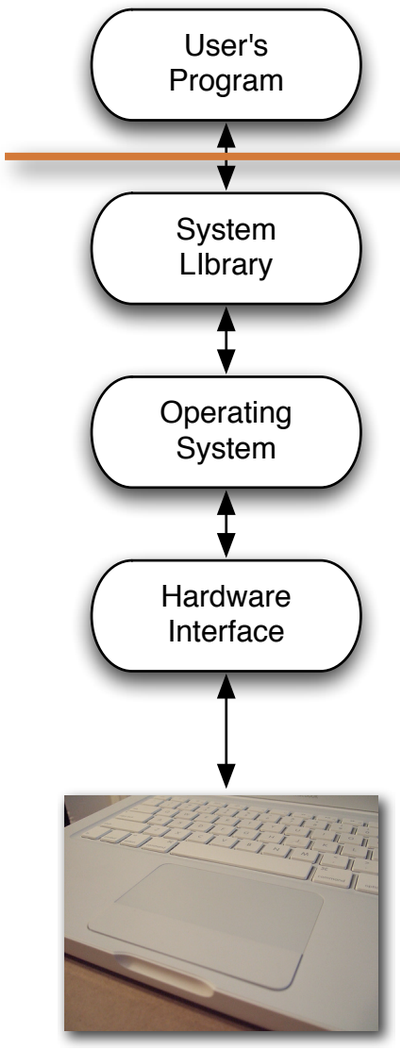
User Interaction: Intro to Multi-Touch Tools

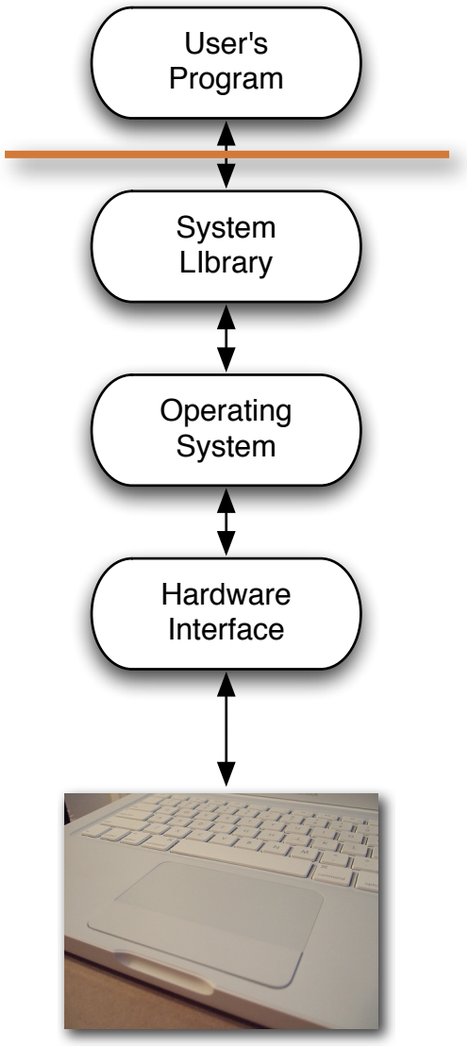
Asst. Professor Donald J. Patterson
INF 133 Fall 2010

Multi-Touch Approach #1



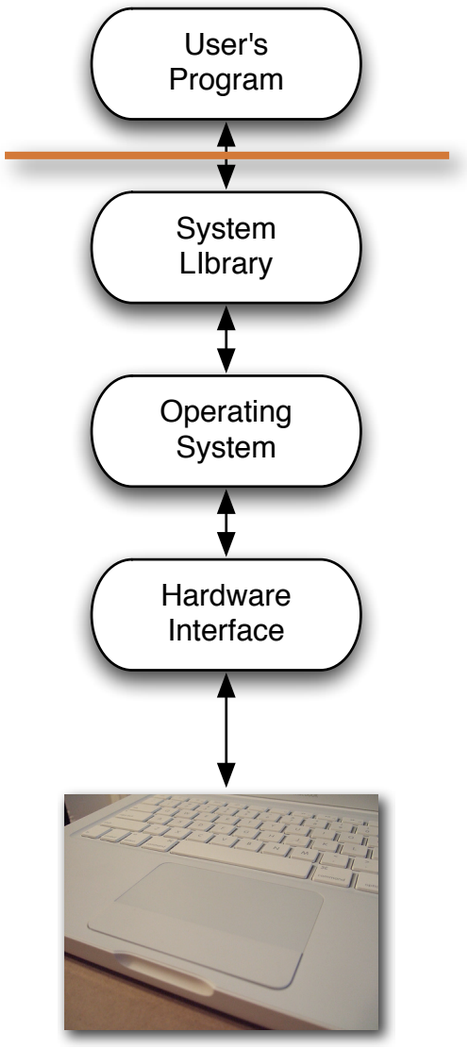
- Design specific multi-touch/gesture events that you can register for:
  - Pinching movements (in or out)
    - meaning zoom out or zoom in
  - Rotate: Two fingers moving in opposite semicircles is a gesture meaning rotate.
  - Swipe: Three fingers brushing across the trackpad surface in a common direction.
  - Scroll: Two fingers moving vertically or horizontally is a scroll gesture.

Multi-Touch Approach #1



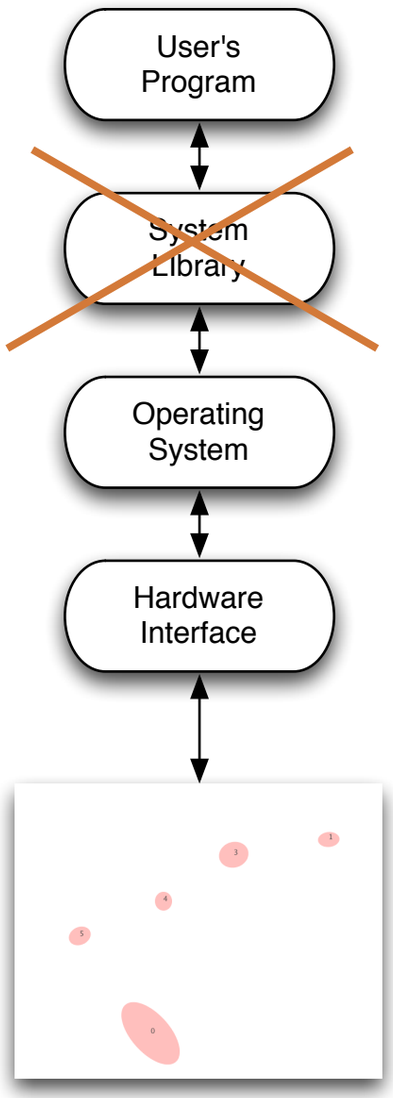
- Advantages:
  - Simple to code
  - Library/OS does all the work
- Disadvantages
  - No flexibility
  - Limited to supported events

Multi-Touch Approach #1



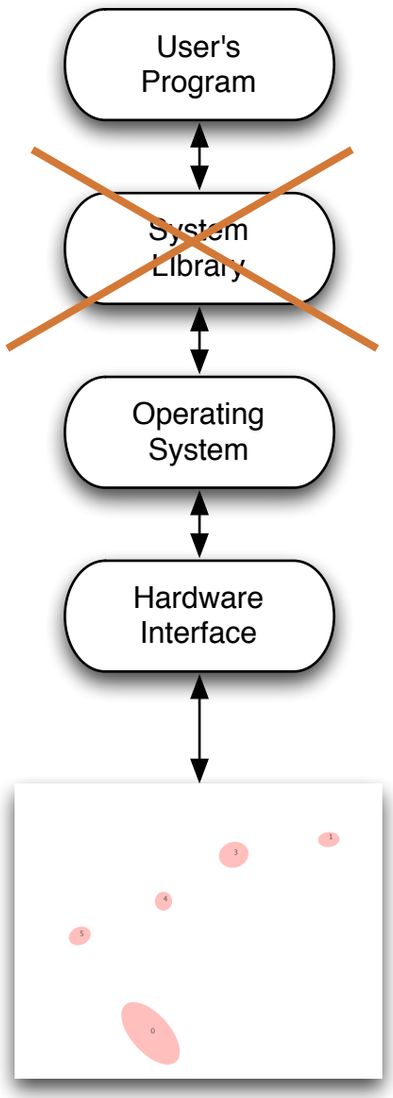
- Examples (demo):
  - Document browsing in Preview
    - Zoom
    - Scale
    - Swipe

Multi-Touch Approach #2



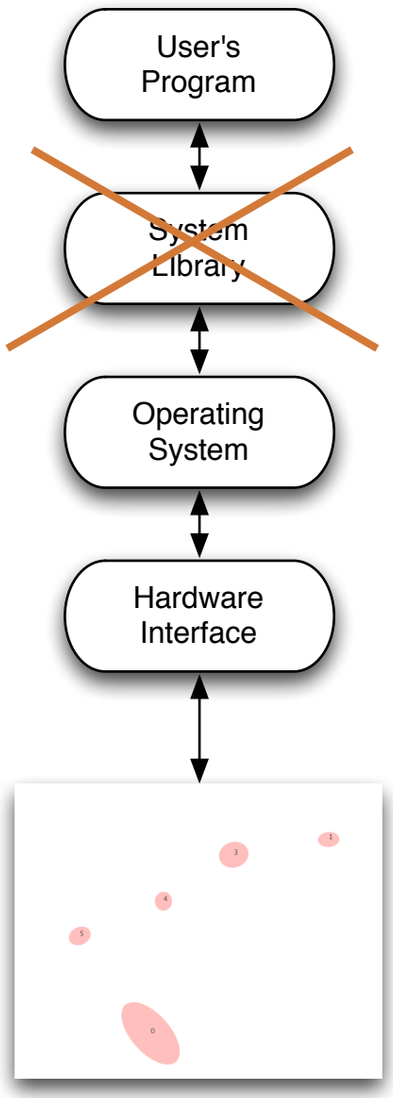
- Blob tracking by program
  - A program receives information about the location/"pressure"/orientation of multiple touches
  - Each touch gets an id to uniquely identify it
  - This is a stream of data
    - continuously updating locations and ids

Multi-Touch Approach #2



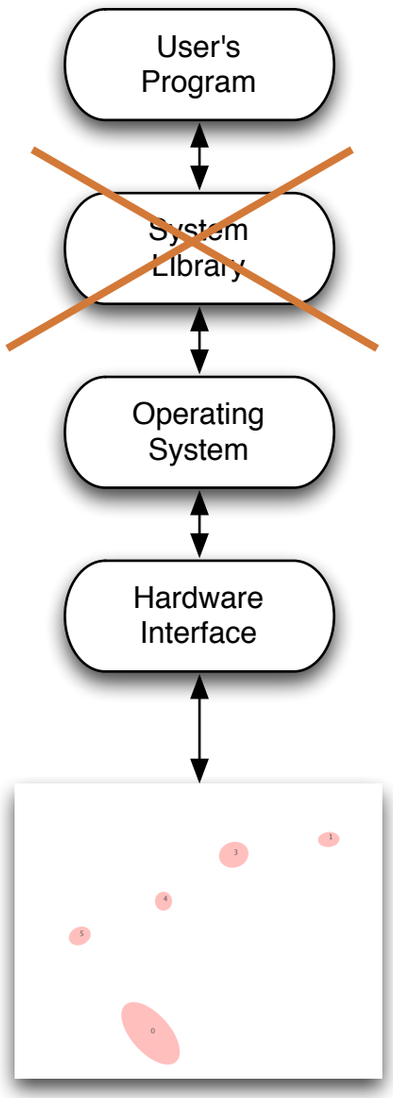
- Advantages
  - Supports unlimited numbers of touches
    - two hands / multiple people
  - Programs can have gestures that make unique sense for them
- OS does a lot of work to find and report blobs

Multi-Touch Approach #2



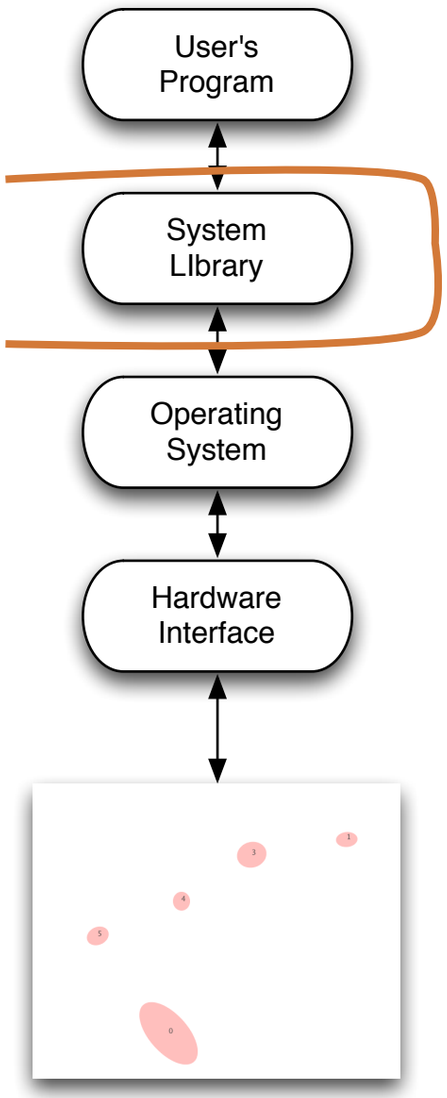
- Disadvantages
  - Each program has to figure out all events itself
    - Was that a pinch?
    - Was that a rotate?
    - Where is the thumb?

Multi-Touch Approach #2



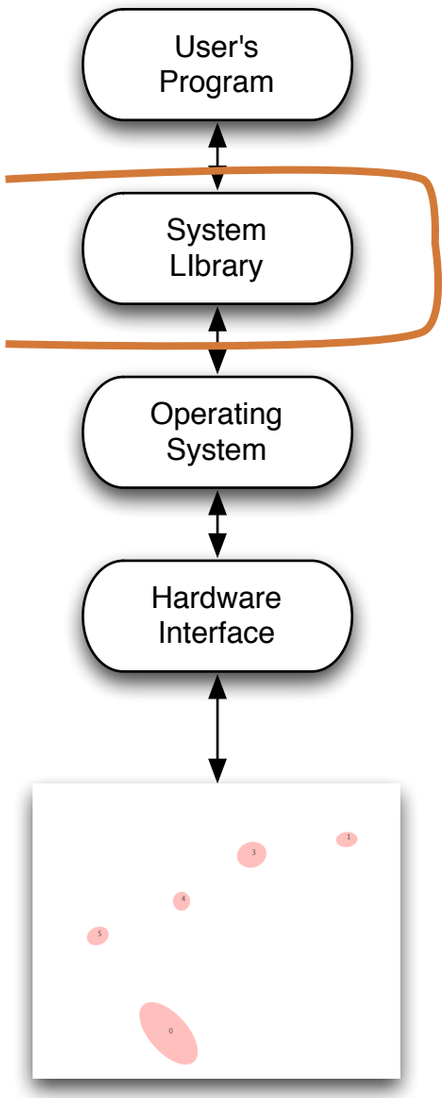
- Examples
  - MacMultitouch Demo
  - FingerMgmt

Multi-Touch Approach #3



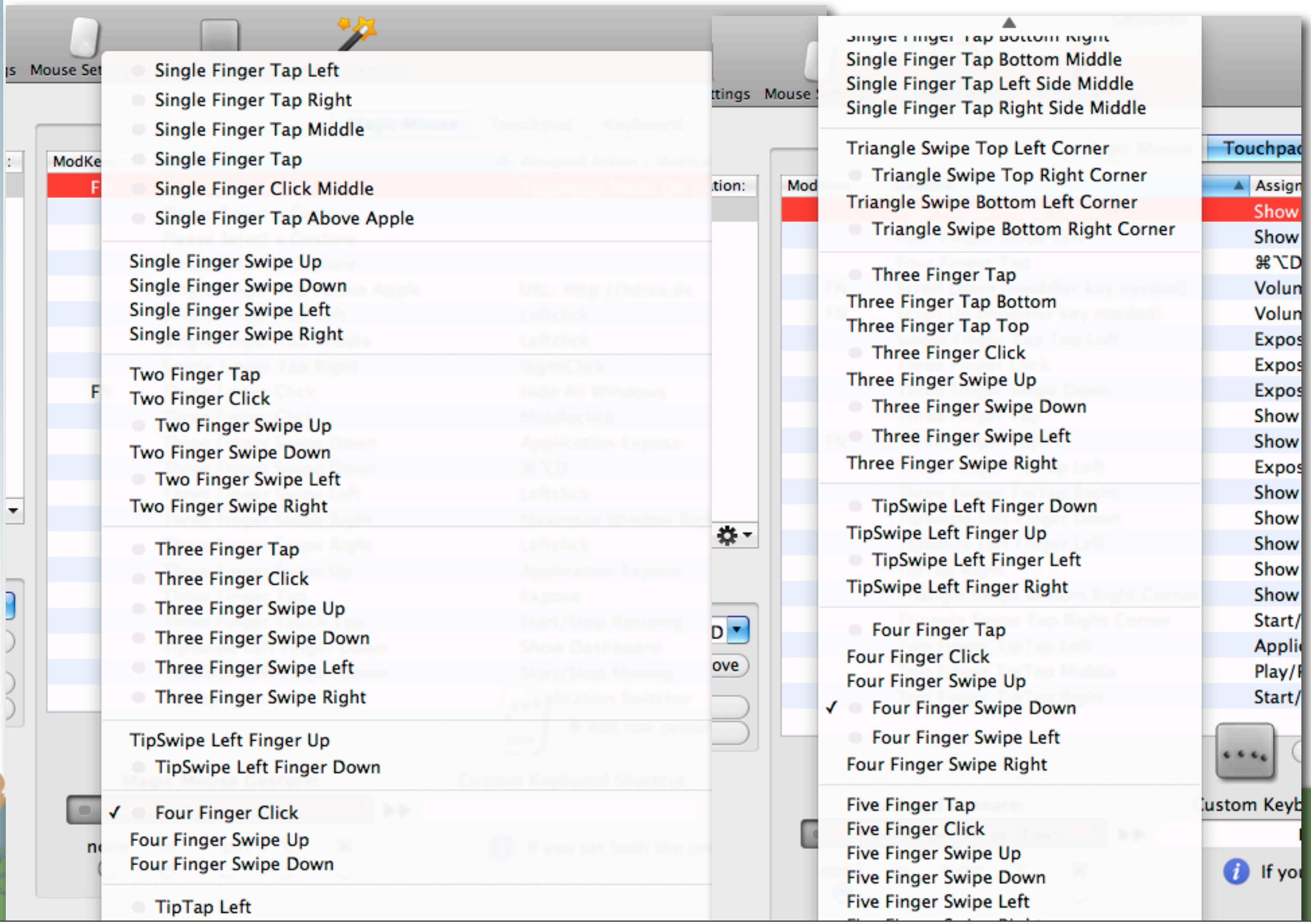
- Create your own event layer
  - Everyone wants to detect triangle touches
  - Everyone wants to interpret for multiple people
  - Everyone nees a “tiptap” interaction

Multi-Touch Approach #3

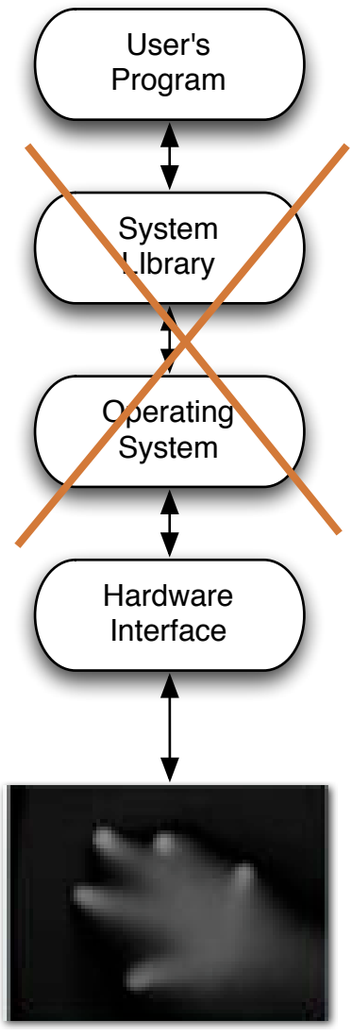


- Advantages:
  - Scalable (Other people can use it)
  - Allows completely new interface design
    - “3-finger pinch”
  - Lots of potential for innovation
- Disadvantages
  - Lots to code
  - Limited application support

Multi-Touch Approach #3: Better Touch Tool (http://boastr.net/)

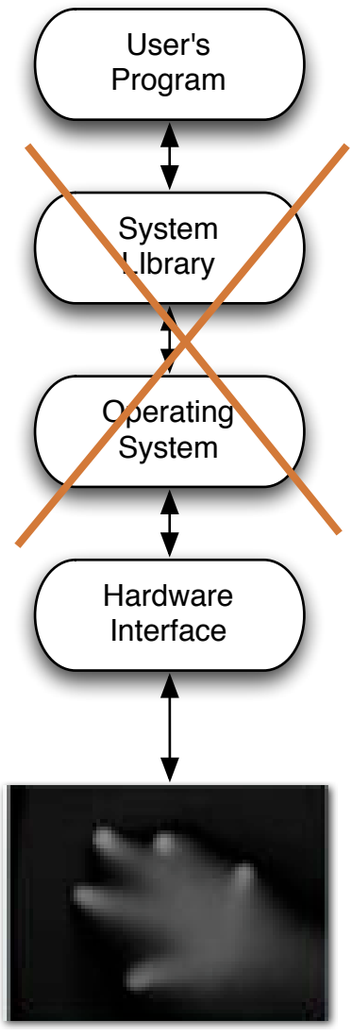


Multi-Touch Approach #4



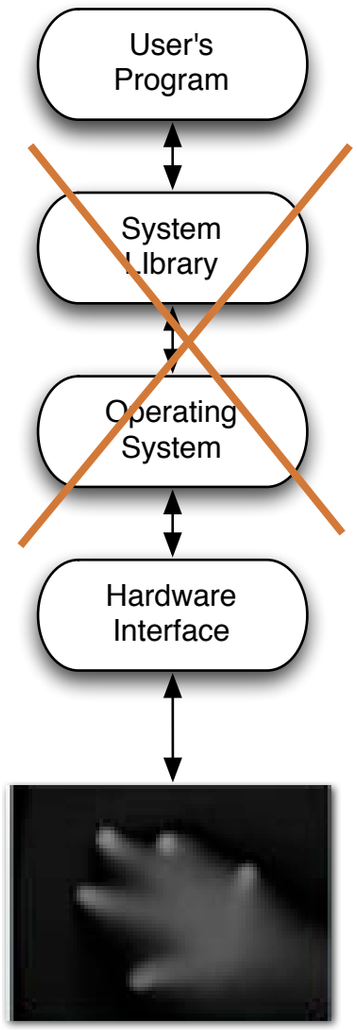
- Grayscale input
  - A program receives a stream of images
  - Darker (or lighter) colors indicates pressure or proximity

Multi-Touch Approach #4



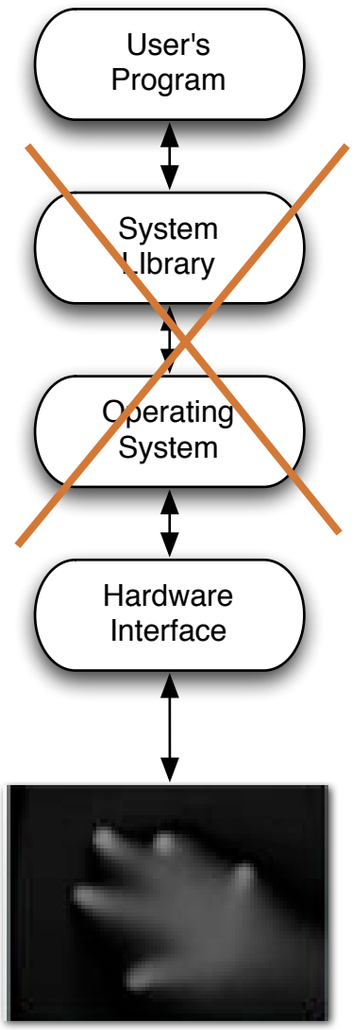
- Advantages
  - Maximum flexibility
  - Not restricted to “finger touch” paradigm
    - Can recognize a “cup down” event for example

Multi-Touch Approach #4



- Disadvantages
  - This is full-fledged computer vision
  - Different technologies generate different quality images
  - Robustly and consistently recognizing events is hard.

Multi-Touch Approach #4



- Examples
  - iShred
  - http://www.youtube.com/watch?v=eZpnzzKbY2I&feature=player_embedded

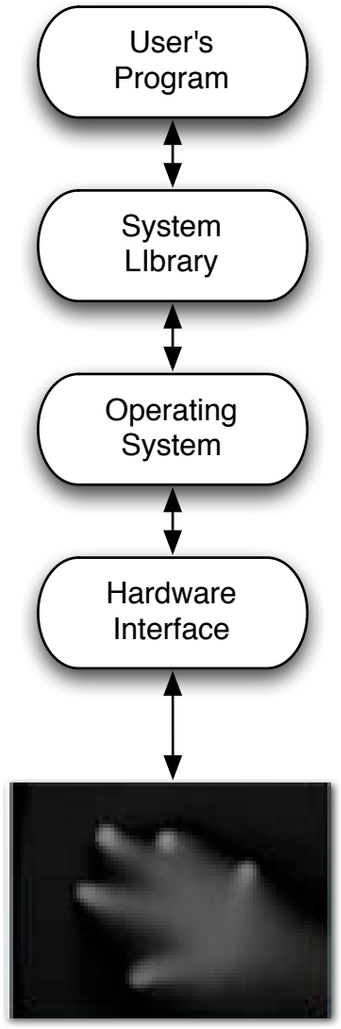
Microsoft Surface

- http://www.microsoft.com/surface/en/us/Pages/Product/WhatIs.aspx

How do you choose?

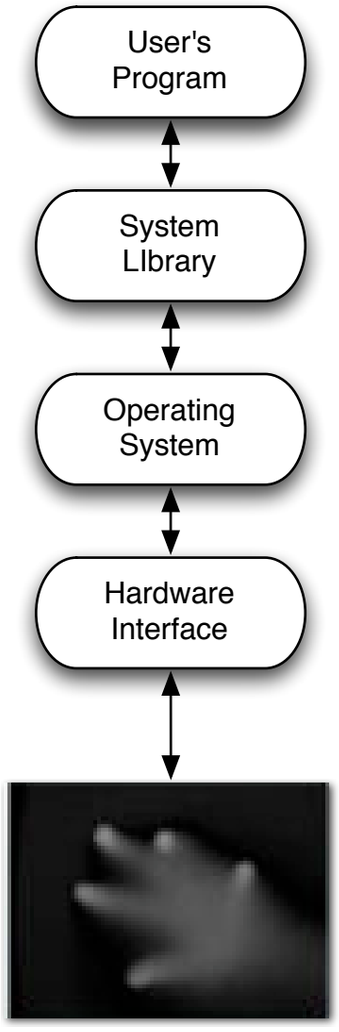
- How fast do you need to get your application done?
  - #1 is fastest, #4 is slowest
- Who are your users?
  - #1 is the most familiar to users, #4 requires users to adapt
- What is your application?
  - #1 is basically point and click extensions
  - #4 supports crazy gaming/applicatinos
- Are you showcasing multi-touch? or supporting a task?

Our assignment



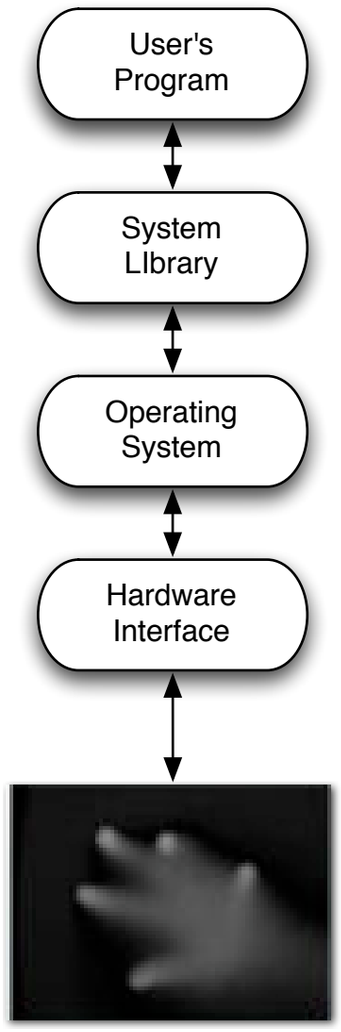
- Build a multi-touch Java paint application
  - No OS support

Our assignment



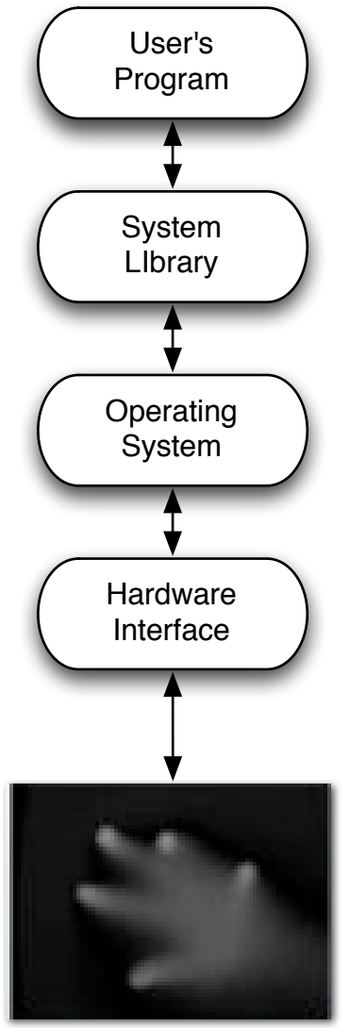
- Where are we going to get a grayscale input?
  - You can build your own
  - You can use prerecorded video

Our assignment



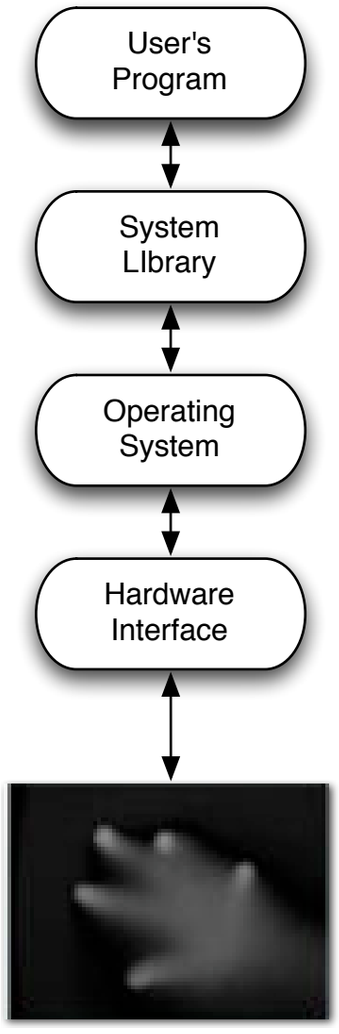
- How will we interface to the computer?
- Use standard camera inputs

Our assignment



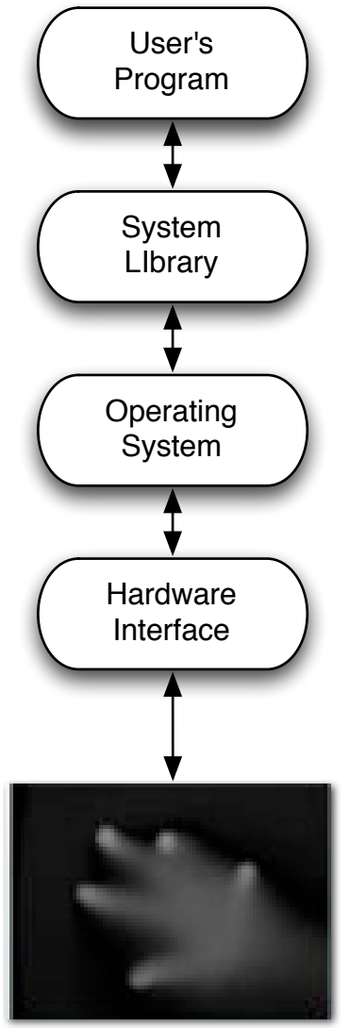
- How will we process it without OS support?
  - We will use Community Core Vision to process the grayscale images

Our assignment

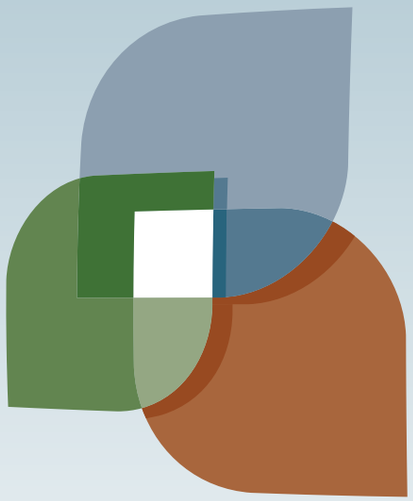


- How will will our application get information about multi-touch events?
  - Using the TUIO standard and a TUIO library for java

Our assignment



- How will will I write a multi-touch application?
  - Register for multi-touch events and then respond when you receive them.



LUCI

Friday, October 29, 2010