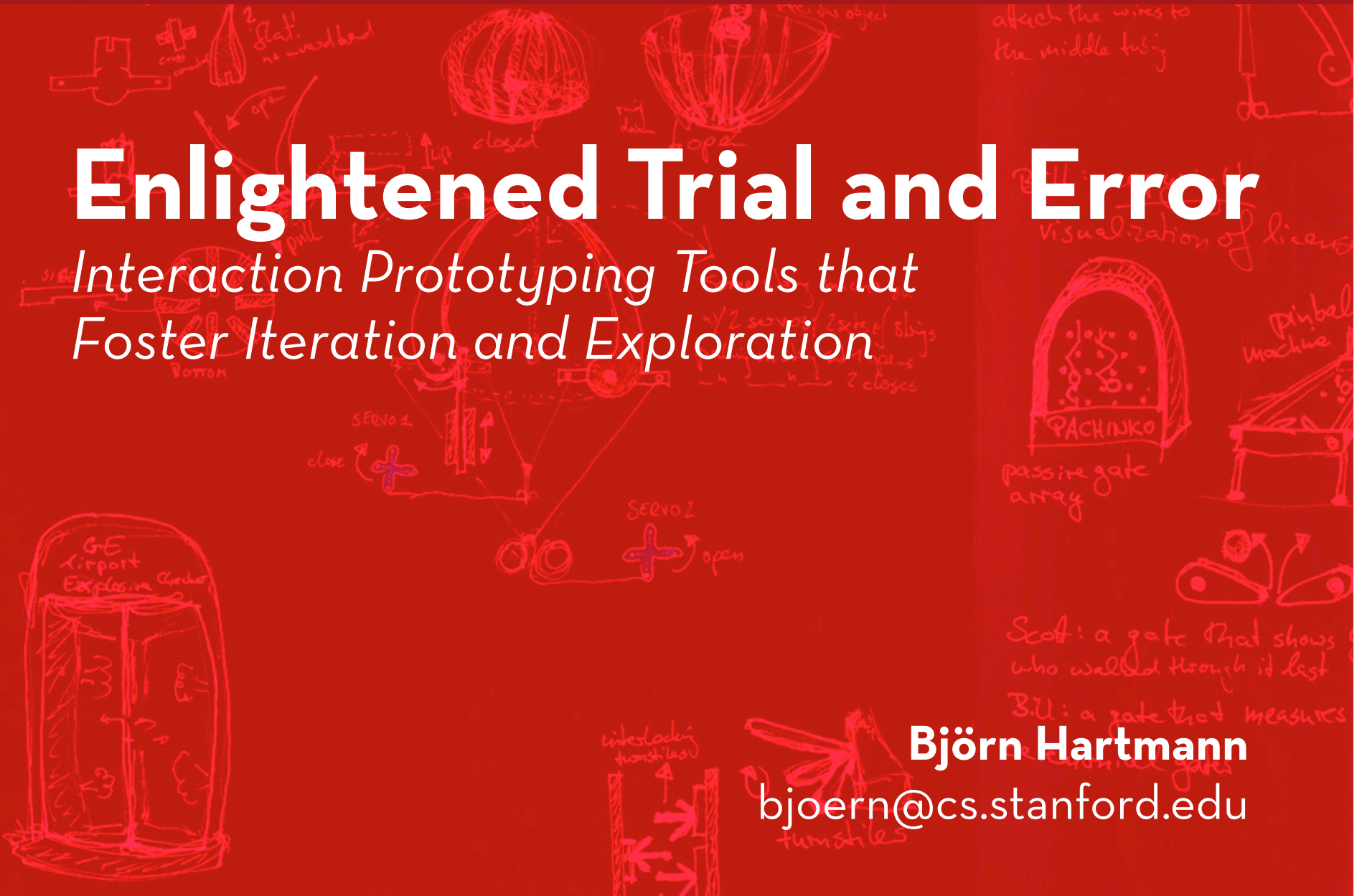


Enlightened Trial and Error

Interaction Prototyping Tools that Foster Iteration and Exploration

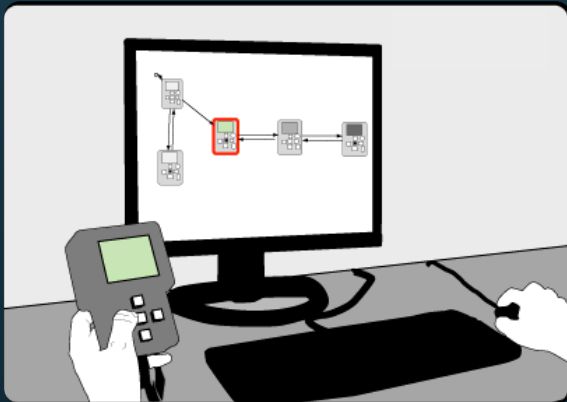
Björn Hartmann

bjoern@cs.stanford.edu

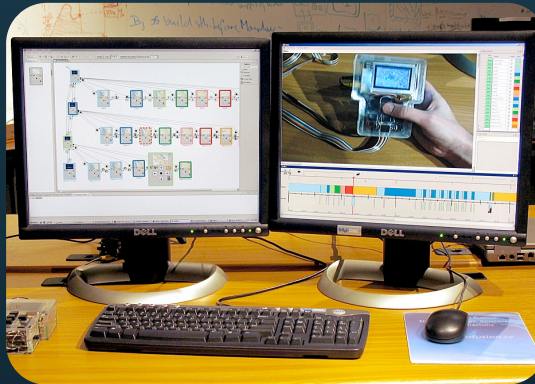


Prototyping Tools

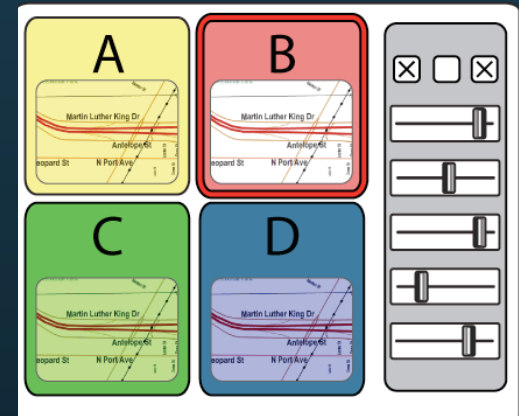
1 Enable **Rapid Authoring**
Off the Desktop



2 Aid **Iteration**
by Managing
Feedback



3 Aid **Exploration**
of Multiple
Alternatives



Prototyping is the central activity
that structures the design process.

Lawson, How Designers Think, Architectural Press
Cross, Designerly Ways of Knowing, Springer
Schrage, Serious Play, HBS Press
Moggridge, Designing Interactions, MIT Press
Buxton, Sketching User Experiences, Morgan Kaufmann

Prototypes enable **exploration** and **iteration** around concrete artifacts.

Lawson, How Designers Think, Architectural Press

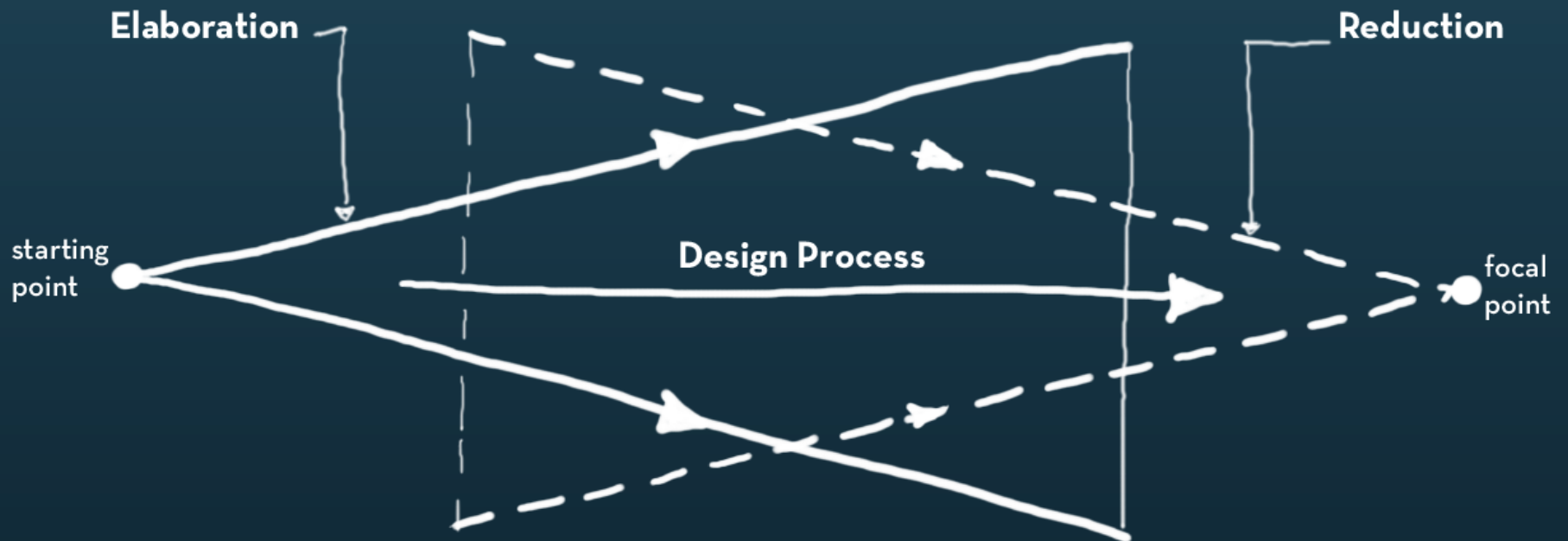
Cross, Designerly Ways of Knowing, Springer

Schrage, Serious Play, HBS Press

Moggridge, Designing Interactions, MIT Press

Buxton, Sketching User Experiences, Morgan Kaufmann

Exploration: Generate Parallel Alternatives

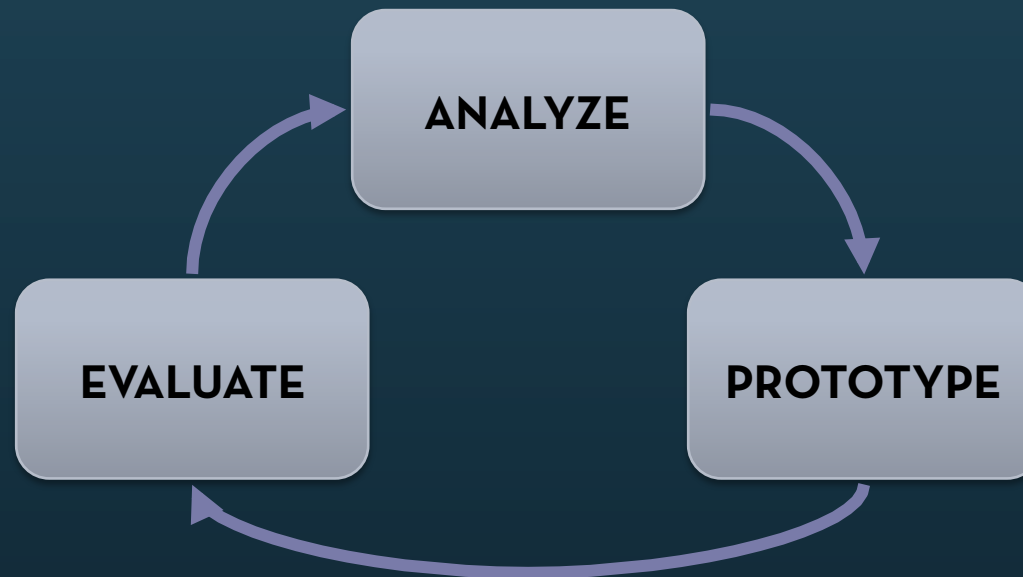


Redrawn from
Buxton, Sketching User Experiences



Prototypes for the
Microsoft mouse
From Moggridge,
Designing Interactions, Ch2

Iteration: Incorporate Insight



Dreyfuss, *Designing for People*, 1955;
Lawson, *How Designers Think*, 1997;
Cross, *Designerly Ways of Knowing*, 2005

Homepage (logged out)

http://drupal.markboultondesign.com/iteration11/homepage_notlog

Most Visited Getting Started Latest Headlines RecentChanges - De... QuickPost to reg:exp D.note survey DWG Share on Tumblr

Get Started Community & Support Documentation Download & Extend Marketplace About

Drupal™

Build for Today. Create for Tomorrow.

What can you make with Drupal? Beautiful, personal blogs or mighty, multi-featured, multi-user corporate sites. Our open source publishing software is the platform you need to create your place on the web.

Search Drupal.org

Search

Refine your search

- Modules and Themes
- Documentation
- API
- Forum posts

Popular searches

- Panels
- WYSIWYG
- Image upload
- Wiki
- CCK

Drupal Homepage Your Dashboard Login / Register

Why Choose Drupal?

Drupal is like Lego. Connect the pieces and build a site limited only by your imagination. Drupal's passionate, vibrant community are always creating new pieces, or improving existing ones. Choosing Drupal means as your needs evolve, so does your site.

Get Started with Drupal

Who Else Uses Drupal?

Magazines: [Fast Company](#), [Popular Science](#).
 Newspapers: [The Onion](#), [Morris Digital](#), [Seattle Times](#) and many, many [more...](#)

Things we made with Drupal



mtv.co.uk

Develop with Drupal

Drupal is extensible, powerful, scalable, and flexible.

Current activity

- 4212 CVS a/c holders
- 612 commits this month

Develop with Drupal

- Drupal API
- Download Drupal
- Security Info
- Issue List

Modules and Themes

Explore Drupal [modules](#) and [themes](#)

DO IT WITH DRUPAL A 3 DAY SEMINAR
 NEW ORLEANS, LA
 DECEMBER 10 - 12, 2008

Advertising helps build a successful ecosystem around Drupal

310,721 people in 24 countries speaking 14 different languages

News ● Docs updates (4) ● Forum posts (4) ● Commits (4) More

Done zotero

Neither **exploration** nor **iteration** are supported in today's user interface design tools.

“[D]esigners frequently wanted to have multiple designs side-by-side [...]

However [...] there is no built-in way in today’s implementation tools to have two versions of a behavior operating side-by-side.”

Myers et al., VL/HCC 2008

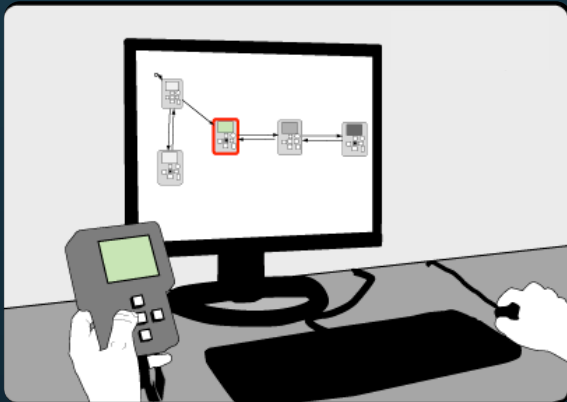
Why should we care?

- Designers tend to hang on to initial solution
(Cross, Designerly Ways of Knowing)
 - Decisions made with insufficient knowledge
 - Gained knowledge is lost or not acted on
 - Resources wasted in working *around* tools
-

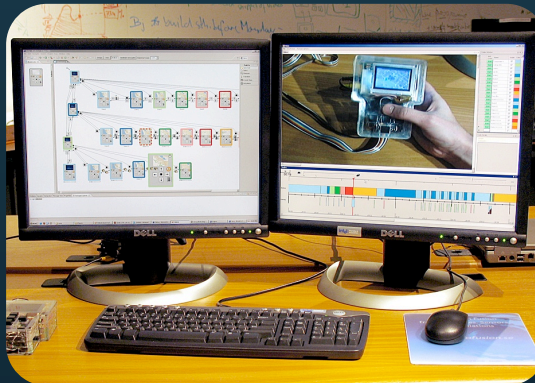
Problems uncovered too late

Poor solutions, failed products

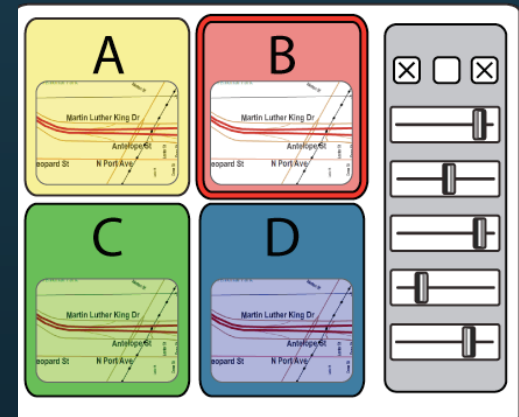
1 Enable **Rapid Authoring** Off the Desktop



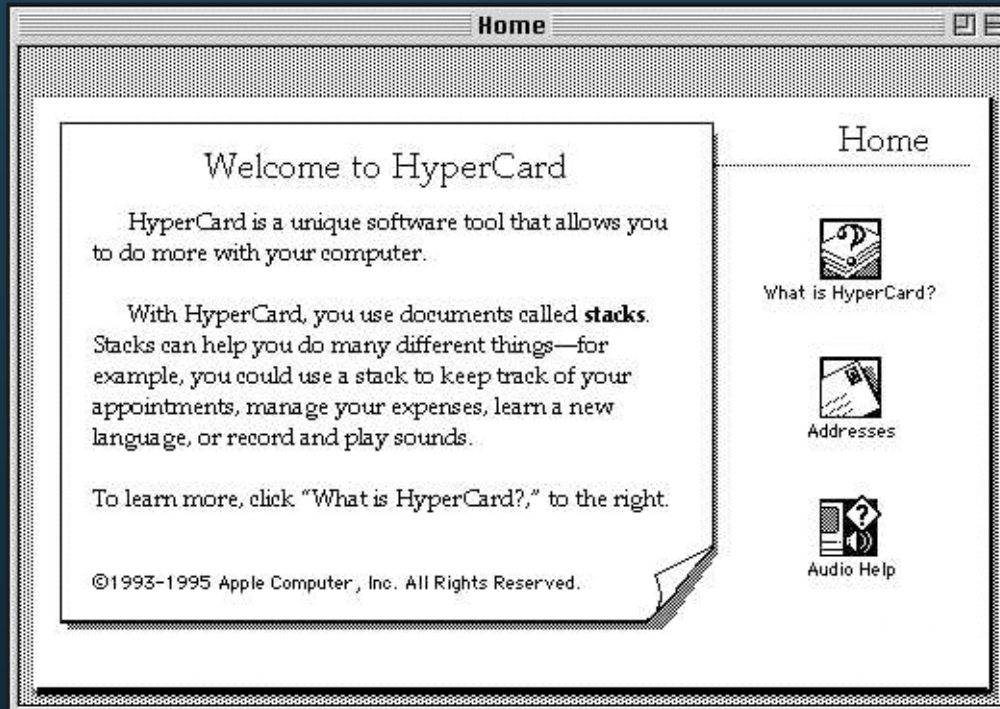
2 Aid **Iteration** by Managing Feedback



3 Aid **Exploration** of Multiple Alternatives



Related Work: Rapid Authoring of GUIs



Also:
SILK, Landay, 1996
DEMAIS, Bailey, 2001
DENIM, Lin, 2000

Hypercard, Atkinson 1985

Related Work: Rapid Authoring of ~~GUIs~~



Phidgets, Greenberg, 2001

Also:

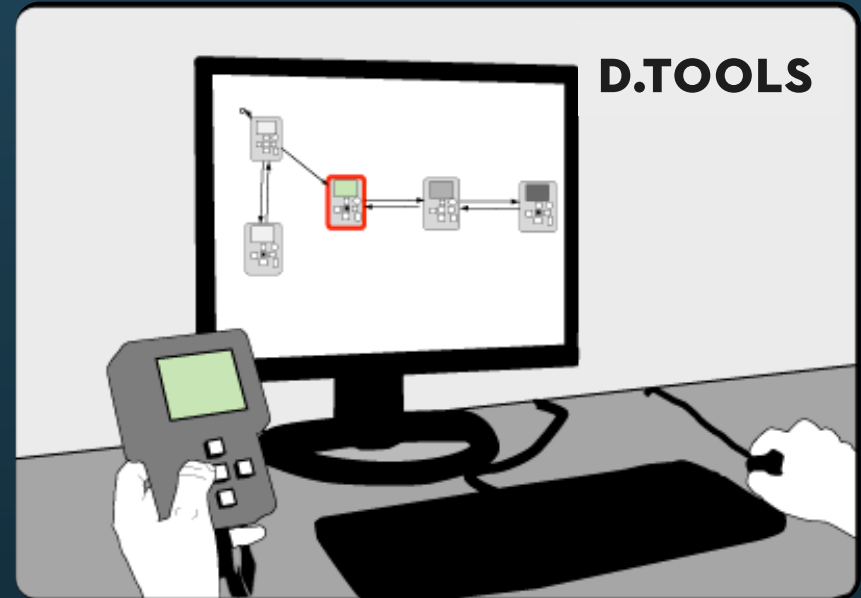
iStuff, Ballagas, 2003

Calder, Lee, 2004

DART, McIntyre, 2004

d.tools

- Visual, storyboard-based authoring
- Unites design of I/O affordances and application behavior
- Exposes hardware & software extension points



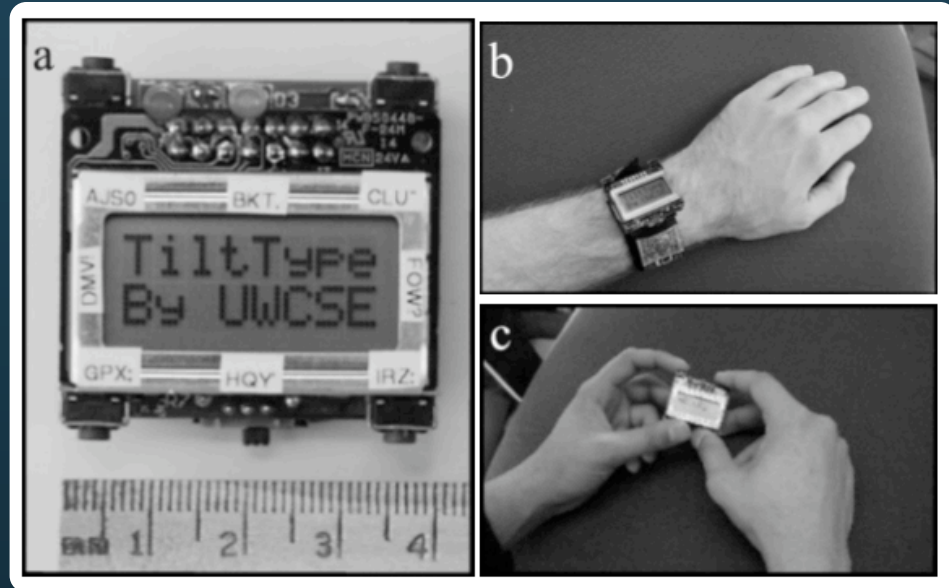
Example: Mobile Text Entry

If $|\text{Buttons}| < |\text{Letters}|$:

1. Select alphabet subset
2. Select letter from subset

Available Controls:

1. X/Y Tilt data
2. Discrete corner buttons



TiltType, Partridge et al., UIST 2002

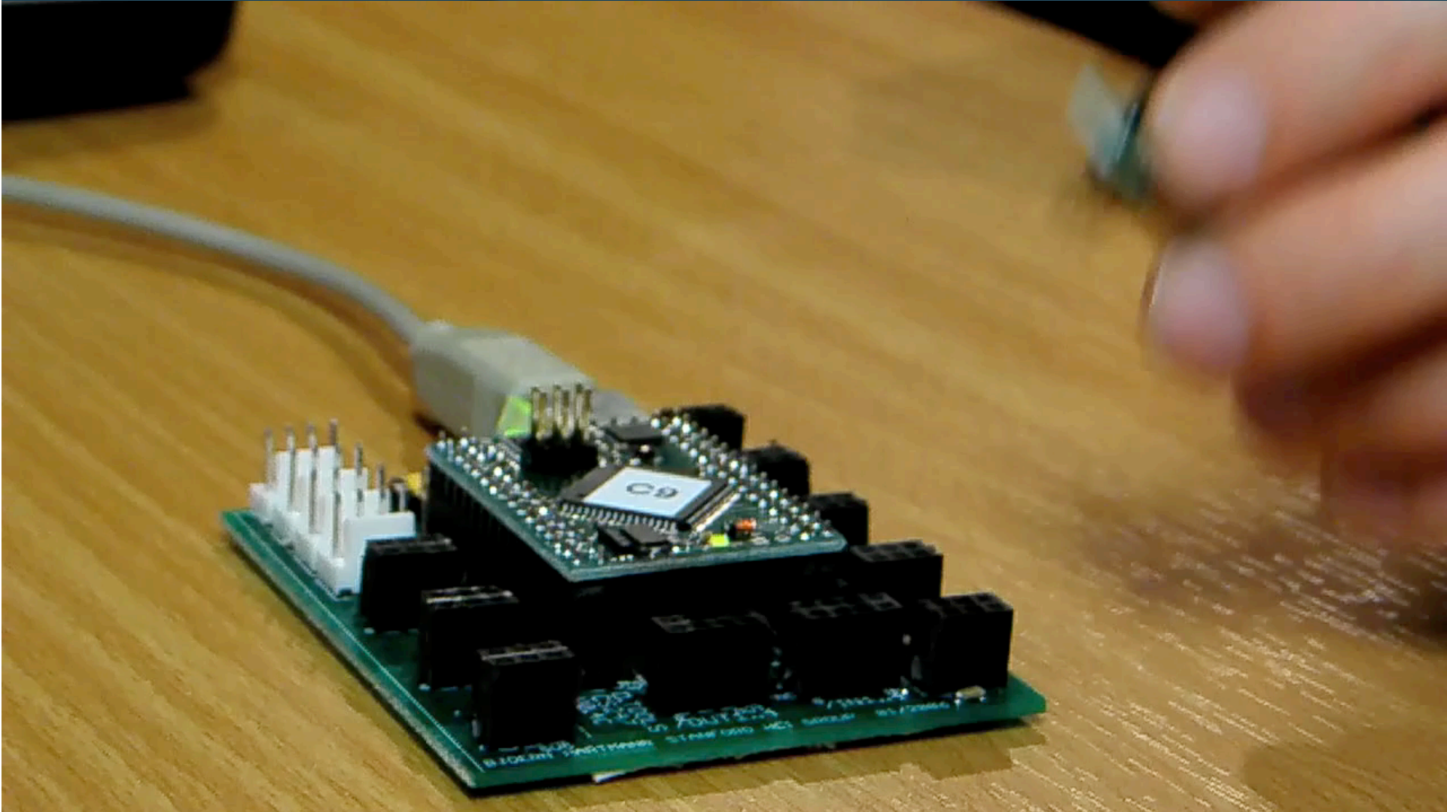
Example: Tilt-based text entry



Option 1:
Select set with buttons,
character with tilt



Authoring with d.tools



Design Environment

The screenshot displays a Design Environment for a camera application. The main workspace shows a state machine diagram with various states and transitions. The states include:

- Starting Up...
- Camera Model (with a red box highlighting a specific state)
- move crop area
- resize crop area
- Play (with sub-states: Auto Play, Red Eye Correction, Sound Menu, Protect)
- No printer found
- Not done yet
- Are You Sure? (with sub-states: Yes, No, Back)

The interface includes several panels:

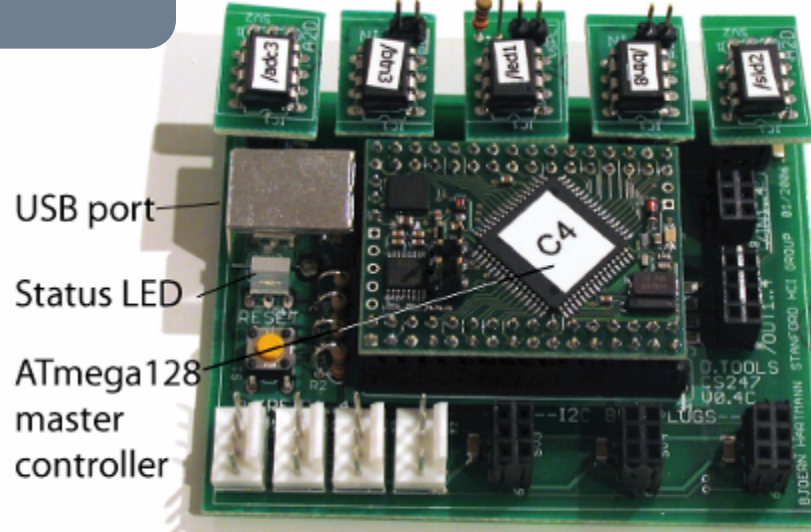
- camera.dvc**: Palette with Select, Simulate, Inputs (Pen, Button, Switch, Slider, Knob), Analog, and Outputs (Display Screen, LED, PWM, Speaker).
- camera.gui**: Palette with Select, Pen, Clip, and Text.
- Properties**:

Property	Value
Hardware Address	/lcd15
Image	edu.stanford.hci.dtools.utility.Path@398258
Name for Script	lcd15
- State Script**:

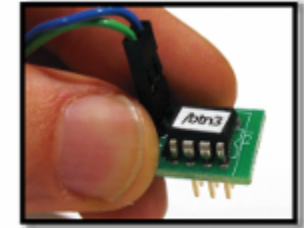
```
loop0 {  
  if(btnDown.getValue()) {  
    cliplmg.setScale(cliplmg.getScale()+5); //scale up  
  }  
  if(btnUp.getValue()) {  
    cliplmg.setScale(cliplmg.getScale()-5); //scale down  
  }  
  //center image on stage  
  cliplmg.setXY(stage.getWidth()/2-cliplmg.getWidth()/2,  
    stage.getHeight()/2-cliplmg.getHeight()/2);  
}
```

Smart Hardware

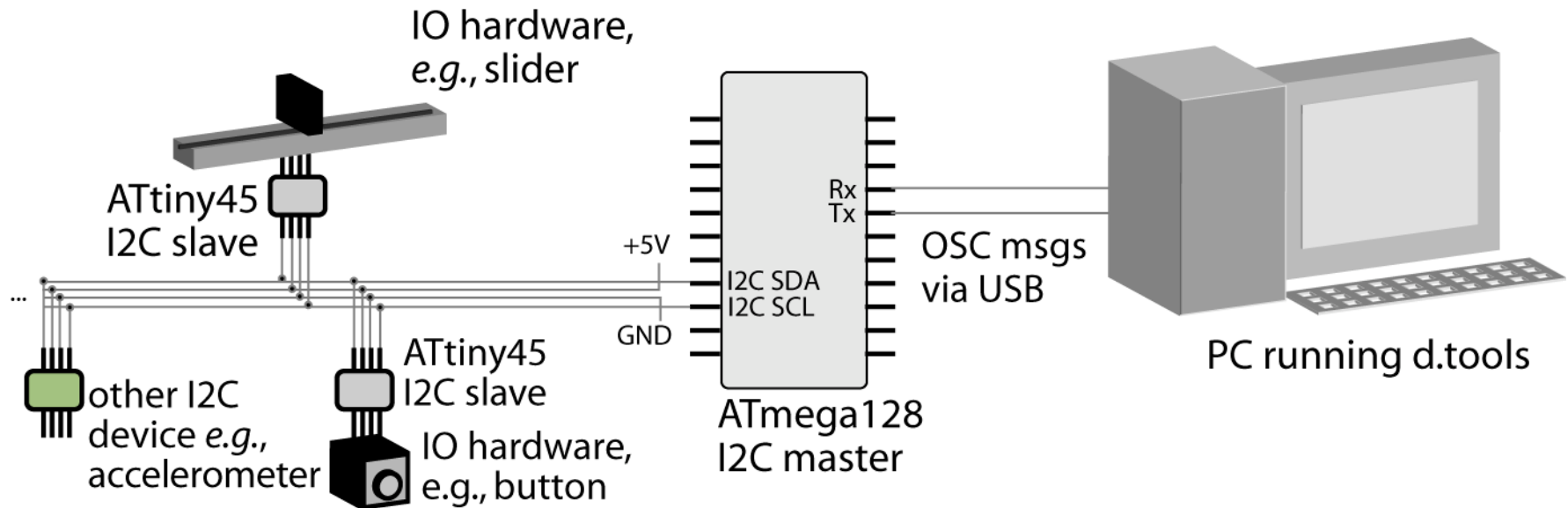
I²C input/output components (ATtiny45)



USB port
Status LED
ATmega128
master
controller



open I²C connectors



IO hardware,
e.g., slider

ATtiny45
I²C slave

+5V
GND

other I²C
device e.g.,
accelerometer

ATtiny45
I²C slave
IO hardware,
e.g., button

Rx
Tx
I²C SDA
I²C SCL
ATmega128
I²C master

OSC msgs
via USB

PC running d.tools

Design Environment

The screenshot displays a design environment for a camera application, showing a state machine diagram and associated components.

Left Panel (camera.dvc):

- Palettes: Select, Simulate
- Inputs: Pen, Button, Switch, Slider, Knob
- Outputs: Display Screen, LED, PWM, Speaker

Bottom Left Panel (camera.gui):

- Palettes: Select, Pen, Clip, Text

State Machine Diagram:

- States include: Starting Up..., Image, move crop area, resize crop area, Play (with various actions like Auto Play, Red Eye Correction, Sound Menu, Protect), No printer found, and Not done yet.
- Transitions are labeled with actions like "OR" and "Are You Sure?".

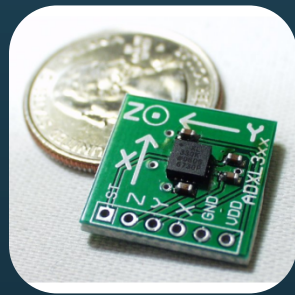
Bottom Panel (Properties):

Property	Value
Hardware Address	/lcd15
Image	edu.stanford.hci.dtools.utility.Path@398258
Name for Script	lcd15

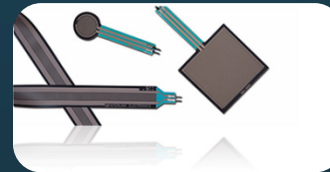
Bottom Panel (State Script):

```
loop0 {  
  if(btnDown.getValue()) {  
    cliplmg.setScale(cliplmg.getScale()+5); //scale up  
  }  
  if(btnUp.getValue()) {  
    cliplmg.setScale(cliplmg.getScale()-5); //scale down  
  }  
  //center image on stage  
  cliplmg.setXY(stage.getWidth()/2-cliplmg.getWidth()/2,  
    stage.getHeight()/2-cliplmg.getHeight()/2);  
}
```

How can we help designers map continuous sensor data to application events?



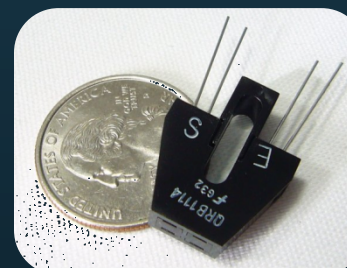
Accelerometers



Force Sensitive Resistors

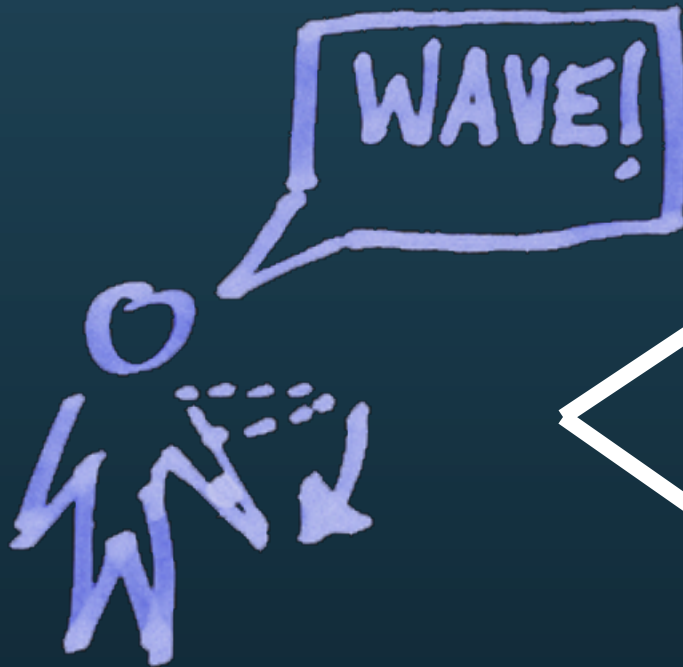


IR/Ultrasonic Ranges



Phototransistors

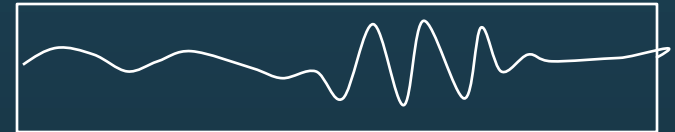
Representation Matters



Accelerometer X axis



Accelerometer Y axis



```
//detect accelerometer peaks

//read data sample
xVal[t++]=readA2DValue(xPin);

//look for changes in derivative
if(((xVal[t]-xVal[t-1]) >= 0
    && (xVal[t-1]-xVal[t-2]) < 0)
    ((xVal[t]-xVal[t-1]) < 0
    && (xVal[t-1]-xVal[t-2]) >= 0)
    //peak detected
    //send message
    oscSendMessageInt("/x/peak",1);
} else {
    //no peak
}
```

Isn't all machine learning example-based ?

- Yes, algorithms enable demonstration-based authoring, but are only the backend.
- What are appropriate interaction techniques that make real-time demonstration and testing feasible for designers?
- Insight: Combine recognition and direct manipulation to leverage the designer's knowledge

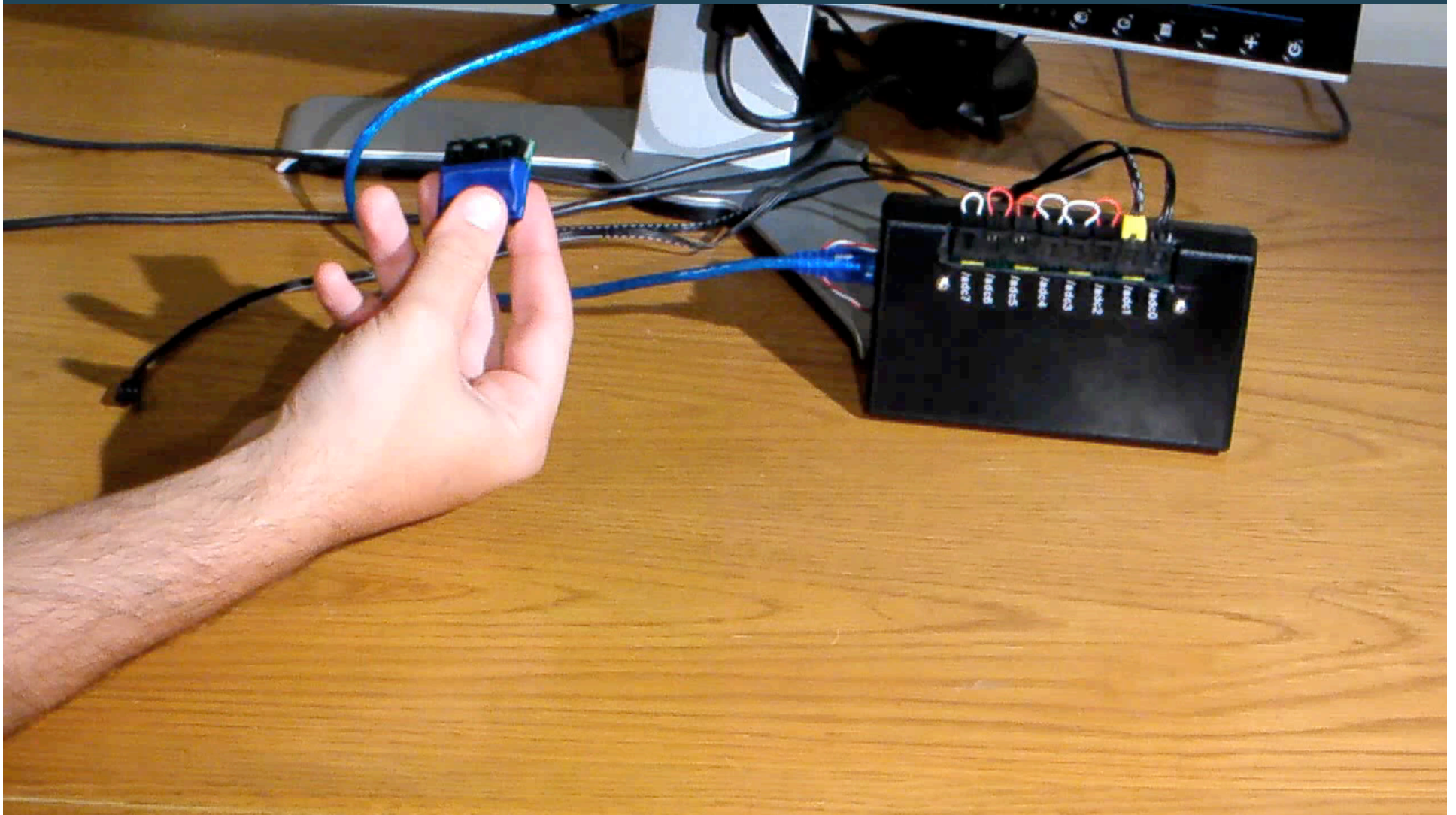
Related Work: Interfaces for Programming by Demonstration



Crayons, Fails, CHI 2003

Also:
Flexigesture, Merrill, CHI 2005
a CAPella, Dey, CHI 2004
Monet, Li, UIST 2005

Authoring by Demonstration



[CHI 2007]

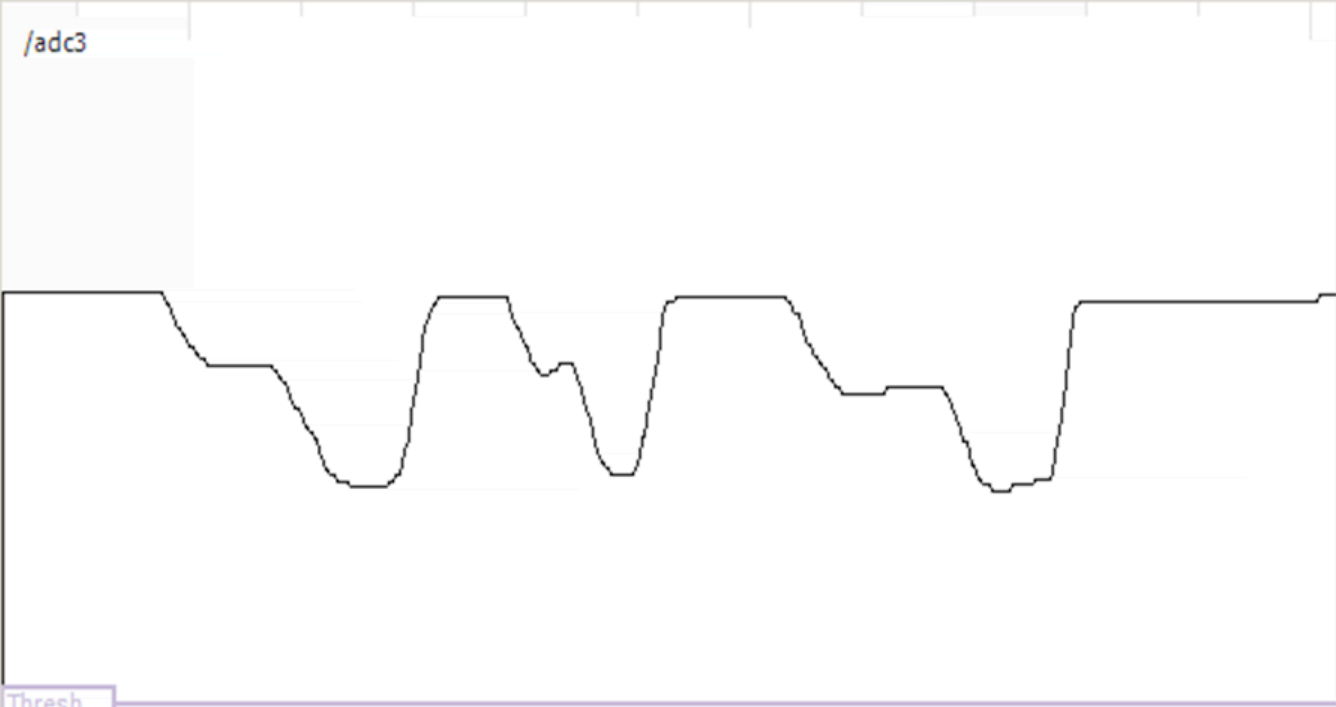
Generalization: Thresholds

Filter for /adc3

- Rate of Change
- Scale Y-Axis
- Offset
- Smooth

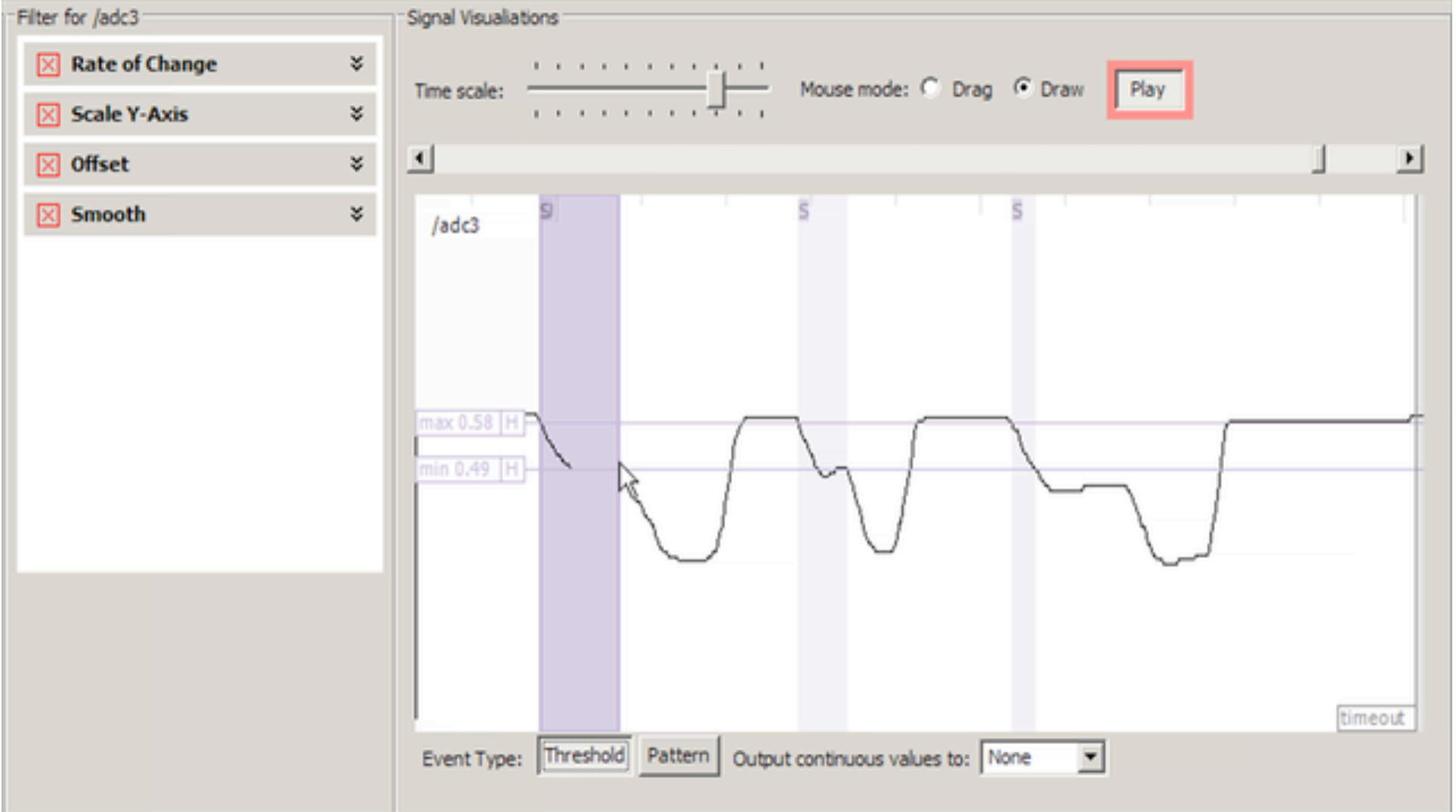
Signal Visualizations

Time scale: Mouse mode: Drag Draw



Event Type: Output continuous values to:

Generalization: Thresholds



Generalization: Patterns

Filter for /adc3

Rate of Change

Scale Y-Axis

Invert Signal ($x=-x$)

Scale [0..1..2]

From: Center Bottom

Offset

Offset Value [-1..+1]

Smooth

Signal Visualizations

Time scale: Mouse mode: Drag Draw **Play**

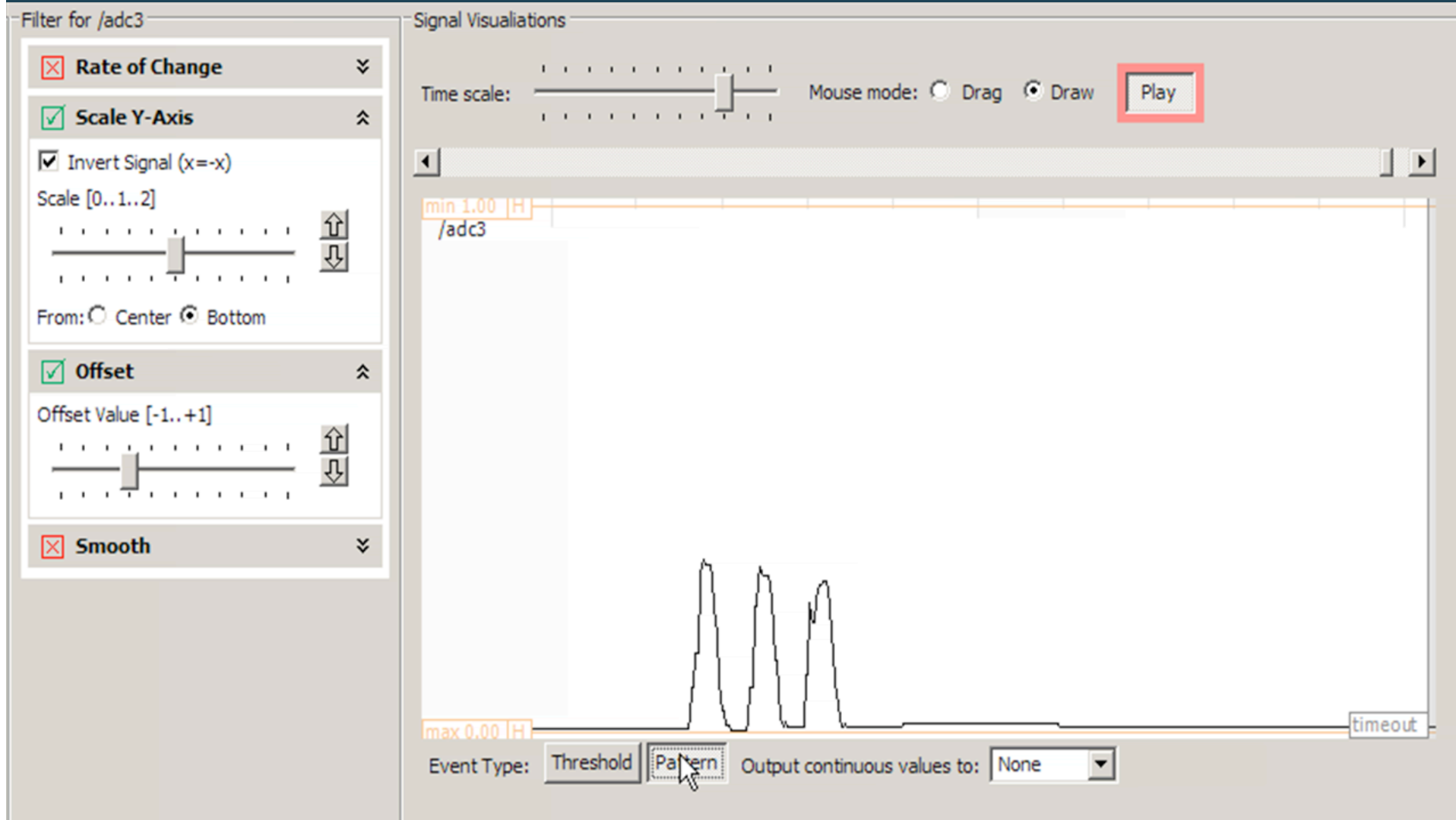
min 1.00 [H]

/adc3

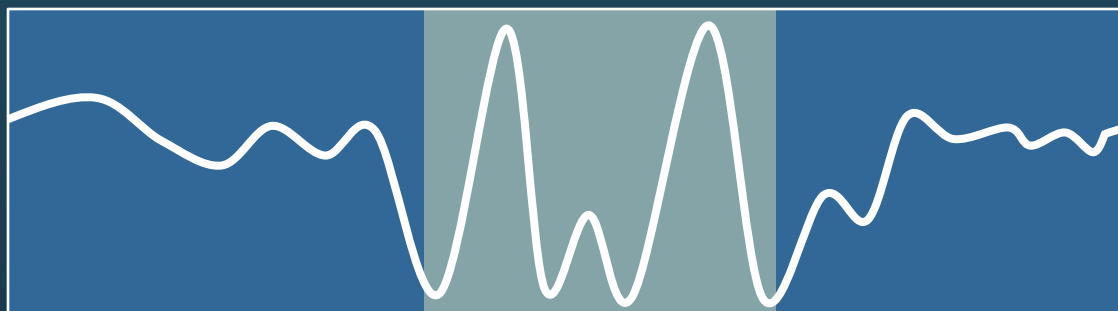
max 0.00 [H]

timeout

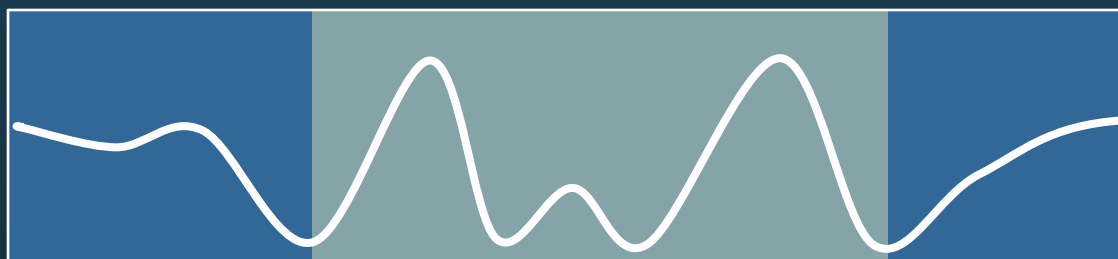
Event Type: **Pattern** Output continuous values to:

The image shows a software interface for signal processing. On the left, there is a configuration panel for a filter named '/adc3'. It includes several checkboxes: 'Rate of Change' (unchecked), 'Scale Y-Axis' (checked), 'Invert Signal (x=-x)' (checked), 'Offset' (checked), and 'Smooth' (unchecked). Below these are two sliders: 'Scale [0..1..2]' and 'Offset Value [-1..+1]'. The 'Scale' slider is set to approximately 1.5, and the 'Offset' slider is set to approximately 0.5. There are also radio buttons for 'From: Center' and 'Bottom', with 'Bottom' selected. On the right, the 'Signal Visualizations' section features a 'Time scale' slider, 'Mouse mode' options for 'Drag' and 'Draw' (with 'Draw' selected), and a red-bordered 'Play' button. Below this is a plot area showing a waveform with three distinct peaks. The plot has a vertical axis labeled 'min 1.00 [H]' at the top and 'max 0.00 [H]' at the bottom, and a horizontal axis labeled 'timeout'. At the bottom of the interface, there is an 'Event Type' section with a dropdown menu currently showing 'Pattern', and an 'Output continuous values to:' dropdown menu set to 'None'.

Dynamic Time Warping



Demonstration Signal



Matching Input Signal

```
int DTWDistance(char s[1..n], char t[1..m], int d[1..n,1..m]) {
  declare int DTW[0..n,0..m]
  declare int i, j, cost

  for i := 1 to m
    DTW[0,i] := infinity
  for i := 1 to n
    DTW[i,0] := infinity
  DTW[0,0] := 0
  for i := 1 to n
    for j := 1 to m
      cost := d[s[i],t[j]]
      DTW[i,j] := cost + minimum(DTW[ i-1, j ], // insertion
                                DTW[ i , j-1 ], // deletion
                                DTW[ i-1, j-1 ]) // match

  return DTW[n,m]
}
```

Quadratic in time & space
But: n is small @100Hz sampling

[Sakoe, H. Chiba, S. '78]

**How should we evaluate design
tool research?**

Triangulation of evaluation methods

What?

Cognitive Dimensions of Notation (CDN) Inspection

First-Use Laboratory Studies

Class & Industry Deployment

Used by authors as design tools for public exhibitions

Why?

Analysis of d.tools as a visual notation

Threshold and usability

Real-world stress test

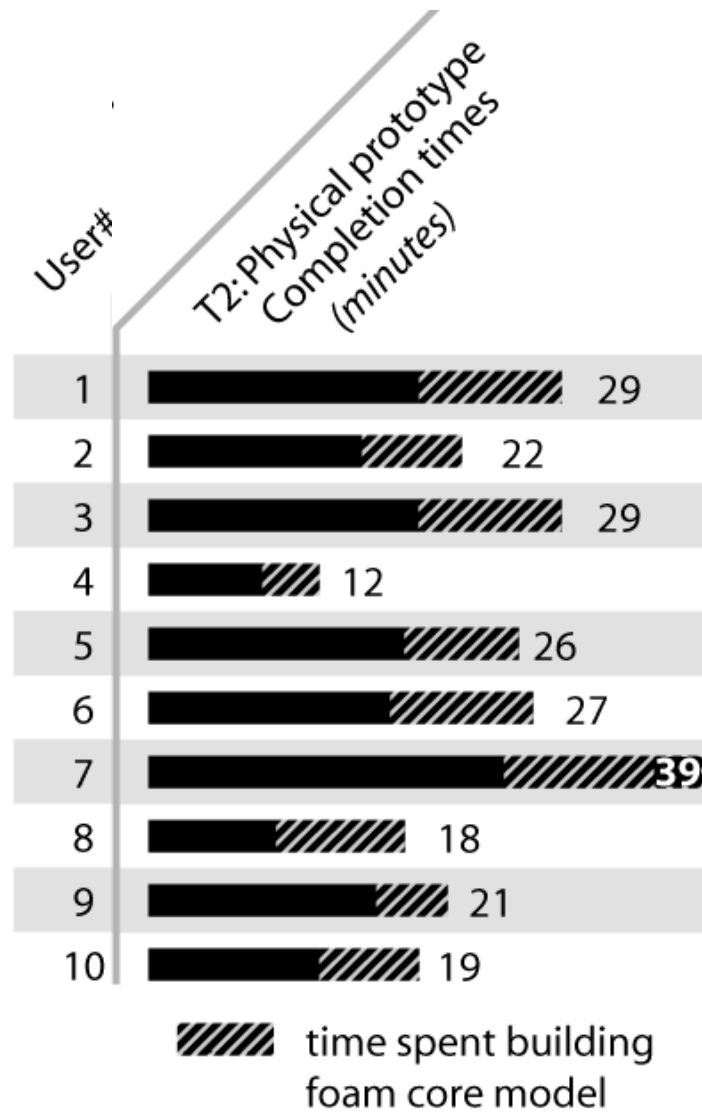
Complexity ceiling for knowledgeable users

Laboratory Studies





Laboratory Studies



Class Deployment

(Hardware manufactured for us by Leapfrog)





What's Important?

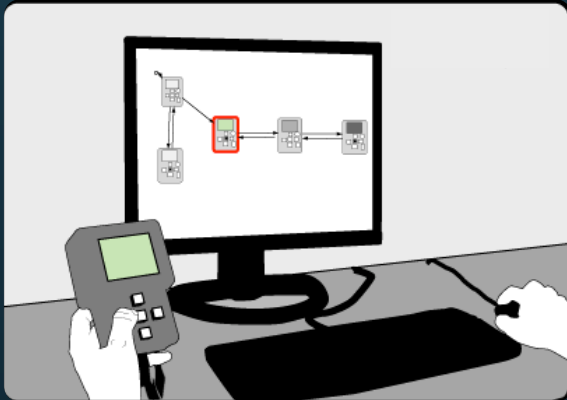
d. Tools successfully enabled non-programmers to prototype novel, sensor-based user interfaces.

Techniques:

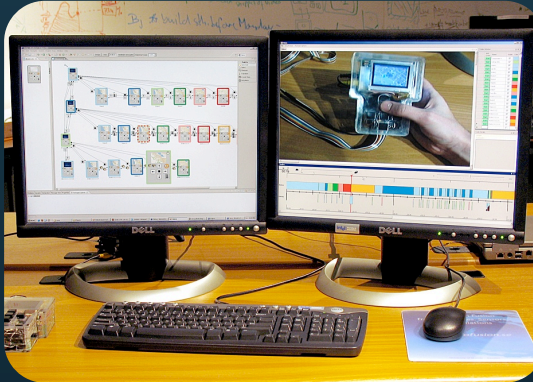
- 1. Virtual modeling of physical device and plug&play hardware enable visual, storyboard-based authoring.**
- 2. Programming by demonstration for sensor events raises complexity of possible interactions.**

Design tool research requires triangulation of evaluation methods

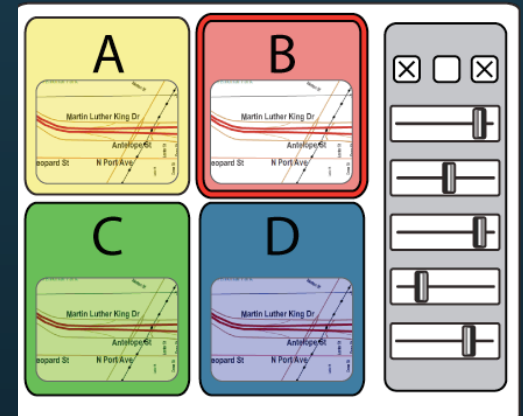
1 Enable **Rapid Authoring** Off the Desktop




2 Aid **Iteration** by Managing Feedback




3 Aid **Exploration** of Multiple Alternatives

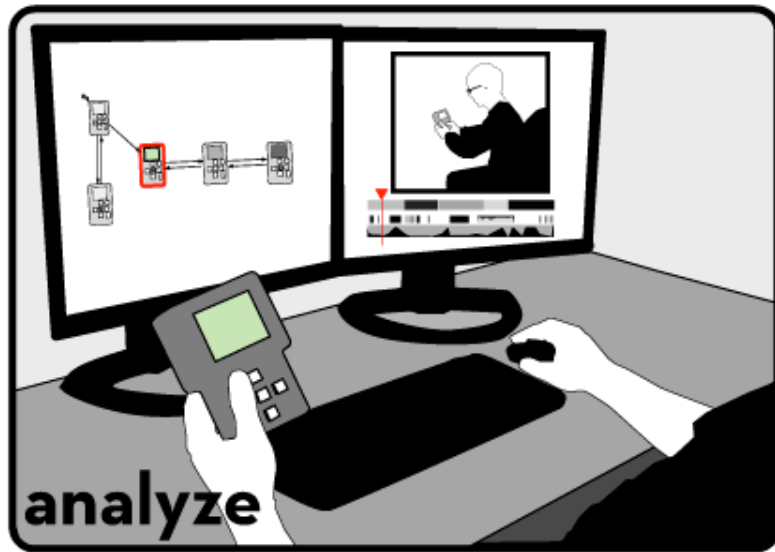
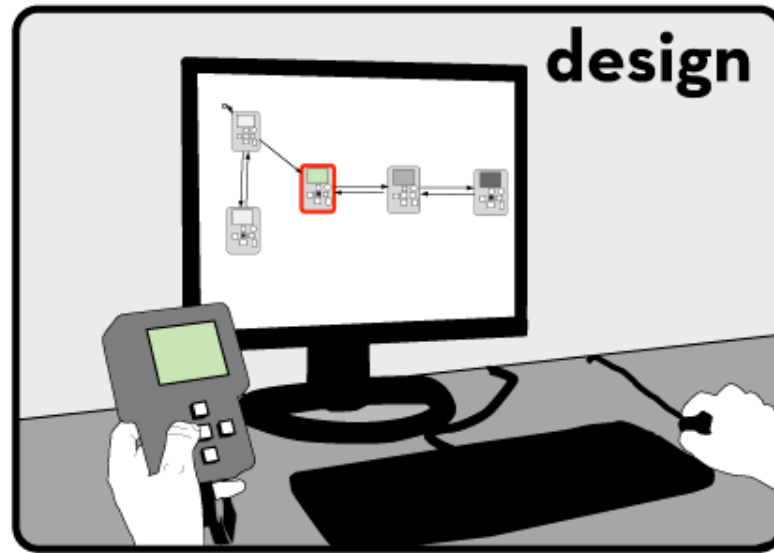




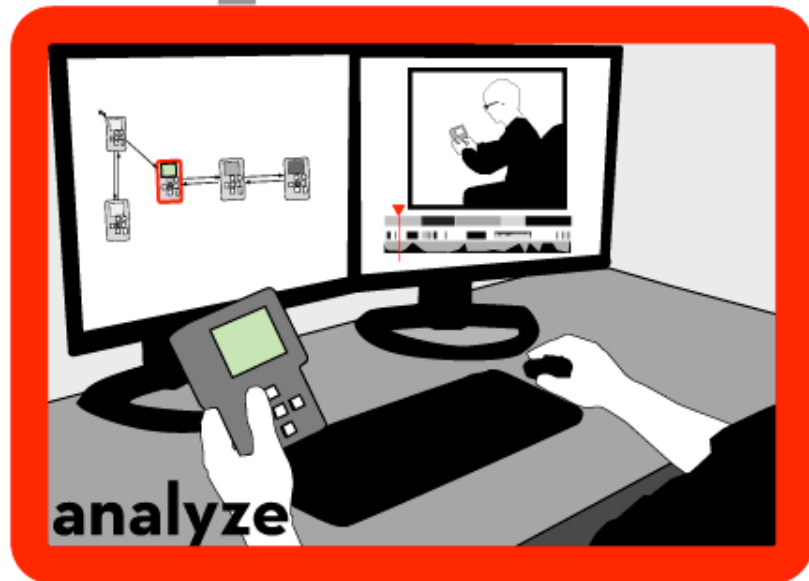
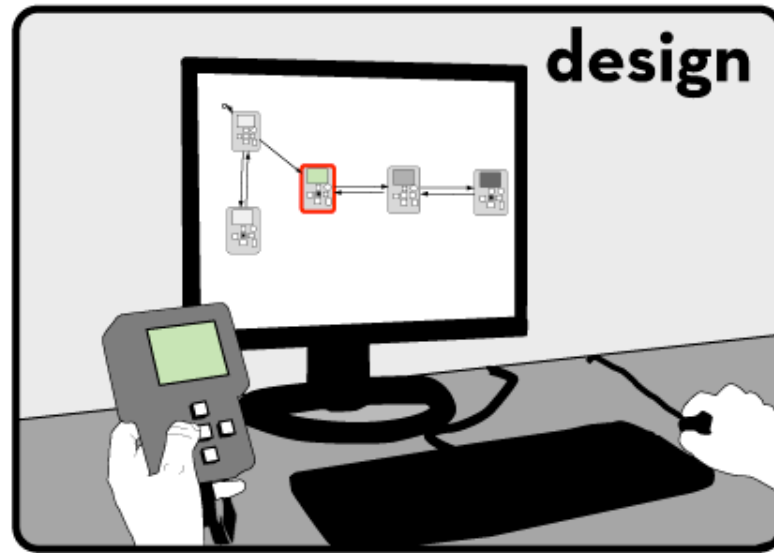
navant  s

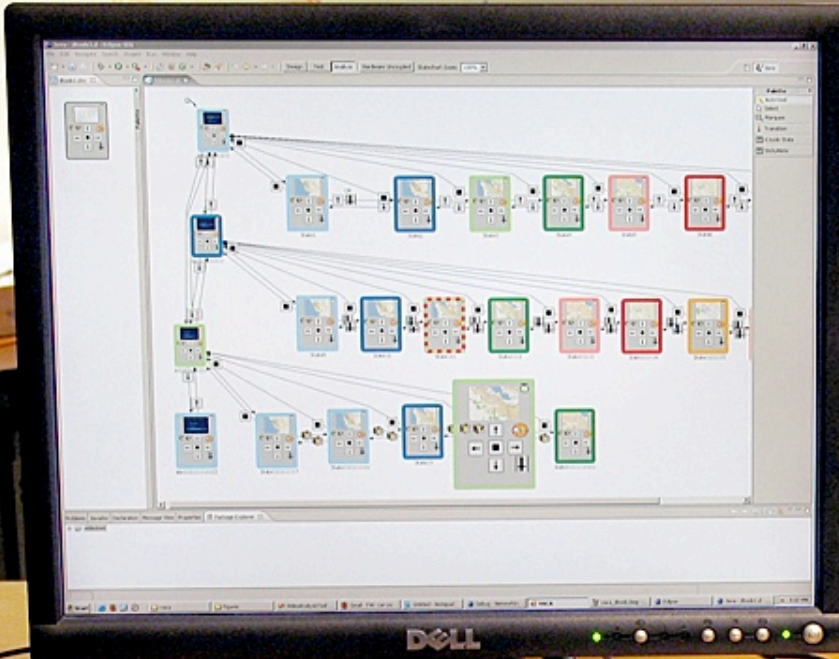
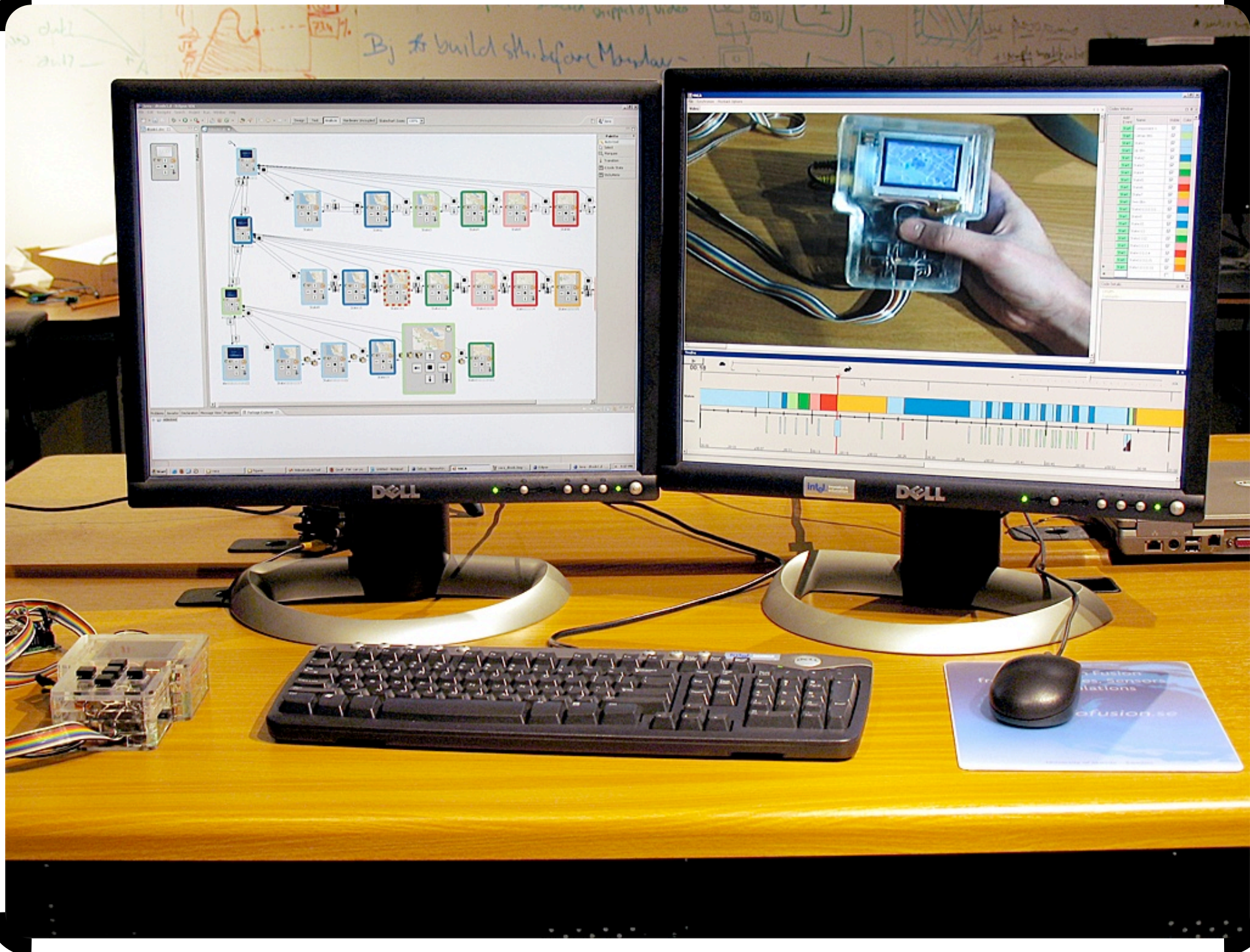
00:15:19:90

Microsoft
and the 





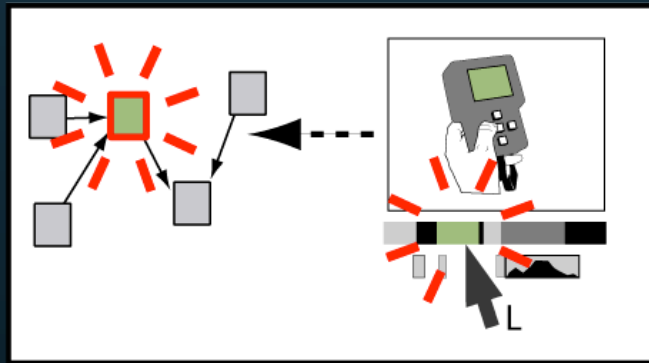




Analyze

The screenshot displays the Eclipse IDE interface for statechart development. The main window shows a statechart with states labeled State1 through State21, connected by transitions. A large orange arrow points from the statechart to a video window on the right, which shows a person holding a handheld device with a screen. Below the video is a timeline window showing execution time from 00:00 to 01:16. To the right of the video is a Code Window with a table of states and their properties.

Add	Name	Width	Color
<input type="checkbox"/>	State Menu	<input type="checkbox"/>	
<input type="checkbox"/>	State Den Btn	<input type="checkbox"/>	
<input type="checkbox"/>	State Menu2	<input type="checkbox"/>	
<input type="checkbox"/>	State Carner Btn	<input type="checkbox"/>	
<input checked="" type="checkbox"/>	State9	<input checked="" type="checkbox"/>	
<input type="checkbox"/>	State Component 1	<input type="checkbox"/>	
<input type="checkbox"/>	State State0	<input type="checkbox"/>	
<input type="checkbox"/>	State State1	<input type="checkbox"/>	
<input type="checkbox"/>	State State12	<input type="checkbox"/>	
<input type="checkbox"/>	State State13	<input type="checkbox"/>	
<input type="checkbox"/>	State State14	<input type="checkbox"/>	
<input type="checkbox"/>	State State15	<input type="checkbox"/>	
<input type="checkbox"/>	State State16	<input type="checkbox"/>	
<input type="checkbox"/>	State Lp Btn	<input type="checkbox"/>	
<input type="checkbox"/>	State State1	<input type="checkbox"/>	
<input type="checkbox"/>	State State2	<input type="checkbox"/>	
<input type="checkbox"/>	State State3	<input type="checkbox"/>	

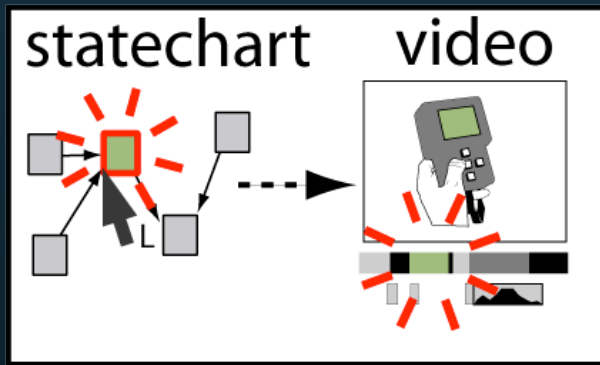


The screenshot displays a video production software interface, likely Avid Media Composer. The main workspace is filled with a complex multi-camera timeline, showing multiple video tracks (V1 through V10) and audio tracks (A1 through A10) for various cameras. The tracks are color-coded and connected by lines, indicating a multi-camera shoot. On the right side, there is a 'Video' window showing a live feed of a hand holding a transparent plastic device with a small screen. The screen displays the text 'BUTTON', 'CLEAR', and 'CONFIRMATION' in blue. Below the video window is a 'Timeline' window showing a multi-track timeline with various colored blocks representing video and audio segments. The interface includes standard software elements like a menu bar, toolbars, and a status bar at the bottom.

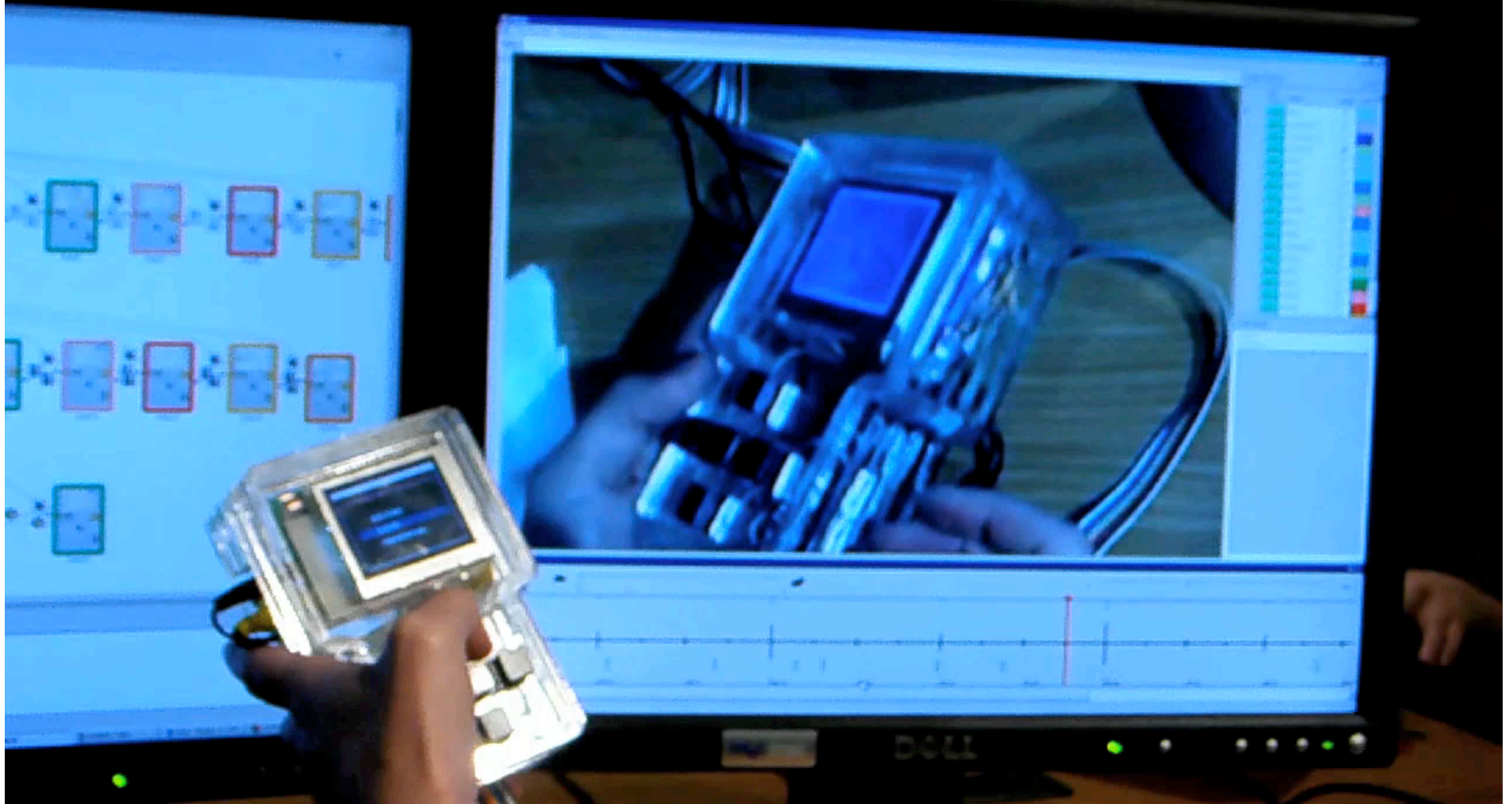
Analyze

The screenshot shows the Eclipse IDE interface. On the left, a statechart diagram is displayed with states labeled State1 through State21. A red dashed box highlights State9. A large orange arrow points from State9 to a video player window on the right. The video player shows a person holding a handheld device with a screen. Below the video player is a timeline with a playhead at 00:05. On the far right, a 'Code Window' is visible, listing various states and components with checkboxes and color-coded boxes.

Add	Name	Width	Color
<input type="checkbox"/>	State Menu	<input type="checkbox"/>	
<input type="checkbox"/>	State Den Btn	<input type="checkbox"/>	
<input type="checkbox"/>	State Menu2	<input type="checkbox"/>	
<input type="checkbox"/>	State Carner Btn	<input type="checkbox"/>	
<input checked="" type="checkbox"/>	State9	<input checked="" type="checkbox"/>	
<input type="checkbox"/>	State Component 1	<input type="checkbox"/>	
<input type="checkbox"/>	State State0	<input type="checkbox"/>	
<input type="checkbox"/>	State State1	<input type="checkbox"/>	
<input type="checkbox"/>	State State2	<input type="checkbox"/>	
<input type="checkbox"/>	State State3	<input type="checkbox"/>	
<input type="checkbox"/>	State State4	<input type="checkbox"/>	
<input type="checkbox"/>	State State5	<input type="checkbox"/>	
<input type="checkbox"/>	State State6	<input type="checkbox"/>	
<input type="checkbox"/>	State State7	<input type="checkbox"/>	
<input type="checkbox"/>	State State8	<input type="checkbox"/>	
<input type="checkbox"/>	State State10	<input type="checkbox"/>	
<input type="checkbox"/>	State State11	<input type="checkbox"/>	
<input type="checkbox"/>	State State12	<input type="checkbox"/>	
<input type="checkbox"/>	State State13	<input type="checkbox"/>	
<input type="checkbox"/>	State State14	<input type="checkbox"/>	
<input type="checkbox"/>	State State15	<input type="checkbox"/>	
<input type="checkbox"/>	State State16	<input type="checkbox"/>	
<input type="checkbox"/>	State State17	<input type="checkbox"/>	
<input type="checkbox"/>	State State18	<input type="checkbox"/>	
<input type="checkbox"/>	State State19	<input type="checkbox"/>	
<input type="checkbox"/>	State State20	<input type="checkbox"/>	
<input type="checkbox"/>	State State21	<input type="checkbox"/>	
<input type="checkbox"/>	State Lp Btn	<input type="checkbox"/>	
<input type="checkbox"/>	State State1	<input type="checkbox"/>	
<input type="checkbox"/>	State State2	<input type="checkbox"/>	
<input type="checkbox"/>	State State3	<input type="checkbox"/>	



Analyze



How might we revise interaction designs?



Tools for revising interaction designs



Movie production

```
Repository (revision 1.18)
<html>
<BODY>
<H1>Debug Demo</H1>
<table border="1" width="700">
<tr bgcolor="red">
<th>Name</th>
<th>Address</th>
<th>Phone</th>
</tr>
<tr>
<td>
<?php
$db = array(
array ("John", "E 10th St., NYC,
array ("Francois", "12 Bd. de Gre
array ("Klaus", "312 Beethoven St.
array ("Dave", "312 Beethoven St.
array ("Kate", "312 Beethoven St.
array ("Klaus", "312 Beethoven St
array ("Shirly", "72 Independence
array ("Bill", "127 Maine St., Sa
);
/**
 * @return string
 * @param i int
 * @desc Returns 'white' for even numbers
 */
function row_color($i)
{
    $bgcolor1 = "white";
    $bgcolor2 = "yellow";
}

Current
<tr bgcolor="red">
<th>Name</th>
<th>Address</th>
<th>Phone</th>
</tr>
<?php
$db = array(
array ("John", "E 10th St., NYC,
array ("Francois", "12 St. de Gre
array ("Klaus", "312 Beethoven St.
array ("Dave", "312 Beethoven St.
array ("Kate", "312 Beethoven St.
array ("Klaus", "312 Beethoven St
array ("Shirly", "72 Independence
array ("Bill", "127 Maine St., Sa
);
/**
 * @param i int
 * @desc Returns 'white' for even numbers
 */
function row_color($i)
{
    $bgcolor1 = "white";
    $bgcolor2 = "yellow";
    if ( ($i % 2) == 0 ) {
```

Programming

- Button
- Switch
- Slider
- Knob
- Analog Sensor
- RFID Reader
- Timer

Outputs

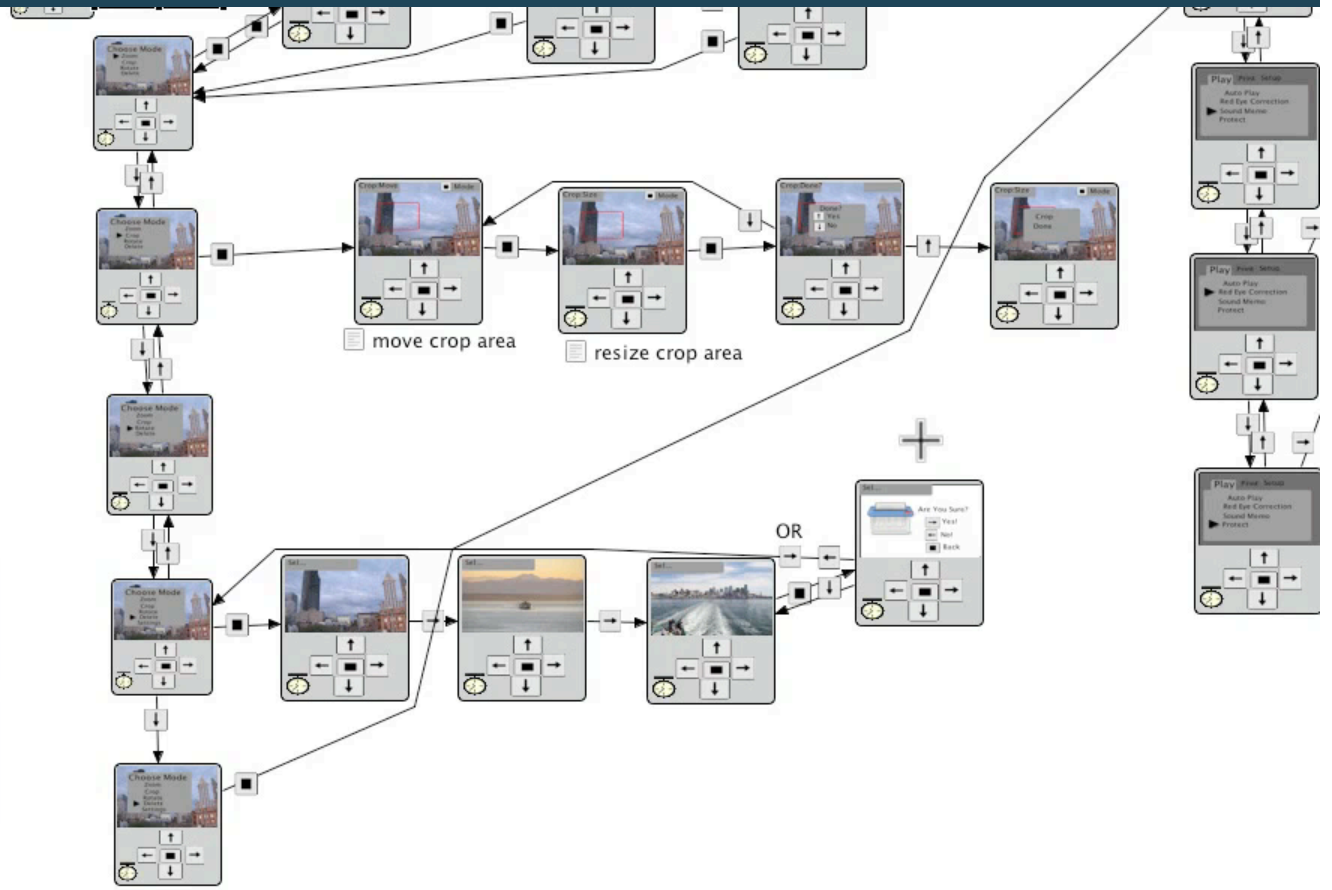
- Display Screen
- LED
- PWM
- Speaker

camera.gui

Starting Up...

Palette

- Select
- Pen
- Clip
- Text



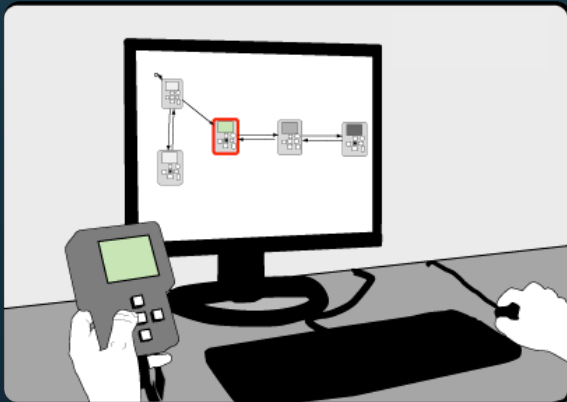
What's Important?

Integrating analysis of test data into a design tool can radically shorten the time to work with test videos.

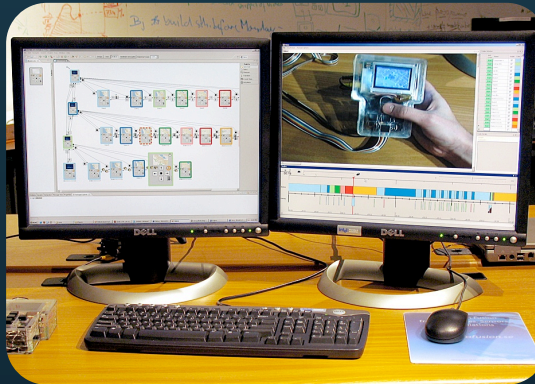
Integration is enabled by a high-level visual representation of the software model.

Design is a team sport – new revision tools are needed to record insight gained during analysis.

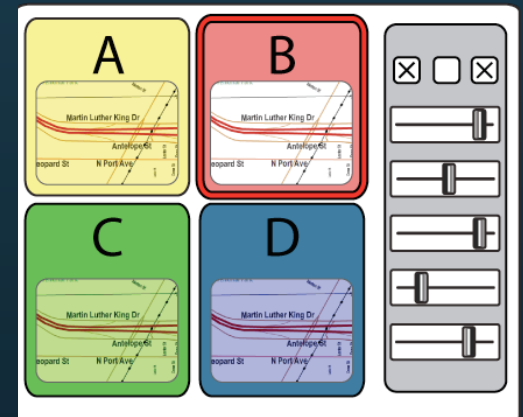
1 Enable Rapid Authoring Off the Desktop



2 Aid Iteration by Managing Feedback

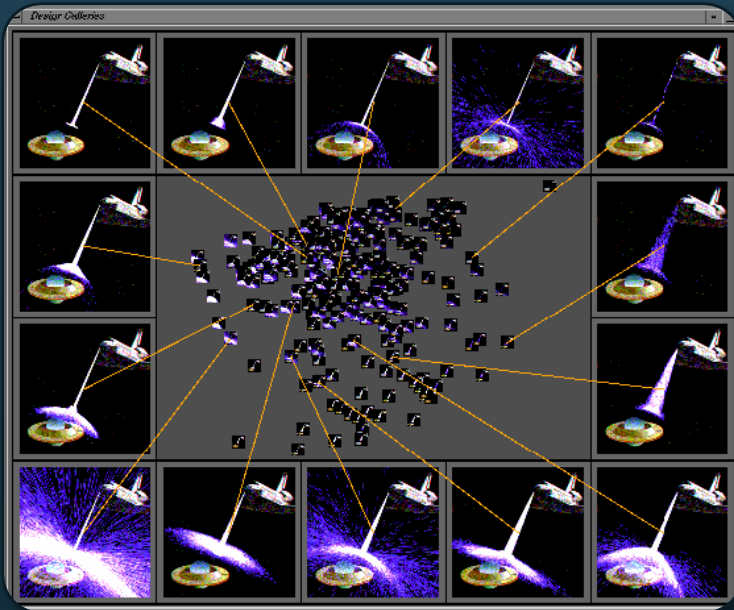


3 Aid Exploration of Multiple Alternatives

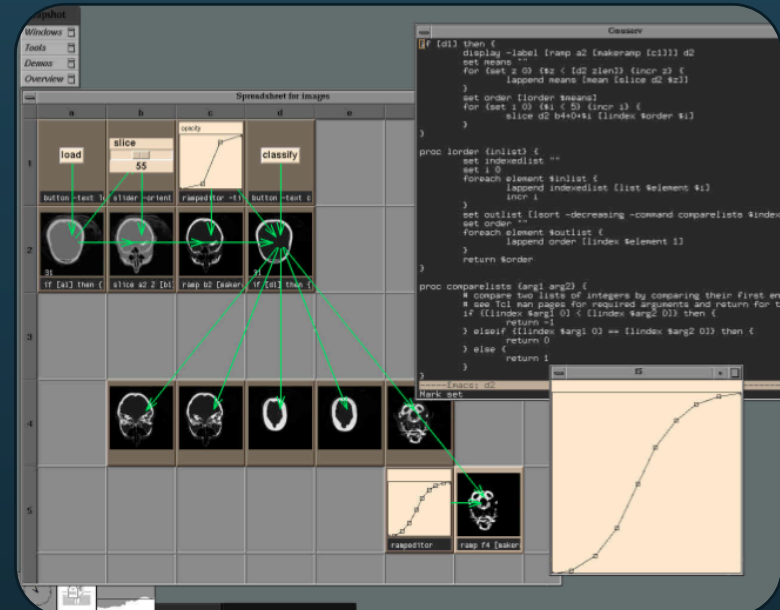


How can tools support the creation of **multiple** user interface **alternatives**?

Related Work: Working with Alternatives



Design Galleries, Marks, 1997



Levoy, Spreadsheets for Images, 1994

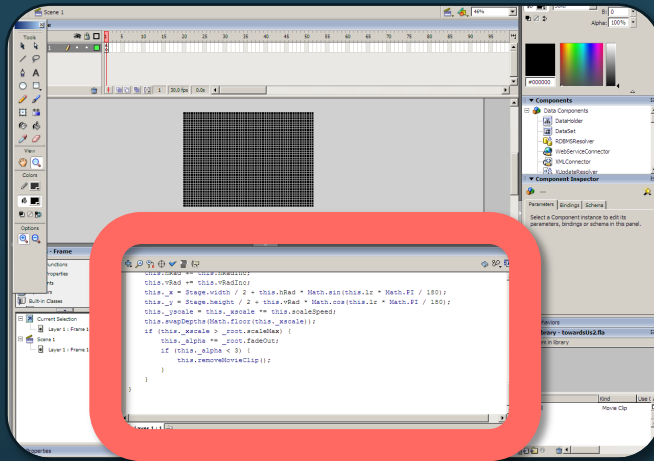
Subjunctive Interfaces, Lunzer, TOCHI 08

Spreadsheets for Visualization, Chi, IEEE CGA 98

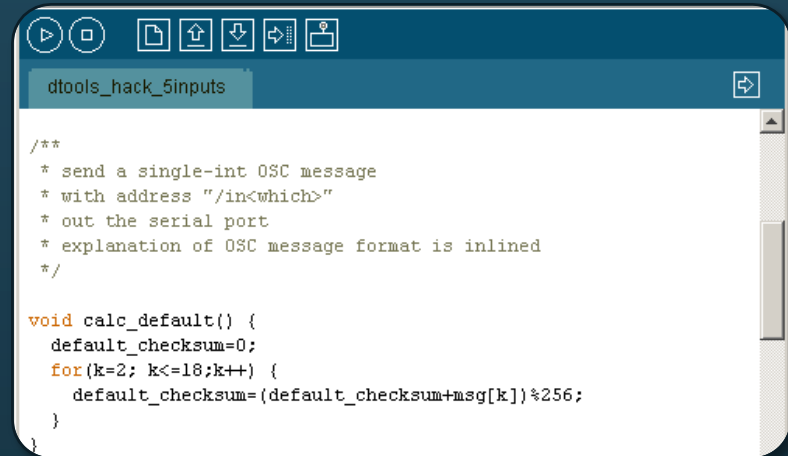
Parameter Spectrums & Parallel Pies, Terry, UIST 02 & CHI 04

Partials, Terry, Phd 2005

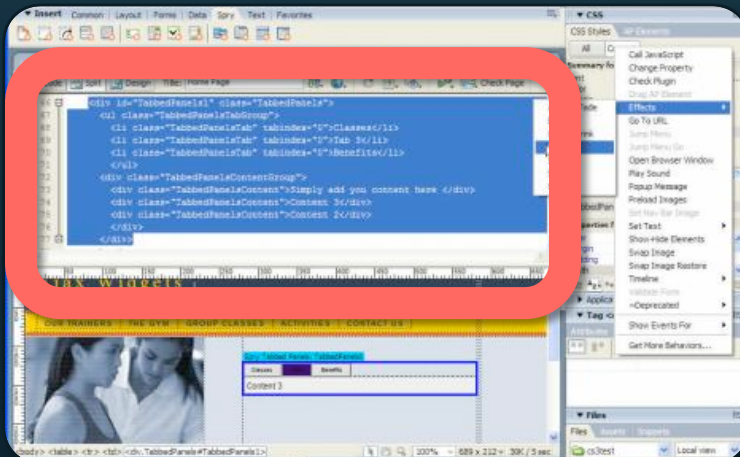
Interaction Designers Write Code



Adobe Flash



Arduino / Processing



Adobe Dreamweaver

3 Requirements for Programmed Interactions

(based on 3 interviews & code inspection)

- Manage parameter variations
- Manage code alternatives
- Access variations & alternatives at runtime

```
public int ATTENUATION = 20; // = 10; // = 50;
```


```
public int BASE_ALPHA = 50;
```

```
public int POSITIVE_RESPONSE = 6;
```

3 Requirements for Programmed Interactions

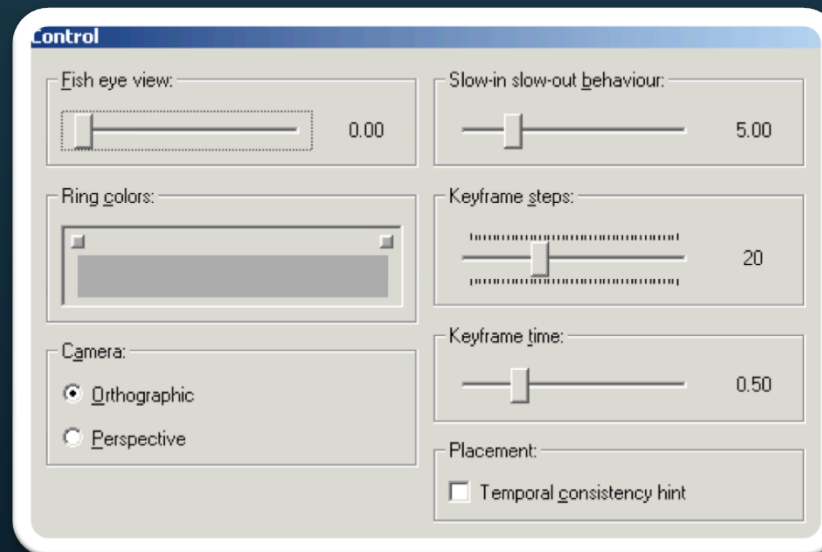
- Manage parameter variations
- Manage code alternatives
- Access variations & alternatives at runtime

```
public calculateNextSize(int [][]currentSizes, int i, int j) {  
    float denominator = 0;  
    int sumOfNeighbors = 0;  
    int maxOfNeighbors = 0;  
    if(i != 0) {  
        sumOfNeighbors += currentSizes[i - 1][j]; denominator += 1;  
        maxOfNeighbors = currentSizes[i - 1][j];  
        // if(j != 0) sumOfNeighbors += currentSizes[i - 1][j - 1]/2; denominator += .5;  
        // if(j != currentSizes[0].length - 1) sumOfNeighbors += currentSizes[i - 1][j + 1]/2; denominator += .5;  
    }  
    if(i != currentSizes.length - 1) {  
        sumOfNeighbors += currentSizes[i + 1][j]; denominator += 1;  
        if(currentSizes[i + 1][j] > maxOfNeighbors) maxOfNeighbors = currentSizes[i + 1][j];  
        // if(j != 0) sumOfNeighbors += currentSizes[i + 1][j - 1]/2; denominator += .5;  
        // if(j != currentSizes[0].length - 1) sumOfNeighbors += currentSizes[i + 1][j + 1]/2; denominator += .5;  
    }  
}
```

 Processing code brought in by an interviewee

3 Requirements for Programmed Interactions

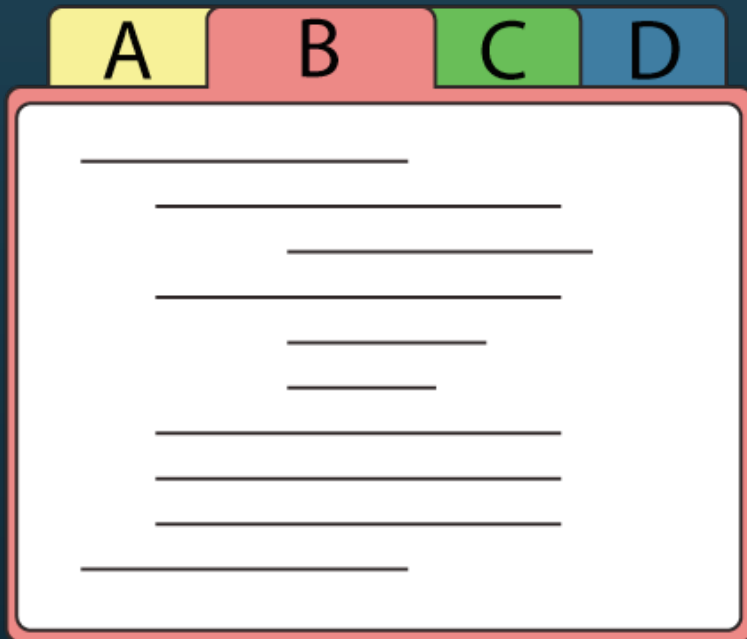
- Manage parameter variations
- Manage code alternatives
- Access variations & alternatives at runtime



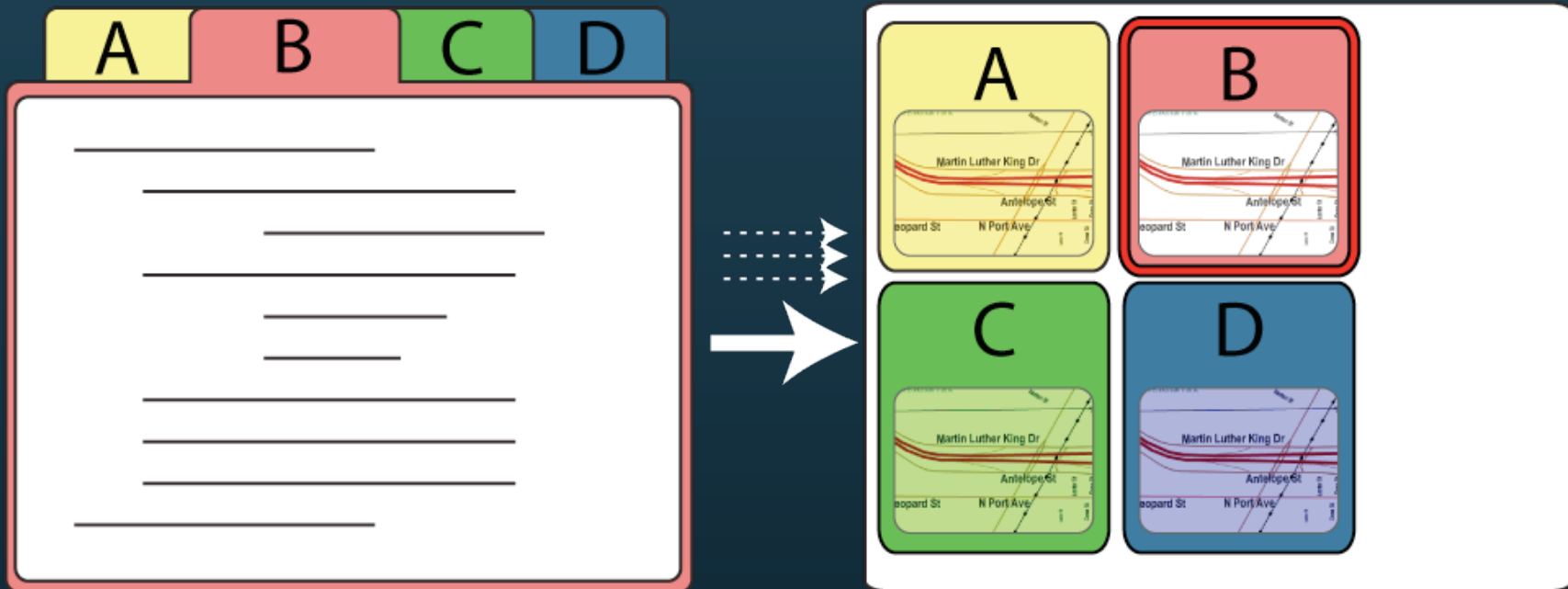
Juxtapose: Source alternatives...



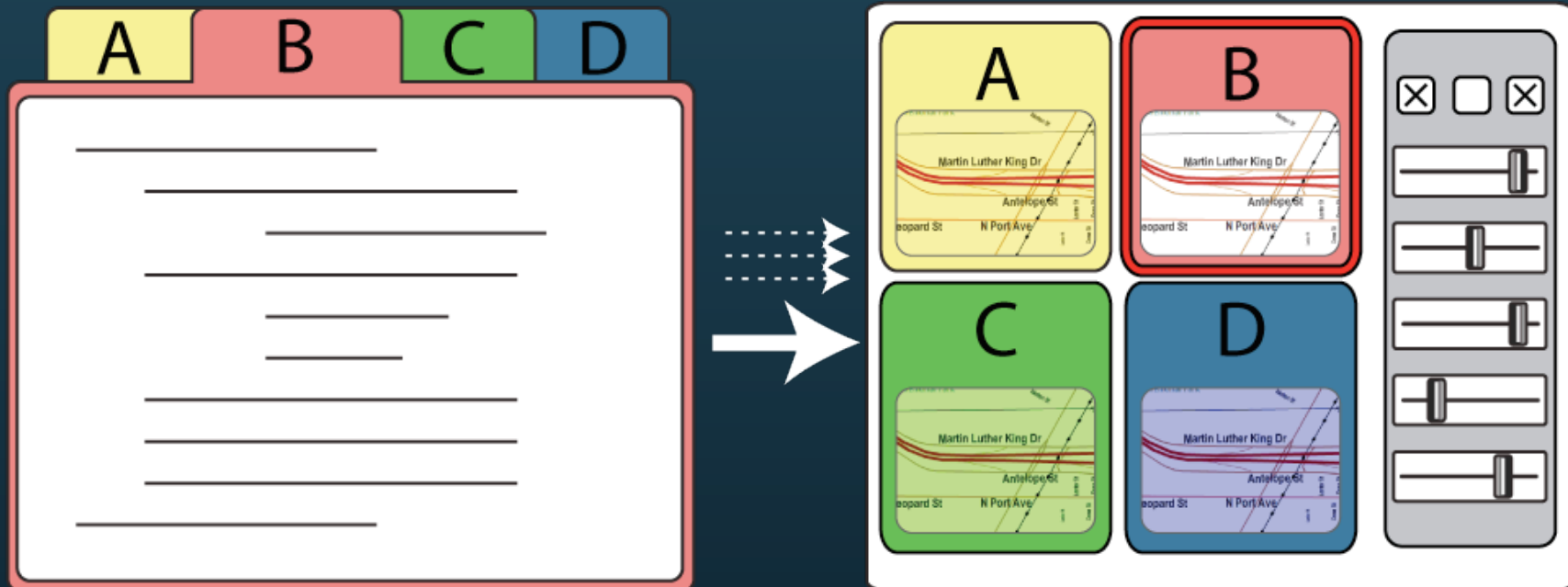
Juxtapose: Source alternatives...



... are executed in parallel,



and tuned through an generated UI.



Parallel Editing

```
File Edit Run User Study
Run Add Alternative Linked Edit
Alternative 1 Alternative 2 Alternative 3 Alternative 4
function FlashApplication() {
    _root["bar"]._xscale = 0;
    _root["title_txt"].text="Slow Wavy";
    _root["percentage_txt"].text="0%";
    //this line sets up the link to the source code editor
    var adapter:MultiplicIDBConnector3= new MultiplicIDBConn

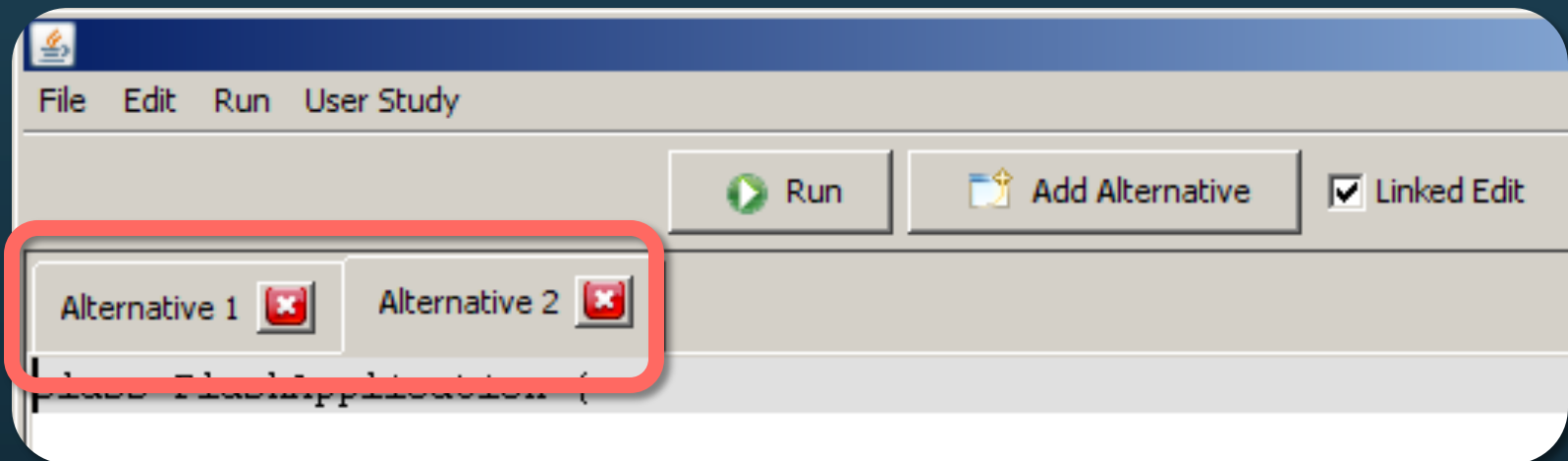
    // set onEnterFrame handler to this object's function
    _root["control"] = this;
    _root.onEnterFrame = function() {
        this["control"].onEnterFrame();
    };
    listenerObject.click = function(eventObj:Object) {
        _root["control"].onClick();
    }
    _root["start_btn"].addEventListener("click", listenerObj,
}

public function map(x:Number):Number {
    return x+Math.sin(x*Math.PI*5)/20;
}

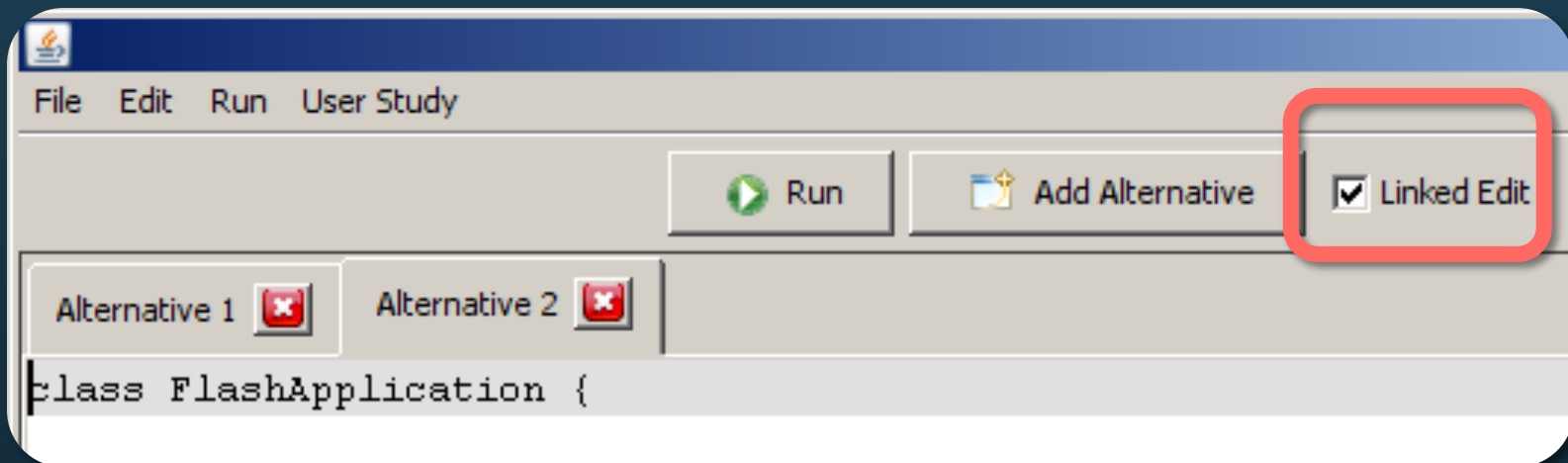
public function onClick() {
```

*Juxtapose editor for
ActionScript 2.
Generates Flash files.*

Parallel Editing

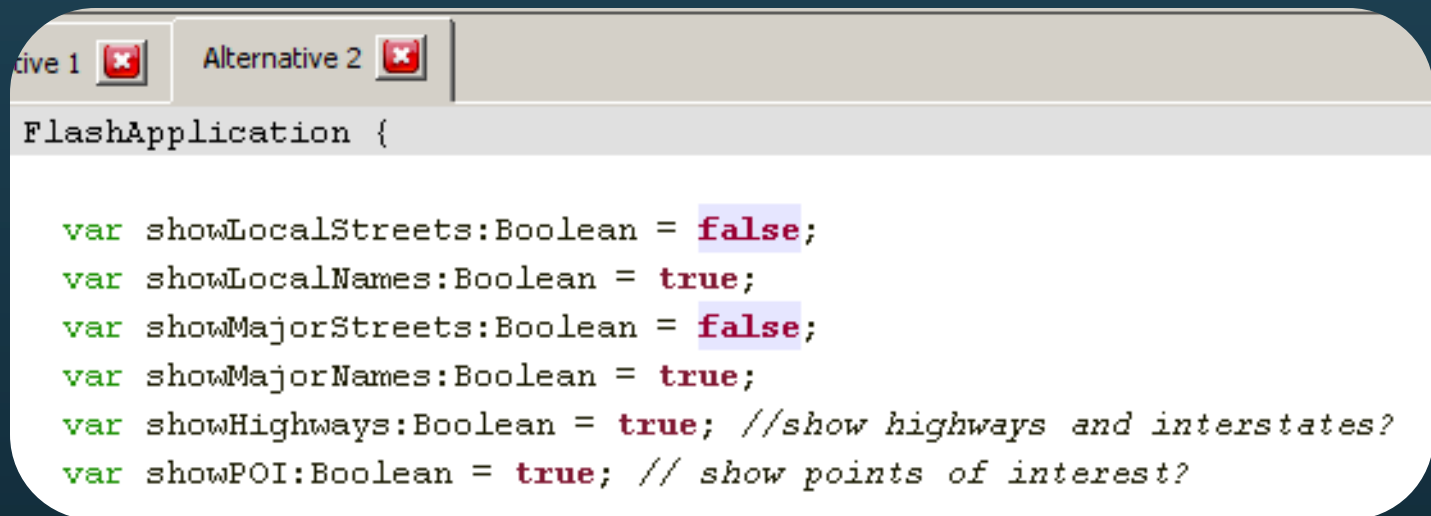


Parallel Editing



Linked Editing: Toomim 2004

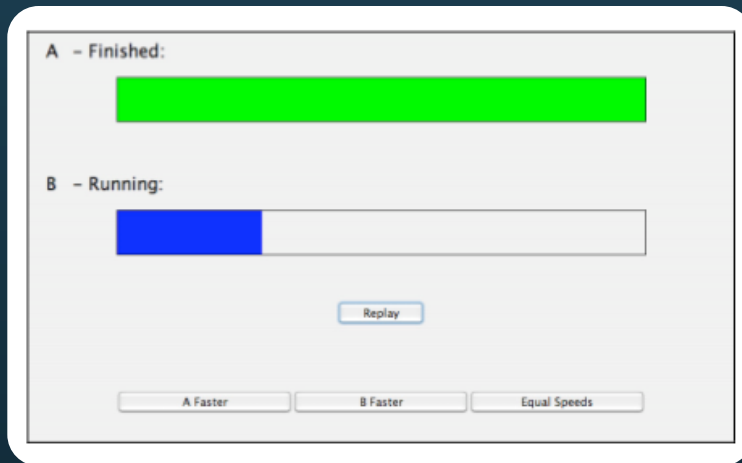
Parallel Editing



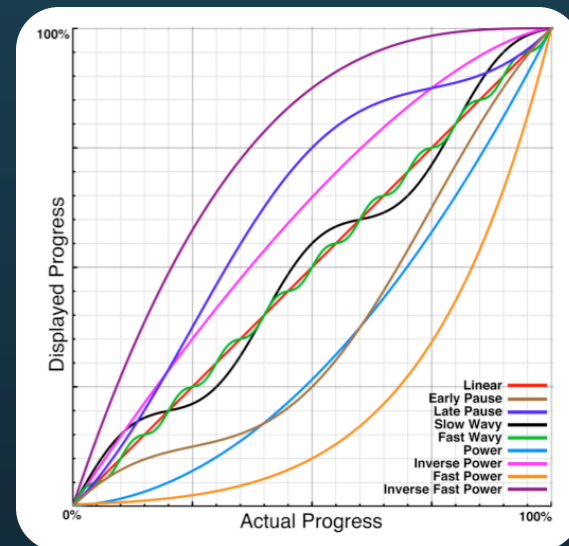
```
FlashApplication {  
  
    var showLocalStreets:Boolean = false;  
    var showLocalNames:Boolean = true;  
    var showMajorStreets:Boolean = false;  
    var showMajorNames:Boolean = true;  
    var showHighways:Boolean = true; //show highways and interstates?  
    var showPOI:Boolean = true; // show points of interest?
```

Example: Rethinking the Progress Bar

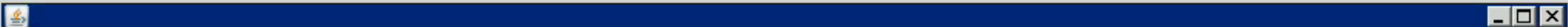
Harrison et al., UIST 2007



Progress bar test application



Progress profiles



Show: One Alternative All Alternatives Parallel Input: On Off

Alternative_1.swf

Alternative_2.swf

Alternative_3.swf

Alternative_4.swf

Linked Tuning

+	VarName	Type
	0	128
Min	50	Max

+	VarName	Type
	0	128
Min	50	Max

Snapshots



Show: One Alternative All Alternatives Parallel Input: On Off

<p>Alternative_1.swf</p> <div style="border: 1px solid gray; padding: 10px; text-align: center;"> <h3>Linear</h3> <div style="background-color: #4CAF50; color: white; padding: 5px; display: inline-block; width: 80%;">100%</div> <input type="button" value="Start"/> </div>	<p>Alternative_2.swf</p> <div style="border: 1px solid gray; padding: 10px; text-align: center;"> <h3>Inv. Fast Power</h3> <div style="background-color: #4CAF50; color: white; padding: 5px; display: inline-block; width: 80%;">100%</div> <input type="button" value="Start"/> </div>
<p>Alternative_3.swf</p> <div style="border: 1px solid gray; padding: 10px; text-align: center;"> <h3>Slow Wavy</h3> <div style="background-color: #4CAF50; color: white; padding: 5px; display: inline-block; width: 80%;">100%</div> <input type="button" value="Start"/> </div>	<p>Alternative_4.swf</p> <div style="border: 1px solid gray; padding: 10px; text-align: center;"> <h3>Fast Power</h3> <div style="background-color: #4CAF50; color: white; padding: 5px; display: inline-block; width: 80%;">100%</div> <input type="button" value="Start"/> </div>

Linked Tuning

± height	number
± running	boolean
± tabChildren	boolean
± tabEnabled	boolean
± width	number
± x	number

Snapshots

Parameter Tuning

```
var angmax:Number = 270;  
var dimin:Number = 61;  
var maxchild:Number = 3;  
var maxrec:Number = 4;
```



The image shows a parameter tuning interface with four sliders, each representing a different parameter. Each slider has a range from 0 to a maximum value, and a current value is displayed in a box below the slider.

Parameter	Min	Max	Current Value
angmax	0	360	270
dimin	0	100	61
maxchild	0	8	3
maxrec	0	8	4

Example: Tuning Phosphor

Baudisch, UIST 2006

Options



steve



patrick

george

ken

ed



Example: Tuning Phosphor

Baudisch, UIST 2006

The screenshot displays a software interface for tuning phosphors. The main window, titled "Alternative_1.swf", contains a control panel with the following elements:

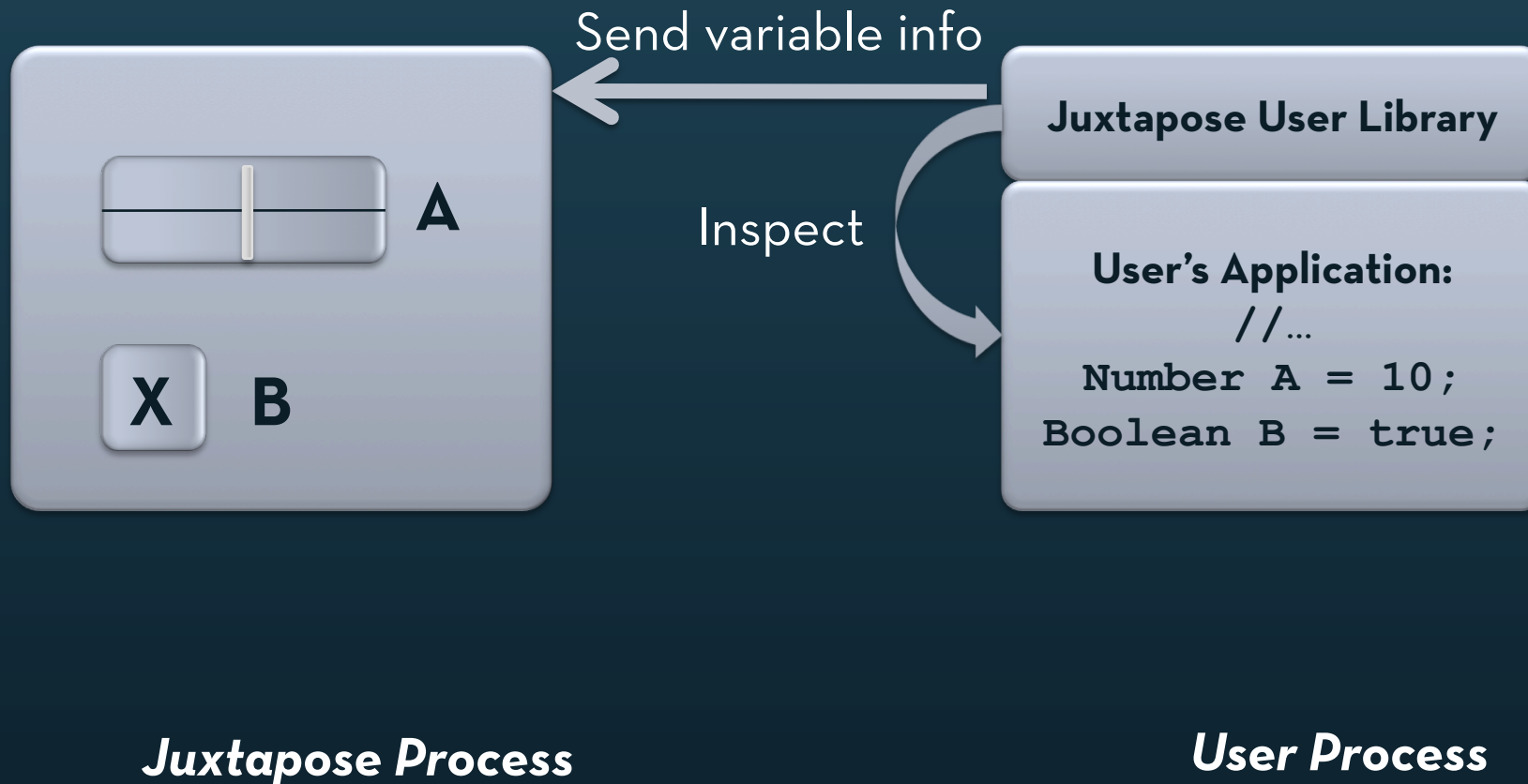
- Four color selection options: Teal, Orange, Cyan, and (unlabeled).
- Three horizontal sliders with green indicators: "Daria" (top right), "Eve" (middle), and "Ana" (bottom left).
- Two dropdown menus at the bottom, showing values "1" and "3".

To the right, a properties window is open, showing the following settings:

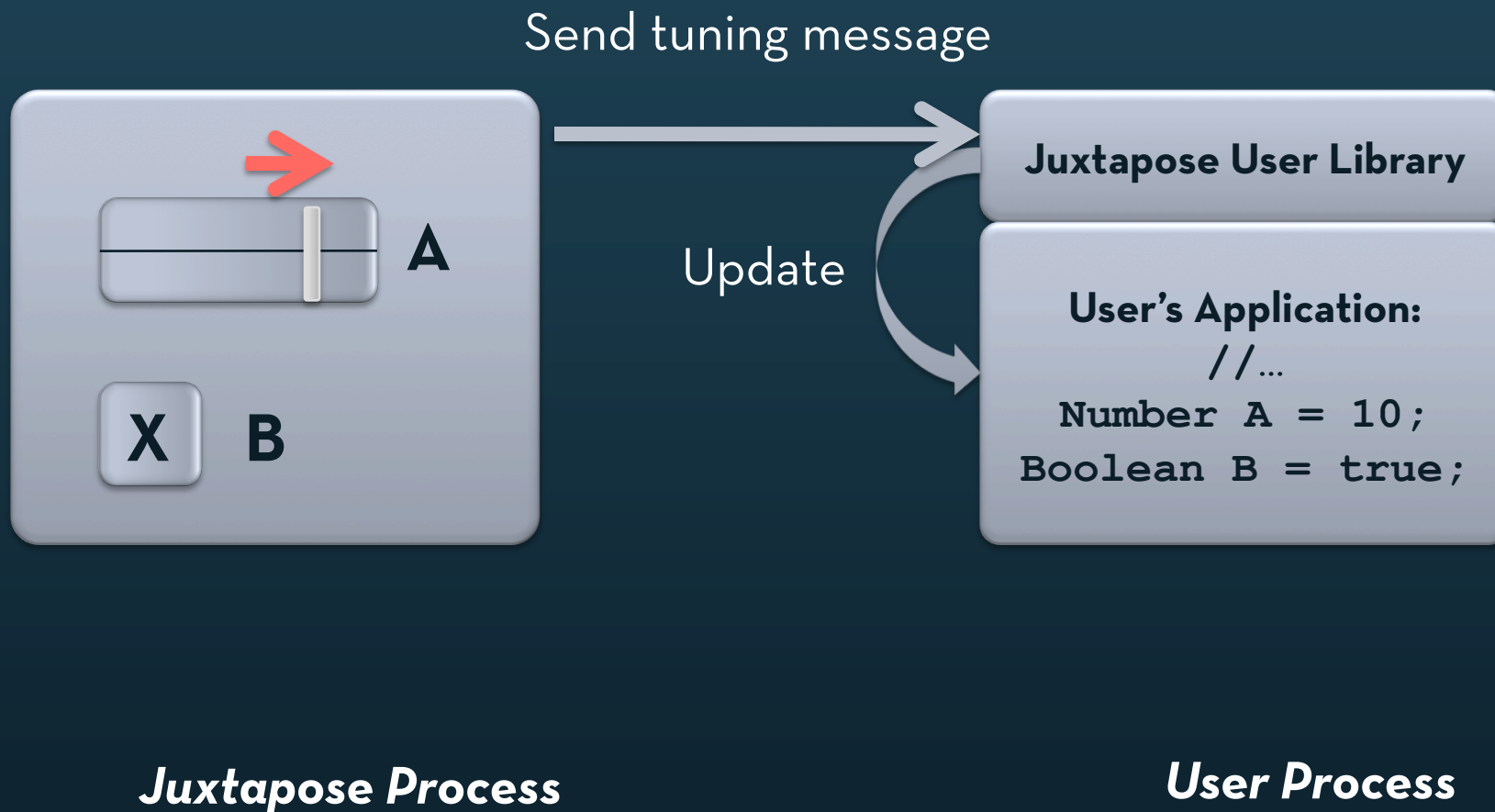
- Linked Tuning
- _childCounter** (number): []
- blueOffset** (number): [-255, 255] with Min: 0, Max: []
- decay** (number): [0, 20] with Min: 10, Max: []
- delay** (number): [0, 20] with Min: [], Max: []
- greenOffset** (number): [-255, 255] with Min: 0, Max: []
- height** (number): []
- redOffset** (number): [-255, 255] with Min: [], Max: []
- tabChildren** (boolean): []
- tabEnabled** (boolean): []

Below the properties window is a "Snapshots" section with a list containing "Initial Configuration".

Generating Tuning Interfaces



Generating Tuning Interfaces



When is tuning most useful?

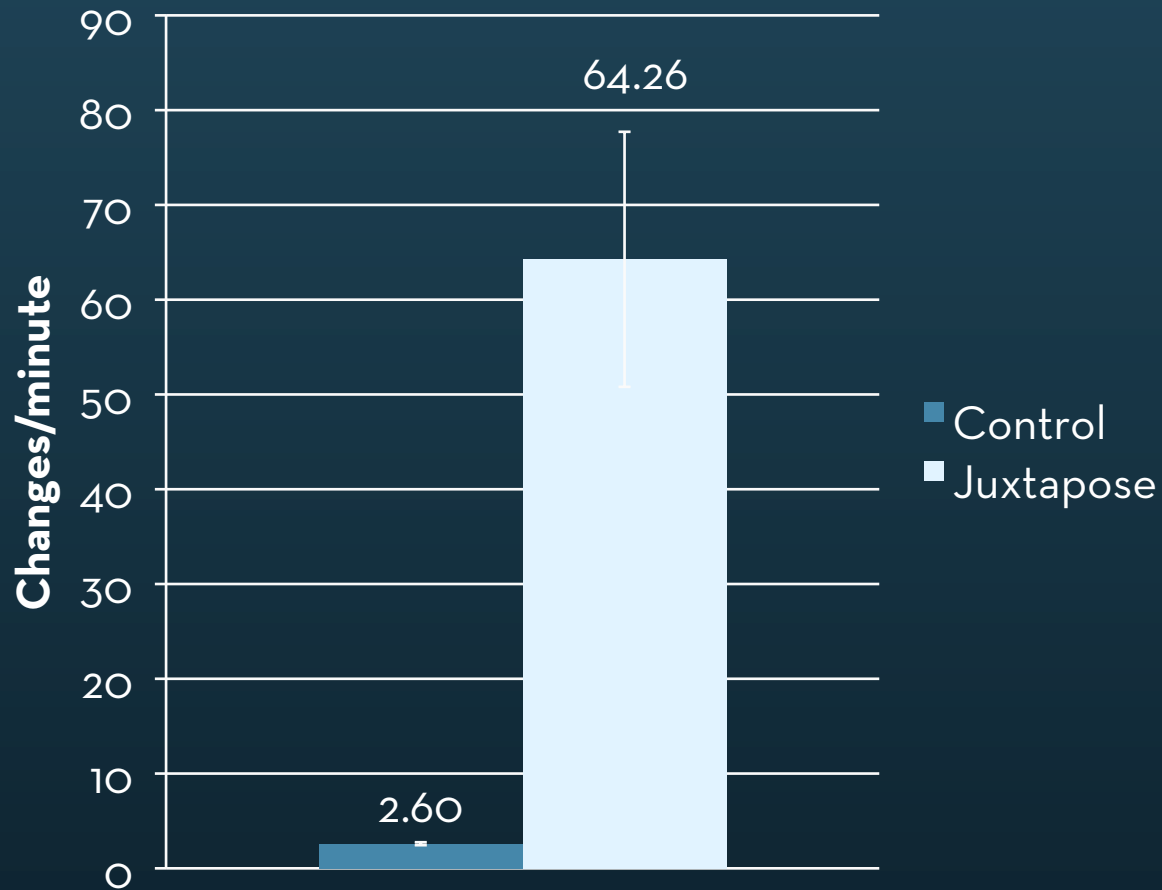
Search in 4D parameter space. Given recursive drawing code for this:



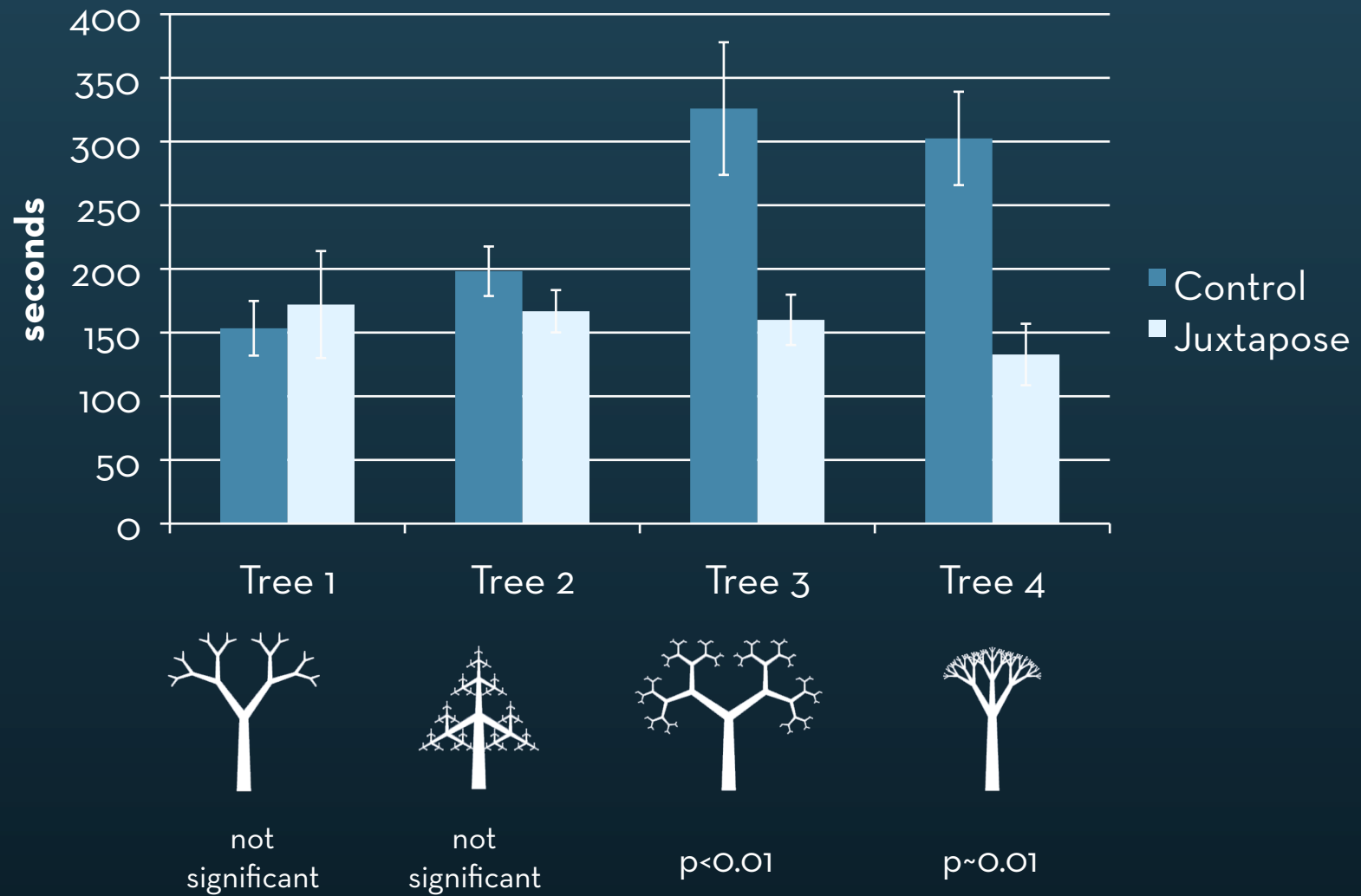
Produce these:



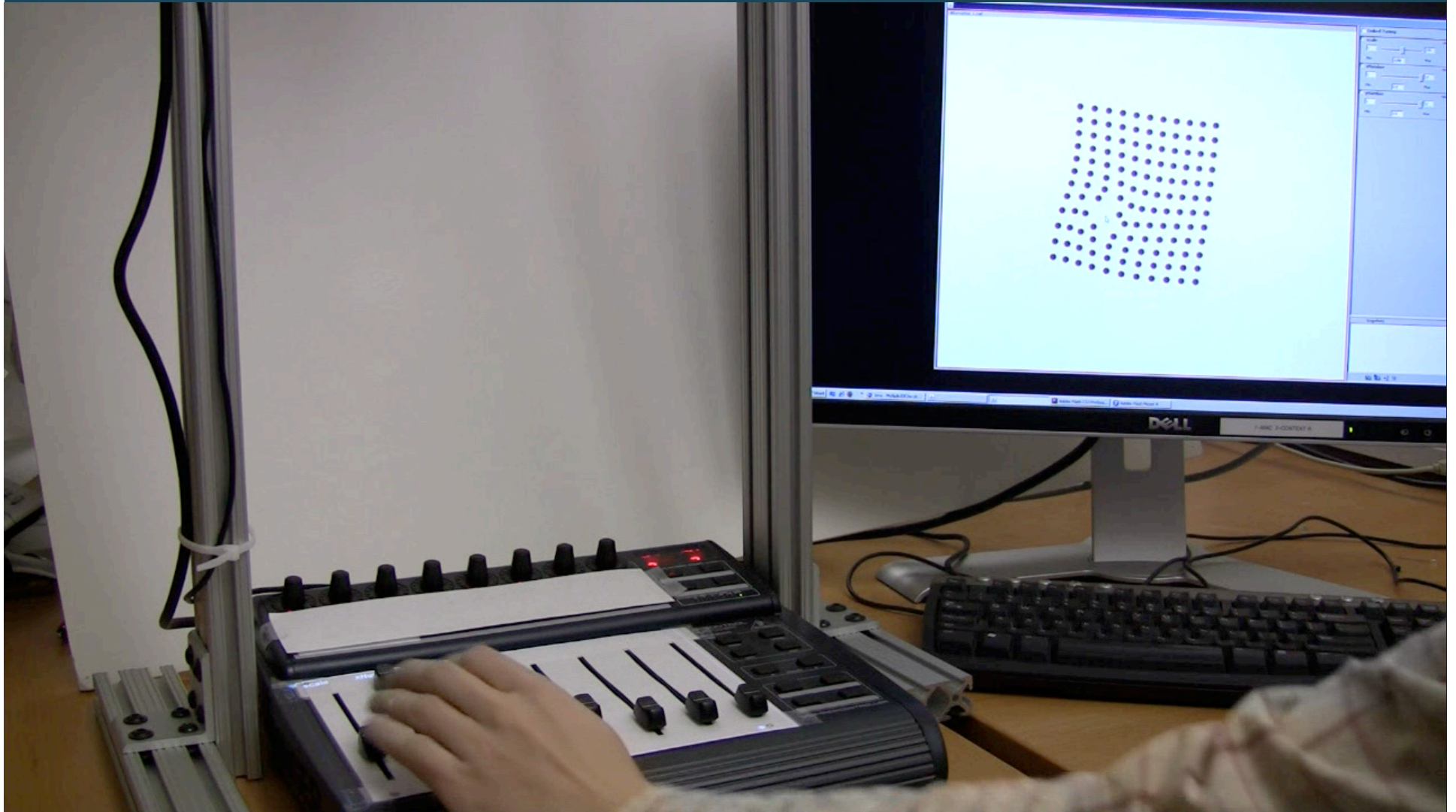
Mean Parameter Changes Tested per Minute



Tree Matching Task: Mean Completion Times by Tree



Tangible Control



**Do these principles hold
beyond desktop GUI
applications?**

Juxtapose Mobile

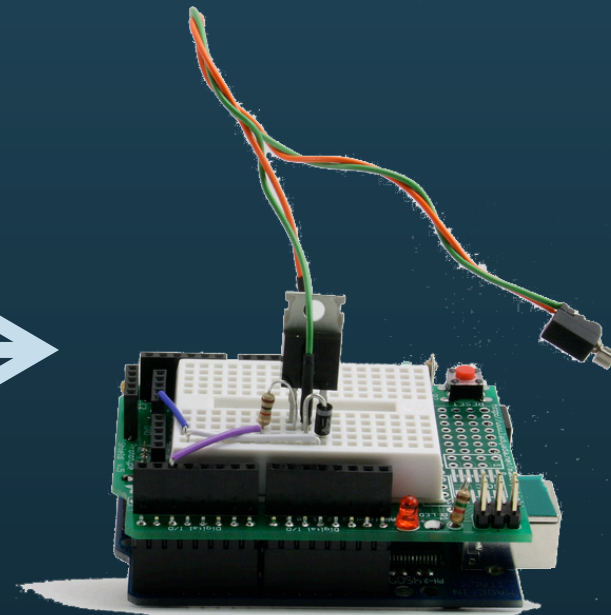
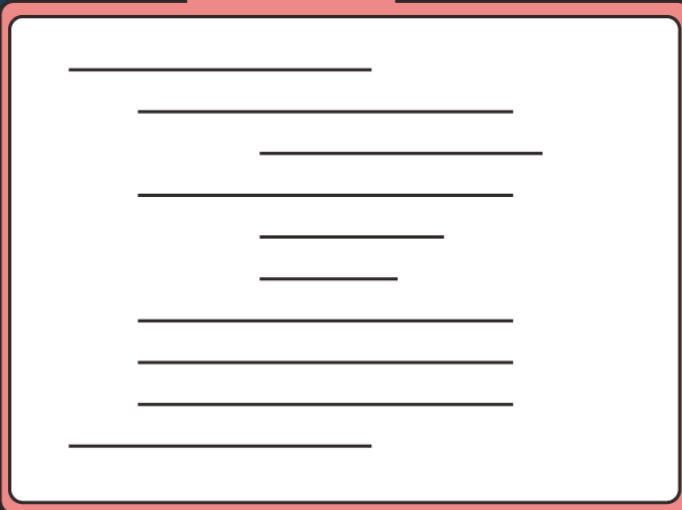
Alt.1 Alt.2 Alt.3



Alternative 1 Alternative 2 Alternative 3

Juxtapose for Microcontrollers

Alt.1 Alt.2 Alt.3



Arduino:
8bit Atmel AVR RISC chip
programmed in C with gcc

*How might one implement variable tuning
in a language without reflection?*

Parse code and build symbol table

```
int dly = 25;  
boolean blink=false;  
void setup() {...}  
//...
```



Var. Name	Type	Pointer
"dly"	int	&dly
"blink"	boolean	&blink

Emit table into code & compile

Auto-generated
table

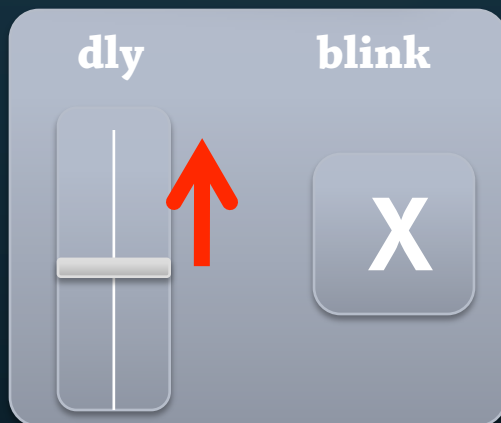
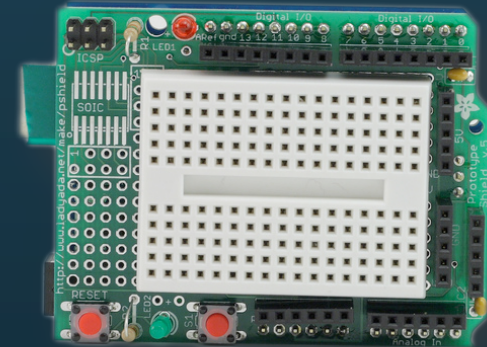


```
void initVarTable(void) {  
    varTable[1].varName = PSTR("dly");  
    varTable[1].varType = VAR_TYPE_INT;  
    varTable[1].varPtr = &dly;  
  
    varTable[2].varName = PSTR("blink");  
    varTable[2].varType = VAR_TYPE_BOOLEAN;  
    varTable[2].varPtr = &blink;  
}  
  
// plus a bunch of communication code...
```

At Runtime



Serial message:
tune "dly" value 100



```
* (varTable["dly"].varPtr)  
= 100;
```

What's Important?

A structured approach to alternatives eliminates craft and enables more exploration.

Tool support requires connecting authoring and execution environments.

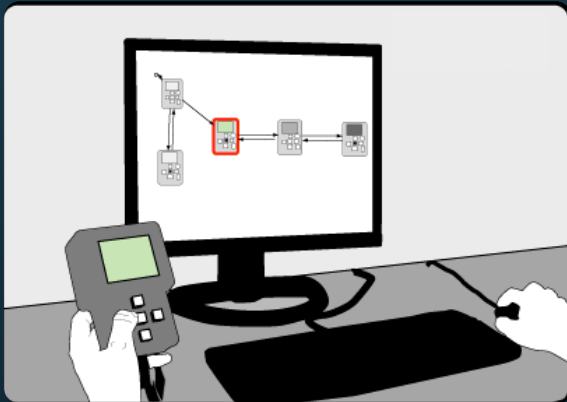
3 Techniques:

- **Linked editing to manage code alternatives**
- **Execute set of programs side-by-side**
- **Auto-generate tuning interface**

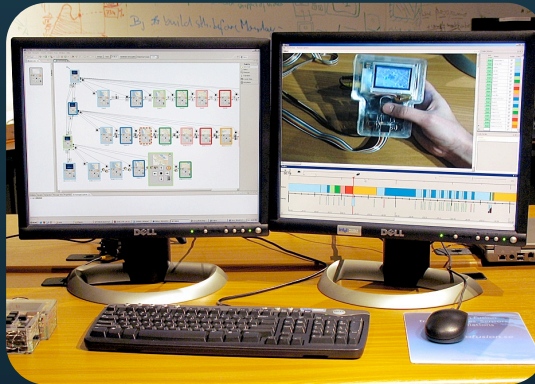
**SUMMARY &
FUTURE AGENDA**

Prototyping Tools

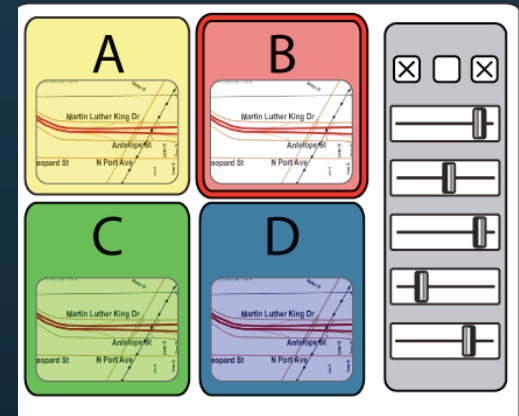
1 Enable **Rapid Authoring**
Off the Desktop



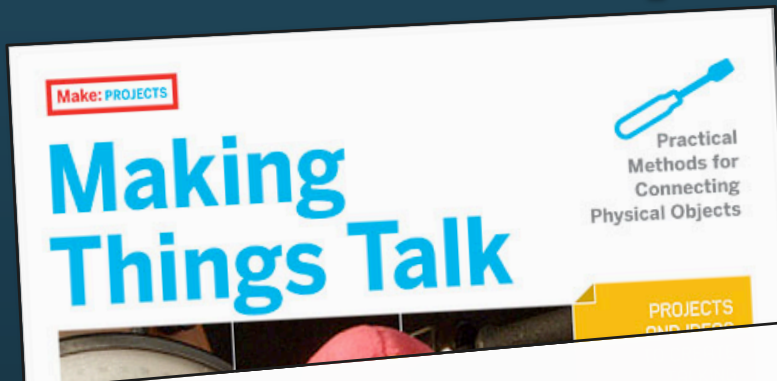
2 Aid **Iteration**
by Managing
Feedback



3 Aid **Exploration**
of Multiple
Alternatives



Community Impact



Integrating User Performance Time Models in the Design of Tangible UIs

Paul Holleis
Embedded Interaction Research Group
University of Munich
80333 Munich, Germany
<http://www.hcilab.org>
pau@hclab.org

Dagmar Kern
Fraunhofer Gesellschaft - IAIS
53757 Sankt Augustin, Germany
dagmar.kern@iais.fraunhofer.de

Albrecht Schmidt
Fraunhofer Gesellschaft - IAIS
53757 Sankt Augustin, Germany
B-IT University of Bonn
53113 Bonn, Germany
<http://iue.bit.uni-bonn.de/>
albrecht.schmidt@acm.org

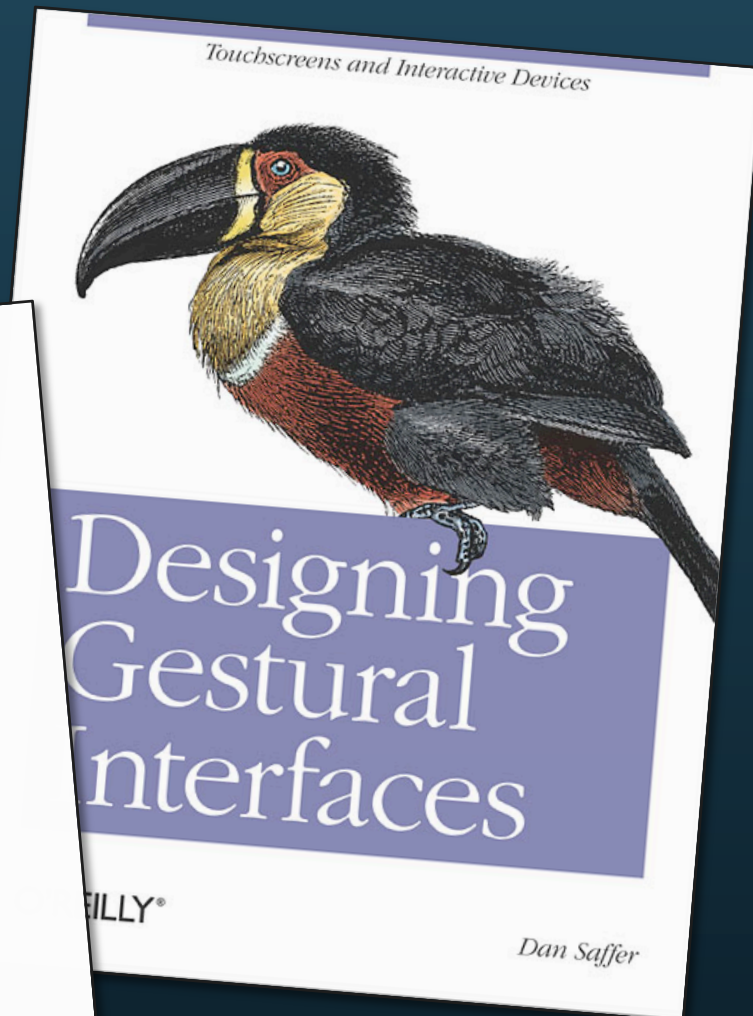
Abstract
One of the aspects that are important for judging an application is the time an experienced user needs to complete a task. This can be assessed by a Keystroke-Level Model (KLM) of that task. In this paper we show a method that allows designers to prototype hardware applications entirely in software and still be able to draw conclusions about the time to completion of given tasks on the envisioned hardware implementation. We provide versatile, easily extensible tools and examples that give developers quick access to KLM data for their prototypes and applications.

Keywords
Platform independent prototyping, user performance times, Keystroke-Level Model (KLM), tool support

ACM Classification Keywords
H.1.2 User/Machine Systems, H.5.2 User Interfaces: Prototyping, D.2.2 Design Tools and Techniques: User interfaces

Introduction / Motivation
The Keystroke-Level Model (KLM) is a specialization of the GOMS [3] family of cognitive modeling approaches. In comparison to other such models, it is relatively easy to apply to application ideas and designs since it is stripped down to only those operators necessary to make user performance time predictions. The KLM

Copyright is held by the author/owner(s).
CHI 2007, April 28-May 3, 2007, San Jose, California, USA.
ACM 978-1-59593-642-4/07/0004.



Research Agenda

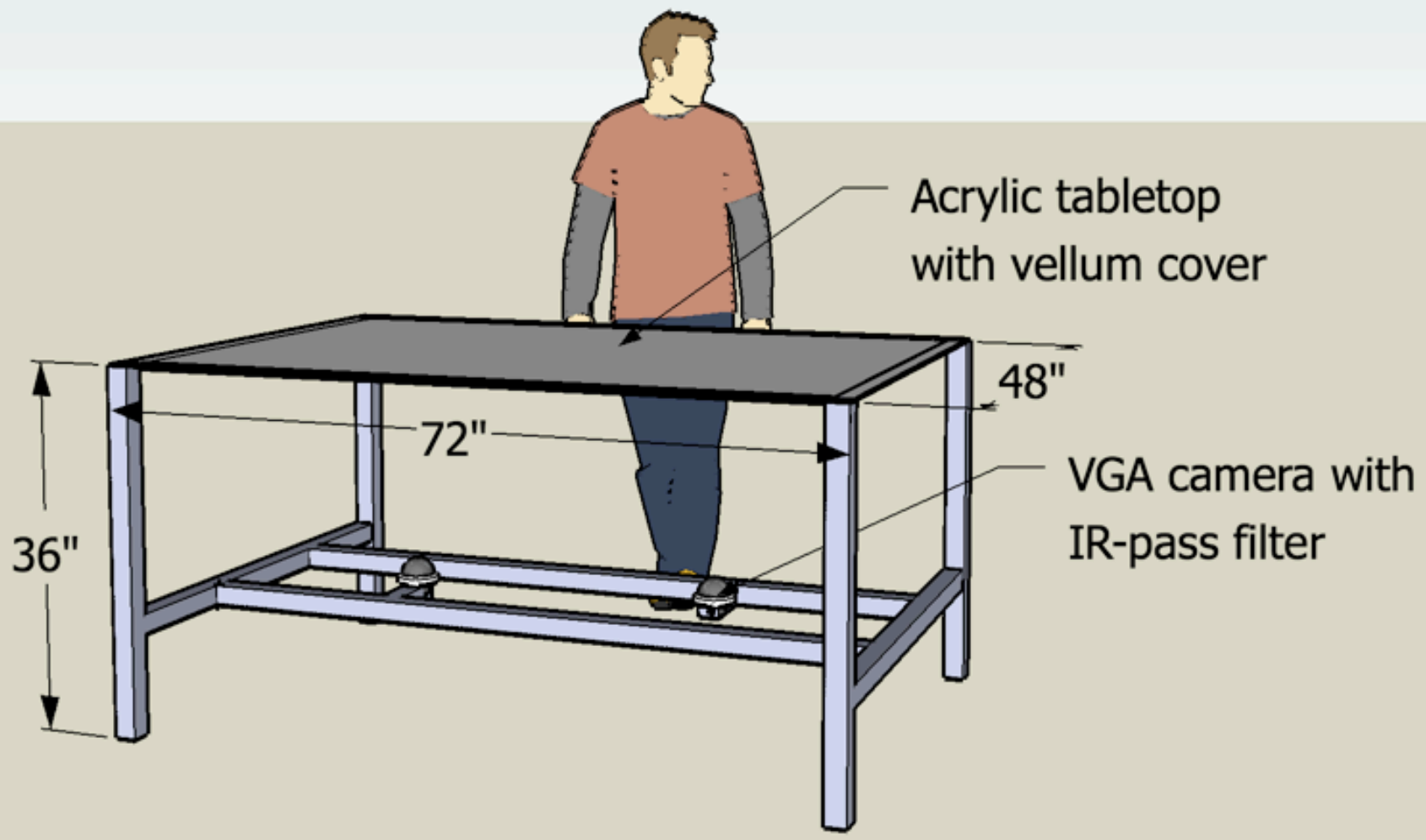
- How can **programming languages & IDEs** be **co-designed** to facilitate prototyping?
- How might authoring environments support the process of design by **example modification**?
- What will the interaction **design studio of the future** look like?





SLR Camera

XGA projector with 45 degree mirror



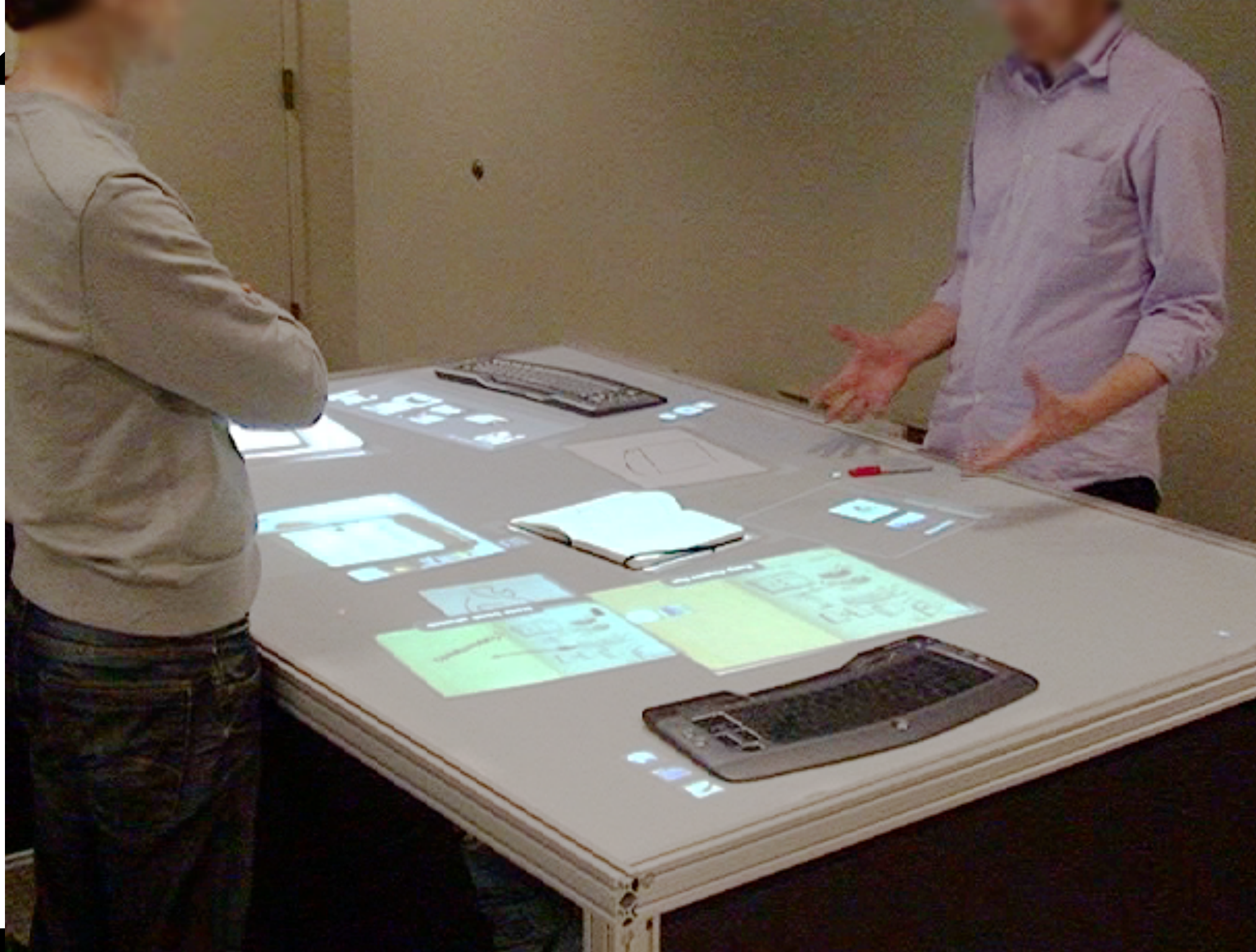
Acrylic tabletop with vellum cover

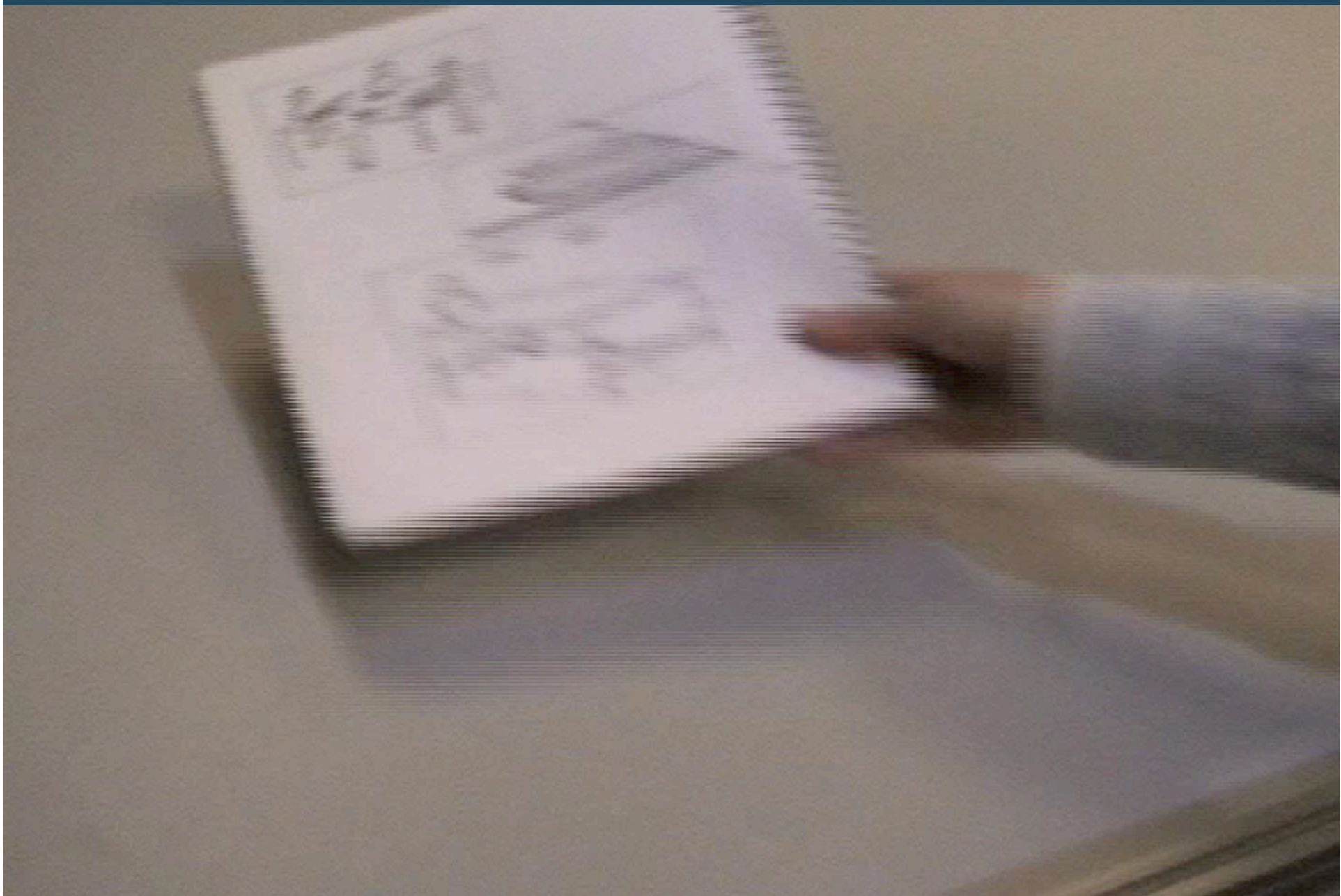
VGA camera with IR-pass filter

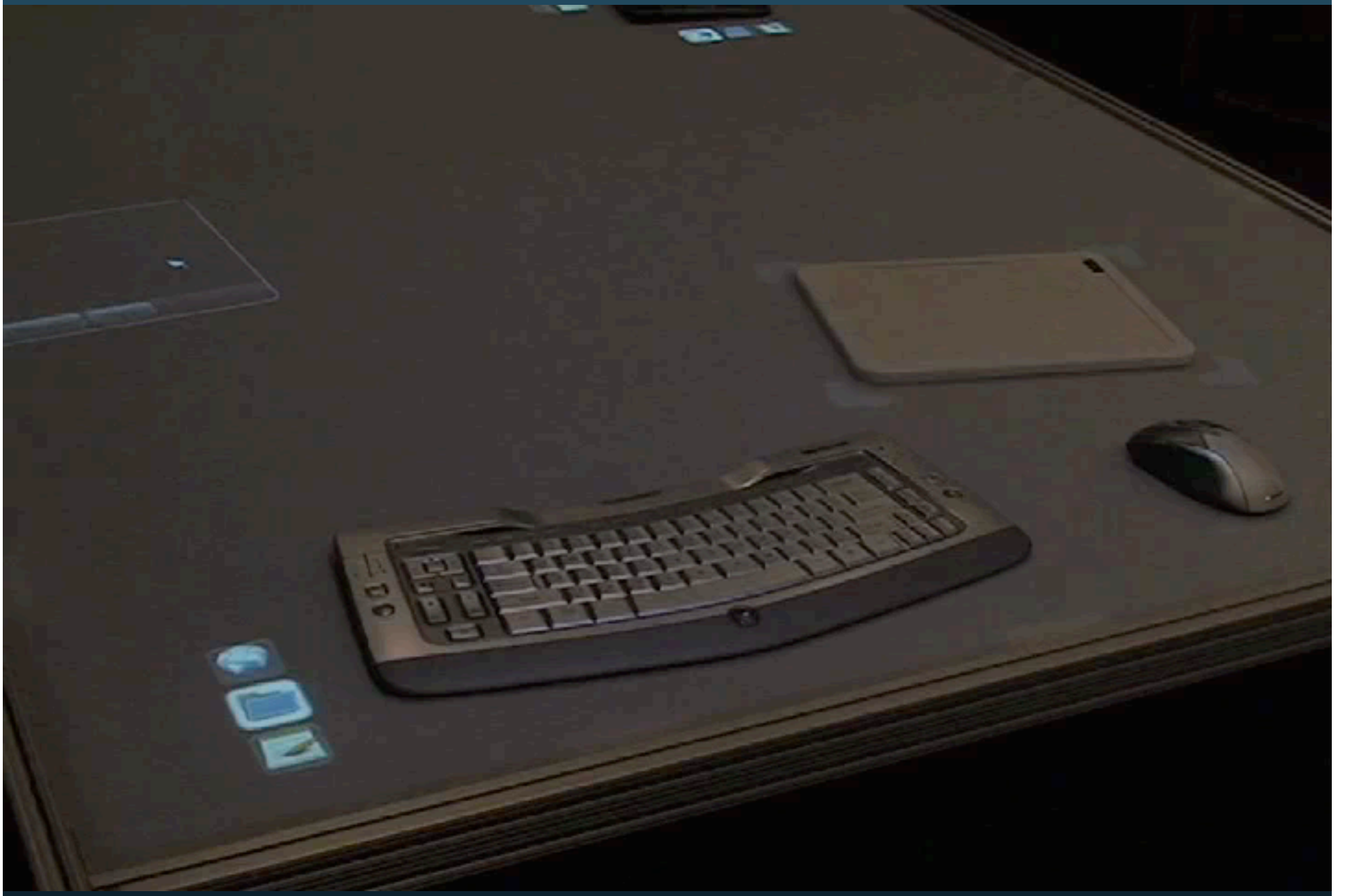
36"

72"

48"









**Leith Abdulla
Abel Allison
Marcello Bastea-Forte
Hrvoje Benko
Michael Bernstein
Brandon Burr
Timothy Cardenas
Kevin Collins
Scott Doorley
Sean Follmer
Jennifer Gee
Scott R. Klemmer
Nirav Mehta
Manas Mittal
Merrie Morris**

**Avi Robinson-Mosher
Anthony Ricciardi
Andrew Wilson
Leslie Wu
Yeonsoo Yang
Loren Yu**

**Frog Design
IDEO
LeapFrog
Nokia
San Jose Tech Museum
San Jose Museum of Art**

Hartmann, Björn, Loren Yu, Abel Allison, Yeonsoo Yang, and Scott R. Klemmer. Design as Exploration: Creating Interface Alternatives through Parallel Authoring and Runtime Tuning. In Proceedings of uist 2008: ACM Symposium on User Interface Software and Technology. Monterey, CA, 2008. **(best student paper award)**

Hartmann, Björn, Leslie Wu, Kevin Collins and Scott R. Klemmer. Programming by a Sample: Rapidly Creating Web Applications with d.mix. In Proceedings of uist 2007: ACM Symposium on User Interface Software and Technology. Newport, Rhode Island, USA, 2007.

Hartmann, Björn, Leith Abdulla, Manas Mittal and Scott R. Klemmer. Authoring Sensor Based Interactions Through Direct Manipulation and Pattern Matching. In Proceedings of chi 2007: ACM Conference on Human Factors in Computing Systems. San Jose, CA, 2007. **(best paper award)**

Hartmann, Björn, Scott R. Klemmer, Michael Bernstein, Leith Abdulla, Brandon Burr, Avi Robinson-Mosher, and Jennifer Gee. Reflective physical prototyping through integrated design, test, and analysis. In Proceedings of uist 2006: ACM Symposium on User Interface Software and Technology. Montreux, Switzerland, 2006. **(best paper award)**



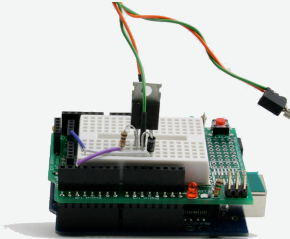
Klemmer, Scott R., Björn Hartmann, and Leila Takayama. How Bodies Matter: Five Themes for Interaction Design. In Proceedings of dis 2006: ACM Conference on the Design of Interactive Systems. State College, PA, 2006.

Funding Acknowledgments:

Stanford MediaX/DNP Grant; NSF grant IIS-0534662
Stanford School of Engineering & SAP Graduate Fellowships
Equipment donations from Intel & Nokia

<http://bjoern.org>

So What's Different?

Target Platform	How are alternatives executed? Why?	How do users interact with alternatives?
	In parallel	Sequentially or in parallel
	In parallel, because commodity hardware is (relatively) cheap	Sequentially
	Sequentially, because custom hardware is expensive to build	Sequentially

Juxtapose Mobile

