# Understanding the Pen Input Modality

Presented at the Workshop on W3C MMI Architecture and Interfaces
Nov 17, 2007

Sriganesh "Sri-G" Madhvanath
Hewlett-Packard Labs, Bangalore, India
srig@hp.com

## Objective

- Briefly describe different aspects of pen input
- Provide some food for thought …

## Unimodal input in the context of Multimodal Interfaces

- Multimodal interfaces are frequently used unimodally

  Based on

  - perceived suitability of modality to task
  - User experience, expertise and preference

- It is important that a multimodal interface provide full support for individual modalities

  “Multimodality” cannot be a substitute for incomplete/immature support for individual modalities

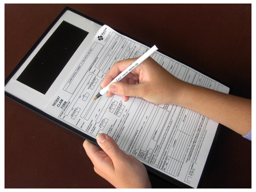
## Pen Computing

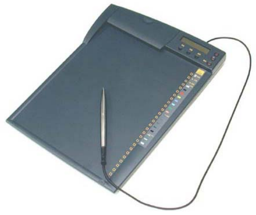
- Very long history … predates most other input modalities
  - Light pen was invented in 1957, mouse in 1963 !
- Several well-studied aspects:
  - Hardware
  - Interface
  - Handwriting recognition
  - Applications
- Many famous failures (Go, Newton, CrossPad)
- Enjoying resurgence since 90s because of PDAs and TabletPCs
  - New technologies such as Digital Paper (e.g. Anoto) and Touch allow more natural and “wow” experiences

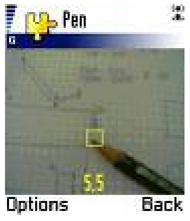
## Pen/Digitizer Hardware …

- Objective: Detect pen position, maybe more
- Various technologies with own limitations and characteristics (and new ones still being developed !)
  - Passive stylus
    - Touchscreens on PDAs, some tablets
    - Capacitive touchpads on laptops (Synaptics)
    - Vision techniques
    - IR sensors in bezel (NextWindow)
  - Active stylus
    - IR + ultrasonic (Pegasus, Mimeo)
    - Electromagnetic (Wacom)
    - Camera in pen tip & dots on paper (Anoto)
- Wide variation in form
  - Scale: mobile phone to whiteboard (e.g. Mimeo)
  - Surface: paper to display to whiteboards to projections

## … and Devices …







## But it all boils down to:

- What operations are detectable ?
    - Contact – up/down
    - Marking – Drawing/Writing
    - Hover? (z-coordinate)
    - Modifiers? (like mouse buttons)
    - Pen identity (which pen used?)
    - Eraser (which tip used) ?
    - Additional modes via software states

- What channels are captured ?
    - x, y, z, force, pen tilt, color, width, …

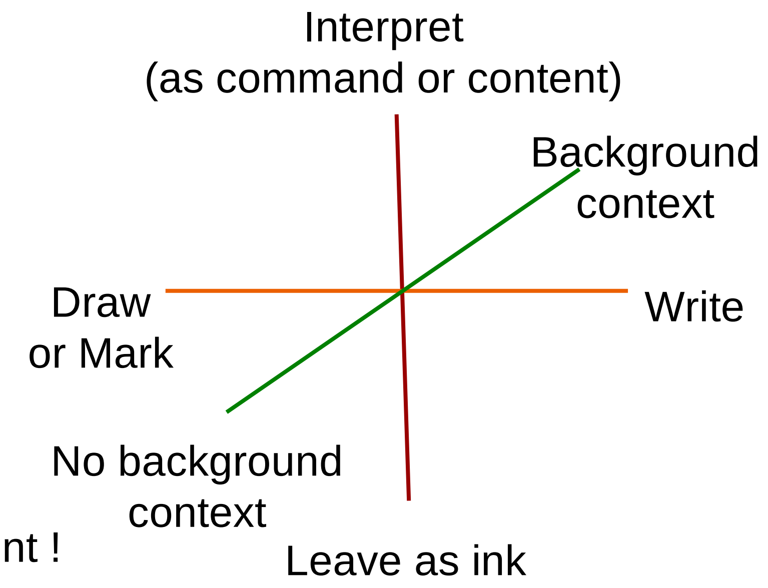
## Pen-based Interfaces

- Interfaces that try to use a pen for accomplishing something useful
  - Extreme “pen computer” view (e.g. slate computer): Enable all interaction via pen alone
  - Customized view (e.g. vertical app): Use pen as integral part of a specific application
  - Normal view: (e.g. TabletPC convertible) Exploit pen affordances to augment a graphical interface

# One Pen, Many Possibilities !

- Very versatile, you can do just about anything !
  - Pro: Fine motor control for precise actions
  - Con: Limited by physical hand movement to doing things sequentially

- Common pen functions
  - Point at things, select from lists
  - Write Ink
  - Draw Ink
  - Gesture
  - Sign
  - Write to enter text
  - Draw to create a figure
  - Either by itself, or on top of other visual content !



## Some Custom Applications
Source: An Overview of Pen Computing, Lee et al, 2005

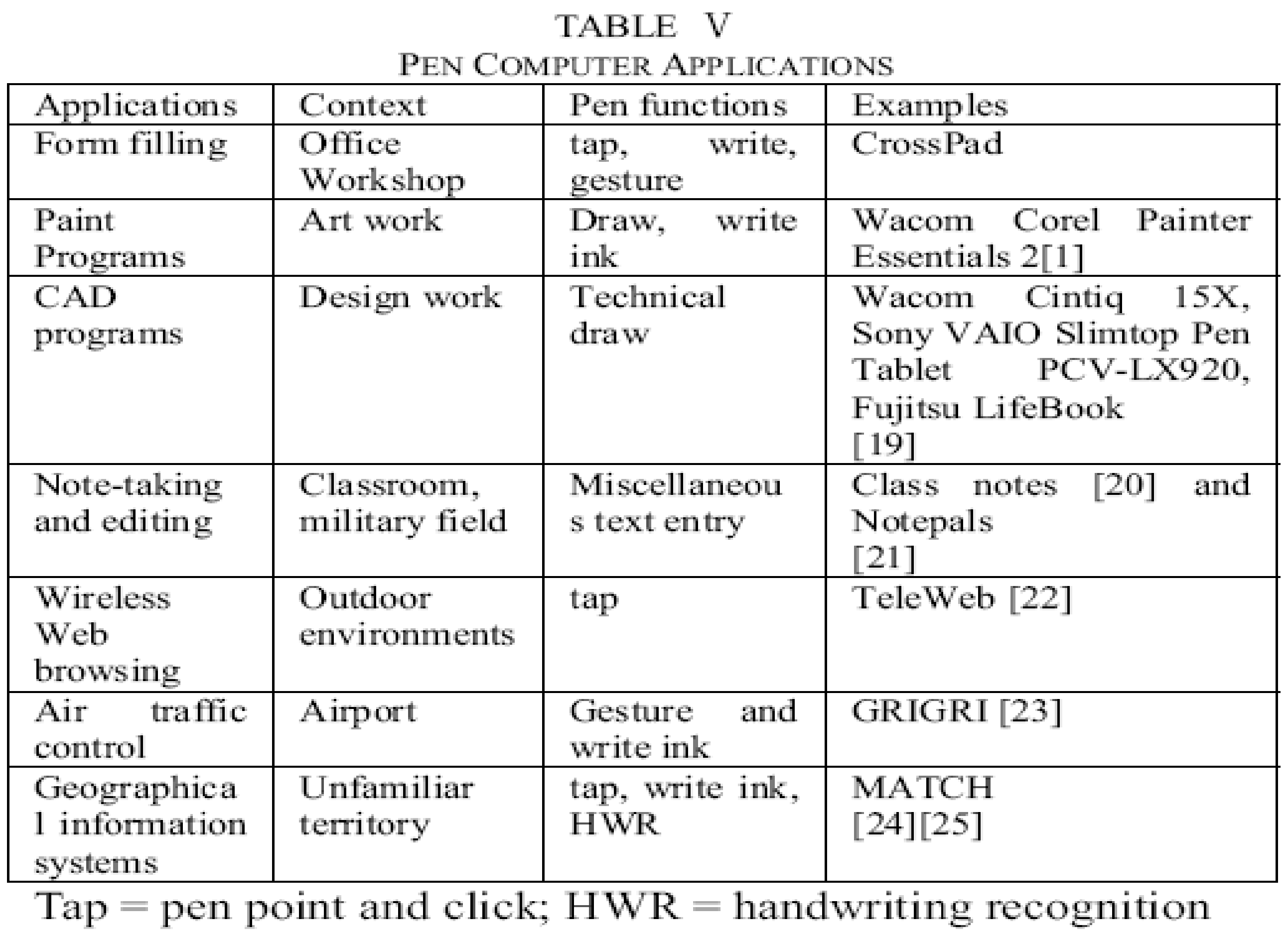


TABLE V
PEN COMPUTER APPLICATIONS

| Applications | Context | Pen functions | Examples |
|---|---|---|---|
| Form filling | Office Workshop | tap, write, gesture | CrossPad |
| Paint Programs | Art work | Draw, write ink | Wacom Corel Painter Essentials 2[1] |
| CAD programs | Design work | Technical draw | Wacom Cintiq 15X, Sony VAIO Slimtop Pen Tablet PCV-LX920, Fujitsu LifeBook [19] |
| Note-taking and editing | Classroom, military field | Miscellaneous text entry | Class notes [20] and Notepals [21] |
| Wireless Web browsing | Outdoor environments | tap | TeleWeb [22] |
| Air traffic control | Airport | Gesture and write ink | GRIGRI [23] |
| Geographical information systems | Unfamiliar territory | tap, write ink, HWR | MATCH [24][25] |

Tap = pen point and click; HWR = handwriting recognition

Pen Input = Distinct modalities enabled by a single device.

Most pen applications are "multi-modal" !

## Pen Input as Pointing

- Tap = Mouse Point and click
- Barrel buttons = Other mouse buttons
- Other capabilities like hover, etc can also be supported
- Often abstracted as a “mouse” device

## Pen Input as Data

- Writing or Drawing
- Uninterpreted, interpreted if needed
- May be

    Archived

    - e.g. Notes capture, drawing capture

        Windows Journal / OneNote, Adobe Illustator, DENIM, …

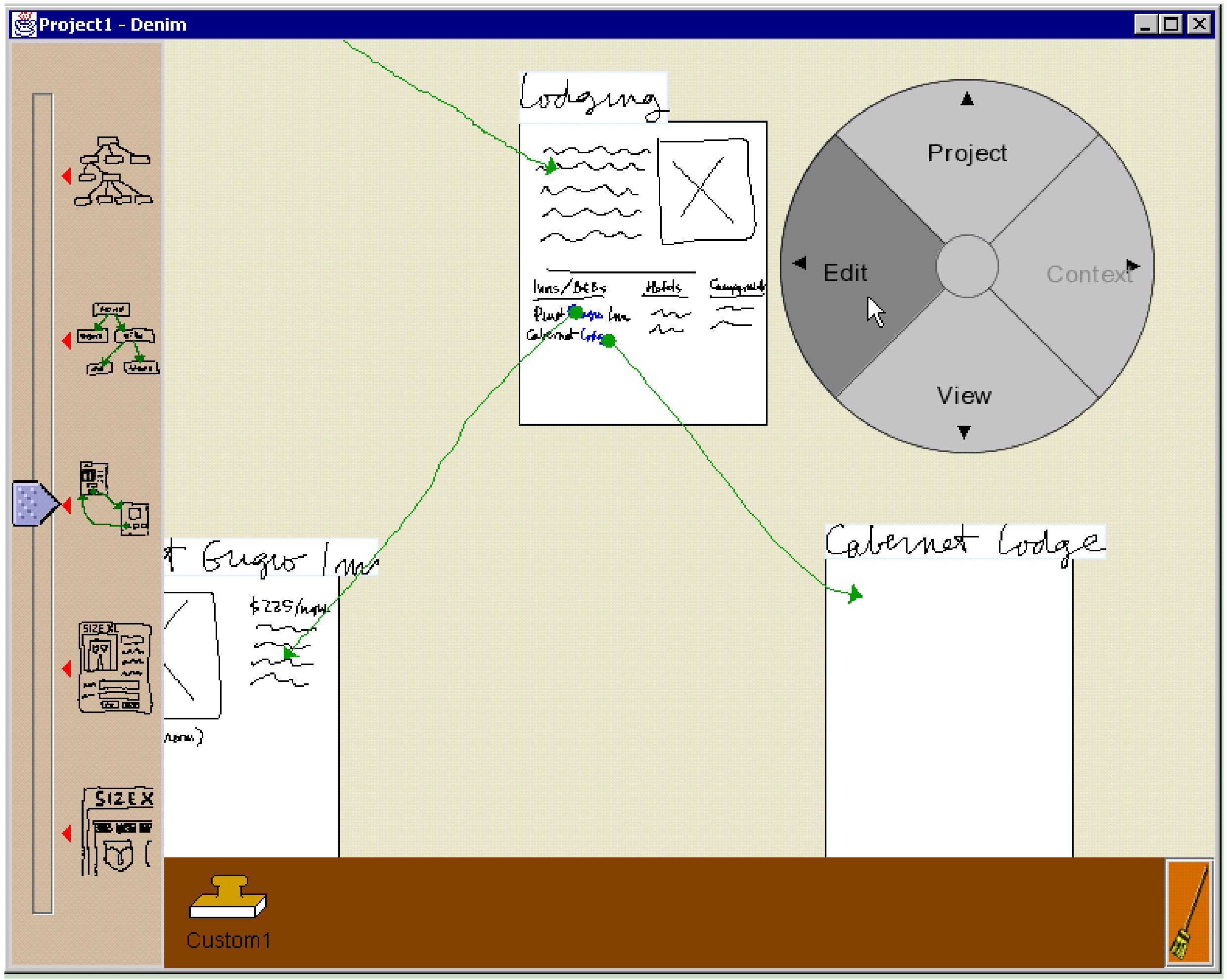
    - Signature capture

    Communicated

    - e.g. Ink Messaging

        Whiteboard sharing applications, Tivoli '93, SmartBoard, Windows Messenger on TabletPCs

## DENIM (Landey et al, UC Berkeley, 1999)



## Pen Input as Annotation

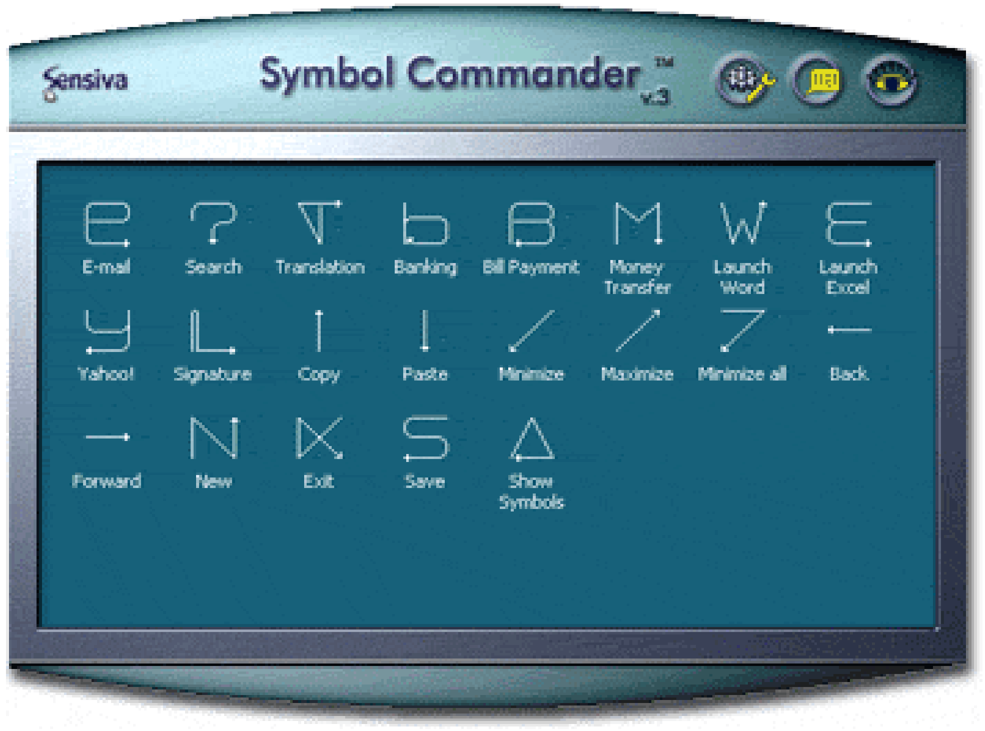
- Almost anything visual can be annotated with ink !
  - Images – photographs, medical images
  - Maps
  - Documents (Word & PDF)
  - Slides
  - Web pages
  - Video frames
- Could be
  - "inline" – writing/markings referring to specific content
  - "attached" – notes referring to content as a whole
- Difficult problem:
  - Figuring out what content is being referred to (even if the ink is not interpreted)

## Pen Input as Gesture

- Instruction to the system/application to do something
- Most popular use of pen input
- Generally application dependent – may also be user defined
- Often have context
    - Context of window
    - Context of content
- Requires gesture recognition

# Example: System Command & Control

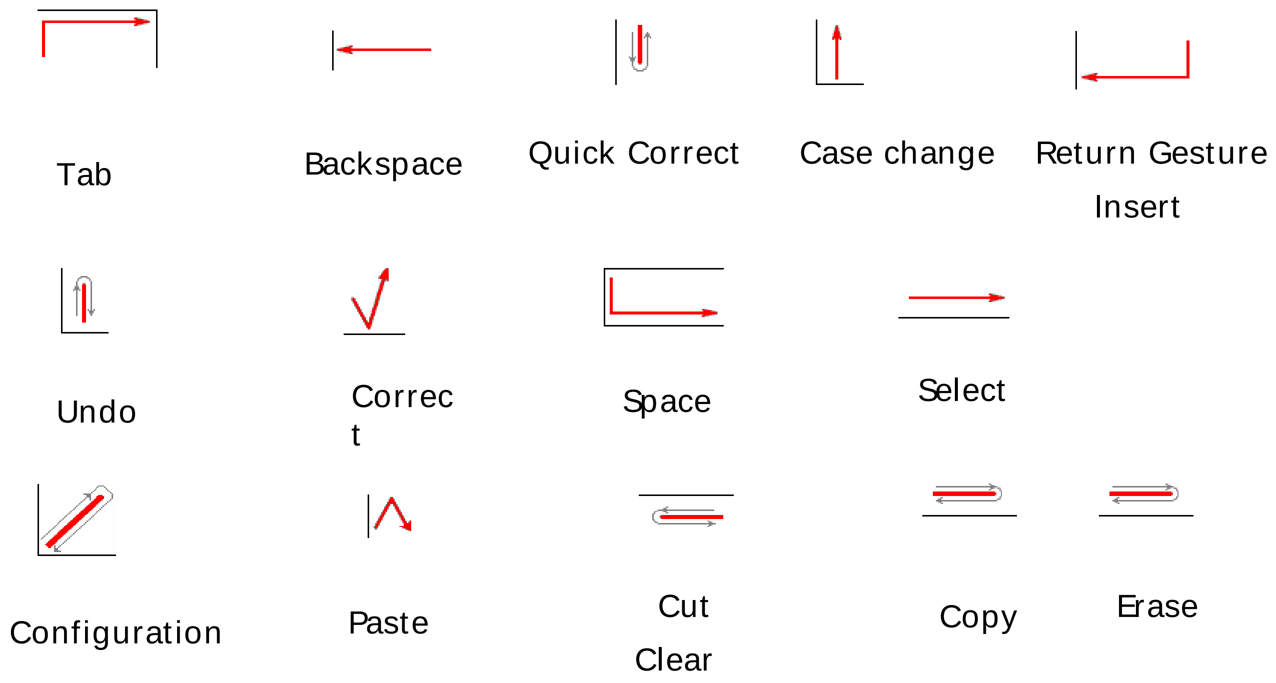
- Launch common applications
- Manipulate windows
- Perform common system actions (“Lock screen”, “toggle app in focus”)
- Perform common application actions (such as “Save” and “Exit”)



Source: Sensiva

# Example: Application specific Gestures

- Editing (word processor)
- Web browsing, e.g. Opera



## Controlling an Avatar

## Gesturing on Content

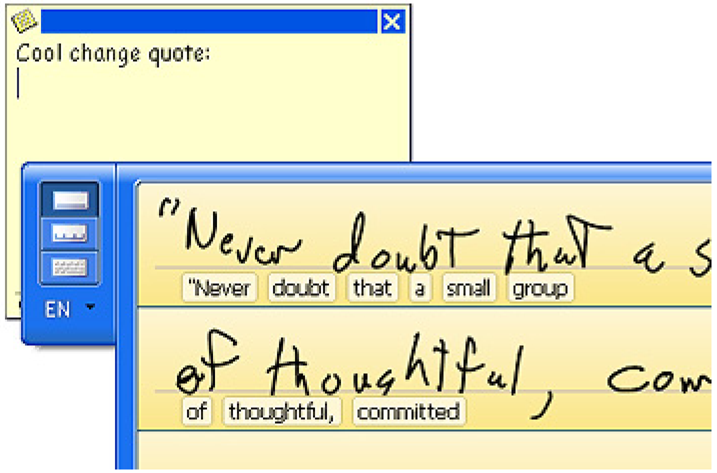
- A totally different experience
  - since gesture now is in a specific visual context
- Examples:
  - ticking a checkbox on a GUI or a printed form to select it
  - circling a city on a map to select it
  - gesturing a ? on a word in a browser to look it up in Wikipedia
  - striking out a word in a word processor to delete it
  - roughly circling a paragraph to select the entire paragraph
- Interpretation requires the context …

## Ink Recognition Systems

- Recognition of content
  - Text: handwriting recognition, simplified textual alphabets
  - Graphics, doodles, figures: sketch-based interfaces
- Recognition of commands
  - Specialized vocabulary of command symbols
  - Modal input of commands
  - Contextual commands: commands distinguished from content only in how they are used

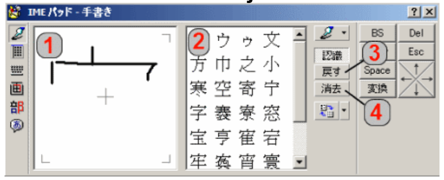
# Pen Input for Text Input

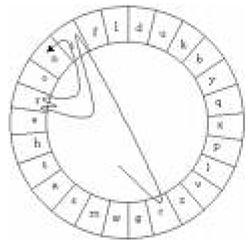
- Integrated or standalone IME
  - Pure handwriting recognition …
    - Requires GUI support for error correctio
  - Soft keyboards
  - Partial handwriting recognition
  - And everything in between !



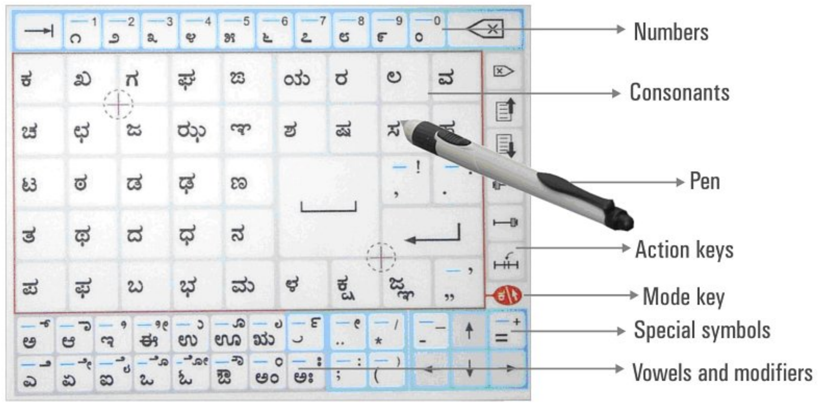
TabletPC Input Panel

Kanji IME - WinXP

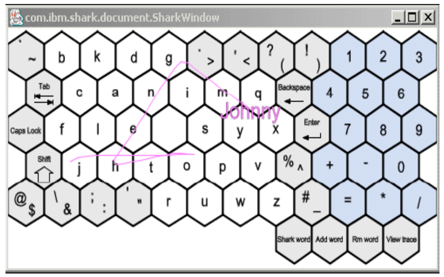




Cirrin



Gesture Keyboard (HP Labs)



Shark (IBM)

## Sketch-based Interfaces

- Interpret pen input as graphics, figures, objects …
- Creating flowcharts and slides
- Fashion design/Clothing virtual characters
- Searching image repositories

## The Fly Pen

- Uses pen as ink (interpreted but not replaced), gesture, mouse to enable:
    - Calculator
    - Algebra
    - Sheduler
    - Music
    - …
- Anoto digital pen and paper
- Recognition built into pen
- Embedded TTS

# Some Multimodal (-input) Pen Applications

- Ink + Speech
  - Note taking (Filochat ‘94, “I’ll get that off the audio”)
    - Modern avatar: LiveScribe ‘07
  - Natural situations: lectures, whiteboard use, brainstorming sessions, photo sharing

- Ink (drawing) + Speech Commands
  - Commands change brush attributes

## Some Multimodal (-input) Applications

- Gesture + Speech (Interpreted)
  - Maps - Put that there
  - GUI – controlling a smart home
- Writing + Speech (both interpreted)
  - Understanding multi-party, multimodal meetings
  - Automatic photo tagging from sharing sessions

## Integrating Pen Input into Applications: Tight coupling with Pen Data

- Multimodal application directly receives pen events from digitizer and decides how to interpret them
- Complex to build, maximum control
- Mode determination is a big problem
  - User makes an ‘O’
    - mouse movement ?
    - selection gesture ?
    - O or 0 ?
    - Leave as ink ?
  - Contextual inference
    - Simple: ink recd in form field is writing, ink on radio-button is gesture, ink on image is ink
    - In general, much more complex logic !!
  - Explicit mode selection
    - Barrel button, stylus inversion, key press (non preferred hand), GUI buttons, …

## Integrating Pen Input into Applications: Loose Coupling with “Pen Functions”

- Application does not directly interpret pen input
- Receives:
  - Mouse events from OS abstraction
  - Text events from IME applications
  - Generic commands (‘open’, ‘save’ etc) from “standard” gesture recognizer
- Where applicable, standard vocabularies of words, commands, etc
  - or grammar specified by application
- Enables ordinary apps to be pen-enabled
- Highly scalable, but no access to rich ink data

## Some New Issues

- What is a pen ?
  - Any device that can be moved to create a trajectory ?
  - How about 3D trajectories ?
    - Finger ?
    - Wii console ?
    - Mobile phones with accelerometers ?
    - Laser pointer ?

- Intersection with touch modality
  - Single-touch has many parallels to pen
    - And some differences – no fine control, different hardware

## Summary

- Pen Input has many aspects
  - mouse, ink (write/draw), input text/graphics, gesture, sign, …
- Pen Input can happen by itself or in visual context
  - GUI, Maps, Documents, etc … access to context is essential for recognition
- Tight coupling provides access to rich ink data; loose coupling via text/mouse events provides scale
- Mode switching between mouse, gesture, ink, and text is a key problem
- Shares characteristics with touch modality and 3-D trajectories

## Thank you