Predictable pointer acceleration

- Overview
- The problem
  - In theory and practice
- The solution
  - Selected details
- Impact
  - Guidelines for input drivers
- Outlook

Ad-hoc census

- Who noticed a change in pointer behaviour ?
- Who changed settings in response ?
- Who even switched profiles or did other experiments ?

From 10.000 feet

- X pointer acceleration previously
  - Very simple
  - Often seen as inadequate
  - It 'feels bad'
- longstanding issues
  - No scaling in dix
    - Leads to driver side scaling
      - distributed buffers
  - Parallel acceleration (synaptics)
  - Sometimes overshoots

➔ One could do better

The problem - in theory

- Useability depends on predictabililty
- The brain knows velocity, the computer knows mickeys
  - Mickeys and velocity correlate
  - (but that's pretty much all there is)
- With acceleration, there's a disconnect
  - The X user is forced to learn how his mouse generates mickeys

➔ Need to restore the feedback loop

  ➔ Talking about the same thing is a good start

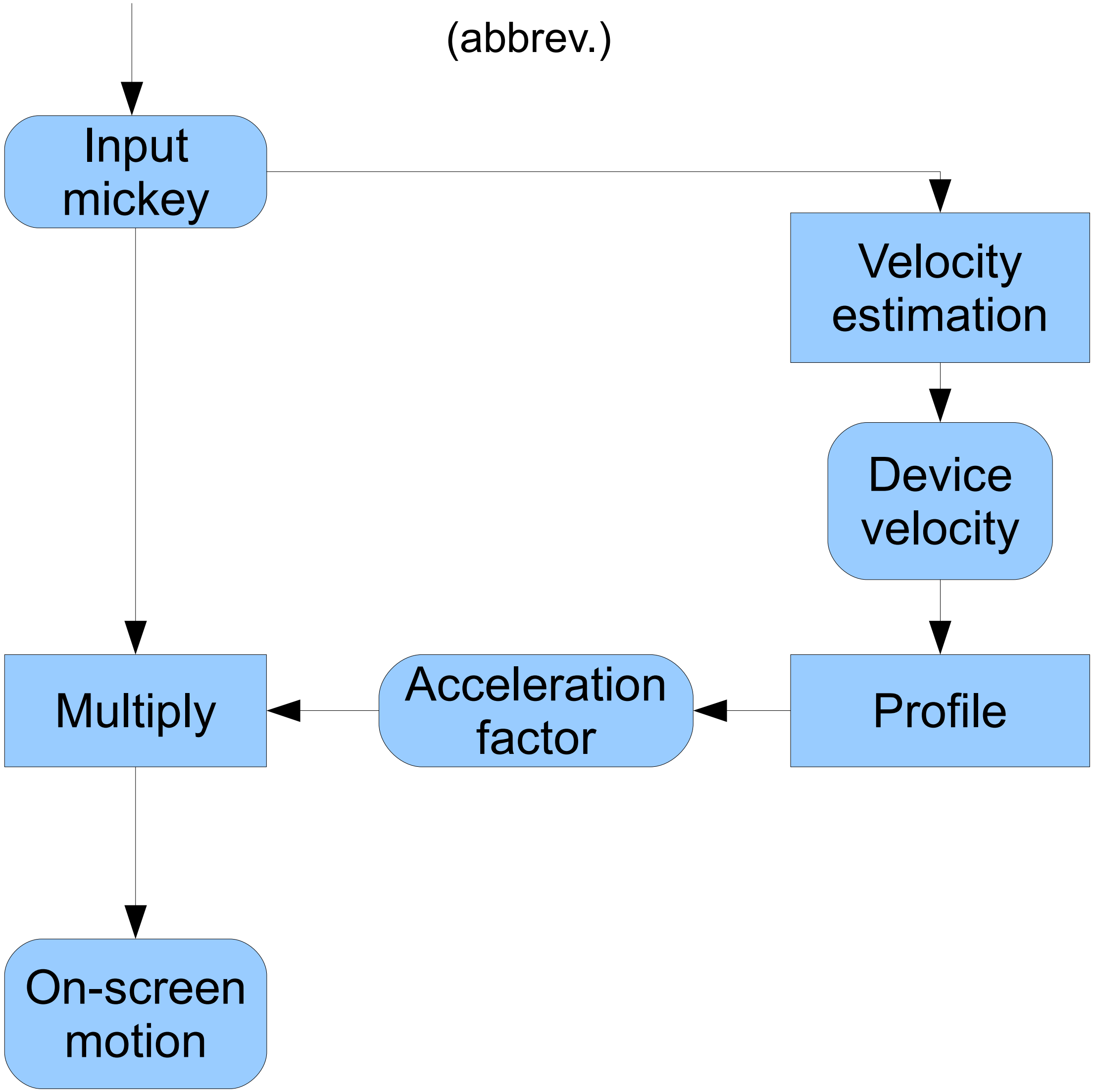
  ➔ users should have more control

The problem - in practice

- Mickeys just don't suffice
  - Mickey is [L], velocity is [L*T^{-1}]
  - Dynamic range is very low
    - Slow motion: ~ 1:3, uneven
      - 'High-Performance' devices: trade dynamic range for responsivity
    - Faster: ~1:15
  - Blocked X jeopardizes mickey
- Resulting acceleration varies
- ➔ We need a proper velocity
  - ➔ Have it or fake it

Data flow

(abbrev.)



Input mickey

Velocity estimation

Device velocity

Profile

Acceleration factor

Multiply

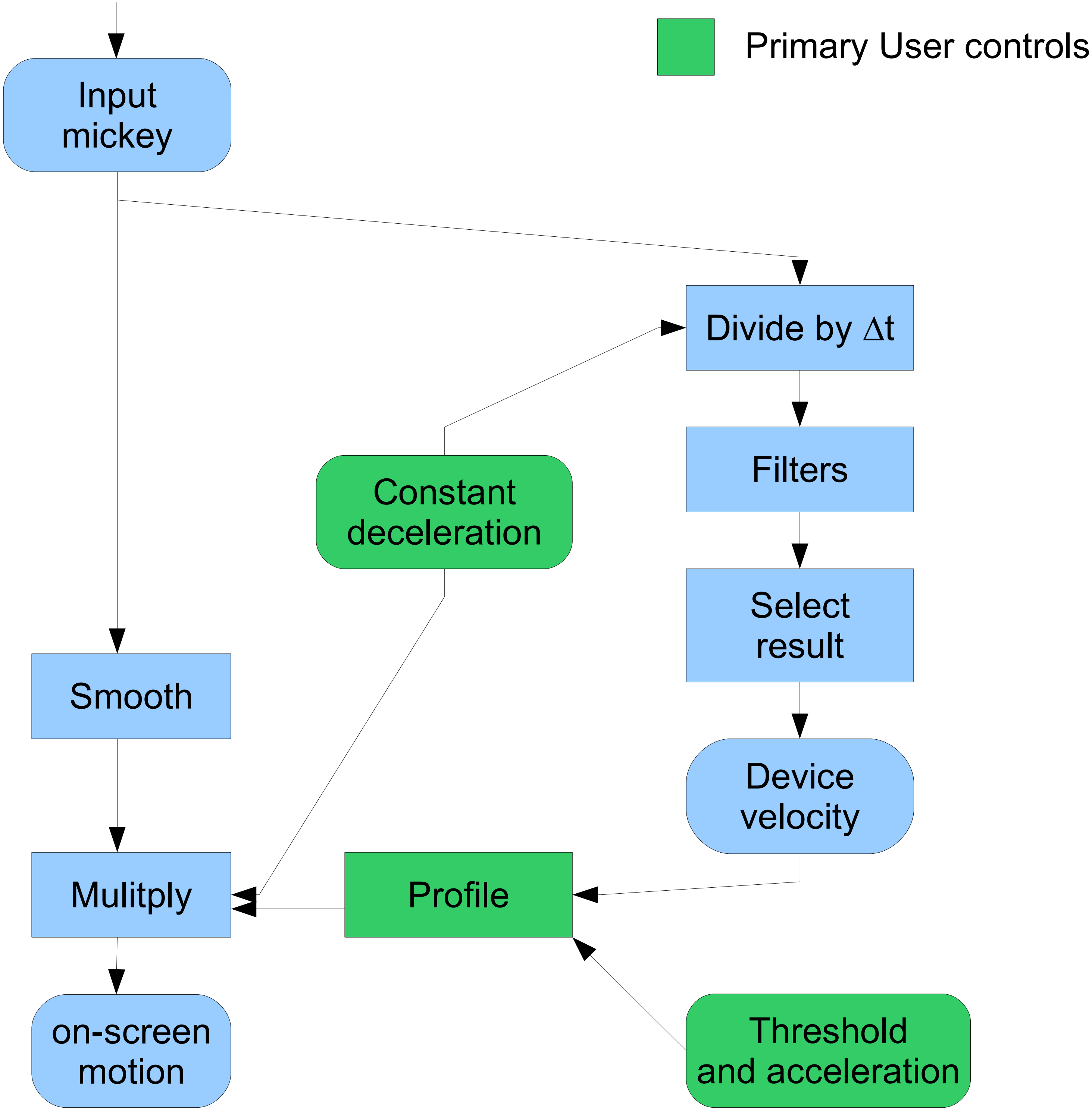
On-screen motion

From mickey to velocity

1) Divide by delta time
   - ✓ Great for estimating slow motion
   - ✓ Bumps dynamic range - 1:50 easily
   - – Still very dependent on individual Mickeys
   - – Creates need to scale estimate
     - Velocity is pixel per scale milliseconds
2) Tracking velocity with filters
   - ✓ Even and dynamic velocity
   - – responsivity
   - ➔ 'Good estimation' becomes 'good filter setup & selection'

Velocity tracking

- Multiple filters
  - Short half-life: tight tracking
  - Long half-life: smooth 'average'
  - Better stability by design
- Select good filter by divergence
  - Details may change
- Sometimes override filters (coupling)
  - Responsive
  - Good compromise esp. for 1 filter
  - Responsive to noise too



🟩 Primary User controls

Input mickey

Divide by Δt

Filters

Constant deceleration

Select result

Smooth

Device velocity

Mulitply

Profile

on-screen motion

Threshold and acceleration

Velocity and then ?

- Profiles
  - Translate device velocity to acceleration factor
  - To be chosen on individual preference
  - Should be smooth to be intuitive
    - Previously they weren't
- Adaptive deceleration
  - Great for precise pointing
- Constant deceleration
  - better adapt to a large device range

Impact

- Scaling in drivers considered harmful
  - Except to suppress errors
  - Better postpone scaling to avoid multiple independent buffers (remainders)
  - precision otherwise unavailable
- API allows to coordinate on scaling or acceleration
  - it's not neccessary for a driver to benefit
  - Main use: postpone scaling
  - driver-specific profile
    - Pressure or other sensor input

Outlook

- Expose device properties
  - Cool UI stuff
  - Upload user-defined profiles
- More numerical stability
  - Change default acceleration
- Accelerate e.g. Z axis
- velocity and sub-pixel position
  - Make some sense now
  - could be of use down the chain
- Move more transforms into dix
  - AngleOffset