ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/228366295

Mobile phone controller based on accelerative gesturing

Article · May 2008

CITATIONS
6

READS
181

4 authors, including:

Susanna Pirttikangas
University of Oulu
113 PUBLICATIONS **1,273** CITATIONS
SEE PROFILE

Jukka Riekki
University of Oulu
279 PUBLICATIONS **3,960** CITATIONS
SEE PROFILE

All content following this page was uploaded by Susanna Pirttikangas on 29 May 2014.

The user has requested enhancement of the downloaded file.

MOBILE PHONE CONTROLLER BASED ON ACCELERATIVE GESTURING

Mikko Kauppila, Tuomas Inkeroinen, Susanna Pirttikangas and Jukka Riekki [1])

Abstract

We describe a system for controlling mobile phones via accelerative hand gestures, where phones' internal accelerometers are used as sensory input. The system, running on Series 60 smart phones, consists of a continuous and trainable Hidden Markov Model (HMM) recognizer component, plus a controller component for mapping recognized gestures into phone commands. In contrast to previously proposed systems, our system also supports automatic spotting of gestures, thus freeing the users from pressing buttons to trigger gesturing.

1. Introduction

For the past decade inertial sensors, such as gyroscopes and accelerometers, have been employed to study novel user interaction modalities for mobile interaction. In these modalities, the users interact with devices not by traditional pressing of buttons, but by rotating and moving the device, effectively turning the device into a physical interface. Such modalities can add new degrees of freedom to interaction, and also make interaction calmer by reducing cognitive load.

The orientation, or tilt, of the mobile device has been utilized to implement simple pointing actions, such as navigating through selections [1, 3, 9]. Linear hand movements, or gestures, have been used to implement discrete command interfaces [1, 2, 3, 4, 5, 7]. A popular framework for gesture recognition is the Hidden Markov Model (HMM) [4, 5, 6, 7, 8, 10], although other algorithms have been suggested as well, for example conditional Gaussian models and support vector machines [2].

In most previous work on accelerometer-based gesturing, e.g., [2, 7], gesture recognition is initiated by pressing a button. Cho et al. [2] suggested automated segmentation, or spotting, of gestures as their future work to allow the users to enter sequential gestures without having to press a button each time. As Bartlett [1] points out, button-based initiation has the obvious problem that users have to synchronize the gesture and the button press, which may be inconvenient.

[1] Intelligent Systems Group, Infotech Oulu, FIN-90014 University of Oulu. Correspondence should be sent to mikko.kauppila@ee.oulu.fi

In our previous work [4] we proposed a two-pass algorithm where gestures are automatically spotted in the first phase based on the magnitudal features of the acceleration signal and classified in the second phase. Unfortunately, the false positive rate of the resulting system is too high to be used in a mobile device. In position and vision based recognition the HMM has been enhanced with garbage states to support simultaneous spotting and classification of gestures [5, 6, 10]. In our proposed system, we use a similar garbage-enhanced HMM. The implications of automatic spotting for user interaction are explored in our controller component that maps gestures into mobile phone commands.

2. System Overview

In the first stage, training data is produced by recording exemplar gestures, for example ten repetitions of each gesture. When performing an exemplar gesture, the user holds down the navigation button of the phone during the whole gesture. This simplifies the following estimation stage, although it has the problem that holding the button constrains the orientation of the hand [1], and in the subsequent button-free recognition stage the hand orientation is less constrained. We use a tilt compensation algorithm, similar to our previous work [4], to alleviate this problem.

In the second stage, the HMM parameters are estimated from the training data. An individual HMM is constructed for each gesture, similar to previous work [4, 7, 8], and these individual HMMs are then combined into a single, sparse garbage HMM.

These stages are performed on a PC, where the input sensor data is sent from the mobile phone over Bluetooth/WiFi connection. Using a PC allowed us to explore parameter estimation algorithms more freely. As future work, these stages could be implemented on a mobile phone.

In the third stage, the HMM parameters are transferred to a mobile phone. A gesture recognizer running on the phone will then use these parameters to recognize gestures from continuous acceleration data produced by the phone's inbuilt sensor. A gesture controller then maps the recognized gestures into various phone commands.

An architectural overview is provided in Figure 1 – arrows denote the direction of data flow. Software components are denoted as ellipses, hardware components as rectangles and file storage as cylinders.

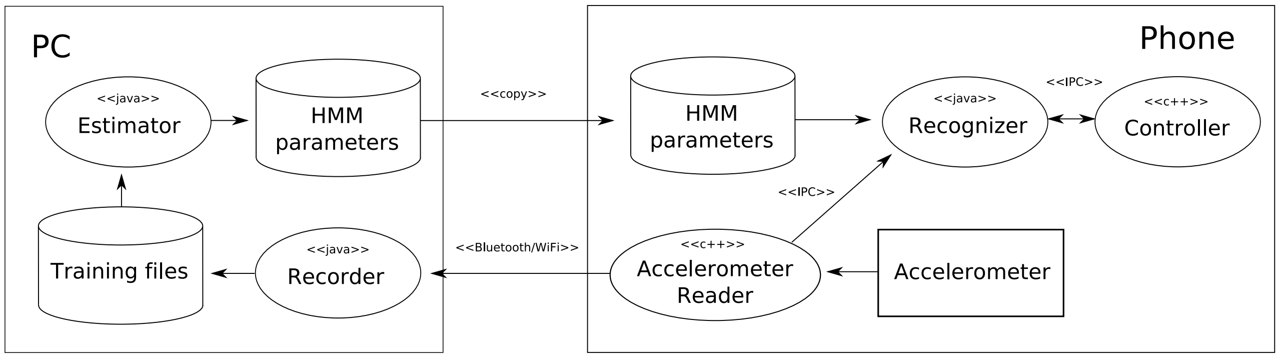

Figure 1. Architectural overview of the system.

3. System Implementation

In this section, the implementation of the recognition system and the controller component are explored further.

3.1. Recognition System

The gesture recognition system, including the recorder, estimator and recognizer components, is written fully in Java 1.5. We use the highly portable Javolution real time class library for collections and serialization, and the JSR14 target mode of the javac compiler to compile Java 1.5 source into Java 1.4 byte code. As a result, the system runs seamlessly on both J2SE and J2ME.

There exists a specification for Mobile Sensor API (JSR256) to access the sensor data from J2ME, but current Series 60 phones don't yet implement this API. Therefore, we use a custom Symbian C++ component to read the sensor data. The component currently supports Nokia 5500 Sport and Nokia N95 phones, which have internal accelerometers.

The set of recognized gestures can be changed on the fly. This is implemented by temporarily modifying the HMM transition probabilities. The motivation is that a lower number of simultaneously recognized gestures lowers both spotting and classification error rates.

3.2. Controller

In order to be able to access low-level features of the Symbian OS, the controller was written in Symbian C++. The controller has been tested with both Nokia 5500 Sport and Nokia N95 phones.

The controller uses four different types of gestures. Six *command gestures* are used to initiate phone commands, currently reading the inbox, composing a message, changing the profile, speed dial, current day's calendar and setting the alarm. Nine *number gestures* are used for selecting the speed dial number. Two *confirmation gestures* ("yes" and "no") are used in some queries. Finally, a single *backlight gesture* can be used to turn on the phone backlight. Below is an example of how the gestures can look like (the large dots denote the start position of the gesture).

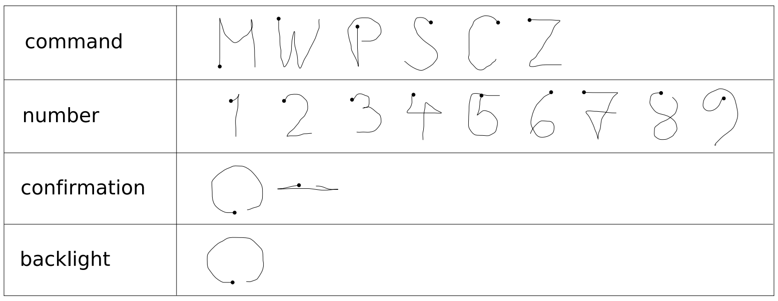

Figure 2. Examples of gestures.

Internally, the controller is implemented as a Finite State Machine (FSM). We have experimented with two FSMs seen Figure 3 – ellipses denote FSM states, and rectangles denote concrete actions. The left FSM represents an entirely button-free controller, where the controller accepts gestures only in the phone's active state where backlight is turned on. The right FSM represents a controller where gesturing is first initiated with button – one button for speed dial and another for the rest of command gestures.

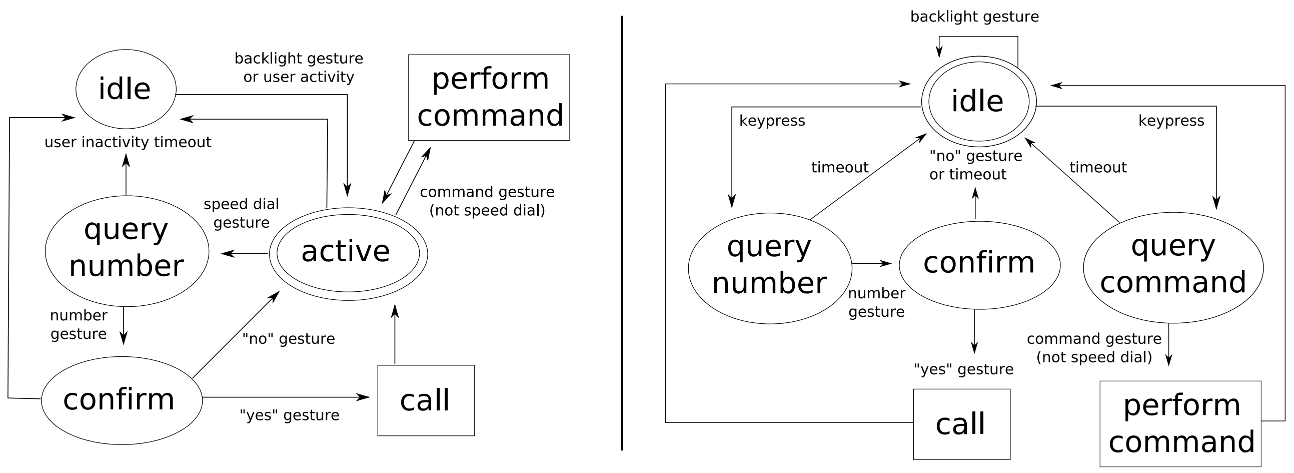

Figure 3. Finite State Machines governing the controller.

4. Conclusion and Perspectives

We have presented a novel controller for Series 60 phones based on continuous gesturing. User studies for both the recognizer and the controller components are underway. Many technical and algorithmic details were omitted due to page limit – these will be elaborated in forthcoming papers.

5. References

[1] BARTLETT, J.F., Rock ‘n’ Scroll is Here to Stay, in: IEEE Computer Graphics Applications 2000, 20(3):40-45.

[2] CHO, S.-J., CHOI, E., BANG, W.-C., YANG J., SOHN, J., KIM, D.Y., LEE, Y.-B., KIM, S., Two-stage Recognition of Raw Acceleration Signals for 3D-Gesture-Understanding Cell Phones, in: Tenth International Workshop on Frontiers in Handwriting Recognition 2006.

[3] HORRY, Y., NAKAJIMA, I., HOSHINO, T., MARUYAMA, Y. A Passive-Style Buttonless Mobile Terminal, in: IEEE Transactions on Consumer Electronics 2003, 49(3):530-535.

[4] KAUPPILA, M., SU, X., PIRTTIKANGAS, S., RIEKKI, J., Accelerometer Based Gestural Control of Browser Applications, in: International Workshop on Real Field Identification (RFId) 2007, pp. 2-17.

[5] LEE, H.-K., KIM, J.H., An HMM-Based Threshold Model Approach for Gesture Recognition, in: IEEE Transactions on Pattern Analysis and Machine Intelligence 1999, 21(10):961-973.

[6] NAM, Y., WOHN, KY., Recognition of Space-Time Hand-Gestures using Hidden Markov Model, in: ACM Symposium on Virtual Reality Software and Technology 1996, pp. 51-58.

[7] MÄNTYJÄRVI, J., KORPIPÄÄ, P., KALLIO, S., Enabling Fast and Effortless Customisation in Accelerometer Based Gesture Interaction, in: Proceedings of the 3rd International Conference on Mobile and Ubiquitous Media (MUM) 2004, pp. 25-31.

[8] PYLVÄNÄINEN, T., Accelerometer Based Gesture Recognition using Continuous HMMs, in: Proceedings of the Second Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA) 2005, pp. 639-646.

[9] REKIMOTO, J., Tilting Operations for Small Screen Interfaces, in: Proceedings of the 9th Annual ACM Symposium on User Interface Software and Technology (UIST) 1996, pp. 167-168.

[10] YANG, H.-D., PARK, A.-Y., LEE, S.-W., Robust Spotting of Key Gestures from Whole Body Motion Sequence, in: Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition (FGR) 2006, pp. 231-236.

View publication stats