Wearables – for Fingers and Ears

Shahriar Nirjon
UNC Chapel Hill
Two Wearables

- **Ring**
  Finger gesture detection
  (*TypingRing, MobiSys ’15)*

- **Earbuds**
  Pulse and motion detection
  (*Musical-Heart, SenSys ’12)*
Typing Ring
A Wearable Ring Platform for Text Input
Shahriar Nirjon, Jeremy Gummeson, Dan Gelb, and Kyu-Han Kim
Hewlett-Packard Labs
MobiSys 2015
Text Input Methods

As computing systems evolve, so do their input methods

**Computing Devices**

**Input Devices**
**Ring – portable, mobile, always with us**

Existing ring based input devices

**Usage of a Ring as a Gesture Interface, NFC tag, Mouse, and for Notifications**

- **Fin** – Numeric pad and gesture interface
- **ThumbTrack** – acts as a mouse
- **NFC Ring** – Two NFC tags to read/write
- **SmartyRing** – alert, notification, and remote control
Typing Ring

Introducing the Typing Ring

Typing Ring

• A wearable, portable, accessory that allows us to input text into computers of different forms.

Specification

• Connects wirelessly as a standard Bluetooth Smart keyboard.
• Works on surfaces such as – a table, a wall, or even your lap.
• Over 98% accurate in detecting typed keys.
• Yields a typing speed of up to 50 keys/min.
• Yields up to 15,500 keys with full charge.
• Weighs ~ 15 gm
Working Principle of Typing Ring
How to type with the Typing Ring

**Wearing It**
The ring is worn in the middle finger.

**Seeking 3-Letter Zones**
As the user hovers his hand on a surface, 3-consecutive keys on a on-screen keyboard is highlighted.

**Typing a Key**
The User makes a typing gesture with one of three fingers and the corresponding key is typed in.
**Special Use Cases**
Special scenarios beside the generic one

**Tiny-Screen Devices**
Devices where we cannot use touch keyboards

**Saving Screen Space**
Typing Ring saves screen space with minimized soft-keyboards.

**Typing On-the-Go**
Wear a keyboard everywhere.

Full-scale Soft KB  
Vs.  
Just Enough Visual Feedback
Hardware Architecture

Hardware components of the Typing Ring

**Microcontroller**
Sensing; Determining and Sending the Key.

**Accelerometer Sensor**
Movement of middle finger; Always On.

**Proximity Sensor**
Determining the typing finger.

**X-Y Displacement Sensor**
Seeking the zone; Optical mouse sensor;

**Bluetooth LE**
Sending the key event.
Firmware Architecture
Software inside of the Typing Ring

**Sensing Layer**
- Read and store 3 types of sensor readings in a bounded circular queue

**Finger/Gesture Recognizer**
- Algorithms to determine 3-letter zone, typing finger, and 3D gesture

**Mapping and Communication**
- Standard key events for a typed key
- Fake key event (ALT+NUM) for zone
- Maps gestures to shortcut keys and sends the key event
Key Stroke Detection

Zone seeking and making a typing gesture

State Machine to Stage Zone Seeking and Typing
Typing Finger Detection
Detecting the typing finger among the three with a HMM

Block Diagram of Algorithm Execution

Given an Emission Sequence
\[ y = (y_1, \ldots, y_T) \]

- Left Finger 3 State HMM
- Middle Finger 3 State HMM
- Right Finger 3 State HMM

Likelihood of Left Finger
Likelihood of Middle Finger
Likelihood of Right Finger

Maximum Likelihood Class
**Gesture Shortcuts**

Simple 3D gestures mapped to short-cut keys

**Gesture to Key Mapping**

<table>
<thead>
<tr>
<th>Gesture</th>
<th>Times Repeated</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pitch</td>
<td>1</td>
<td>Space Bar</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Enter</td>
</tr>
<tr>
<td>Roll</td>
<td>1</td>
<td>Shift</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Caps Lock</td>
</tr>
<tr>
<td>Yaw</td>
<td>1</td>
<td>Delete (letter)</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Delete (word)</td>
</tr>
</tbody>
</table>
Gesture Shortcuts
3D roll, pitch and yaw detection

Variance of 3-axis Accelerometer Readings
Prototype Implementation

Hardware, communication, and visual feedback

**Hardware**

- TinyDuino boards (20 mm x 20 mm)
- MCU – Atmel ATmega328P MCU (8MHz, 32KB Flash, 2KB RAM, 1KB EEPROM)
- Accelerometer – Bosch BMA250 3-axis accelerometer shield.
- BLE – Bluegiga BLE112 module
- Proximity – QRE 1113 IR line sensor (3 mm sensitivity)
- Displacement - ADNS 9800 optical motion sensor (high precision)
- Total wt. 15.5 gm
- Could be miniaturized by 2x-3x
Prototype Implementation

Hardware, communication, and visual feedback

Communication
- Bluegiga BLE112 Bluetooth LE SoC
- BGLib API
- HID over GATT profile
- Two types of HID reports for reporting zone changes and key values.

Visual Feedback
- Android Custom Keyboard
- Two types of visual feedbacks – regular full-scale and 3-key only (for tiny screen devices)
Prototype Evaluation
Evaluating the Typing Ring prototype with micro and macro benchmarks

**System Measurements**
Measuring the executing time and energy consumption

**Empirical Evaluations**
Collecting raw sensor data for analysis and parameter tuning of algorithms

**User Study**
Evaluating the performance (e.g. speed) of Typing Ring
Execution Time
Computation and communication delay

Methodology – Precise Time Measurement

digitalWrite (pin, HIGH);
// Ring firmware code
// segment to time
digitalWrite (pin, LOW);
Execution Time
Computation and communication delay

Results – Execution Times of Major Computation and Communication Components

![Graph showing execution times of various components (msec)]
Energy Consumption

Energy profile and estimated lifetime

Results – Energy Consumption of Various Components

- Arduino (3.6 mW): 4.14 mJ per key
- Accelerometer (0.45 mW): 0.52 mJ per key
- Proximity (60 mW): 24 mJ per key
- Displacement (75 mW): 73 mJ per key
- BLE TX/RX (80 mW): 0.34 mJ per key

Total energy consumption: 102 mJ per key

Energy profile:
- 0-20: Arduino
- 20-60: Accelerometer
- 60-80: Proximity
- 80-120: Displacement and BLE TX/RX

125 mAh battery capacity:
- 13,500 – 15,500 Keys
- 3.8 – 4.3 Hours
Empirical Evaluation
Data collection for offline analysis and parameter tuning

Goal
• Collecting raw sensor readings
• Use data for training the classifier

Data Collection Settings
• 18 Participants
• Each types 50 random characters, 5-15 phrases, and makes 30 gestures.
• Full on-screen keyboard for visual feedback
• Bootstrap classifier (for data collection)
• Sensor sampling at 100 ms interval

Data Collection Program
(Running on a laptop connected to the ring over USB)
Video Demo
Typing Finger Detector’s Accuracy

Different models of HMM

Comparing HMMs with 2, 3, 4 and 5 states

**HMM**
- Empirical Dataset
- 1000 training iterations
- Randomized initialization
- Repeated 10 times
- 70% training, 30% test

![Diagram showing HMM states and observable states](image)
Gesture Shortcut Detector’s Accuracy

Detecting roll, pitch, and yaw

Result – Sampling rate vs. Accuracy

![Graph showing sampling rate vs. accuracy](image)
User Study
Typing speed and experience

Participants
• 7 participants
• 2 sessions each (10-15 min sessions)

Text and Typing Settings
• Concatenated phrases from MacKenzie set
• Manual corrections and gestures allowed
• Full on-screen keyboard for visual feedback
• No auto-corrections

Two Baselines
• Android on-screen soft keyboard.
• Win7 mouse click-based on-screen keyboard.
Typing Speed - Comparison

Rate of valid key entries

Result – Typing speed on a Soft KB, with Mouse Clicks, and Typing Ring
Typing Speed – Learning Effect

Learning effect – it gets better with time

Result – Session 1 vs. Session 1 with Typing Ring
User Survey

Understanding user experience

Result – Survey on various usability aspects of Typing Ring

![Bar chart showing scores for various usability aspects](chart.png)
Musical Heart
A Hearty Way of Listening to Music
Shahriar Nirjon, Dezhi Hong, John Stankovic, + 7 more
University of Virginia and Microsoft Research
SenSys 2012
The Musical Heart System

A biofeedback-based, context-aware, and automatic music recommendation system for smartphones.

Sensor Equipped Earphones (Septimu) + Android App
Musical Heart: Wearable Sensors

- **Sensors:**
  - IMU
  - Microphone
  - IR Reflective Sensor
  - Thermometer

- **Communication:**
  - Audio Jack
  - Bluetooth

- **Power:**
  - Li-Polymer battery
Musical Heart: Smartphone App

Detect: Heart Rate, Activity level, & Context

Sensing

Music Recommender

Proxy
Algorithm – Heartrate Detection

Filtering: A low pass filter to remove non-heart beat signals.

Signals from Ear (music + heart beats)

After filtering (heart beats)
Algorithm – Heartrate Detection

Detection: simple thresholding does not work

Large Threshold:
too few candidates (8)

Small Threshold:
too many candidates (27)
**Algorithm – Heartrate Detection**

**Detection: as an optimization problem**

Step 1 – *Use a small threshold to pick initial candidates and score each based on their peak-peak distance and resemblance to a heart beat.*

6 Candidate for Heartbeats
Step 2 – Maximize the sum of scores, while minimize the variance of time-gaps. *(for an assumed number of beats)*
Algorithm – Heartrate Detection

Detection: as an optimization problem

Step 2 – Maximize the sum of scores, while minimize the variance of time-gaps. (for an assumed number of beats)

For example, to select 5 out of the 6 candidates:

Max Sum = 3.8
Min Variance = 0, if we select the red ones.

Repeat Step 3 for HR = [40, 220]
Algorithm – Activity Level Inference

Activity Levels: Low \((L_1)\), Medium \((L_2)\), High \((L_3)\)

Example: \(L_1 \rightarrow L_2 \rightarrow L_3 \rightarrow L_1 \rightarrow L_3 \rightarrow L_2 \rightarrow L_1\)

We use \(k\)-means clustering to learn the thresholds
Algorithm – Biofeedback and Music Player

\[ u = [\alpha_1 \alpha_2 \alpha_3] \times [\text{Tempo Pitch Energy}]^T \]
Use Case – Cardio Exercise Program

Cardio Chart

<table>
<thead>
<tr>
<th>Time</th>
<th>Desired Intensity (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 min</td>
<td>65%</td>
</tr>
<tr>
<td>3 min</td>
<td>75%</td>
</tr>
<tr>
<td>2 min</td>
<td>85%</td>
</tr>
<tr>
<td>3 min</td>
<td>75%</td>
</tr>
<tr>
<td>2 min</td>
<td>85%</td>
</tr>
<tr>
<td>5 min</td>
<td>65%</td>
</tr>
</tbody>
</table>

Avg. Deviation: 11.4%
Download Musical Heart 2.0

www.cs.virginia.edu/~smn8z/musicalheart.html
Thank You
# Typing Finger Detector’s Accuracy

Comparison of different classifiers

## Setup – Classifier Configurations

**HMM**
- Empirical Dataset
- 1000 training iterations
- Randomized initialization
- Repeated 10 times
- 70% training, 30% test

**Decision Tree**
- Empirical Dataset
- Quantized Features:
  - Proximity Values
  - 3 axis Acceleration

**Naïve Bayesian**
- Empirical Dataset
- Quantized Features:
  - Proximity Values
  - 3 axis Acceleration

\[
P(C|X_1, X_2, ..., X_n) = \frac{P(X_1, X_2, ..., X_n|C) P(C)}{P(X_1, X_2, ..., X_n)}
\]
Typing Finger Detector’s Accuracy
Comparison of different classifiers

Result – Accuracy of HMM, Decision Tree, and Naïve Bayesian