Badger Remote Mouse

Yuye Wang, Xiufeng Xie, Chun Wang, Jack Chiu

11/12/2012

Agenda

- Motivation
- Our Application vs. Related Work
- User Interface
- System Implementation
- Challenges & Limitations
- Summary

Motivation

Motivation: 3D Mouse

- Touchpads in notebook is not easy to use
- Always bringing mouse is not convenient.
- Traditional mouse needs a solid surface

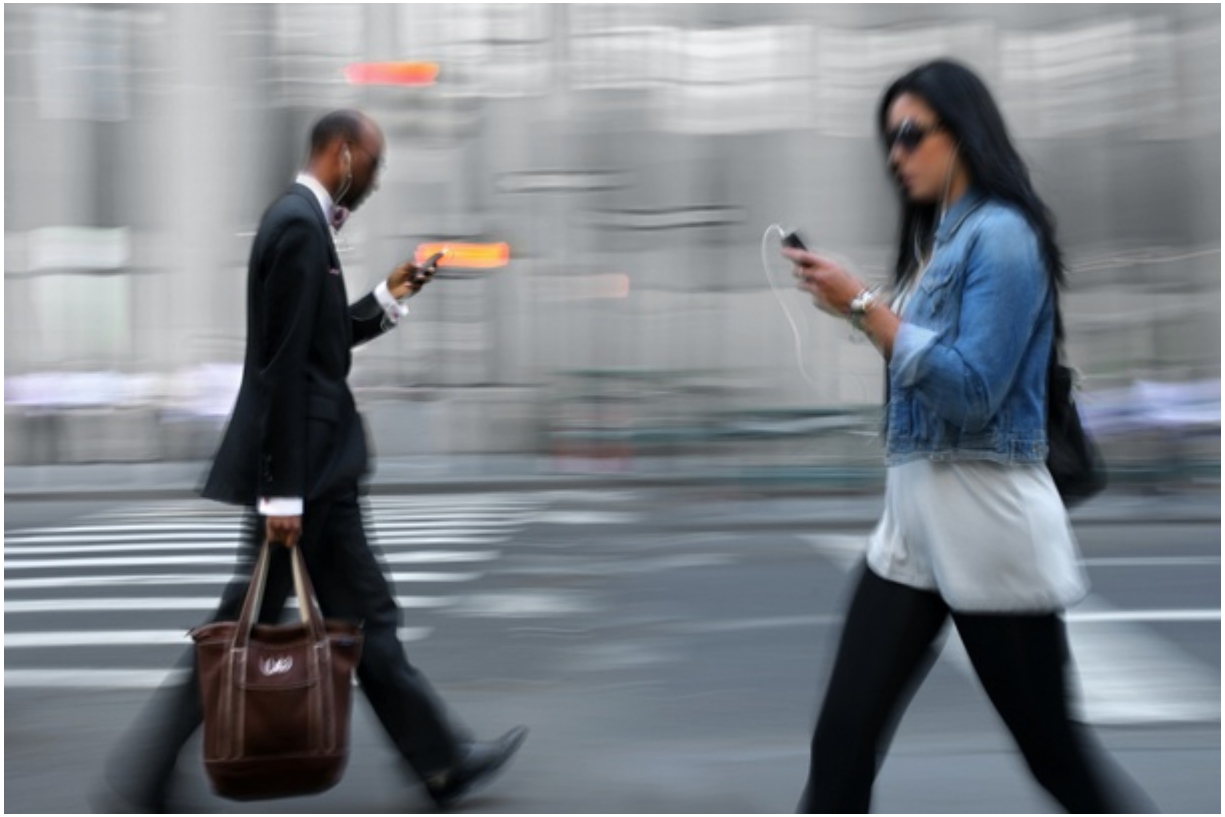
Motivation: Leg Control While Typing

- Switch between keyboard and mouse

- Keyboard typing takes both of your hands

Motivation: Always at Hand

- People will carry their phones all the time



Motivation: Help disabled people

- Also for people with hand disability

- They can use their leg to control a PC

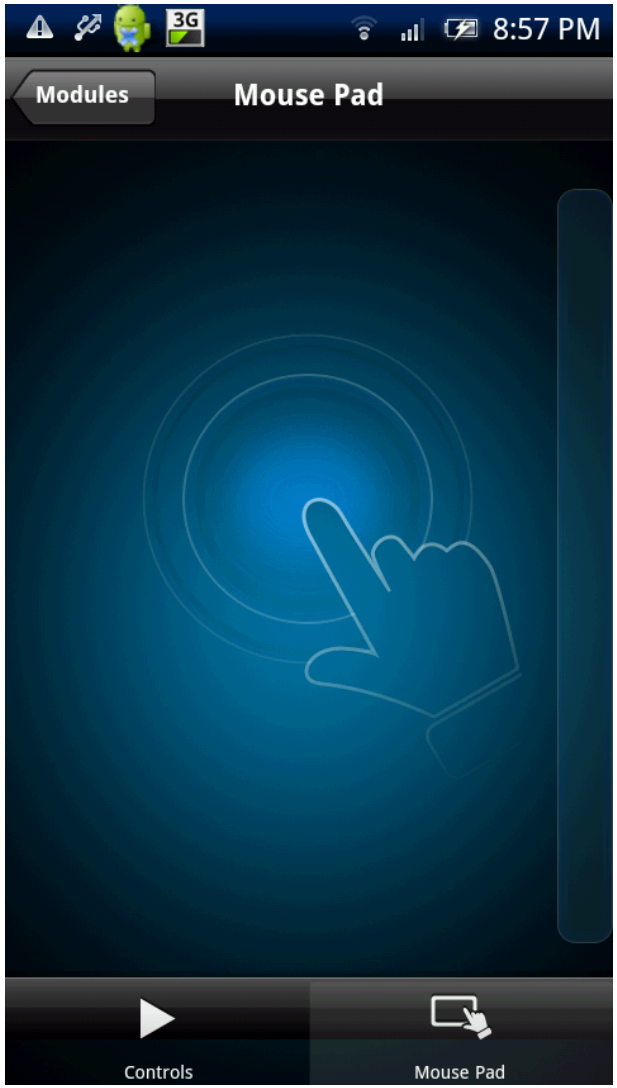
Our Application vs. Related Work

Related Work (i)

- There are many remote mouse application
- Most of them are for Android Phone/iPhone

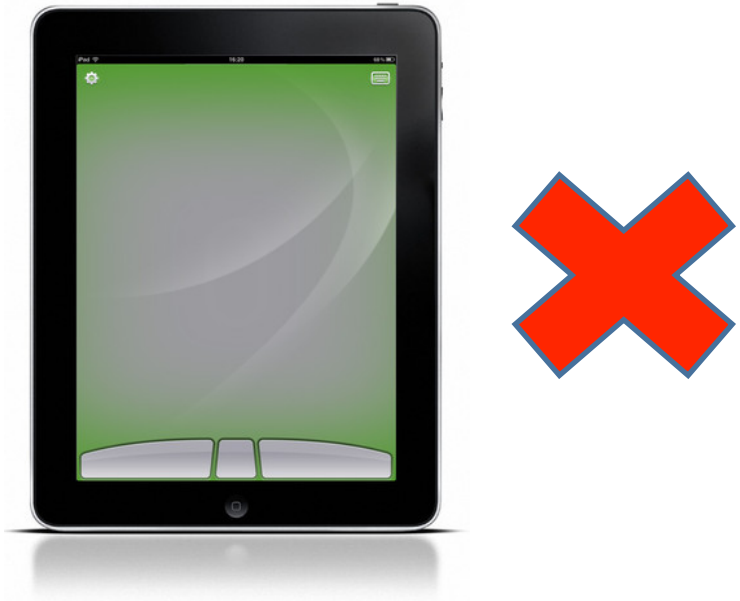
Related Work (ii)

- Not so many for Windows Phone
- Most are simply touchpad
  - Use touch screen on the phone as the touch pad



Our Work: Not Only A Touch Pad

- Use the reading of the accelerometer and gyroscope on the phone to control mouse cursor



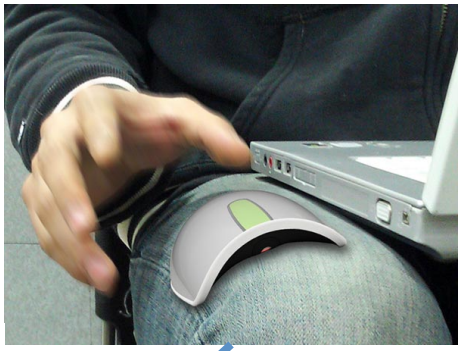
- No need to use your finger

Our Work: Release Your Hand

- Keep on typing, use your leg to control the mouse cursor
- Programmers will be happy about this when busy typing

Our Work: Multiple Operation Modes

- All in one
  - Leg Control
  - 3D mouse
  - PPT clicker
  - 2D mouse
  - Touchpad

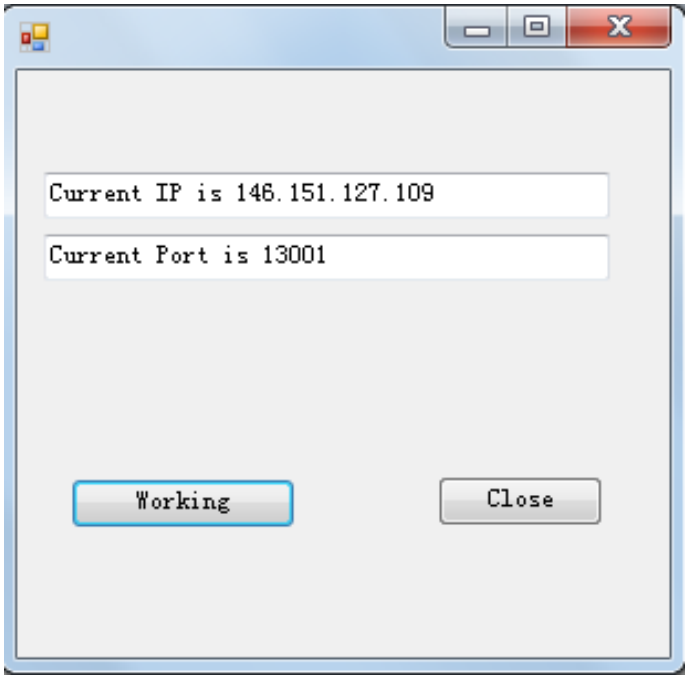


Demo

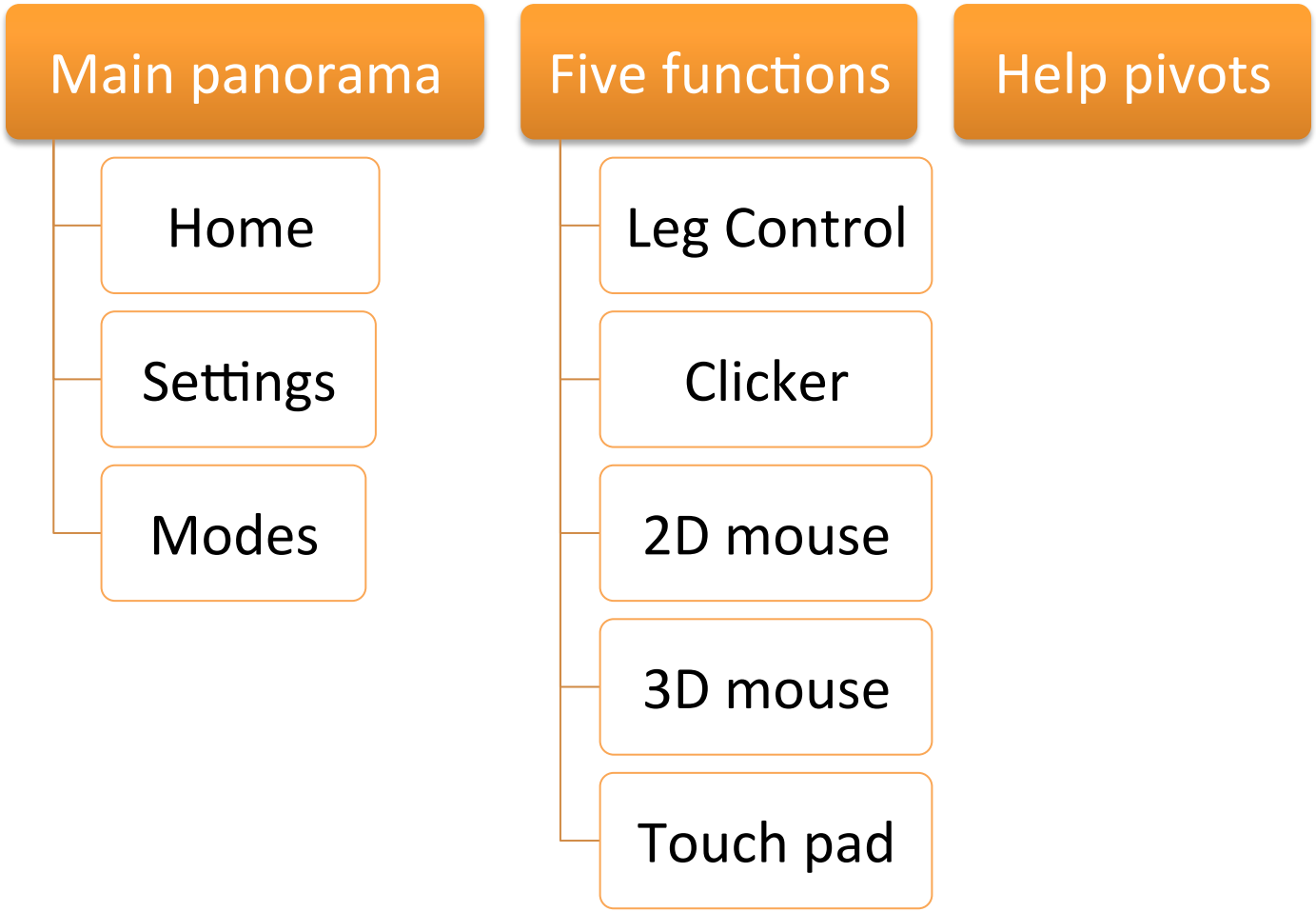
User Interface

Server UI

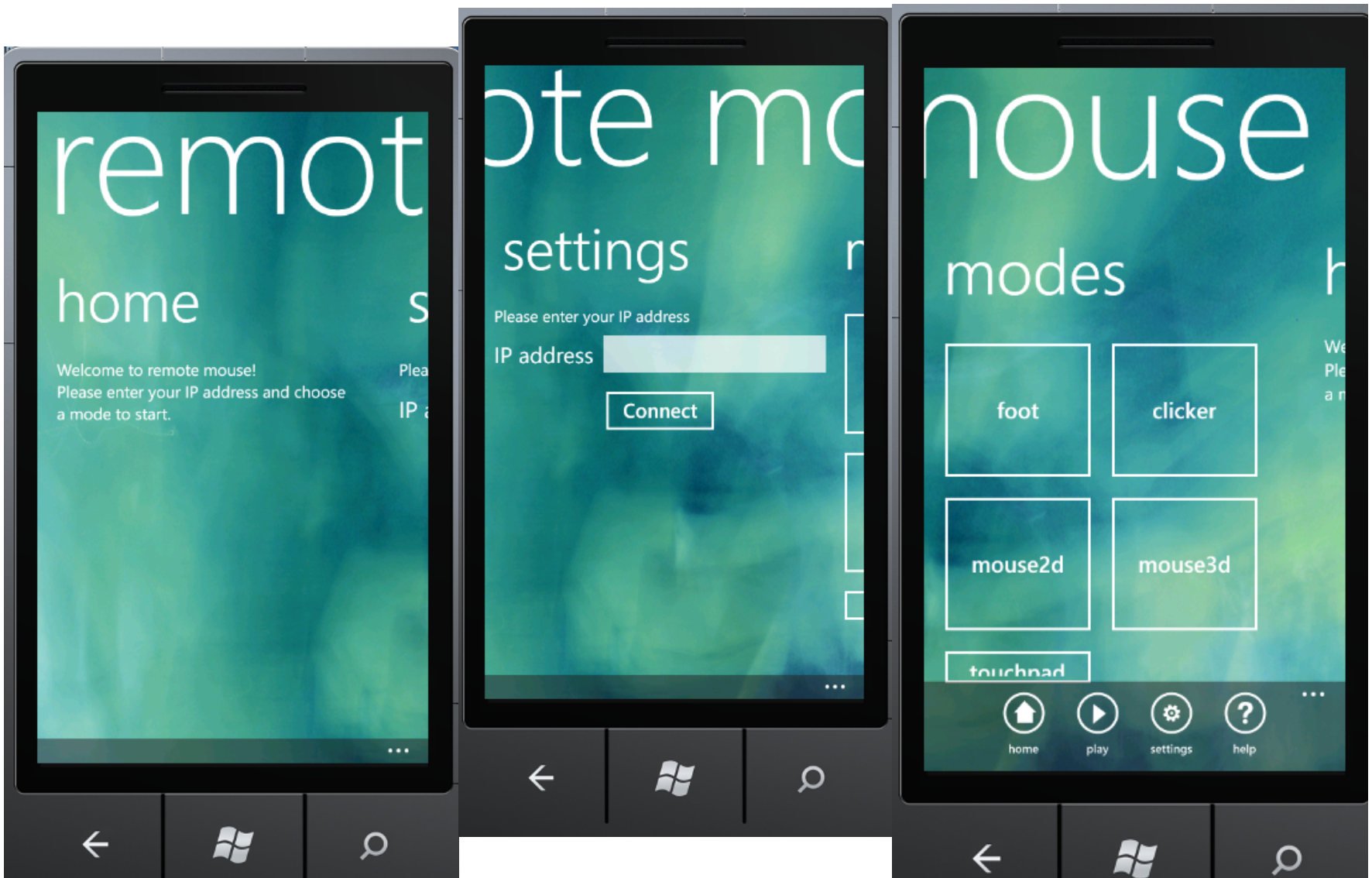
- Windows Form Application
  - More friendly to normal users
- Windows Console
  - For debugging



Client UI


Main panorama
Home
Settings
Modes
Five functions
Leg Control
Clicker
2D mouse
3D mouse
Touch pad
Help pivots

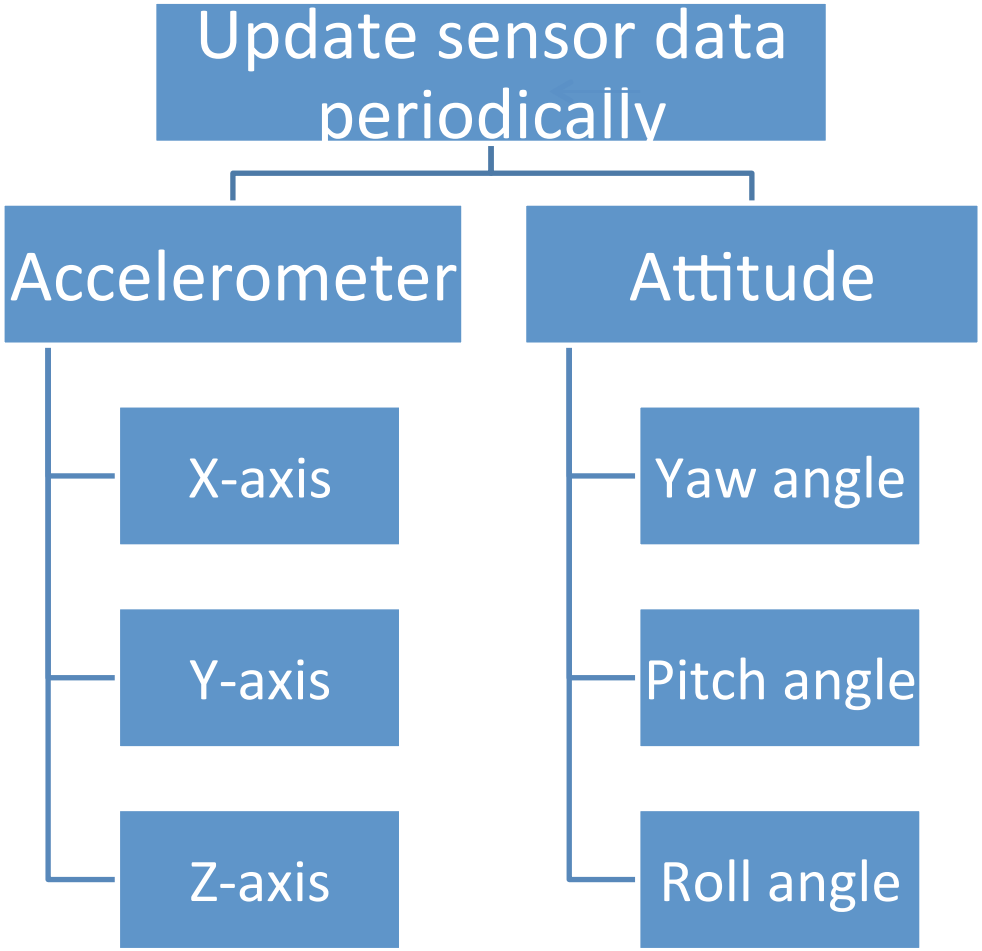
Panorama Page



Functions Page

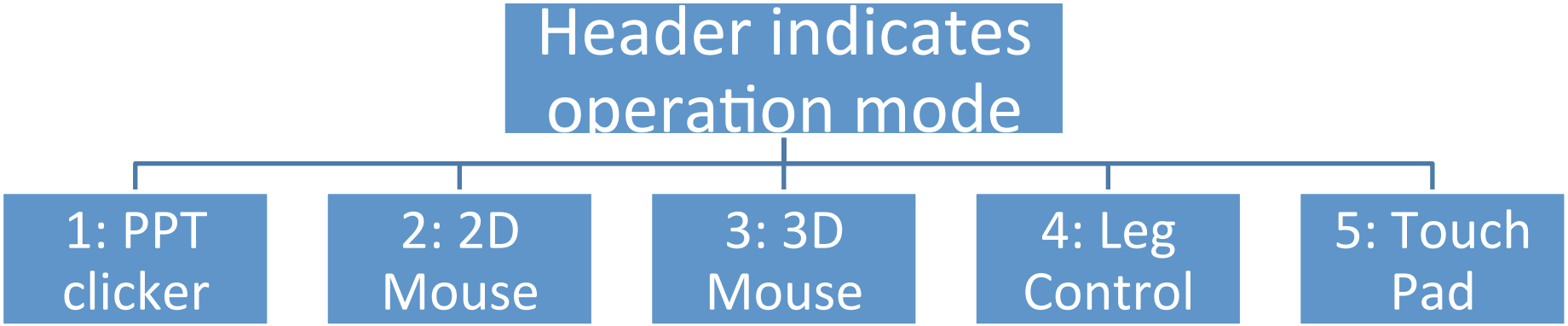
System Implementation

Update Sensor Data



Prepare Data to Send

- Pack sensor data and other control data (e.g. mouse button click control) into a data string



Send Data: How to send?

- Send control data to the laptop (server)
  - Wi-Fi socket (UWNet will block our application)
  - TCP/IP
- Set up your laptop as a virtual AP
  - Open “cmd” in administrator mode
  - Enter (example): netsh wlan set hostednetwork mode=allow ssid=xxf key=12345678
  - Enter: netsh wlan start hostednetwork
- Shut down the virtual AP
  - Enter: netsh wlan stop hostednetwork

Send Data: When to send?

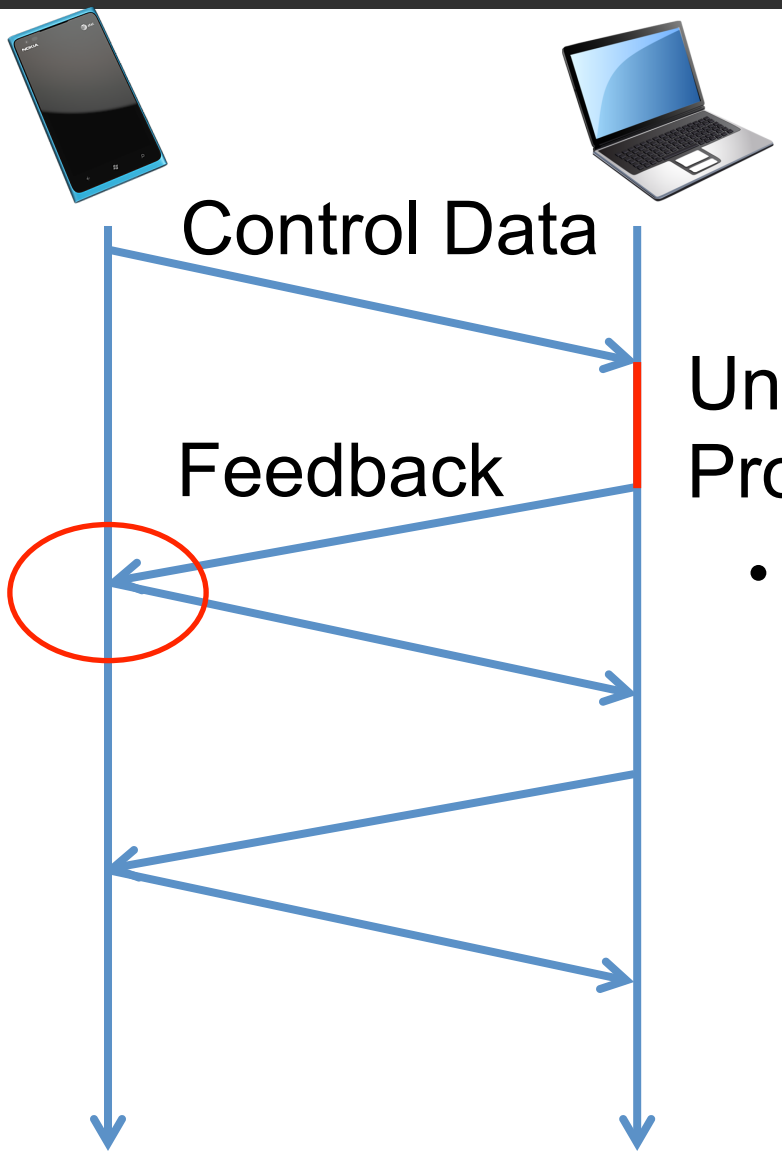
- Send control data periodically?
- Problem:
  - The application always crashes with the exception:

    "An existing connection was forcibly closed by the remote host socket“
- Why: Real-time application vs. Windows
  - If the server cannot deal with old data before new data come, the thread will be blocked, then the application crashes.

Send Data: Less Burden?

- Transmit less data to server
  - Less digits for sensor data
  - Increase interval between transmissions
  - Each operation mode only transmit necessary data for its own functionality
- Larger buffer in server side

Send Data: Use Feedback



Unpredictable
Processing Delay

- Windows is not a real-time system

Send Data: Feedback Triggered

- The sending frequency of control data will never exceed the processing ability of the server
- Advantage
  - More stable
  - More fluent:
    - no need to be conservative for the worst case processing time

Laptop receive data from WP

- Extract control parameters from received data string
  - Operation Mode
  - Accelerometer: X, Y, Z
  - Attitude: Yaw, Pitch, Roll
  - Mouse click indicator

Mode Selection

- Server side program will figure out the operation mode base on the header of received data string.
  - 1: PPT clicker
  - 2: 2D Mouse
  - 3: 3D Mouse
  - 4: Leg Control
  - 5: Touchpad

Smooth received sensor data

- Ripple in sensor data:
  - unconscious movement
- First order low pass filter

$$y(n) = (1 - \alpha)y(n-1) + \alpha x$$

$$\alpha = 0.08$$

Sensor data ⟶ cursor location

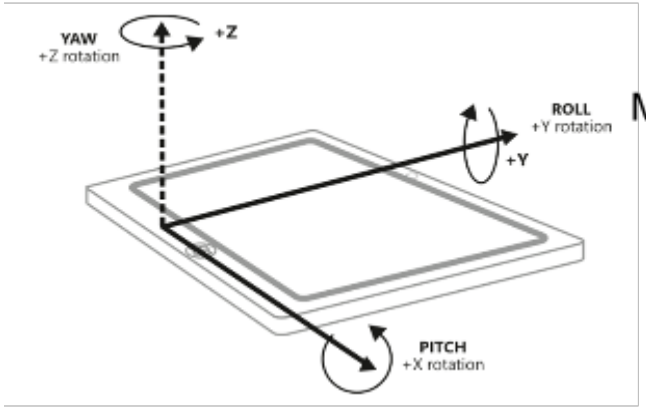
- Mathematic model
  - Cursor location is (x,y)
  - x: location of cursor in the x-axis of the screen

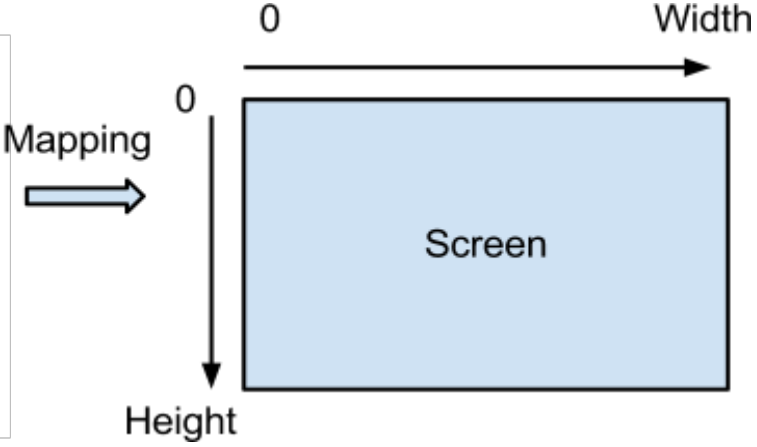
$$x \in [0, Width]$$

  - y: location of cursor in the x-axis of the screen

$$y \in [0, Height]$$

  - Roll: smartphone rotation angle
  - Pitch: smartphone pitch angle





Linear Mapping Algorithm

$$(Roll_{current}, Pitch_{current}) \rightarrow (x, y)$$

– X axis: use the rotation of the phone

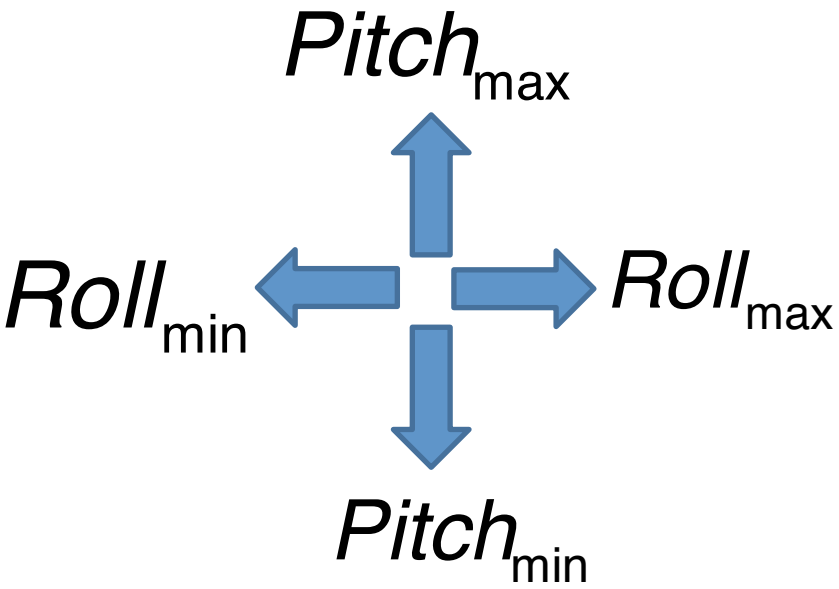
$$x = K_x \cdot Roll_{current} + b_x$$

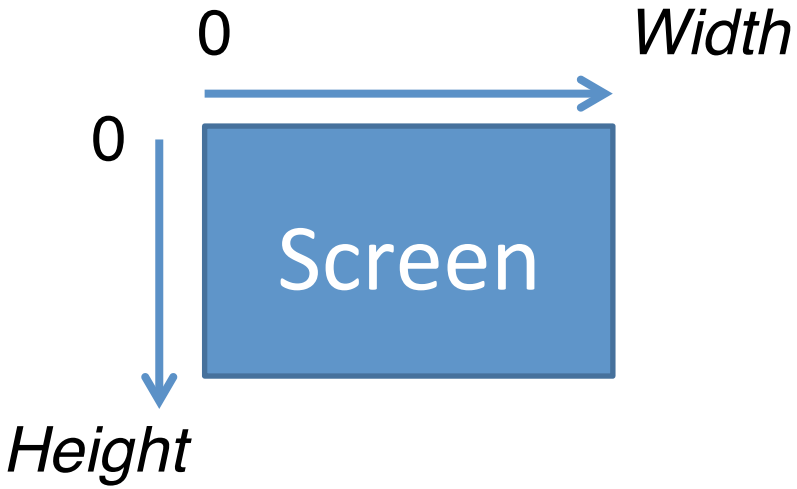
– Y axis: use the pitch up and down

$$y = K_y \cdot Pitch_{current} + b_y$$

– K_x, K_y, b_x, b_y are predetermined parameter

Design Mapping Parameters



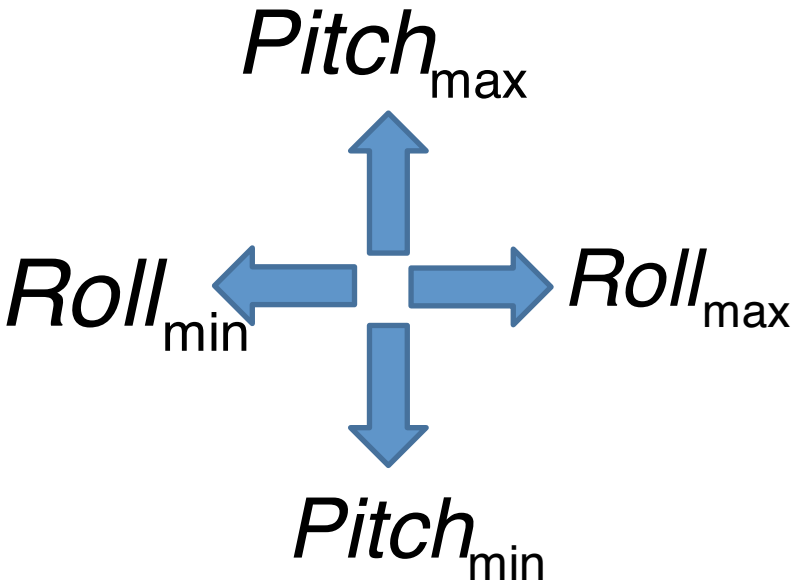


$$K_x = \frac{Width}{Roll_{max} - Roll_{min}} \qquad b_x = -K_x Roll_{min}$$

$$K_y = \frac{Height}{Pitch_{min} - Pitch_{max}} \qquad b_y = -K_y Pitch_{max}$$

Learning Algorithm

- Problem
  - Users have different bodies & habits
  - The range of sensor reading will be different
- Solution
  - Learning algorithm
  - Gather user data in the training phase
  - Use learnt data to update mapping parameters



Set mouse cursor to the position

- Easy to implement using windows API
- Import “user32.dill”
  - Get cursor position
  - Set cursor position
  - Mouse Events
    - Mouse button up & down
    - Mouse wheel

Simulate Mouse Button Click

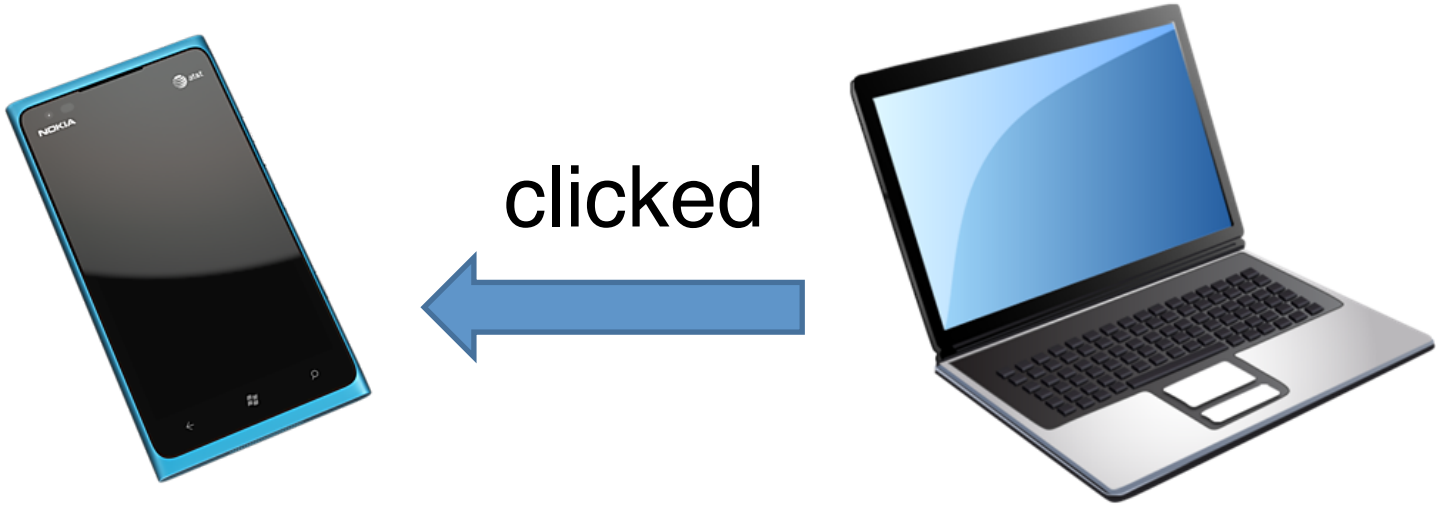
- Leg control
  - Set a threshold for acceleration
    - If acceleration on Z-axis (up/down) exceeds the threshold, perform left button click
    - If acceleration on X-axis (left/right) exceeds the threshold, perform right button click
  - Do not interfere with cursor position control, as long as you do it slow enough

Simulate Mouse Button Click

- 2D/3D mouse
  - Display mouse button on the touch screen of windows phone
    - Use “Tap” Event rather than “Click” Event
    - Double Tap
  - Pack mouse button click variable in the data string transmitted to the laptop
  - Laptop perform mouse click based on the received mouse button click variable.

Send feedback to smartphone

- Transmit a data string back to the phone
  - In normal case, transmit “connected”
  - If mouse button is successfully clicked, transmit “clicked”

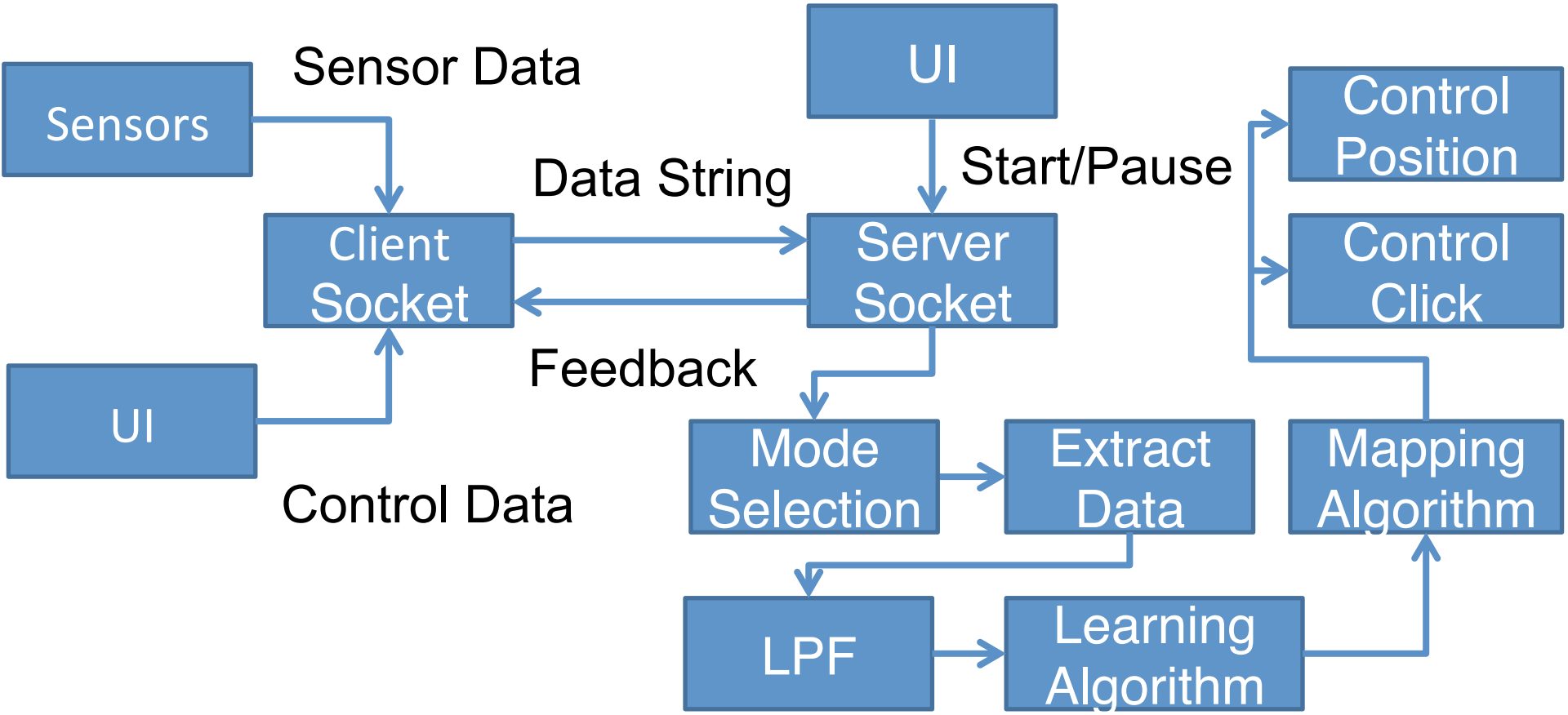


Phone gives feedback to user

- After receiving the "clicked" signal from the server, WP will vibrate as a feedback that the button is successfully clicked.

- On receiving the feedback, WP will send current sensor data and other control data to the server

System Diagram



Challenges & Limitations

Selection of Click Threshold

- Problem
  - Simulate mouse button click without interfering with cursor position control
  - Use acceleration on different axis to control click
    - Threshold to large: difficult to trigger click
    - Threshold to small: always interfere with position control

Feedback to User

- Problem
  - Users want feed back
  - Physical button feedback:
    - Feeling on the hand
    - Click sound
  - No physical button in our application
- Solution
  - Vibration after click
    - After a mouse button click, server sends a "clicked" signal back to WP.
    - After receiving the "clicked" signal, WP will vibrate.

Smooth Data

- Problem
  - You thigh is not a stable platform
  - Sensor reading will ripple
- Solution
  - A first order low pass filter is used to smooth sensor data
  - Eliminate the rippled caused by the inadvertent movement of users.

What is the proper place

Method	**Phone location**	**User status**	**User behavior**	**Easy to use?**
Hand control	At hand	No limit	Rotate phone with hand	Good
Thigh control	Tied above thigh	Sitting	Move thigh	Good
Ankle control	Tied above ankle	Sitting	Move ankle	Fair
Body control	Tied above ankle	Standing	Move body	Fair

Precision of the sensor

- Sensors equipped on Windows Phone are not precise enough.
- Use integration of acceleration, we cannot get current velocity of the phone.
- Fatal for 2D mouse

Take away

Take away

- Fast
- Simple
- User friendly
- Integration

Summary

Thank you!