The Tablet PC: Designing Pen-based Applications

Richard Anderson
Slides derived from
Rob Jarrett
Microsoft Corporation
robjarr@microsoft.com

Topics
- Tablet PC introduction
  - What and why
- Usability issues
  - Hardware requirements
  - Display
  - Pen as mouse
  - Digital ink entry, editing, gestures, and recognition

What’s a Tablet PC?
- Pen-driven computer with various form-factors
  - Pure slate
  - Convertible
  - It's a full PC
    - Fast processor, lots of memory, hard-drive, high-resolution graphics
    - Runs Windows applications
    - "Laptop with a pen"

Why Tablet PC?
- Desktop and laptop computing doesn’t fit well with certain common situations
  - Meetings: laptops can be annoying and/or remove user from discussion
  - Relaxing (e.g. leaning back in chair, lying on couch): laptops must be on a flat surface
  - On-the-go (walking, standing waiting): laptops must remain statically positioned for use

Why Tablet PC?
- Using a pen is natural
  - Task adapted to human physiology rather than the other way around
  - Evolved over thousands of years
  - Easy
    - Most people can do it
    - Direct interaction/input
    - "Always on"
Why Tablet PC?

- Ink is a great data type
  - Expressive
    - Text, graphics, diagrams, etc.
  - Personal
    - ASCII text is "cold" :-)
- Free-form input
  - Write anywhere

- Hardware finally makes it doable
  - Small enough: 8.5” x 11”, under 3 lbs, less than 1” thick, good battery life
  - Cheap enough: shouldn't cost more than a nice laptop
  - Fast enough: real-time ink, handwriting recognition
- Software is good enough
  - Handwriting recognition doesn't suck
  - "Ink as ink"

Why past efforts failed

- Hardware was cumbersome
  - Units were slow and heavy
  - Screens were black and white, low resolution
- Handwriting reco was bad, but the usage paradigm needed it to be great
  - Often, ink was instantly converted to text, making bad reco obvious
- Usage paradigm was unnatural
  - Conversion to text often required
  - Little use of free-form input
  - Ink "gestures"

Why past efforts failed

- No standard software development platform
  - Low market penetration, tough business justification for 3rd parties
  - Therefore no "killer apps"
- Portable computers were niche
  - Networking wasn't common for PCs
  - Tradeoffs were significant in comparison to today (e.g. screens, upgradability, speed)

What about palm-sized devices?

- Great at what they do
  - Contacts, scheduling, jotting down short notes, etc.
  - Saving grace (IMO): synchronization with desktop PCs
- Form factor is awesome
  - Fits in your pocket or hand!

What about palm-sized devices?

- We still have to adapt to them
  - Small screens
  - "Graffiti" input
- Not good at desktop tasks
  - Unsatisfying for web surfing, email composition, document viewing, etc.
  - Syncing is a pain for some
  - Peripheral device for many users
Windows XP Tablet PC Edition

- Version 1.0 released November '02
- Superset of Windows XP Professional
  - Runs all apps XP Pro can
- Culmination of many years of work
  - Ink recognition software
  - End-user studies
  - Learning from the past
- Evolution rather than revolution

Windows XP Tablet PC Edition

- The goal: the simplicity of paper combined with the power of the PC
  - "Ink as ink" / "Ink as a first-class type"
  - Data lives life as ink
  - Editable, searchable, persistable
- Natural feel
  - Ink flows out of the pen quickly and smoothly
  - Free-form input
  - Pages of paper instead of infinite canvas

Windows XP Tablet PC Edition

- The goal: the simplicity of paper combined with the power of the PC (cont'd)
  - Using existing ("legacy") apps with the pen
    - Mouse input
    - ASCII text input

About the digitizer

- Want accurate ink: looks more "real", and has better recognition results
  - Therefore need high sampling rate with high resolution
- Must be low power consumption
  - Narrows range of technologies that can be employed
  - Electromagnetic is popular choice, but not without tradeoffs

About the digitizer

- Pen hovering capability
  - Important to connect the user interface with the pen even when it’s not touching the screen
- Capture other data from pen besides x,y position
  - Pressure, tilt, rotation, roll, etc.
  - Great ink and data manipulation

About the digitizer

- Tradeoffs: sensitive to interference
  - Hard drive, CPU, battery, and other components can alter where the digitizer thinks the pen actually is
  - Calibration system (i.e. software correction) helps tremendously, but still not perfect
Display hardware

- User can’t write directly on the LCD surface
  - Psychedelic color blooming occurs because of squishing liquid crystals; very distracting
  - Bad for the display
- Solution: glass overlay
  - Doesn’t allow any “give” across the display
  - Protects the LCD

Tradeoffs:

- No “give” means it feels unnatural
- Glass is slippery to a hard plastic pen
  - Pen skids a little, making writing and targeting a bit more difficult
- Parallax
  - Thickness of glass causes visual disconnect from ink/cursor/etc. when pen tip touches the display
  - These will get better

Pen/stylus design

- Very personal piece of hardware!
  - People play with it, chew on it, etc.
- Should be as close to a ballpoint in size and weight as possible
  - Pocket clip is a good thing too, even if people don’t use it for their pocket
- Needs rugged design
  - Most people on the tablet team have broken a pen because the design was fragile

- Pen tip
  - Some pens have their tip act as a momentary switch so the digitizer knows when the pen is touching vs. hovering – feels strange
- Pen button
  - Very useful trigger for non-ink functionality
  - Right-button click, erasing, selection, etc.
  - Some designs are very prone to accidental clicks by users
    - Causes undesirable behavior – very frustrating!

High DPI displays

- Displays are small but high resolution – increased DPI
  - Ink and eBooks look great
- Tradeoff: Legacy apps suffer
  - Many hard-coded pixel sizes, meaning text, icons hard to see
  - Targeting areas get smaller
  - Can only expect evolution here; platform makes this a pain today, tomorrow will be easier

Portrait-mode display

- Support portrait mode; just like paper
  - Great for web surfing, reading eBooks and most other document types
  - Hot-switch to landscape and back is great for convertibles
- Tradeoff: Legacy apps suffer
  - All written assuming horizontal > vertical resolution
  - Toolbars, menus, etc. can be cut off
Pen as mouse: Hovering

- Hovering still with a pen is tough
  - While in-air, our control of a pen is considerably worse than when the pen is touching a surface
  - Mouse is intrinsically static, pen is not
  - Legacy applications typically assume cursor must be perfectly still for e.g. tooltips to appear
  - Software help needed to "smooth" hover location of cursor

Pen as mouse: Hovering guidelines

- Features requiring hover should have generous tolerances
  - For instance, tooltips
  - Use COMCTL32 provided ToolBars
  - Test hover-triggered features for ease of use

Pen as mouse: Obstruction

- Viewing obstruction
  - Pen is a direct pointing device and the hand covers screen; mouse does not
  - Menus and tooltips display towards the right
    - Most people are right-handed!
  - Lefties have it tough with scrollbars

Pen as mouse: Obstruction guidelines

- Ensure state changes do not occur under the hand
- Ensure that consequences of actions do not appear under the hand
- Respect user handedness system setting
  - Check SPI_GETMENUDROPALIGNMENT via SystemParametersInfo()
  - Apply it to Tooltips, menus, popup menus

Pen as mouse: Targeting

- Because hovering still with a pen is tough, targeting is tough
  - Operations such as resizing windows, selecting ASCII text, clicking push buttons, checkboxes, scrollbar buttons, etc. were designed for the mouse
  - Downward motion of pen alters x,y location of cursor
  - Very frustrating for users to "miss"
  - Software help would have great benefits

Pen as mouse: Clicking

- Clicking with a pen is tough
  - Legacy applications typically assume during a click the mouse doesn't move
  - Pen taps are more like little strokes or stabs because of pen skidding and high-precision digitizers
  - Detecting the difference between tap and a drag is an interesting problem!
  - Double-clicking is even tougher
    - Quick motion means sloppier result
Pen as mouse: Pen positioning

- Pen positioning can be arduous work if UI requires targeting all over the display
  - Mouse can move the cursor far without much arm/hand movement, but the pen requires a lot of arm/hand movement
  - Menus and toolbars are typically at the top of a window; editing often occurs mid-way or toward the bottom
    - Lots of physical arm/hand movement results – a real pain for users
    - More local UI is desirable (e.g. context menus)

Pen as mouse: Targeting guidelines

- Cursor feedback
- Bigger, easily-targeted controls
- Generous tap, double-click, and hover tolerances
- Keep related objects in proximity

Pen as mouse: Right-clicking

- Need to be able to right-click with the pen
  - While not used by majority of Windows users, still an important capability for backwards compatibility and contextual UI
  - Solutions: “Press-and-hold” and pen barrel-button invocation
    - P & H was fairly controversial because some felt it got in the way

Digital ink realism

- Ink should look smooth
  - No “jaggies” -> antialiased
  - No straight lines -> curve-fitted
- Use pen pressure information
  - Vary stroke width (more pressure means wide stroke)
- Support pen tips
  - Round/ballpoint vs. rectangular/ highlighter

Digital ink performance

- Writing requires uninterrupted inking
  - Users have difficulty with delays in ink appearance
  - Users are frustrated with delays in inking
- Guideline
  - Ensure fast efficacy
  - Is it as fast as writing on paper?

Pen modes & cursors

- Explicit vs. implicit input modes
  - A.k.a. Modal vs. modeless
- How to allow things to be efficient but not confuse users
  - Select mode uses “right-mouse button” for implicit mode as well as utilizing an explicit mode
  - Erase mode uses pen’s eraser tip (if available) for implicit mode as well as utilizing an explicit mode
**Pen modes & cursors**
- Need feedback as to the mode of the pen
  - Indicates actions available to the user
- Guideline
  - Develop a set of cursor feedback to indicate the different modes of the pen
  - Careful attention to cursor design
    - Either symmetric or use handedness setting

**Pen gestures**
- Gestures need precise tuning
  - Trade-off between accidental activation vs. not getting when wanted
  - False activations are annoying and distracting to the task!
- Guideline
  - Use gestures guardedly
  - Error on the side of having “zero” incidence of false activation
  - Non-destructive consequences are better

**Writing location**
- Users have an initial expectation that they can write digital ink anywhere
- Guideline
  - Communicate clearly where users can ink
  - Ink enabled controls should be self-evident

**Ink selection**
- Traditional rectangular selection tools are inadequate
- “Lasso” selection is much more natural
  - Percentage-based stroke tolerances
  - Employ word-based selection
  - Visual feedback is essential, real-time is much better than static

**Recognition expectations**
- Handwriting recognition is highly variable by person
  - Errors are expected
  - Perceived good or bad handwriting effects expectation of accuracy
- Guideline
  - Be realistic about recognition accuracy rate, don’t rely heavily on it for authoring