overall activity completion time is minimized subsequently. Today's smart phones come with various technologies such as fingerprint recognition, proximity sensing, motion sensing etc in order to provide their customers a better experience which is closely related to the physical world. Most of the smart phones today are loaded with inertial sensors like gyroscopes, magnetometers or accelerometers that can sense the motion of the device in 3D. This technology has been successfully used in games providing real world simulation of the events and actions to the user. In this paper we propose a user friendly hand gesture recognition for android smart phones based on the readings of built-in 3-axial accelerometer. In this process we have designed an android application that allows users to set some easy gestures to open frequently used apps such as Phone, Contacts, Camera, Gallery etc. For Example, When the user performs a predefined gesture such as placing a phone onto his ears triggers the call for the user selected contact.**

***Index Terms*— Accelerometer, Android, DTW, HMM, Motion based gesture recognition.**

I. INTRODUCTION

Human Computer Interaction (HCI) has been gaining attention of a lot of researchers and developers. Gesture recognition [1] is one such method or technology that detects as well as tracks users movement either through a camera (Computer Vision - based) or Motion Sensor -based. For many years, Vision based gesture recognition has been studied and optimized for gesture recognition however this type of model cannot be applied on smart phones as this model requires high computational and processing requirements. In addition, factors such sensitivity and lighting conditions, camera facing angles etc. limit the applicability of vision based gesture recognition in mobile devices.. Motion Sensor based gesture recognition, on other hand, makes use of the built-in accelerometer which is typically low cost, low power and is not constrained by external factors. This has enabled the development of new input modalities where interaction between human and a smart phone is realized by physical manipulation of the device.

Gesture Recognition can be viewed as a special case of pattern recognition. Works in [2]-[5] utilize Hidden Markov Model (HMM) based recognition. HMM requires extensive training data for effective matching. Works in [6] and [7] use Dynamic Time Warping (DTW) for recognition algorithms. DTW has proven its use with a limited training data in contrast to HMM. Comparing these two methods, Niezen et al. [8] reached to a conclusion that DTW performs faster than HMM since it does not require any preprocessing of raw sampled data. Data preprocessing is very much needed for HMM. DTW-based methods, on the other hand, have to maintain templates for matching and it requires additional storage requirements as the number of gestures increase.

In this paper we present a hand gesture recognition application for android platform. The user can define a set of gestures by moving the phone in any particular direction. Keeping the usability constraints in mind, trials required to define a particular gesture are limited to 5. These set of gestures are used as templates for the matching of the real time gesture performed by the user. We have used DTW technique for matching the gestures to the templates. On a successful match the application performs the predefined action, such as, launching predefined applications. We implemented a prototype on an android phone, evaluated and tested with real users. The applications perform excellently in terms of processing speed and accuracy.

The paper is organized as follows: Section II reviews some basic accelerometer gestures that are possible with mobile phones. Section III describes the current implementation of the Android App we had developed. Section IV shows the results of the current implementation in terms of cpu and memory usage. Conclusions are presented in section V.

II. ACCELEROMETER BASED GESTURES

A 3-axial accelerometer is hardware that measures the acceleration forces as well as rotational forces in m/s2 along 3 physical axes namely x, y and z, including the force of gravity. Some of the commonly used gestures are shaking, tilt etc. Figure 1 shows a 3-axial accelerometer coordinate system.

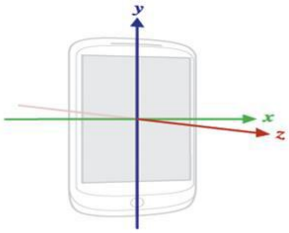
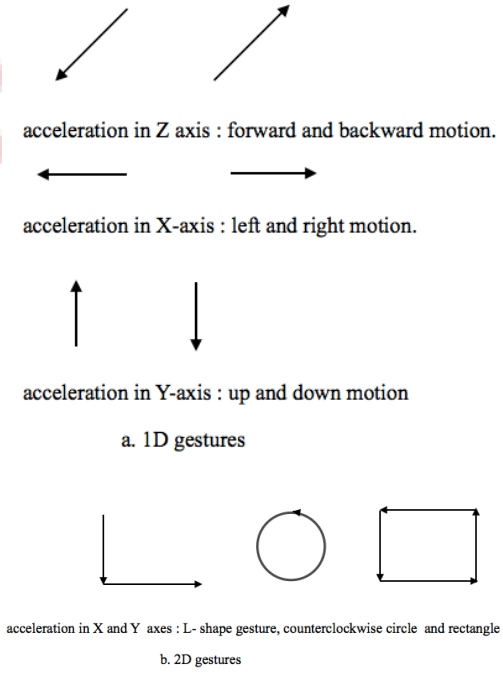


Figure 1: Triaxial Accelerometer Coordinate system [16]

The triaxial Accelerometer provides the user with gesture vocabulary in 1D, 2D and 3D. 1D gestures are linear accelerations made by the movement of the phone in either of the three axes either in positive direction or negative direction as shown in Figure 2.a. 2D gestures are generated when the movement of the phone results in acceleration in two axes. Figure 2.b shows some of the possible 2D gesture with a triaxial accelerometer. Movements that result in acceleration in all three axes are termed as 3D gestures as shown in Figure 2.c.



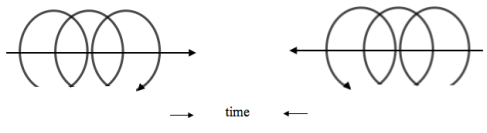


Figure 2: Gestures with 3-axial accelerometer.

Practically we have limited our discussion and development to 1D and 2D gestures keeping in mind various usability and processing constraints. 1D and 2D gestures are easier to perform as compared to 3D gestures. 3D gestures find limited applicability as the time required to perform them is higher than the time required to perform the same action manually, for which the gesture is intended for.

III. PROPOSED SYSTEM OVERVIEW

In this section we propose a gesture recognition system for android mobile phones, solely based on accelerometer readings. We have developed a sample android application (App) using Android Studio 1.5.1 as the IDE and Android SDK.SQLite, which is a software library that implements server less SQL database engine is used to store the templates within the app. Android provides different time delays in which accelerometer readings can be read. Table 1 shows the delay modes and the delay in us (micro seconds).In this work SENSOR_DELAY_UI which sets the sampling rate of 16.67 Hz is used.

Modes	Delay in microseconds
SENSOR_DELAY_NORMAL	2,00,000
SENSOR_DELAY_GAME	20,000
SENSOR_DELAY_UI	60,000
SENSOR_DELAY_FASTEST	0

Table 1: different sensor time delays for accelerometer

The app works in three stages which are described in Figure 3. In stage 1, the user is required to define a set of gestures he/she wishes to perform in order to launch some applications (Call, Gallery etc.). The user needs to perform the gesture only 5 times in order to save it as a template, which is very less as compared to the same action in [2]. After the fifth input, the normal of all these templates is calculated and saved in a local SQLite DB as time series data set. These are used at the recognition stage by the DTW to match the sensed gesture. The user inputs different gestures for different operations.

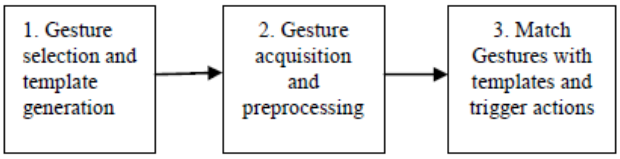


Figure 3: Application workflow

Once the templates are defined, the application can automatically sense and detect the motion or change in accelerations. The captured gesture in stage 2 is also modeled as a time series data set as in stage 1. This allows effective comparison between the templates and the real time gestures. The user performs predefined gesture which is captured in the application and then is forwarded to the next stage for recognition.

The app uses DTW technique for pattern matching which is a widely used algorithm for gesture recognition. Dynamic time warping (DTW) is a technique that calculates the optimal alignment between two time series X and Y by aligning ith point of the first time series X to the jth point of second Y [9] by either stretching or shrinking it along its time axis. It uses dynamic programming approach to find the minimum distance warp path.

The algorithm first calculates a two demential cost matrix A of size |X| by |Y|.where each element A(i,j) is the minimum distance warp path that can be constructed from time series X'=x1,...,xi and Y'=y1,...,yj. The value at D(|X|, |Y|) will contain the minimum-distance warp path between time series X and Y. Any gesture that is "close" (in terms of warp distance) to the template data is considered to be of the same gesture type.

The above two steps provide a value representing the similarity between one sample data set and one template (training) data set. Then all of these steps are completed for all of the sample/template data pairs. The pair that has the smallest "warp distance" indicates the predicted gesture. Figure 4 indicates basic gesture as time series and their similarity using DTW.

Figure 4: gestures in terms of series (a and b) and wrap distance between them (c)

IV. RESULTS

We have tested this application for 4 different gesture types (circle, L shape, I shape, square,). Results are shown in Table 2.The average accuracy we achieved by using DTW instead of HMM is > 90% .HMM provides over 97 % accuracy in matching a real time gestures to the templates [2] but requires high number of training samples to achieve this accuracy. DTW has proved the said accuracy for linear and 2D gestures only. For more complex gestures more training samples are necessary in which case it is desirable to use HMM for pattern matching.

DTW and android environment complement each other as DTW runs faster on big data sets and requires fewer numbers of samples. This is very much needed in mobile environments such as android phones that have comparatively less processing power and memory. During the execution of the app it is observed that it consumed negligible amount of CPU (<5% of cpu) and <10 Mb of RAM which included template generation and pattern matching memory requirements.

Gesture Type	Number of tests	Number of accurate matches	Number of inaccurate matches
I -Shape	20	20	0
L-shape	20	19	1
Circle	20	18	2
Square	20	19	1

Table 2: Test results for different gesture types.

V. CONCLUSION AND FUTURE SCOPE

We have installed the gesture recognition application package (APK) into an android smart phones to test the algorithms performance and usability but there are some features provided by the android framework that are private to the android operating system itself. For example, Power Manager that allows the sleeping and waking up of the phone is private and cannot be accessed by the installed APK.

Simple Gestures can be used such as user keeping his/her phone on the table can be used to make the device sleep as the phone is not in use, this may save a lot of battery power along with proper management of resources. Alternatively when the User picks up the sleeping phone from the table, the phone can be made to wake up automatically providing the user a better experience of an intelligent device.

Accelerometer in conjunction with some other sensors has proved to be a comfortable way for the users to interact with the device [11]. Accelerometer based reading can also be used to define real time human motion like whether the user is standing, sitting or walking. Algorithms can be designed atop this technology to identify critical situations such as falls, accidents etc.

A gesture based Android OS can accommodate all the above possibilities as the operating system can itself access most of the services and make a balanced use of gestures and other user inputs for a better user experience.

REFERENCES

[1] S. Mitra and T. Archary, "Gesture recognition: A survey," IEEE Trans Syst Man Cybern Part C (Applications Rev), vol. 37, no. 3, pp. 311–324, 2007.

[2] Tea Marasovic´ , Vladan Papic´ ,"User-Dependent Gesture Recognition on Android Handheld Devices" 22nd International Conference onSoftware, Telecommunications and Computer Networks (SoftCOM), 2014.

[3] T.Pylvänäinen,"Accelerometer based gesture recognition using continuous hidden Markov models," Pattern Recognit Image Anal, pp. 639–645, 2005.

[4] M. Kauppila, T. Inkeroinen, S. Pirttikangas, and J. Riekki, "Mobile phone controller based on accelerative gesturing," in Proc. 6th Int Conf Pervasive Comput, 2008, pp. 130–133.

[5] T. Schlömer, B. Poppinga, N. Henze, and S. Boll, "Gesture recognition with a Wii controller," in Proc. 2nd Int Conf Tangible Embed Interact, 2008, pp. 11–14.

[6] J. Liu, L. Zhong, J. Wickramasuriya, and V. Vasudevan, "uWave: Accelerometer-based personalized gesture recognition and its applica- tions," Pervasive Mob Comput, vol. 5, no. 6, pp. 657–675, 2009.

[7] A. Akl, C. Feng, and S. Valaee, "A novel accelerometer-based gesture recognition system," IEEE Trans Signal Process, vol. 59, no. 12, pp. 6197–6205, 2011.

[8] G. Niezen and G. P. Hancke, "Evaluating and optimising accelerometer- based gesture recognition techniques for mobile devices," in Proc. IEEE AFRICON Conf, 2009, pp. 1–6.

[9] E. Keogh, C. A. Ratanamahatana, "Exact Indexing of Dynamic Time Warping", In Proceedings of International Conference on Very Large Data Bases (VLDB), pp. 406-417, 2002.

[10] T. Marasovic´ and V. Papic´, "A novel feature descriptor for gesture classification using smartphone accelerometers," in Proc. 18th IEEE Symp Comput Commun, 2013.

[11] Miroslav Takács and Valentino Vranić, "Creating, composing, and recognizing multisensor gestures in mobile devices", IEEE 19th International Conference on Intelligent Engineering Systems (INES), 2015.pp237 - 242.

[12] K. Q. Weinberger, J. Blitzer, and L. K. Saul, "Distance metric learning for large margin nearest neighbour classification," J Mach Learn Res, vol. 10, pp. 207–244, 2009.

[13] T. Marasovic´ and V. Papic´, "Accelerometer-based gesture recognition system using distance metric learning for nearest neighbour classifica- tion," in Proc. 2012 IEEE Int Work Mach Learn Signal Process, 2012.

[14] J. Kela et al., "Accelerometer-based gesture control for a design environment," Pers Ubiquitous Comput, vol. 10, no. 5, pp. 285–299, 2006

[15] Overview of sensors on Android platform, Available on[URL:http://developer.android.com/guide/topics/sensors/sensors_overview.html], Accessed on: 31 March 2016