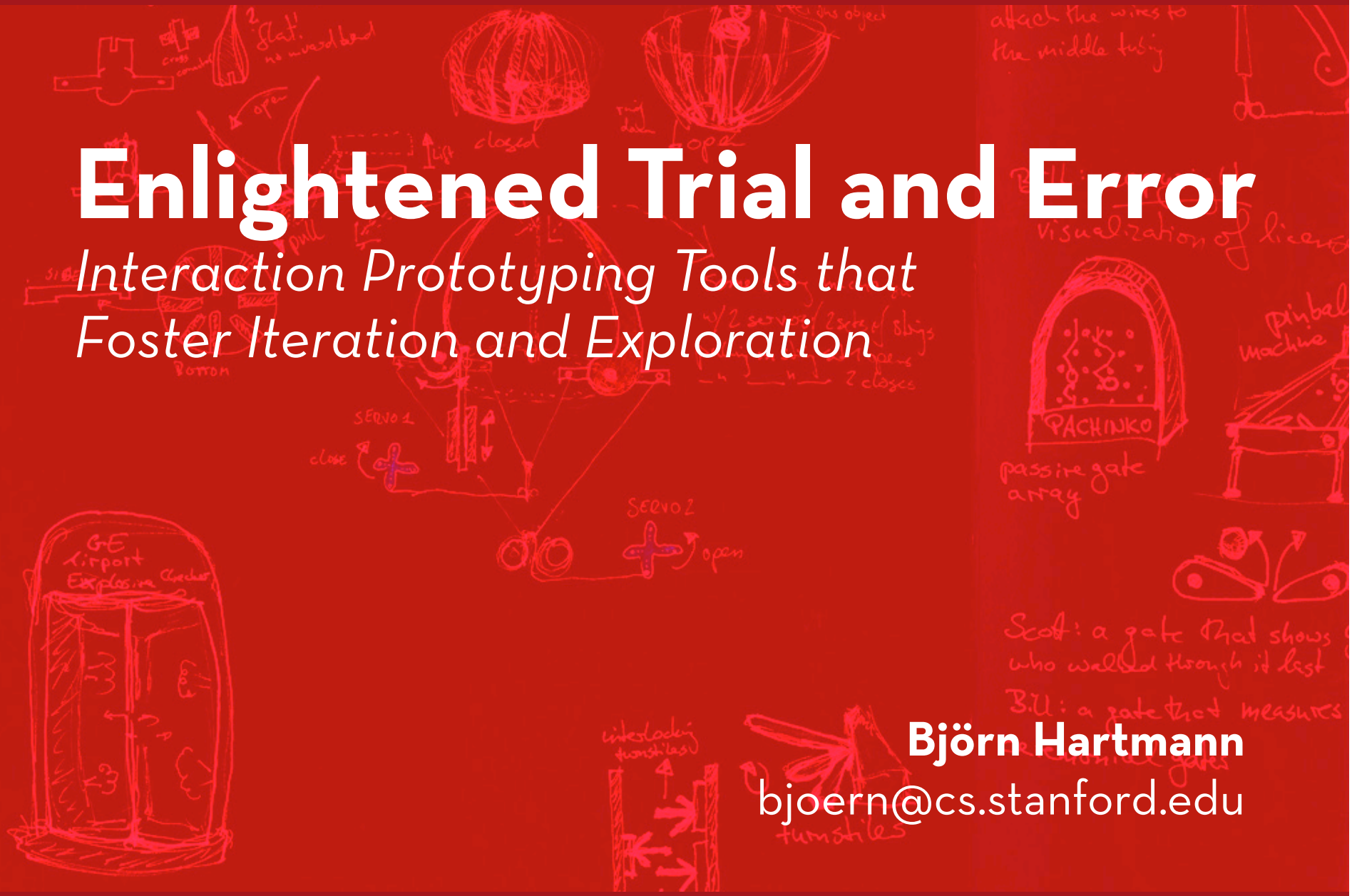
STANFORD HCI GROUP

FEB 11, 2009

# Enlightened Trial and Error

## *Interaction Prototyping Tools that Foster Iteration and Exploration*

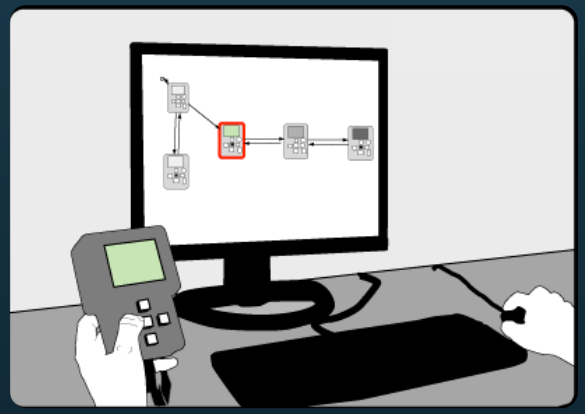



**Björn Hartmann**

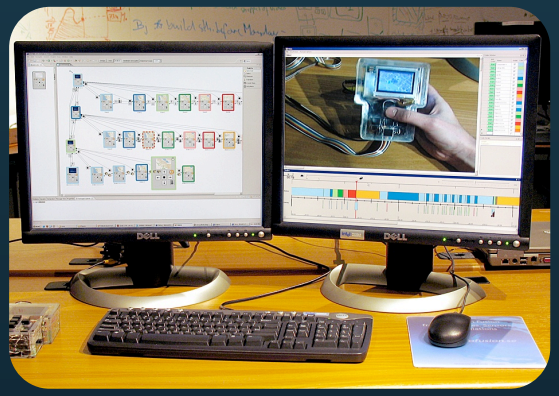
bjoern@cs.stanford.edu

# Prototyping Tools

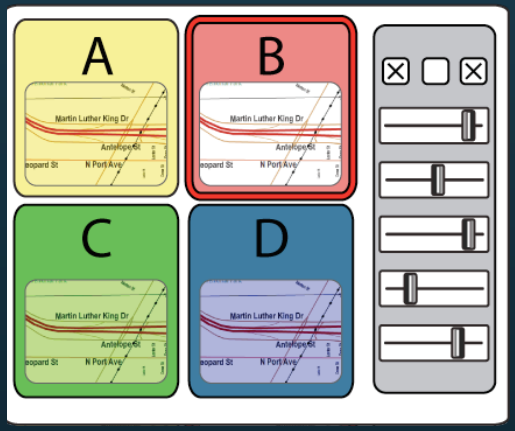
**1** Enable **Rapid Authoring** Off the Desktop



**2** Aid **Iteration** by Managing Feedback



**3** Aid **Exploration** of Multiple Alternatives



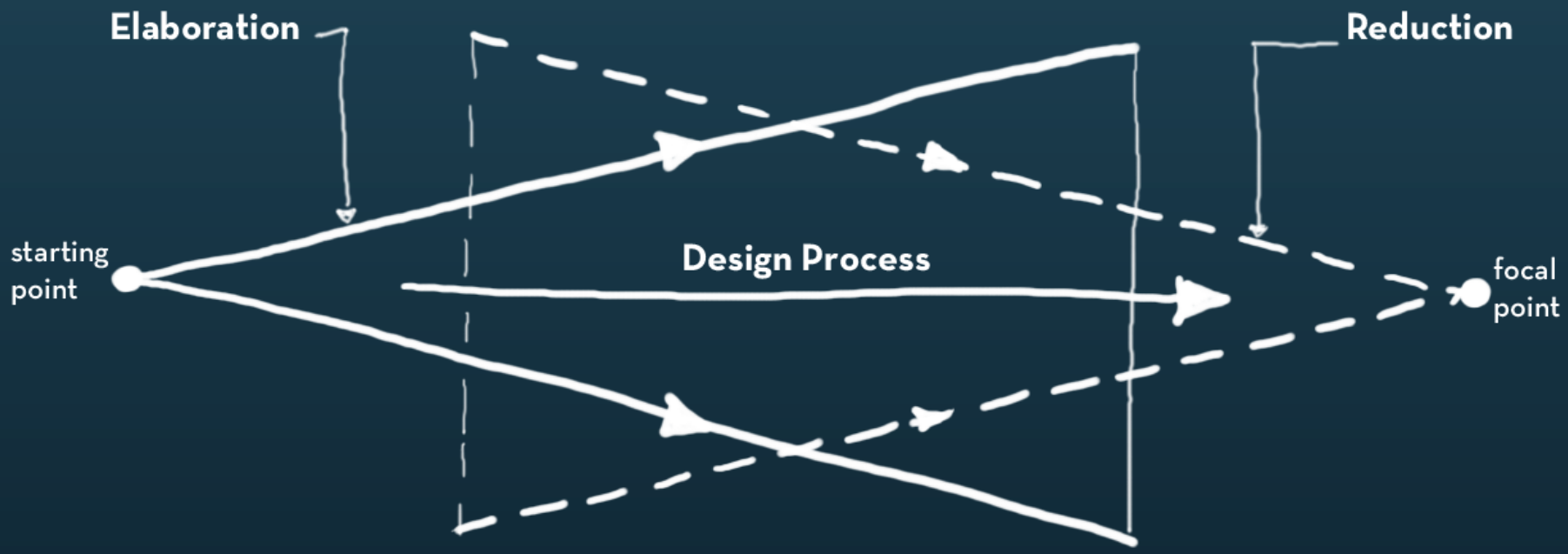
**Prototyping** is the central activity that structures the design process.

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

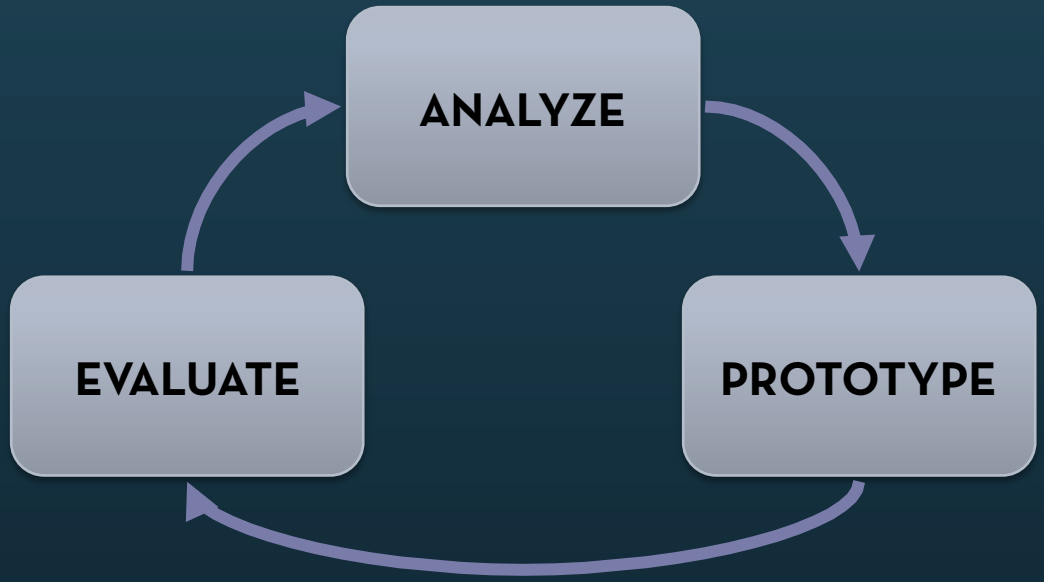
# Exploration:
# Generate Parallel Alternatives




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
prototype iterati
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
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

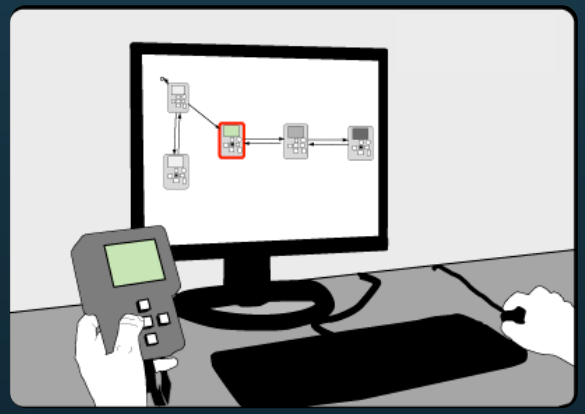
  (Cross, Designerly Ways of Knowing)
- Decisions made with insufficient knowledge
- Gained knowledge is lost or not acted on
- Resources wasted in working *around* tools

---

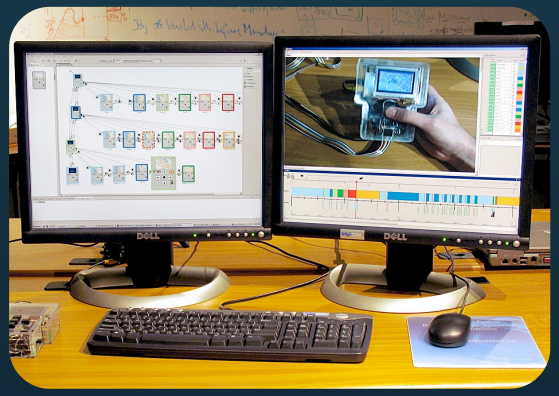
Problems uncovered too late

**Poor solutions, failed products**

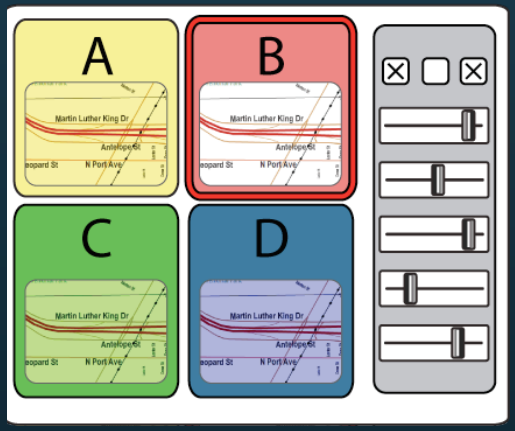
## 1 Enable **Rapid Authoring** Off the Desktop



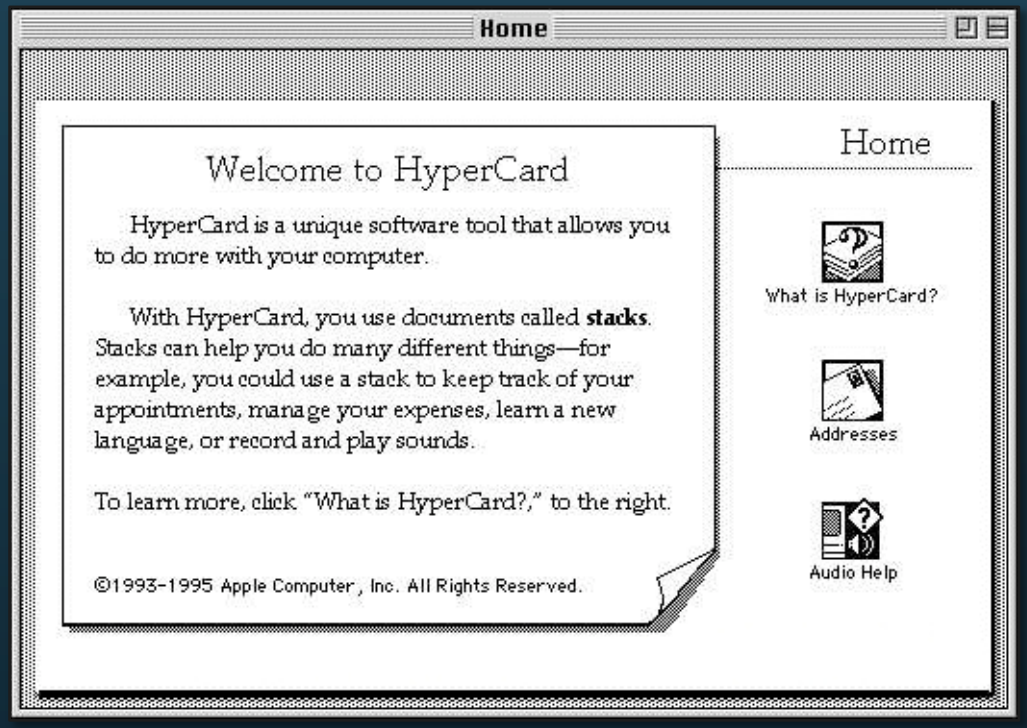
## 2 Aid **Iteration** by Managing Feedback



## 3 Aid **Exploration** of Multiple Alternatives



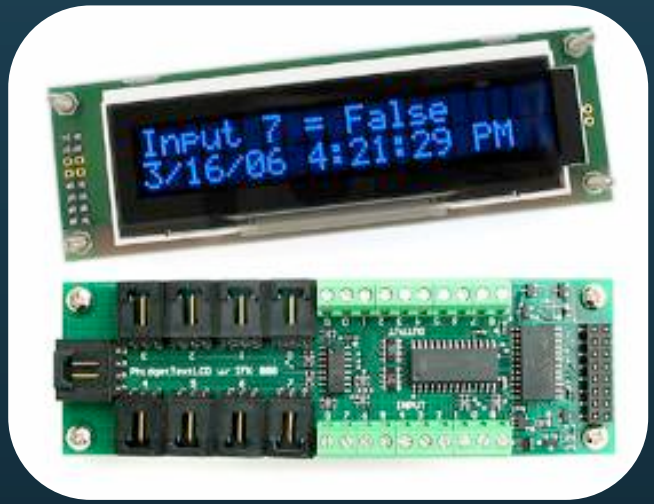
# Related Work: Rapid Authoring of GUIs




***Hypercard, Atkinson 1985***

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

# Related Work: Rapid Authoring of ~~G~~UIs



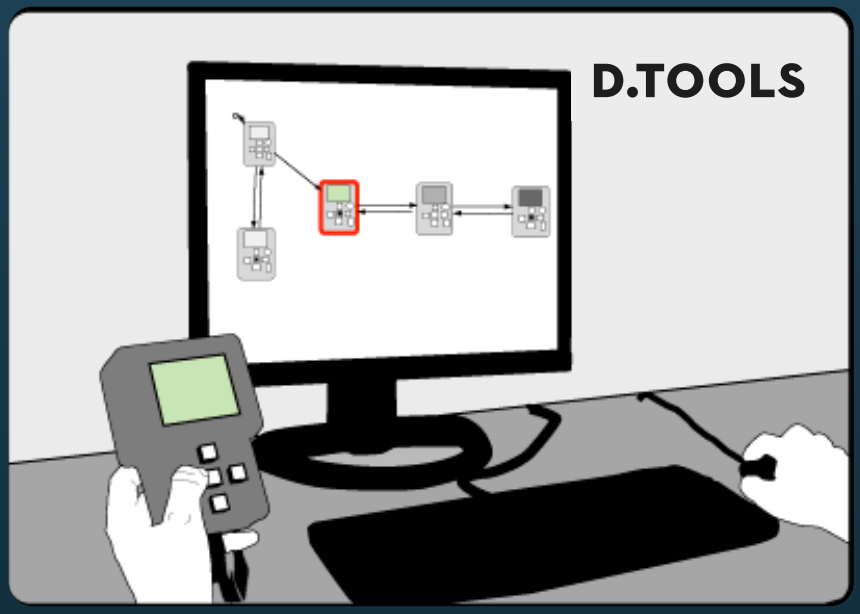

***Phidgets, Greenberg, 2001***

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



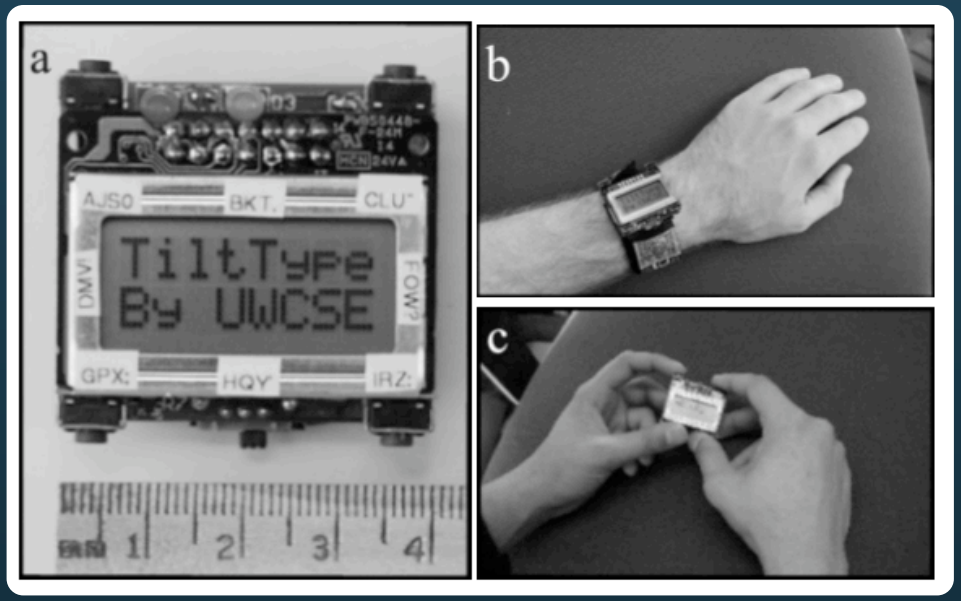
# Example: Mobile Text Entry

**If |Buttons| < |Letters|:**

1. Select alphabet subset
2. Select letter from subset

**Available Controls:**

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

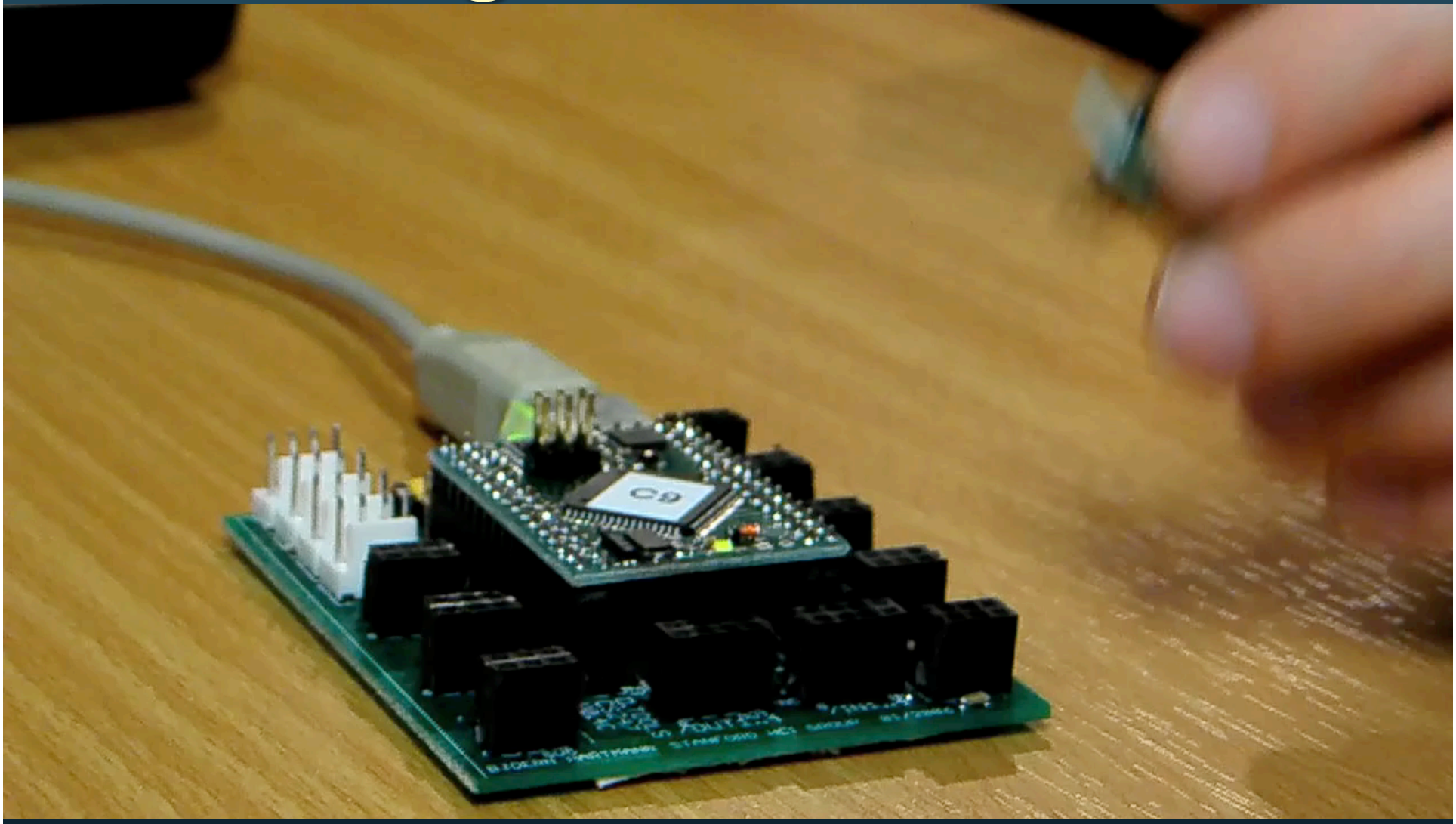



TiltType, Partridge et al., UIST 2002

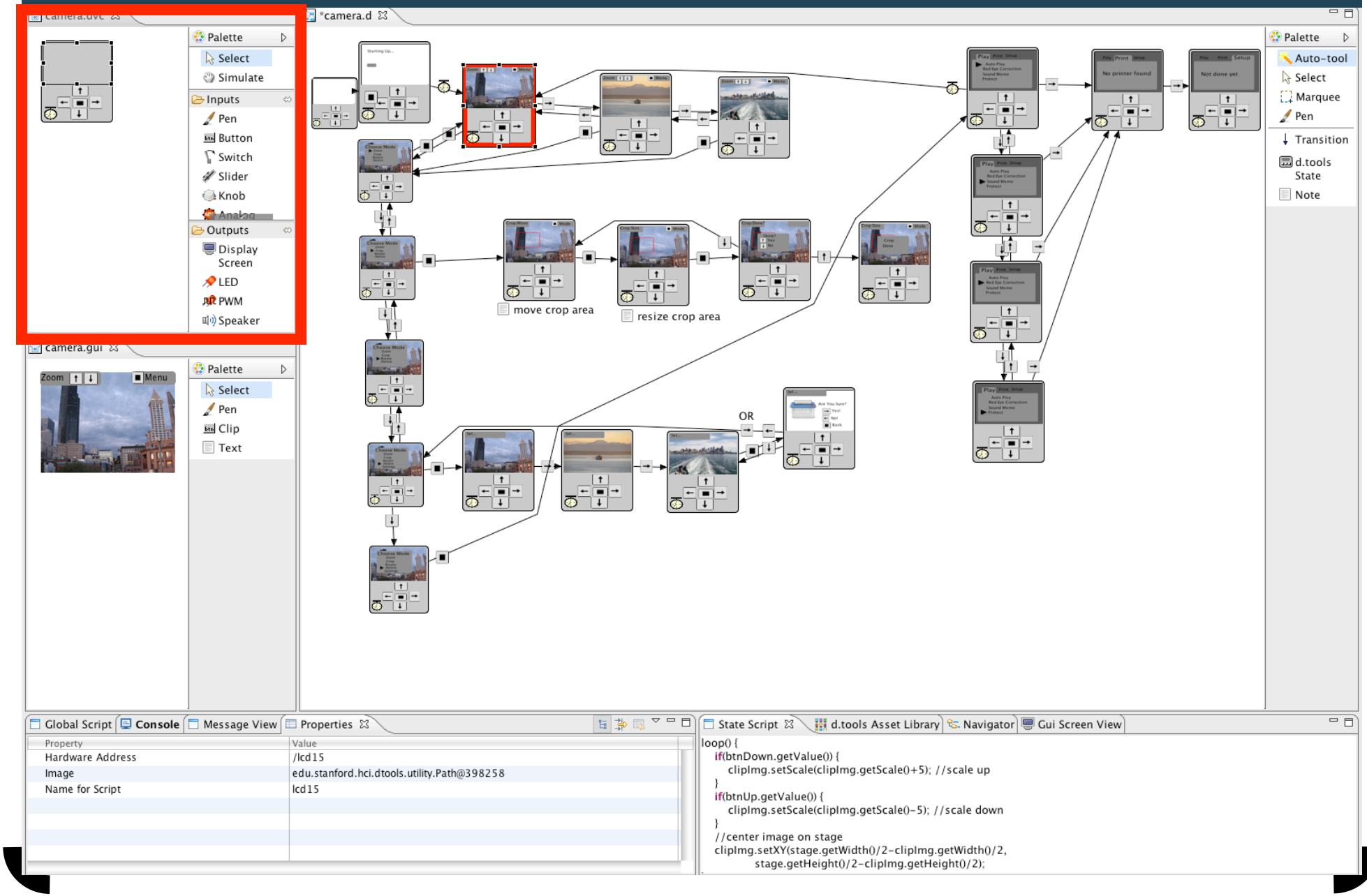
# Example: Tilt-based text entry


Option 1:
Select set with buttons,
character with tilt
A B C
D E F
G H I
J K L
M N O
P Q R
delete
TiltType
m

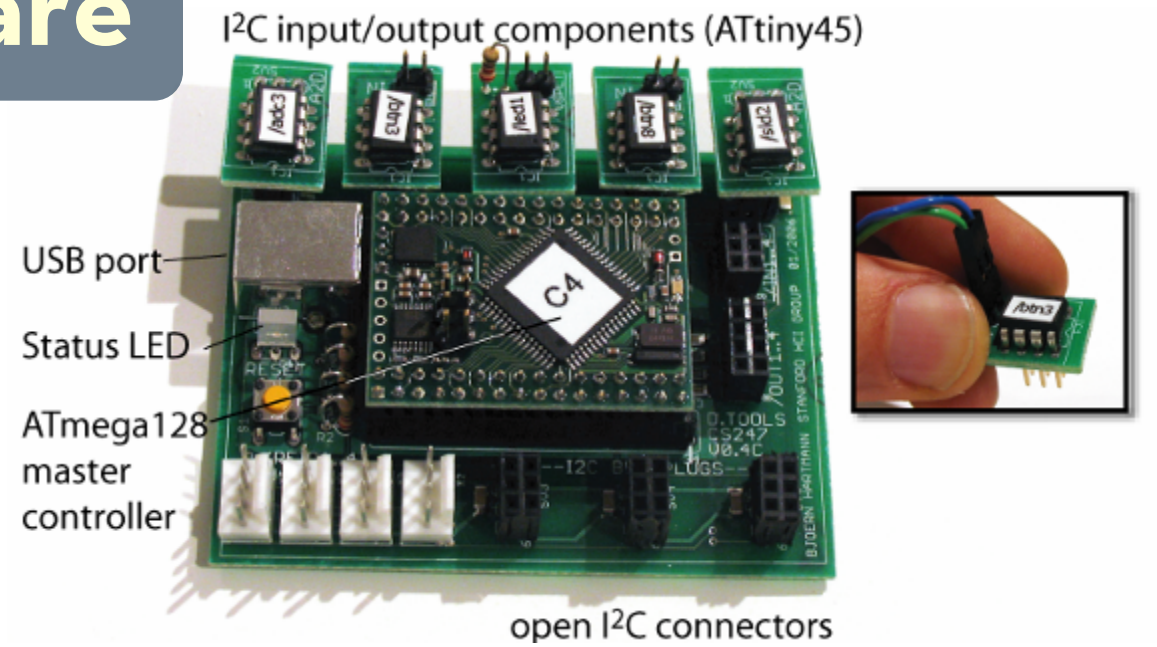
# Authoring with d.tools

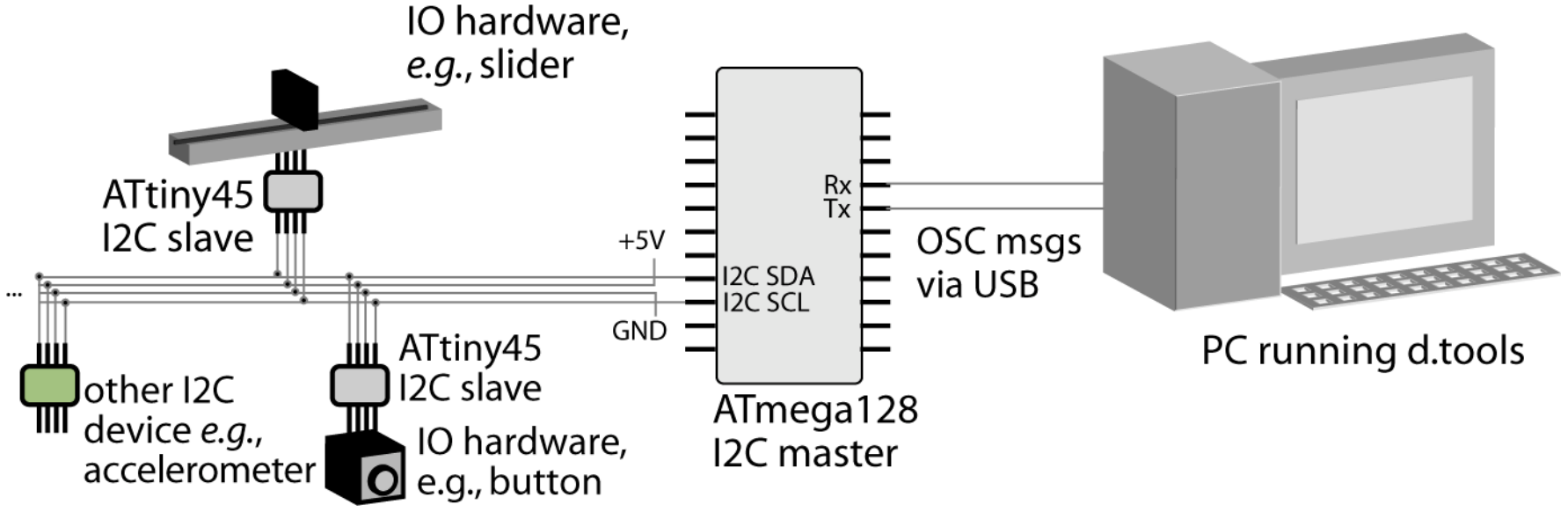


# Design Environment

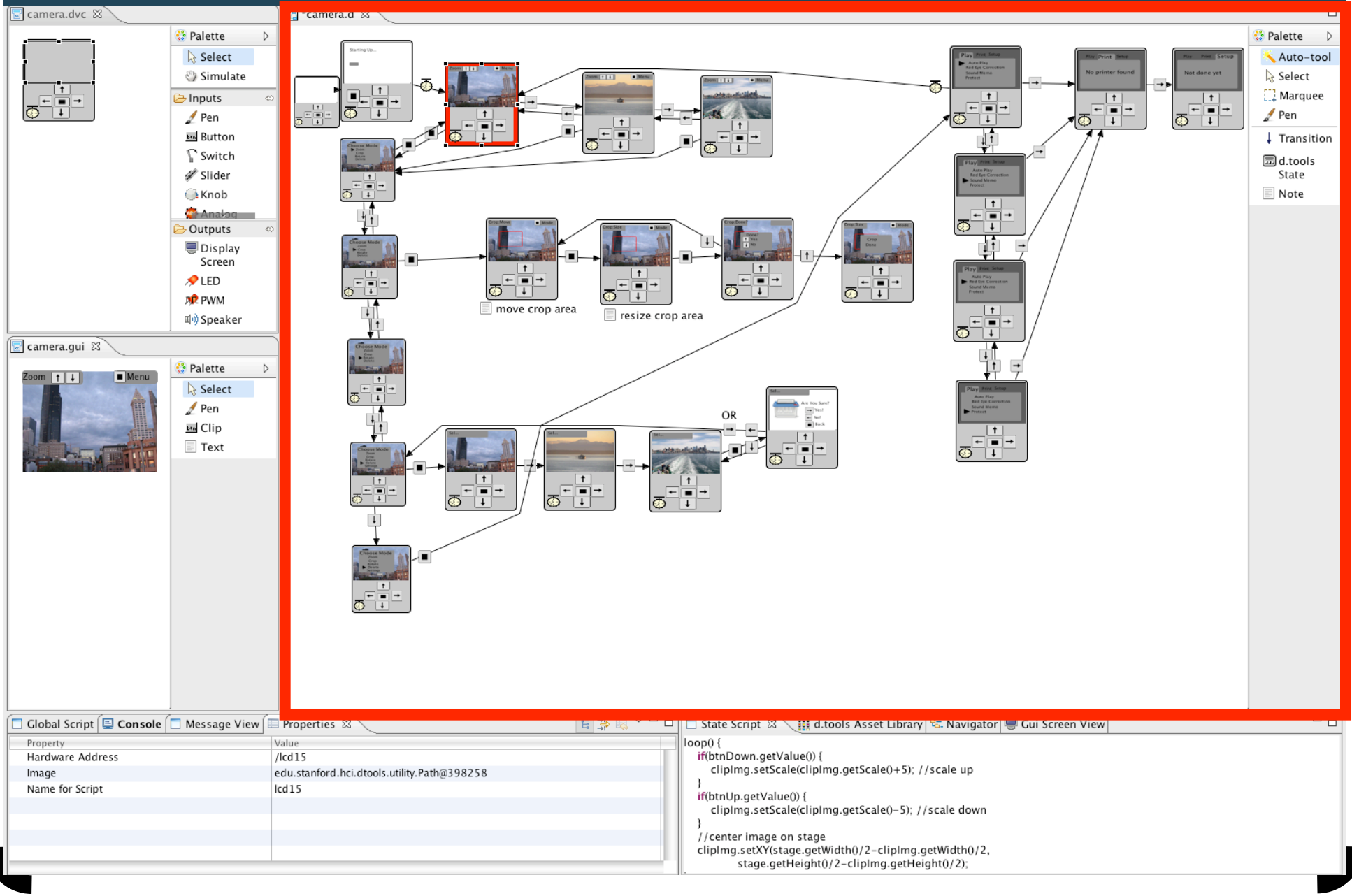


# Smart Hardware

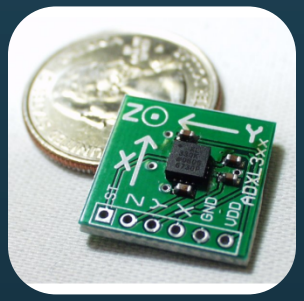


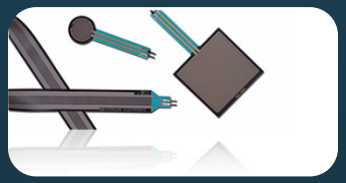
# Design Environment



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

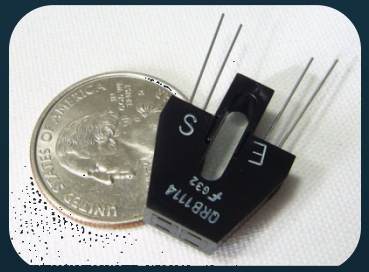



Accelerometers



Force Sensitive Resistors

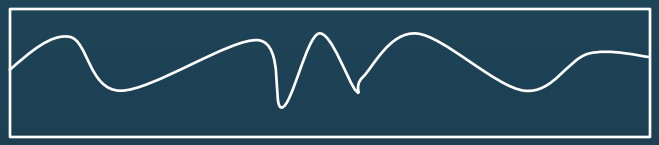

IR/Ultrasonic Rangers



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 ?

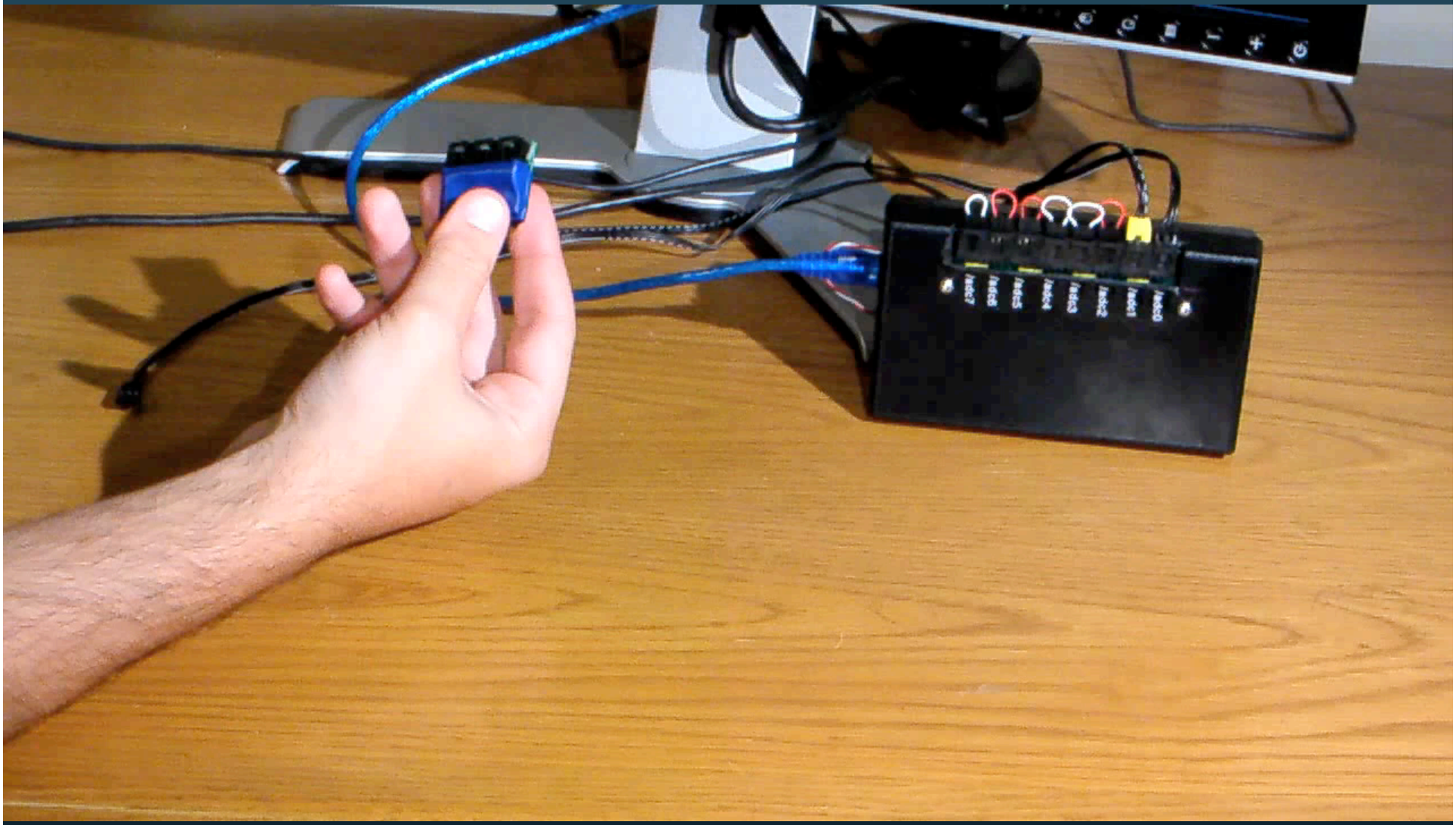
- Yes, algorithms enable demonstration-based authoring,  but are only the backend.
- What are appropriate interaction techniques that make real-time demonstration and testing feasible for designers?
- Insight: Combine recognition and direct manipulation to leverage the designer's knowledge

# Related Work: Interfaces for Programming by Demonstration

Crayons, Fails, CHI 2003

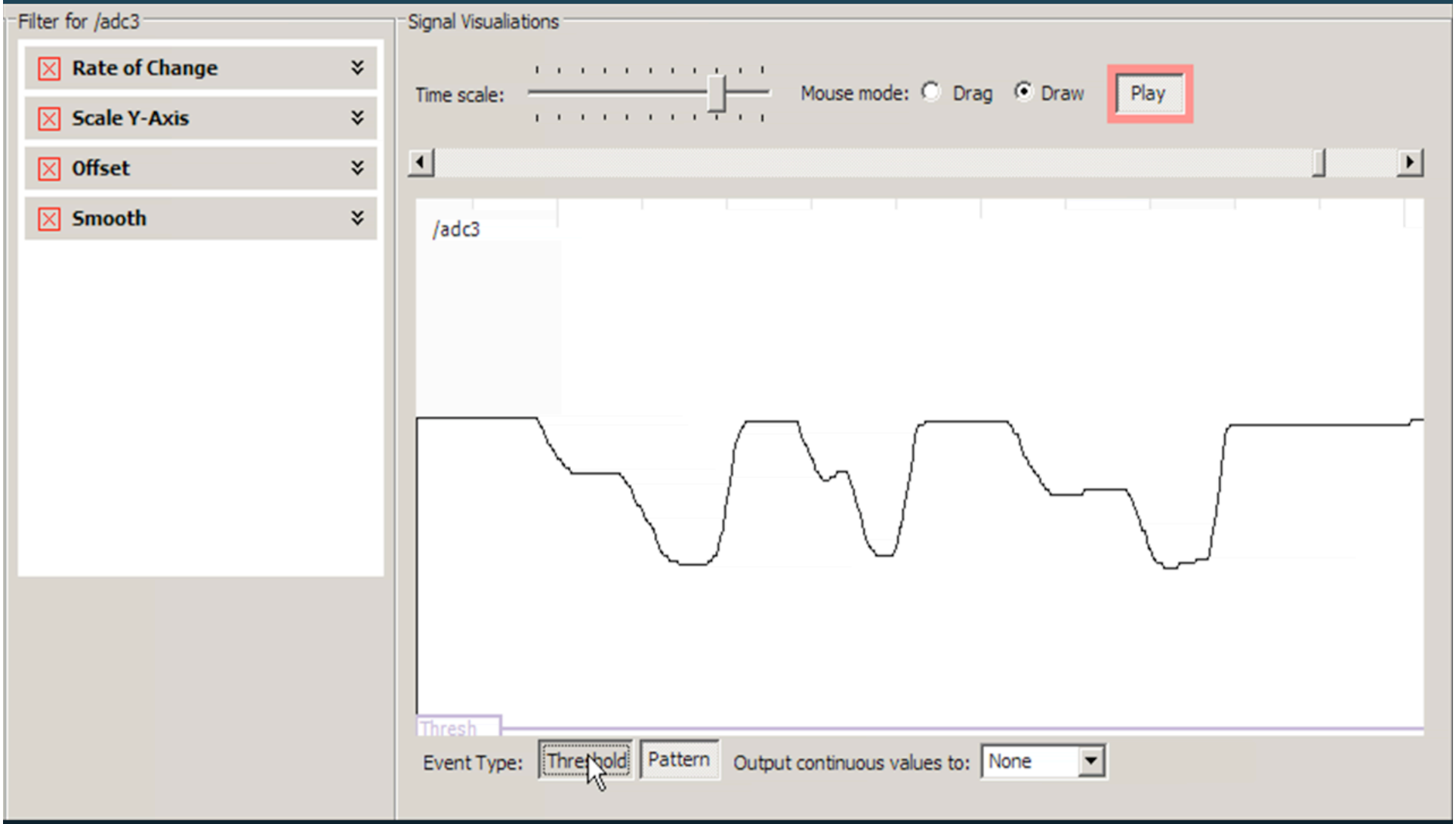
Also:
Flexigesture, Merrill, CHI 2005
a CAPella, Dey, CHI 2004
Monet, Li, UIST 2005

# Authoring by Demonstration

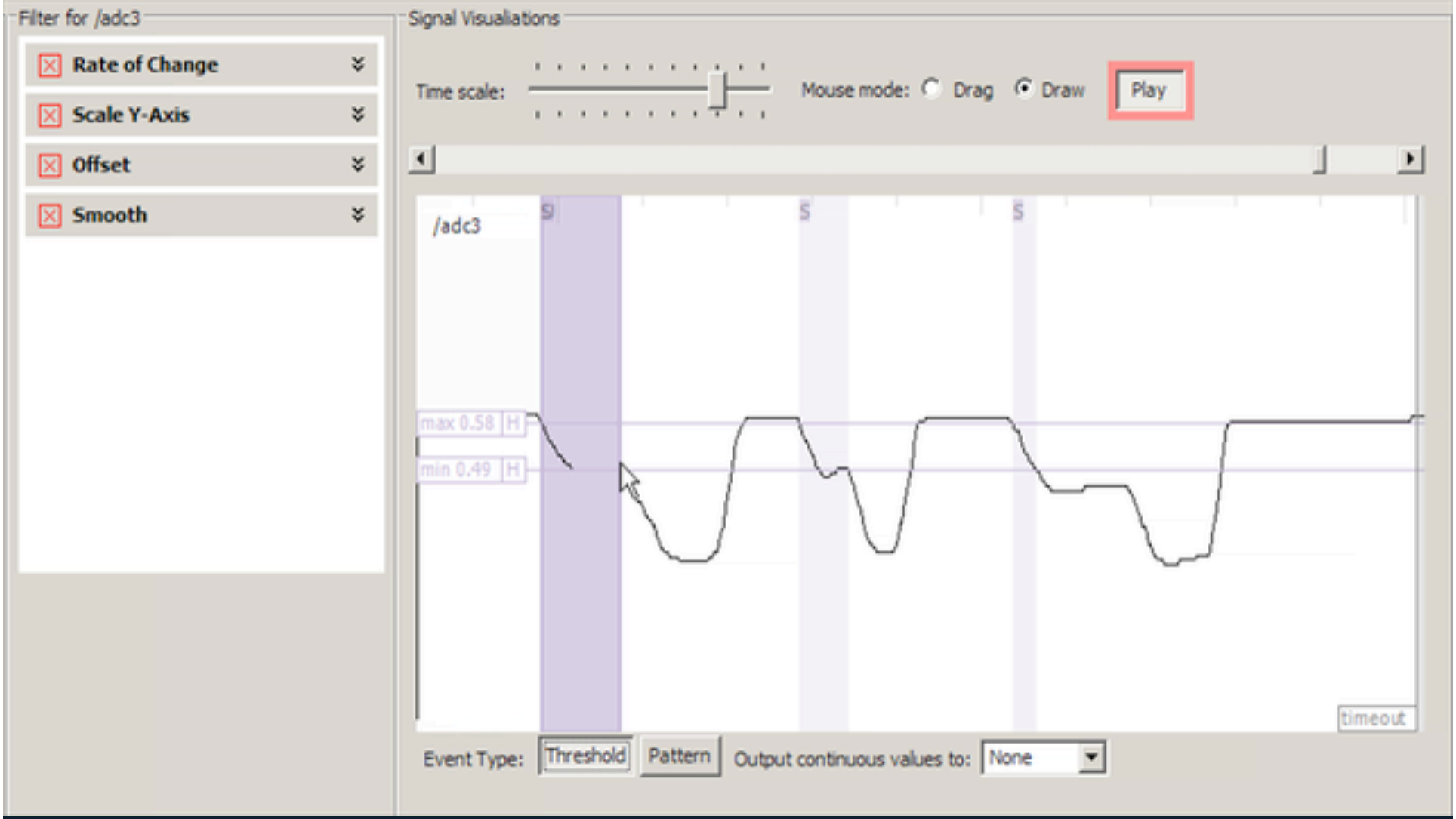


[CHI 2007]

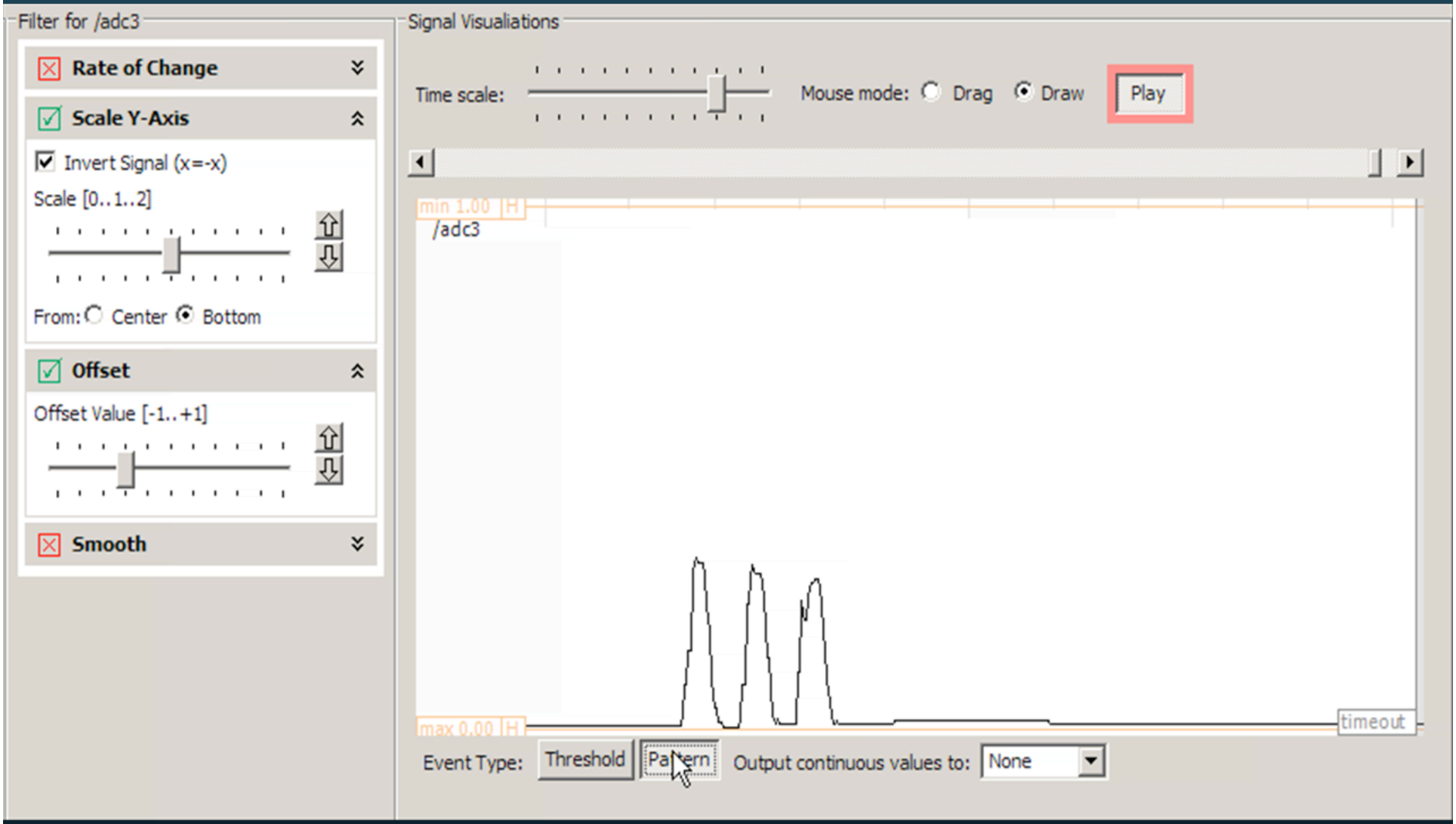
# Generalization: Thresholds



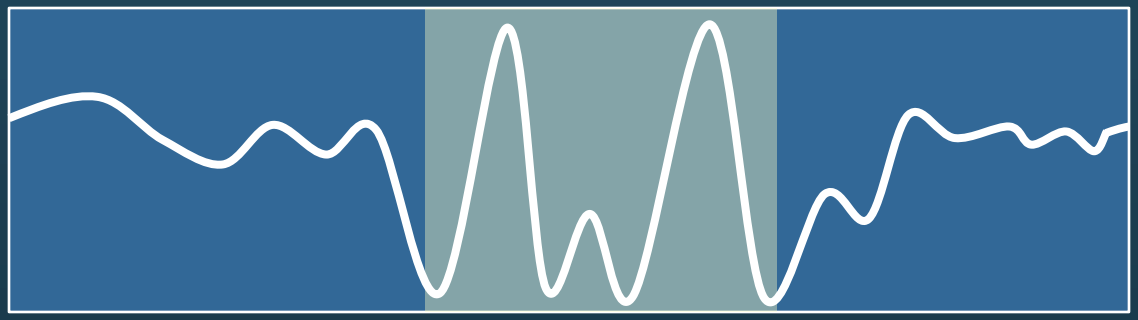
# Generalization: Thresholds



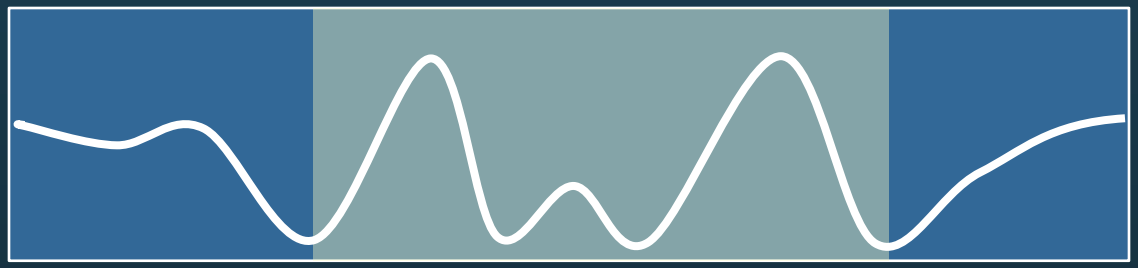
# Generalization: Patterns



# 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?

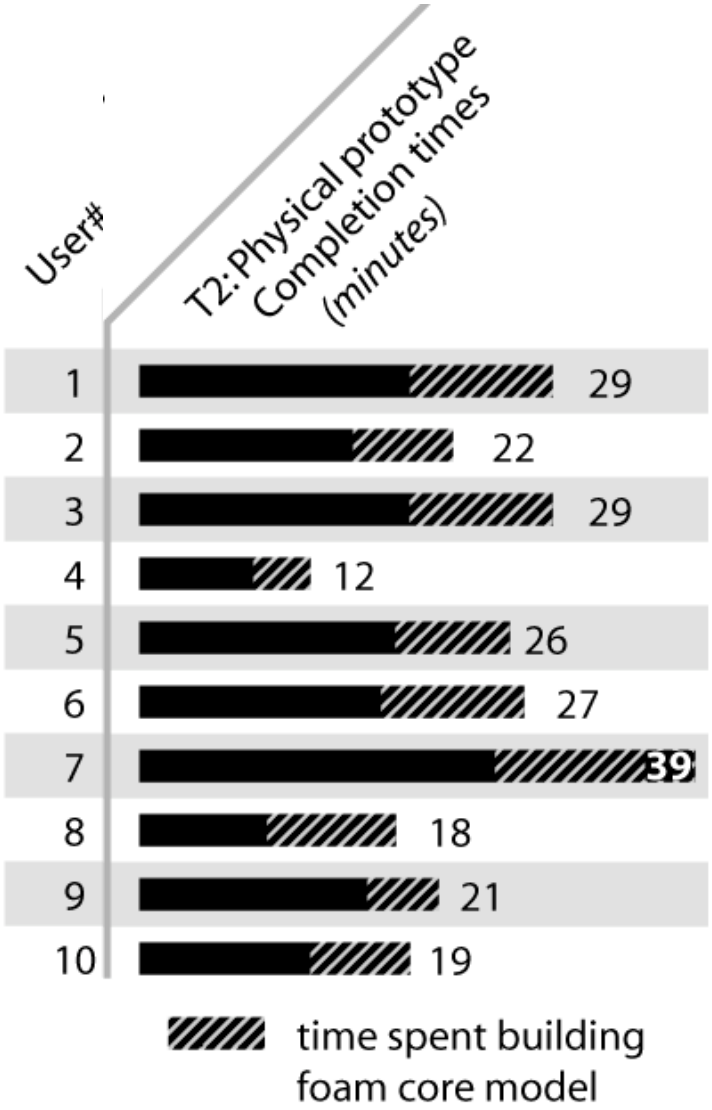
# Triangulation of evaluation methods

| What? | Why? |
| --- | --- |
| Cognitive Dimensions of Notation (CDN) Inspection | Analysis of d.tools as a visual notation |
| **First-Use Laboratory Studies** | **Threshold and usability** |
| **Class & Industry Deployment** | **Real-world stress test** |
| Used by authors as design tools for public exhibitions | Complexity ceiling for knowledgeable users |

# Laboratory Studies


Stanford iRoom

# Laboratory Studies



# Class Deployment

(Hardware manufactured for us by Leapfrog)


Drive
Color
Drive
Color

# What's Important?

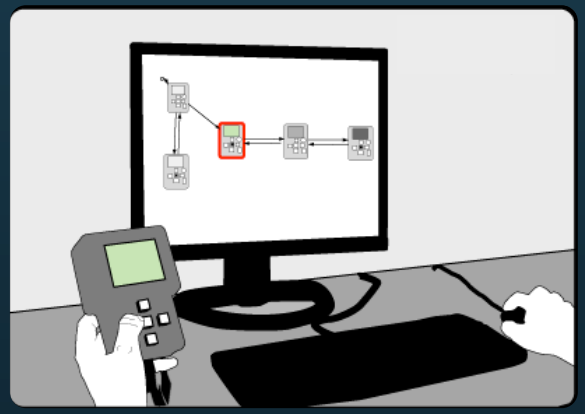
d.Tools successfully enabled non-programmers to prototype novel, sensor-based user interfaces.

Techniques:

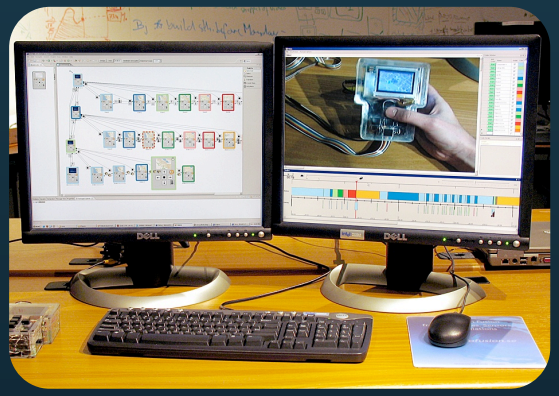
1. Virtual modeling of physical device and plug&play hardware enable visual, storyboard-based authoring.
2. Programming by demonstration for sensor events raises complexity of possible interactions.

Design tool research requires triangulation of evaluation methods

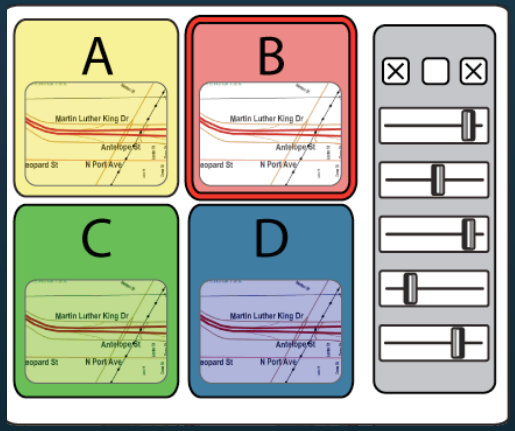
**1** Enable **Rapid Authoring** Off the Desktop



**2** Aid **Iteration** by Managing Feedback

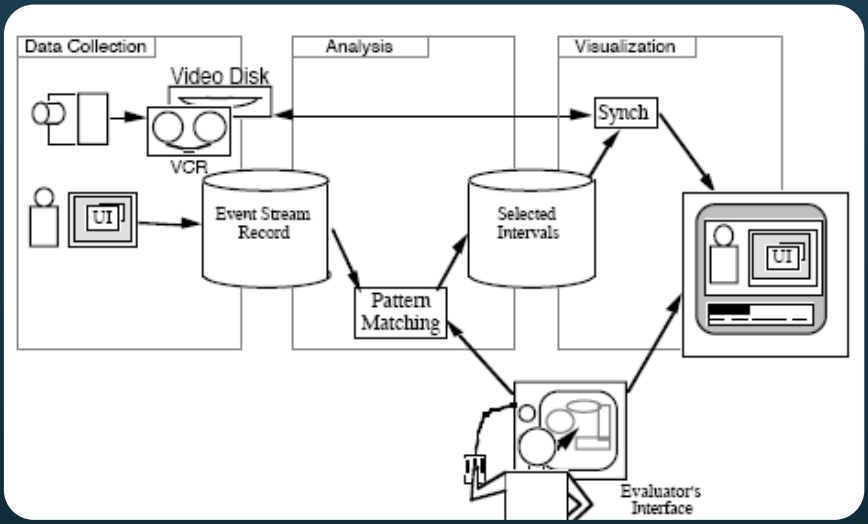


**3** Aid **Exploration** of Multiple Alternatives

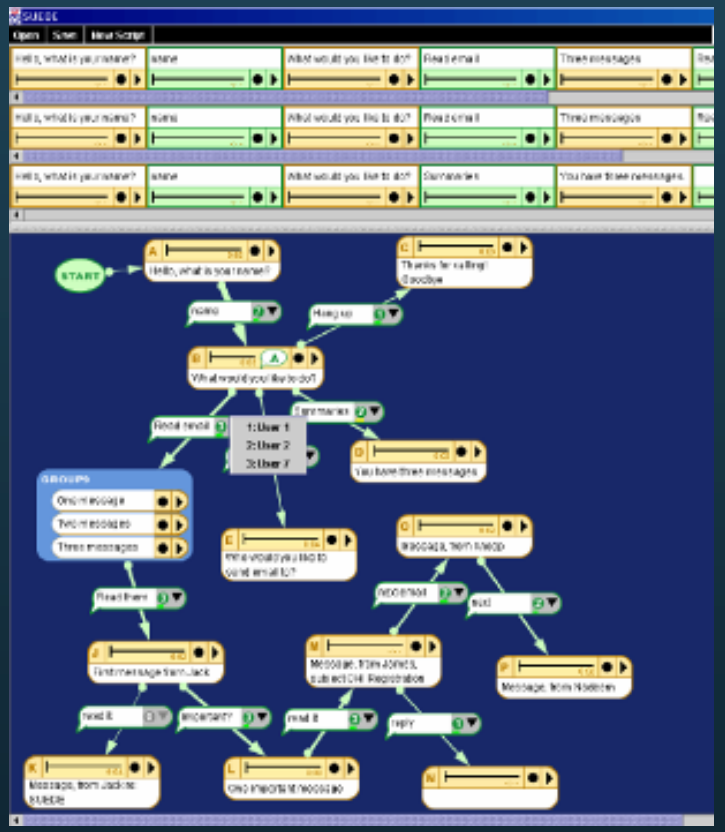



navantis
00:15:19:00
Microsoft
and the

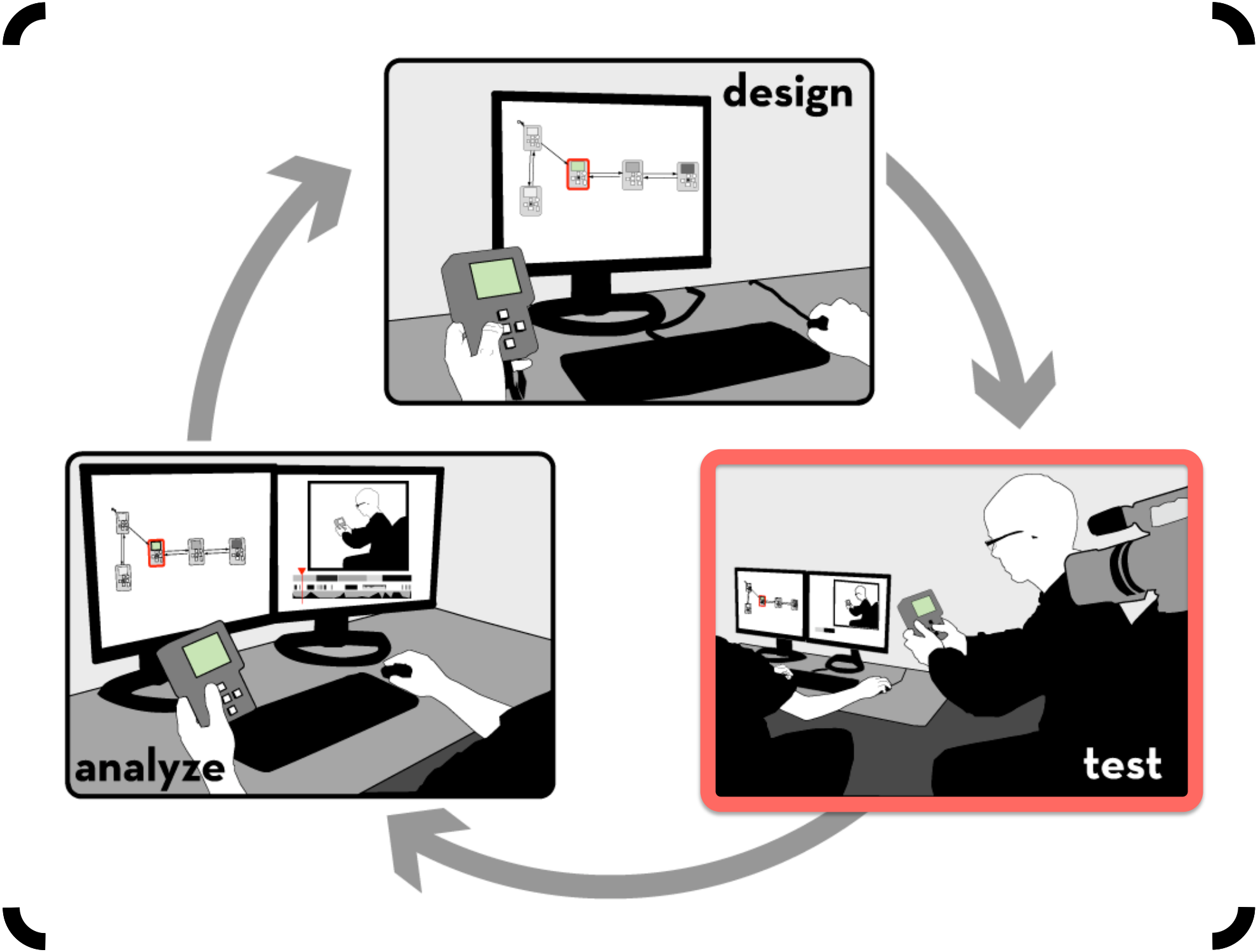
# Related Work: Capturing & Managing Feedback

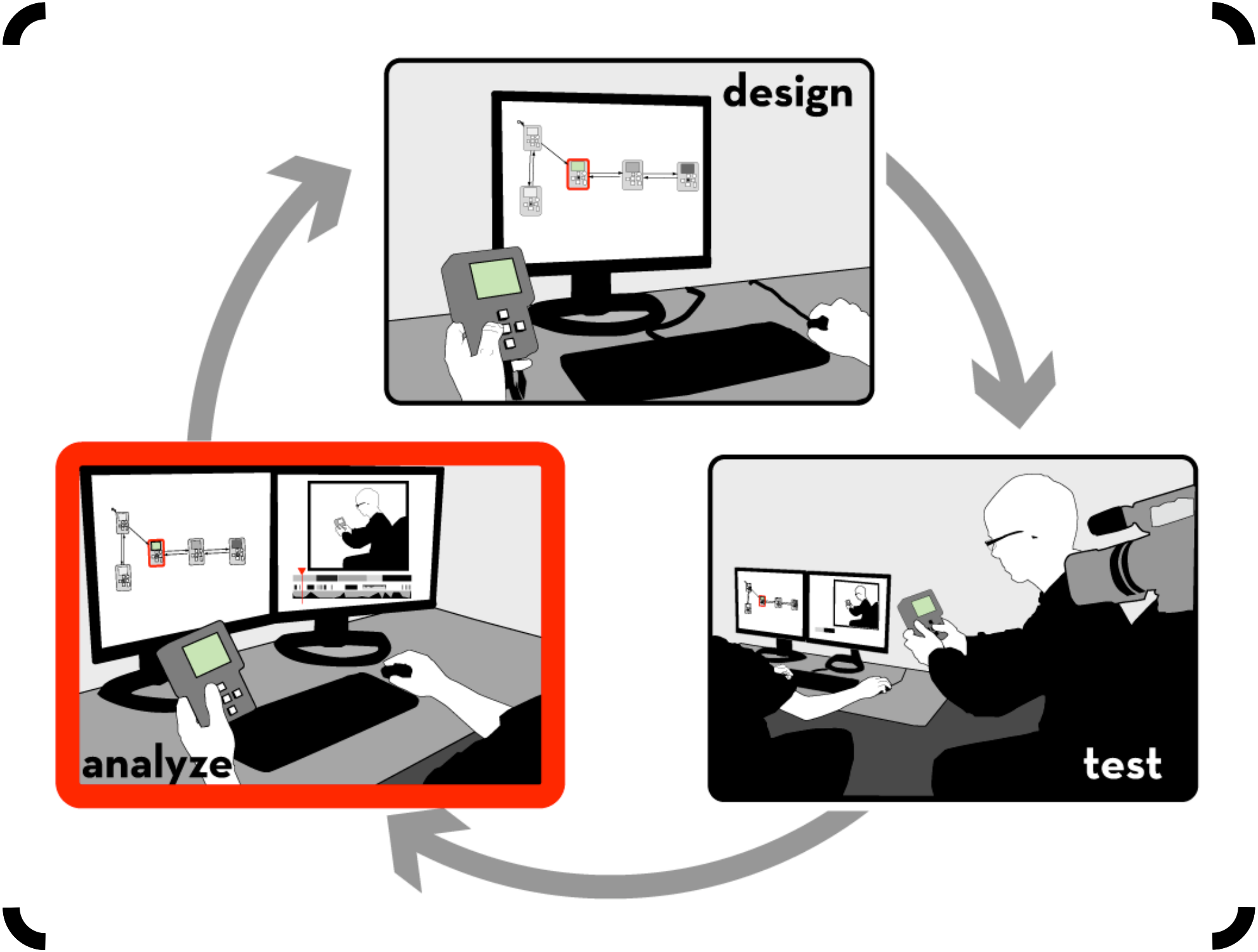


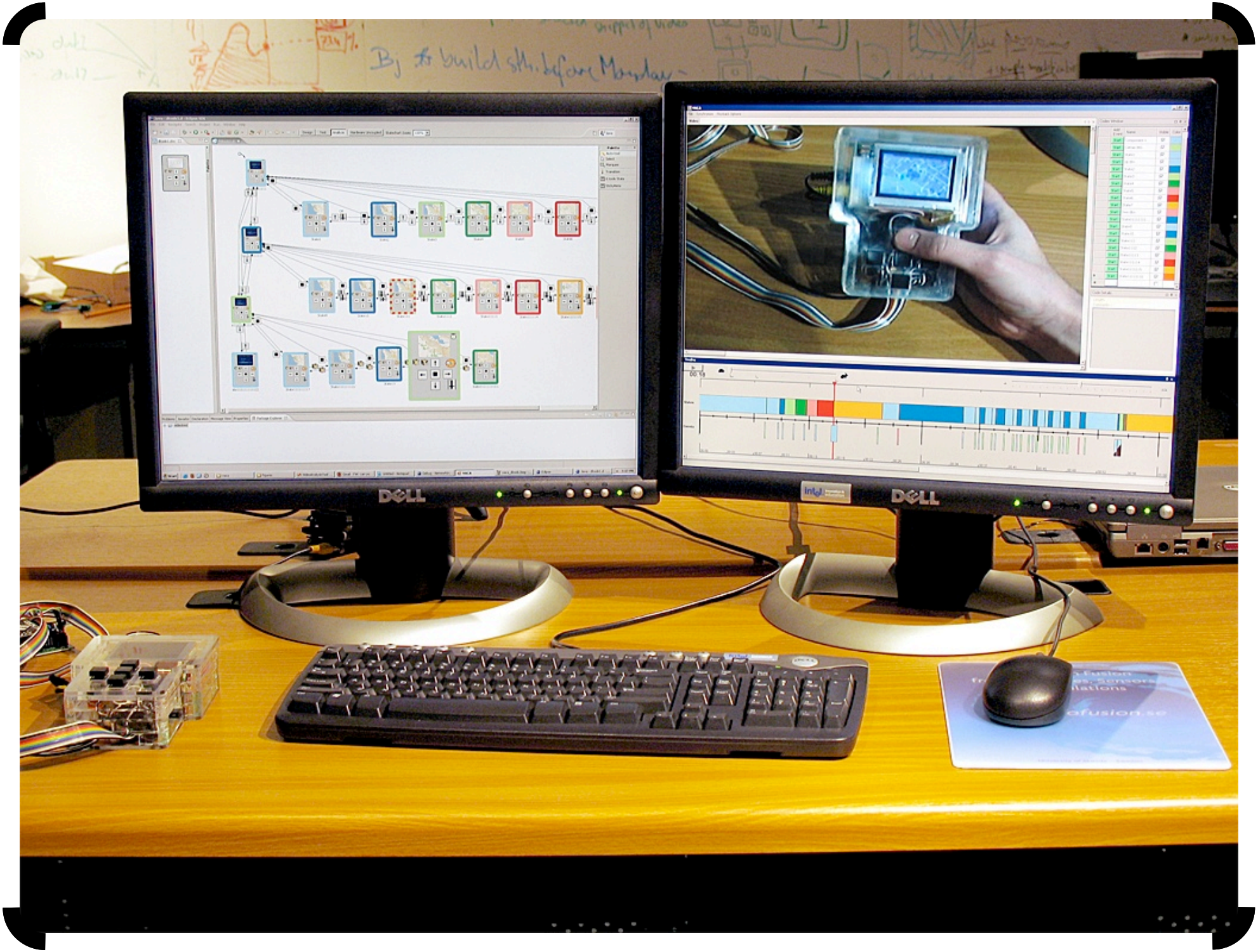

I-Observe, Badre, 1995



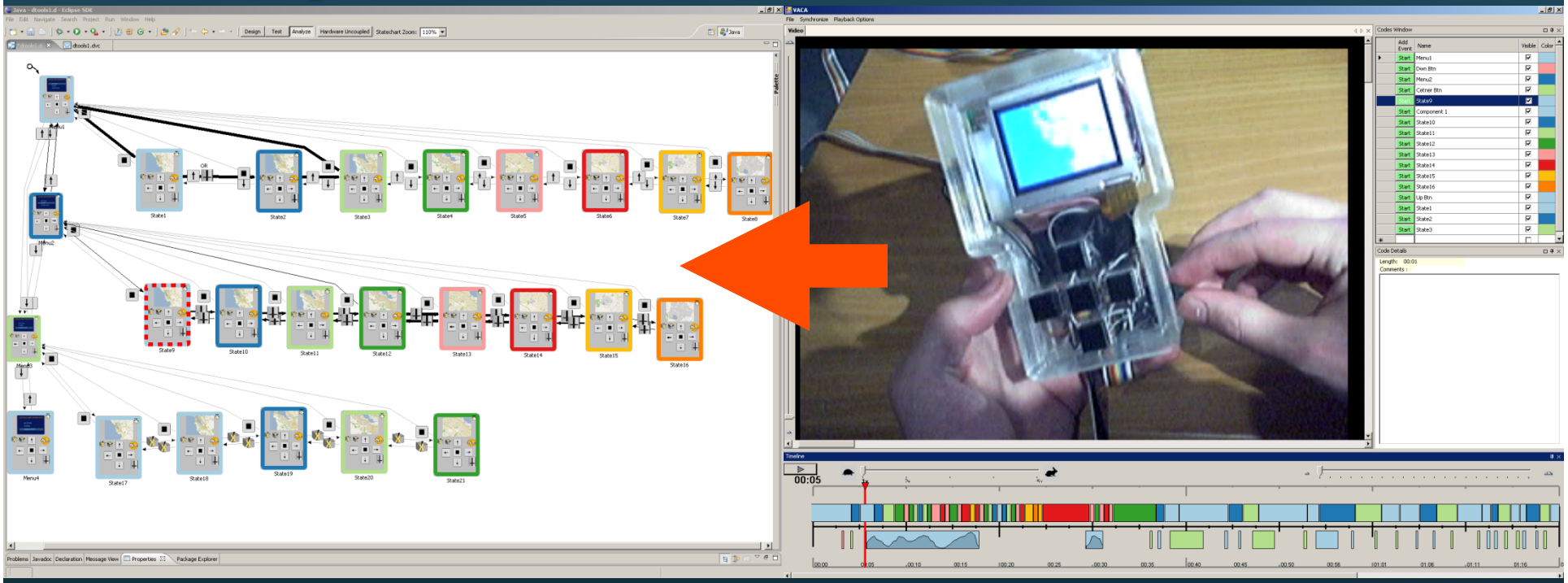
SUEDE, Klemmer, 2000

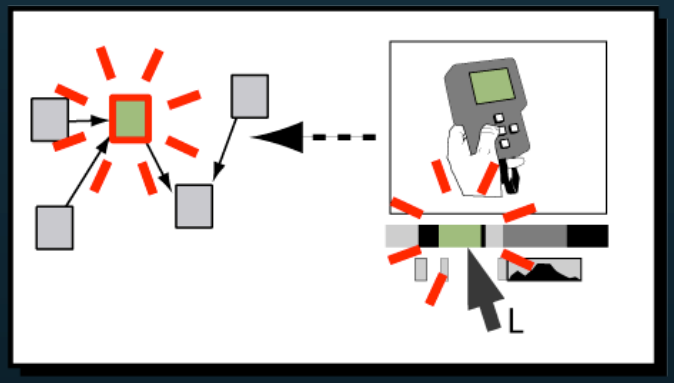

design
test
analyze

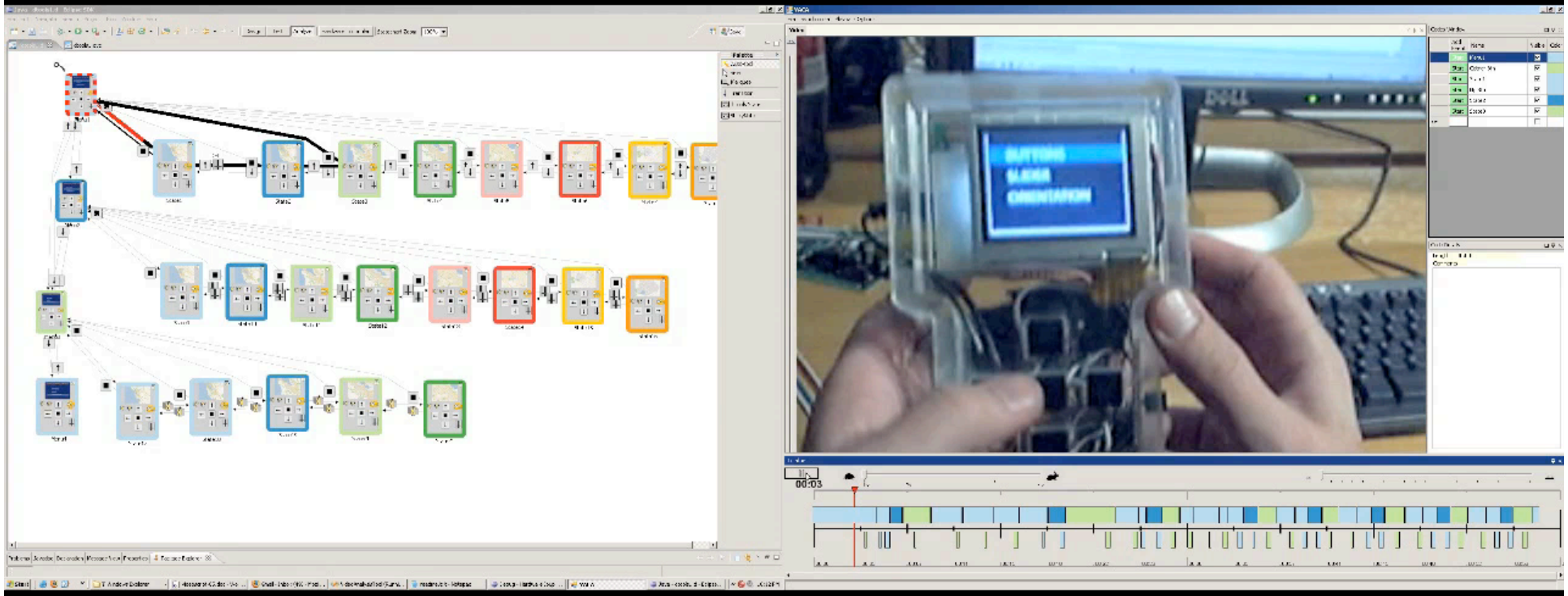

design
test
analyze


DELL
DELL

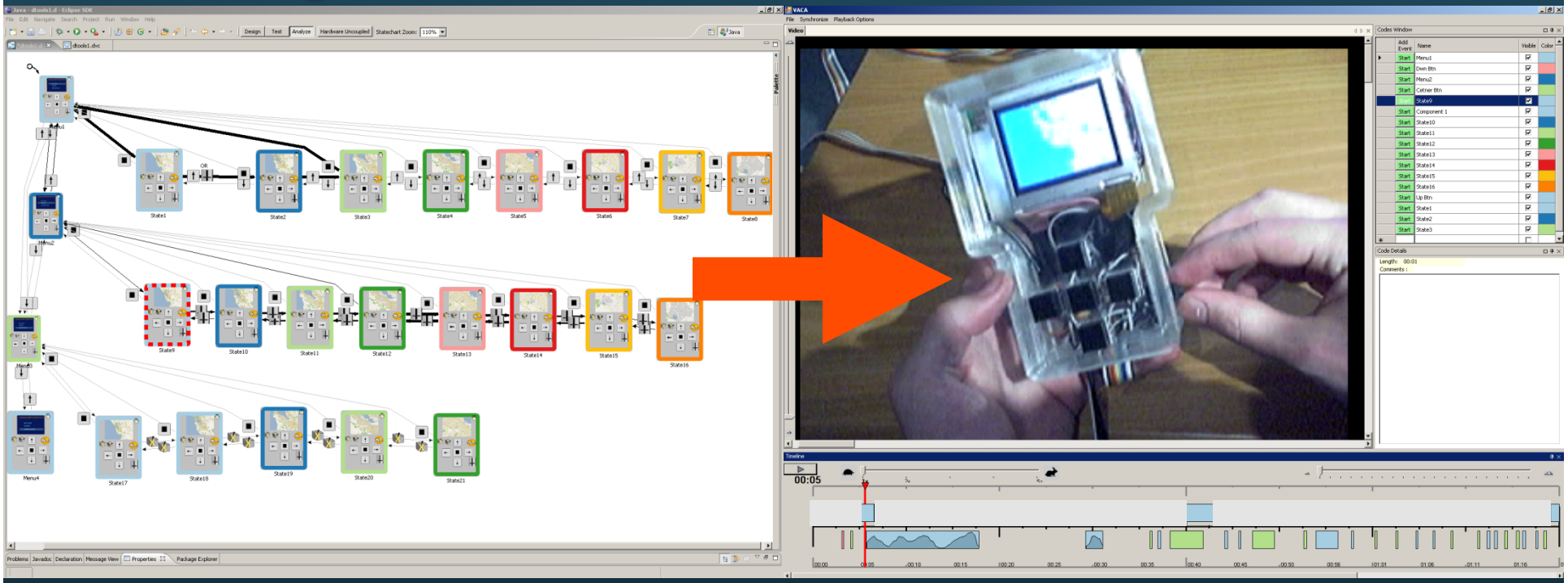
# Analyze

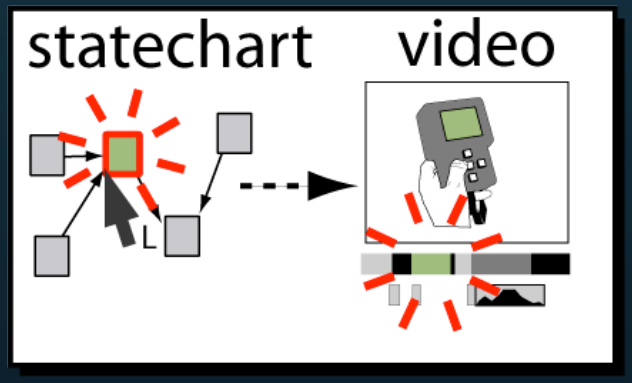




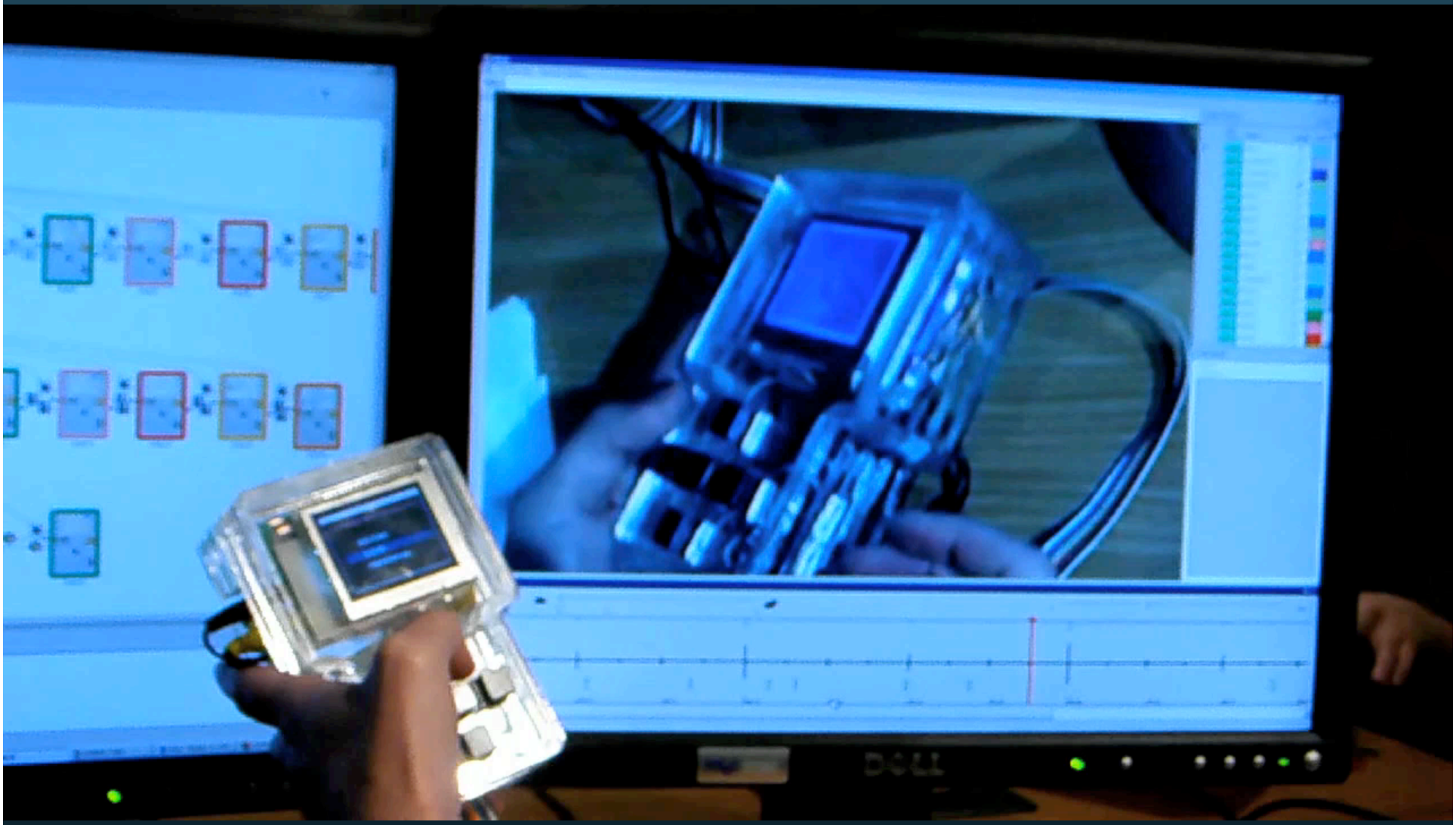


# Analyze





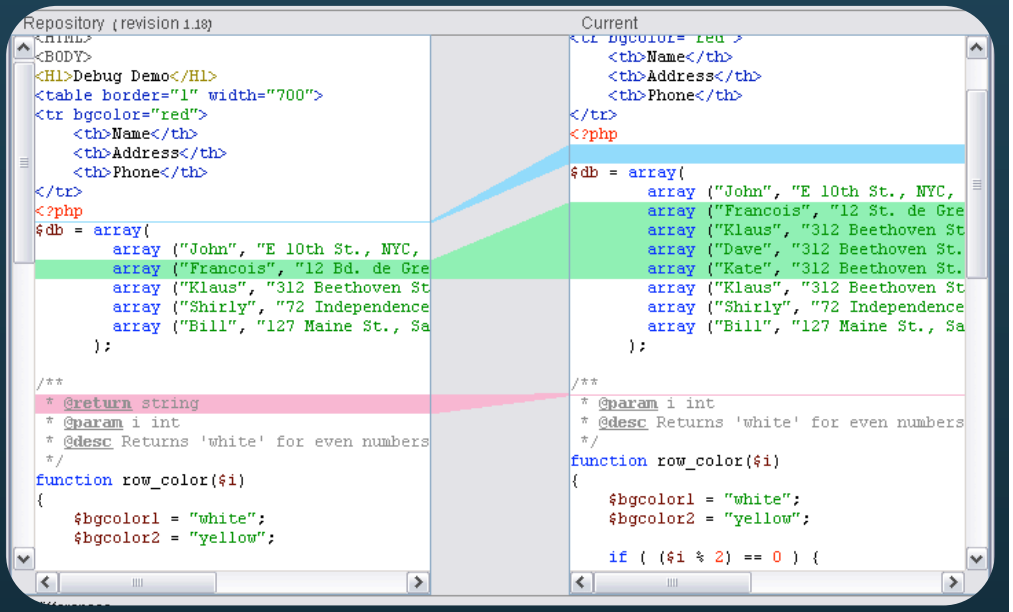
# Analyze



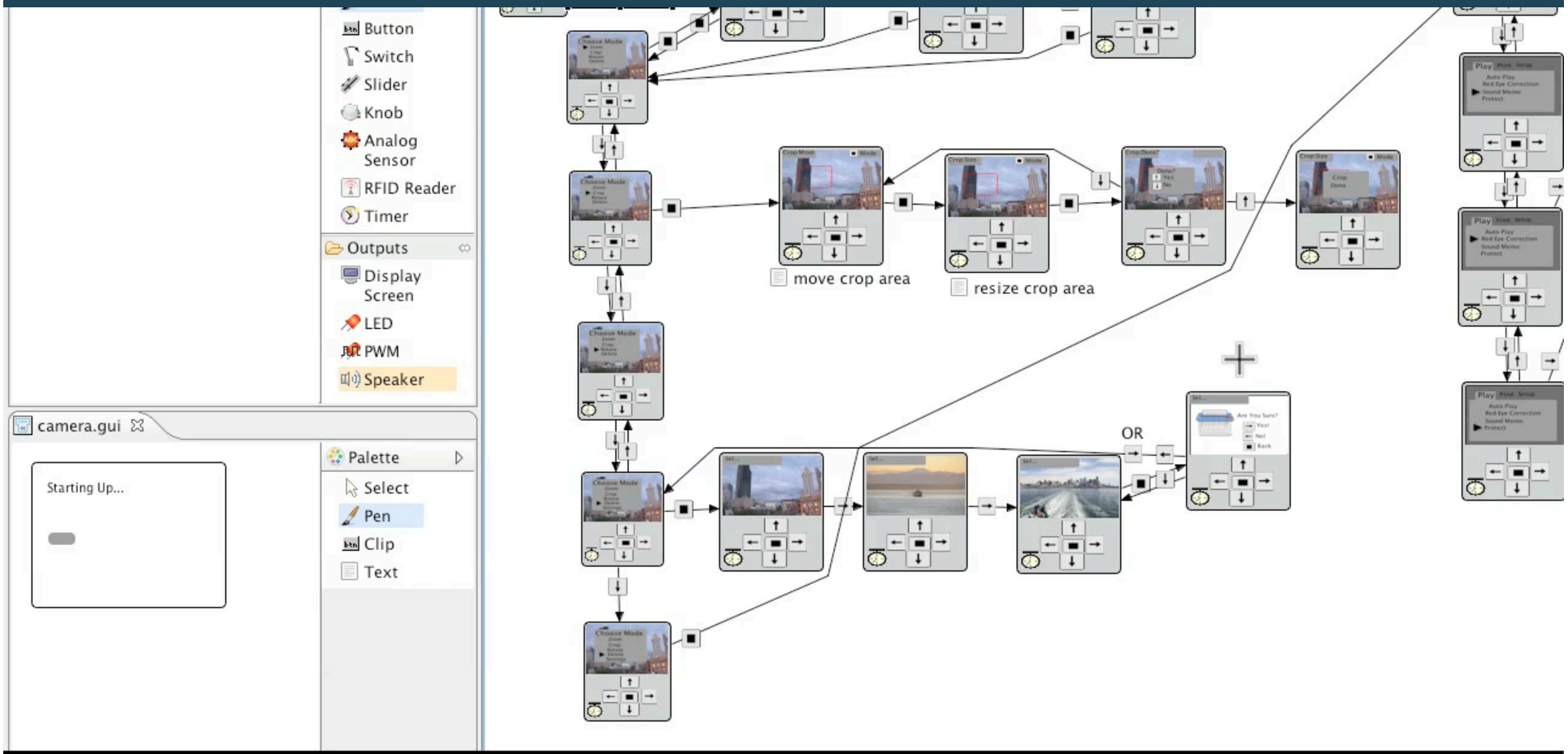
# How might we revise interaction designs?

# Tools for revising interaction designs

Movie production




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
move crop area
resize crop area
OR

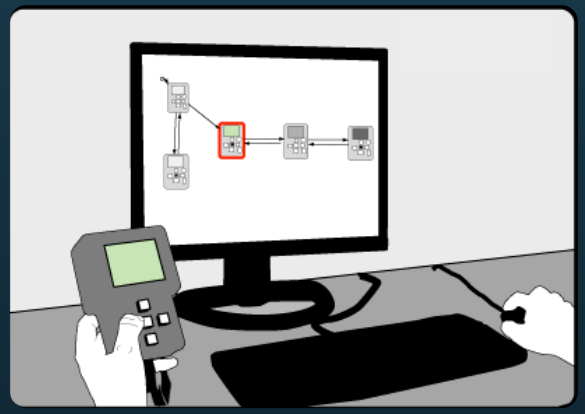
# What's Important?

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

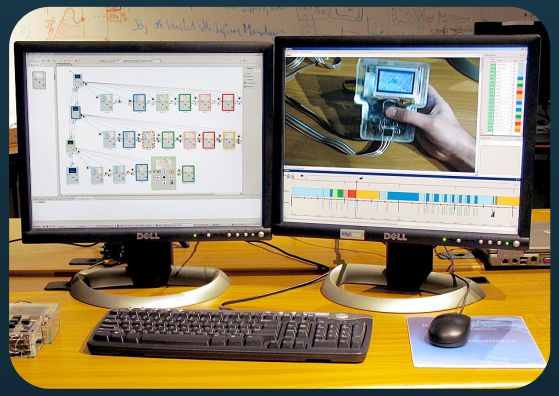
Integration is enabled by a high-level visual representation of the software model.

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

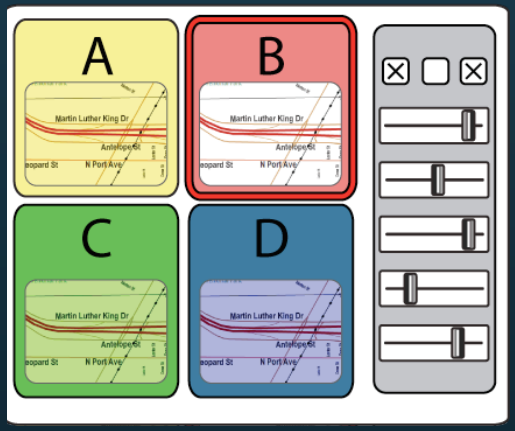
**1** Enable **Rapid Authoring** Off the Desktop



**2** Aid **Iteration** by Managing Feedback

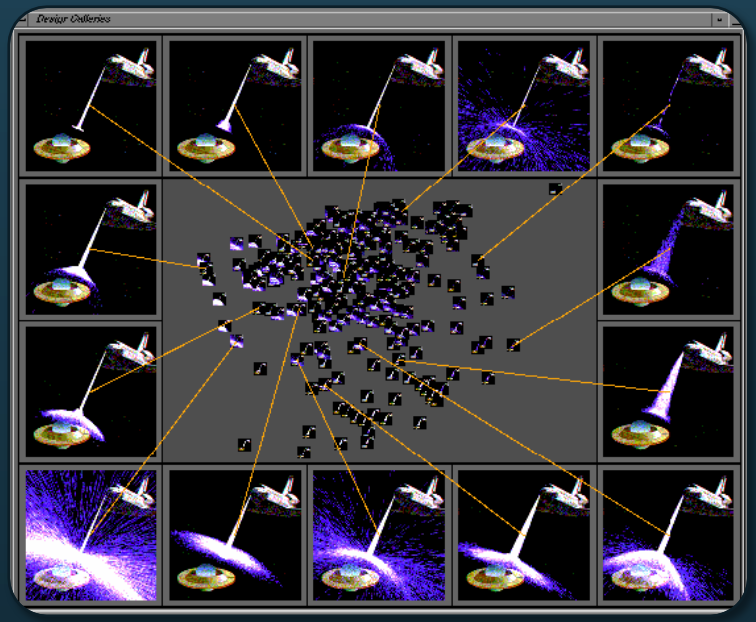


**3** Aid **Exploration** of Multiple Alternatives

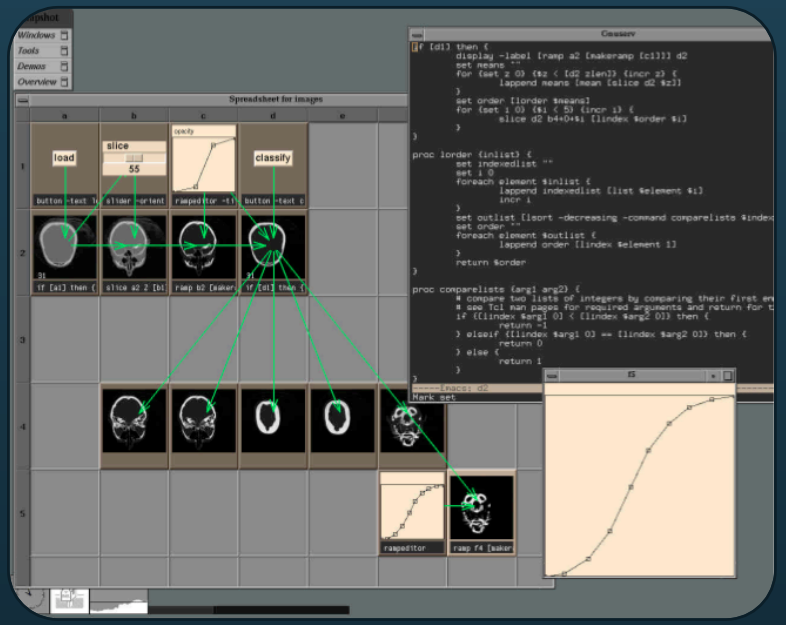


How can tools support the creation of **multiple** user interface **alternatives**?

# Related Work: Working with Alternatives



Design Galleries, Marks, 1997




Levoy, Spreadsheets for Images, 1994

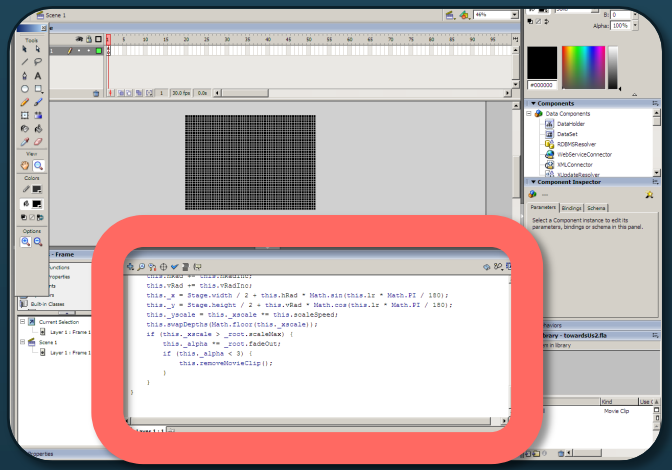
Subjunctive Interfaces, Lunzer, TOCHI 08

Spreadsheets for Visualization, Chi, IEEE CGA 98

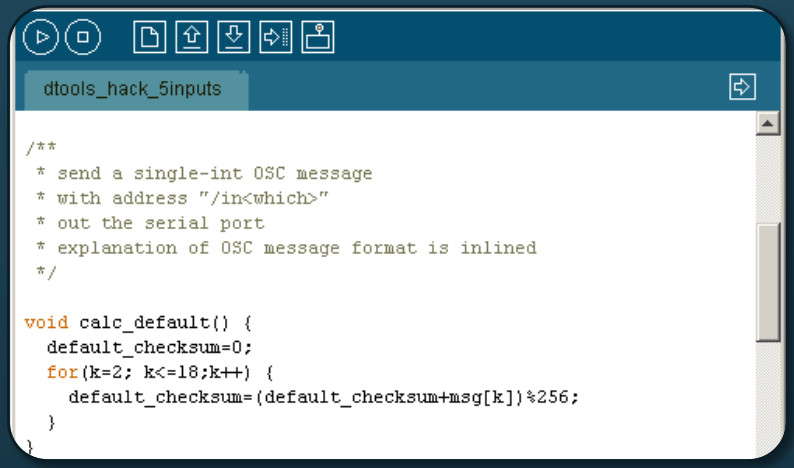
Parameter Spectrums & Parallel Pies, Terry, UIST 02 & CHI 04

Partials, Terry, Phd 2005

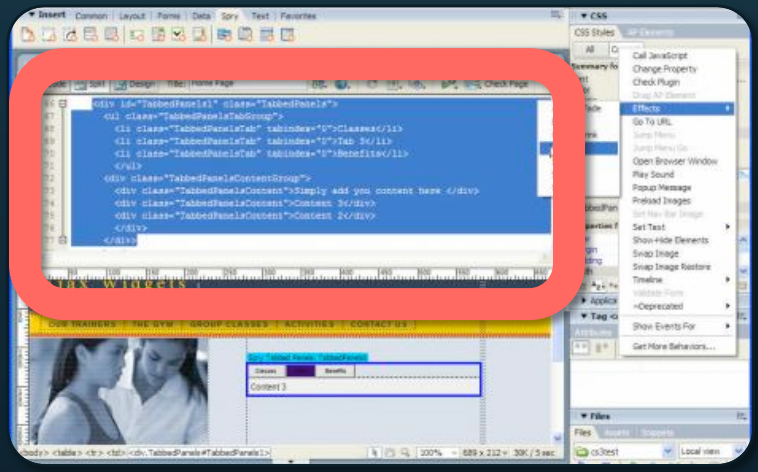
# Interaction Designers Write Code



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

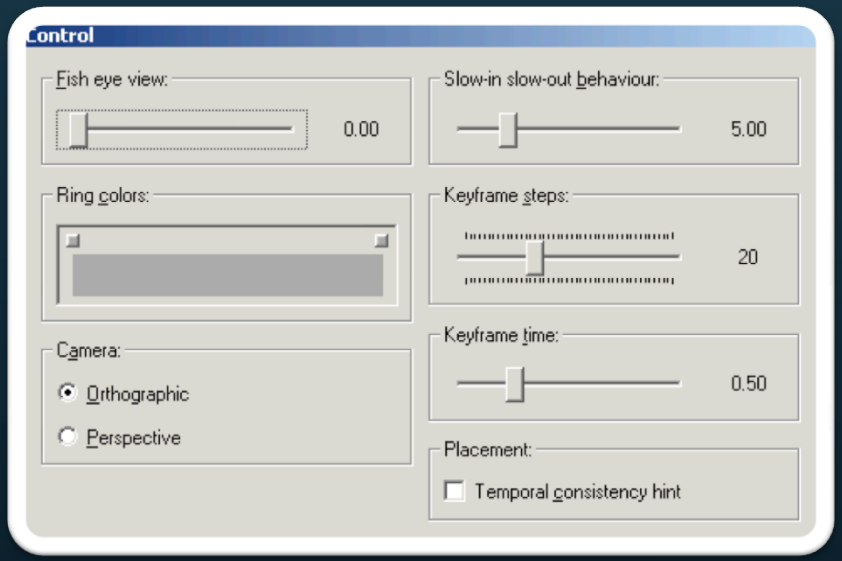
- Manage parameter variations
- Manage code alternatives
- Access variations & alternatives at runtime

```
t 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*

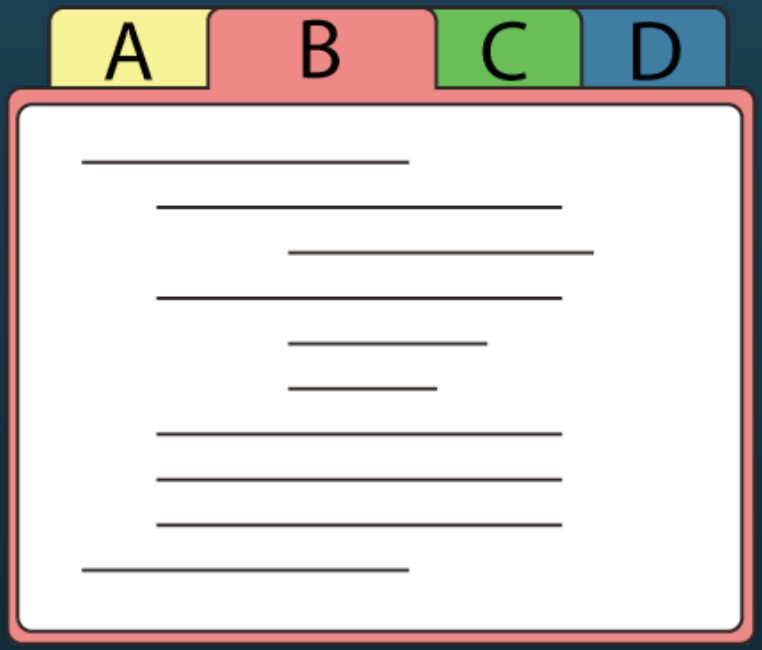
# 3 Requirements for Programmed Interactions

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

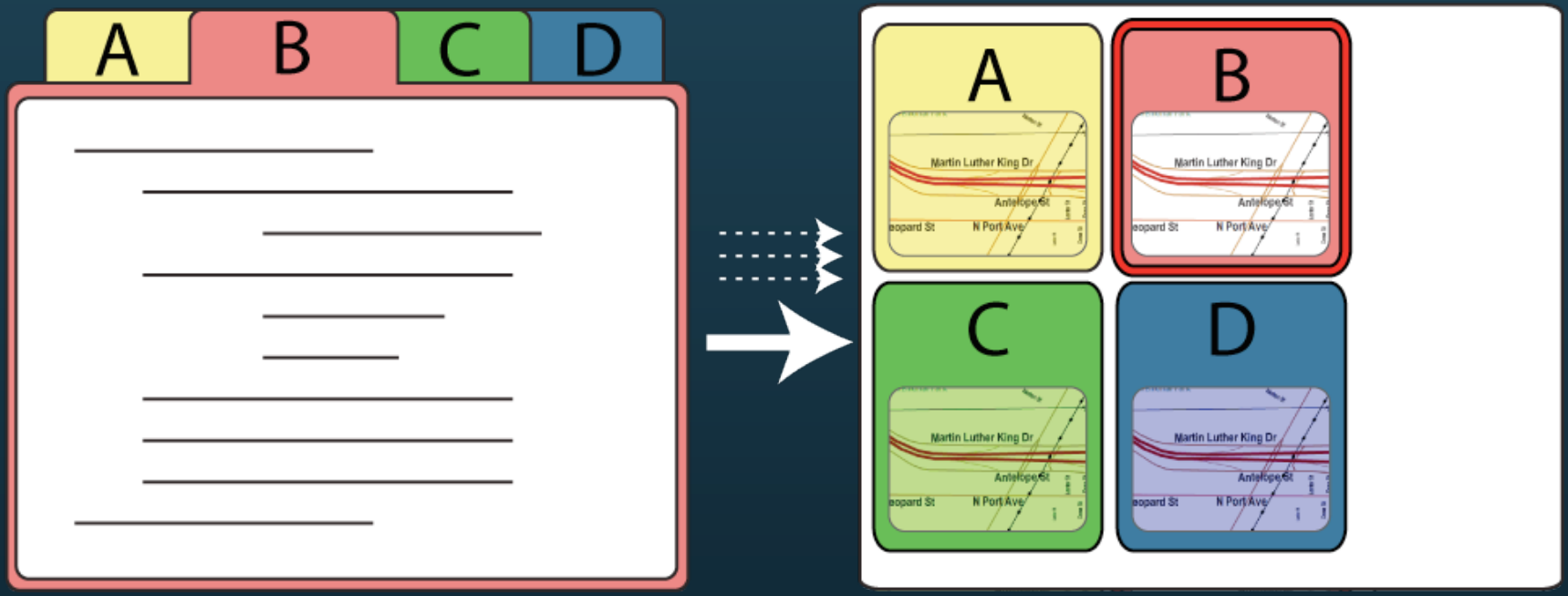


# Juxtapose: Source alternatives...

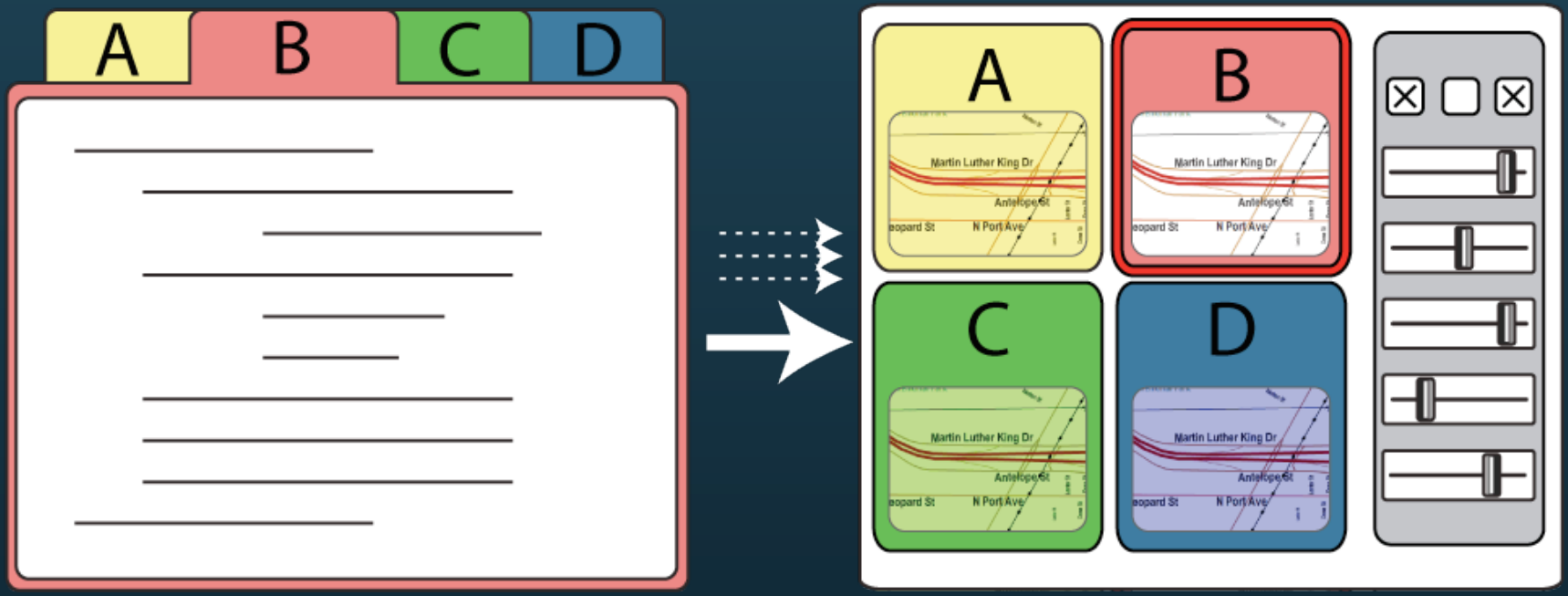
# Juxtapose: Source alternatives...



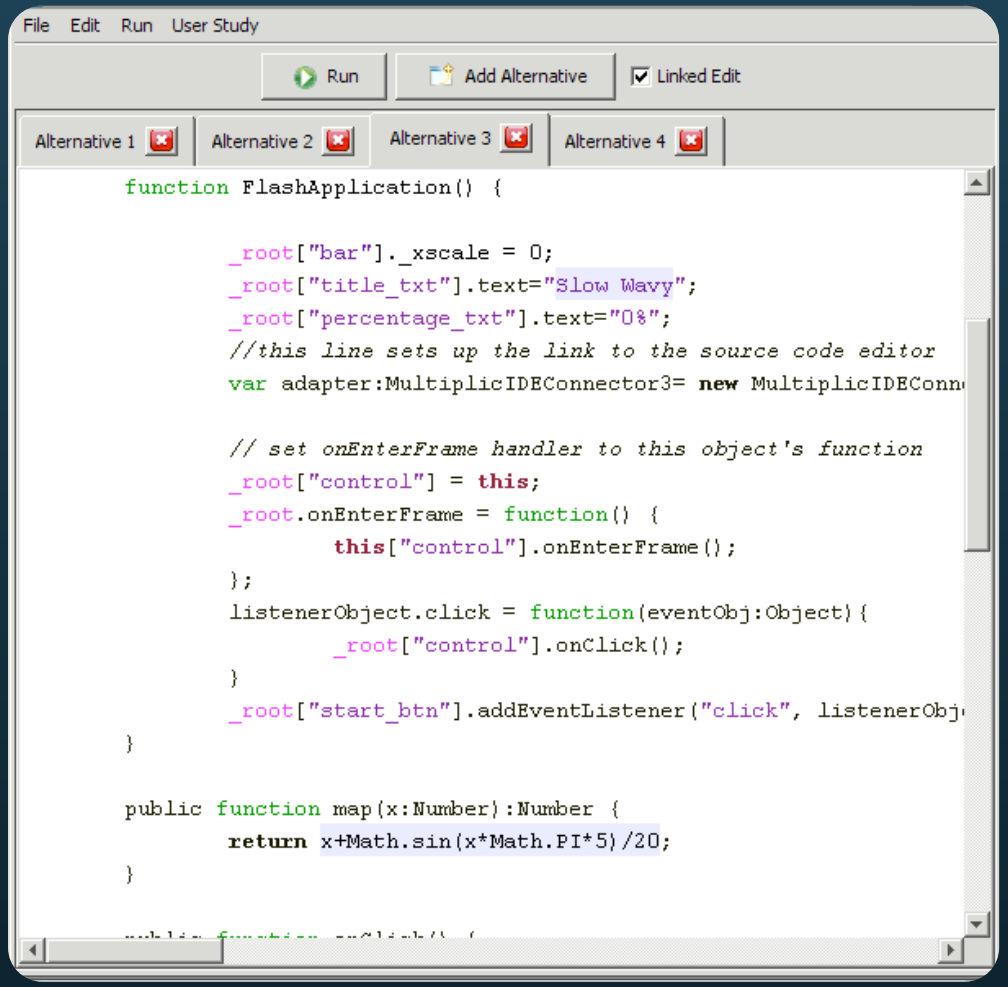
# … are executed in parallel,



# and tuned through an generated UI.

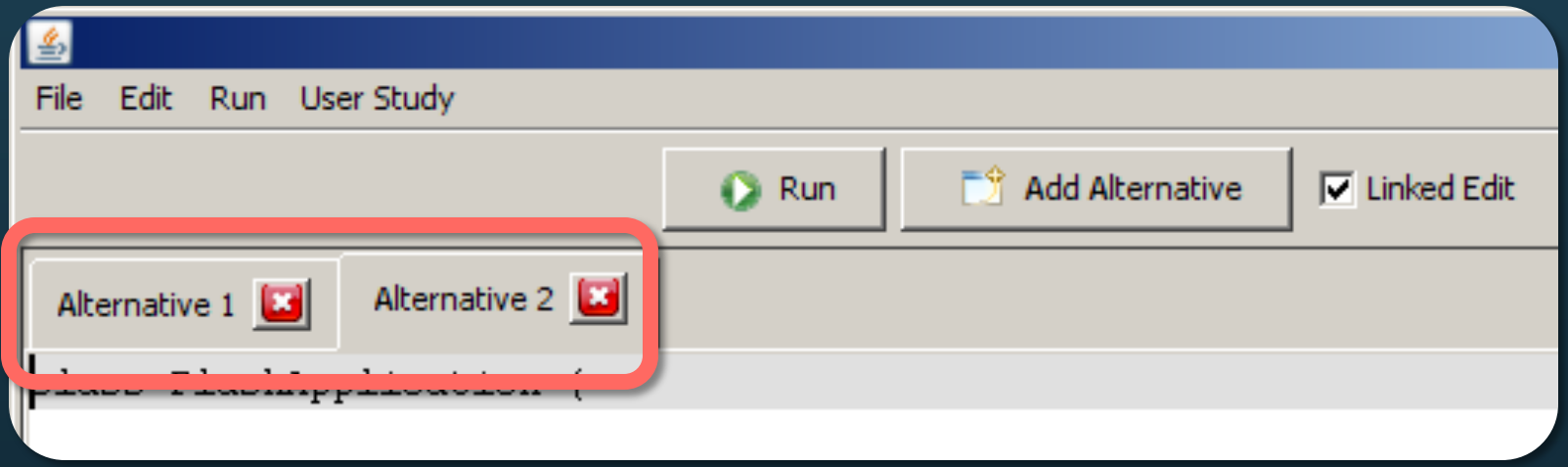


# Parallel Editing

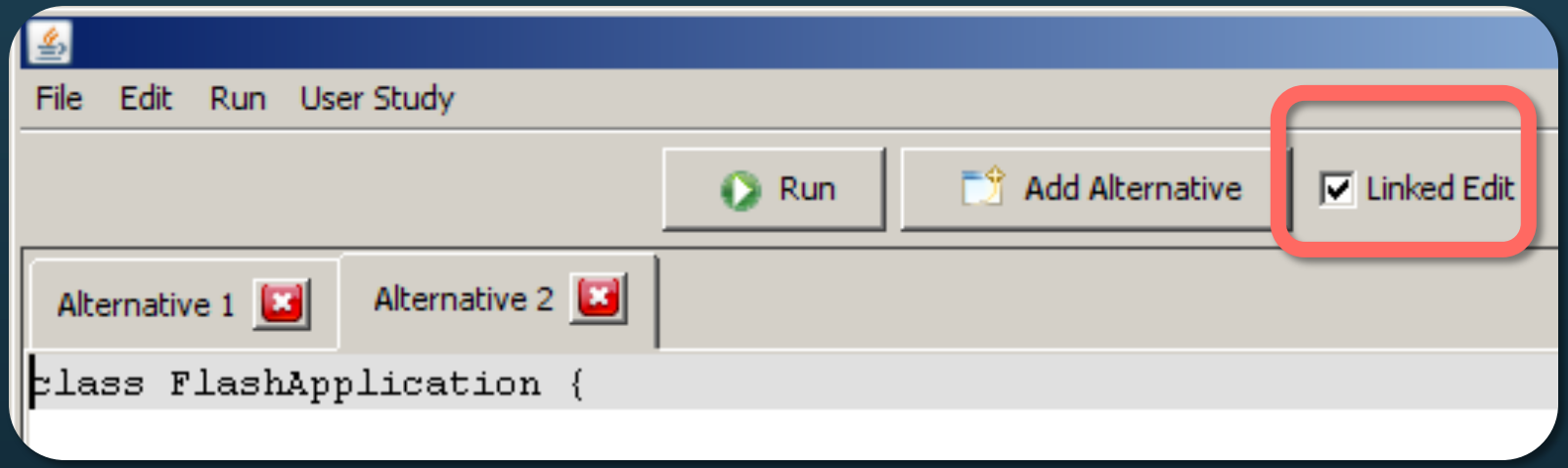



*Juxtapose editor for ActionScript 2. Generates Flash files.*

# Parallel Editing

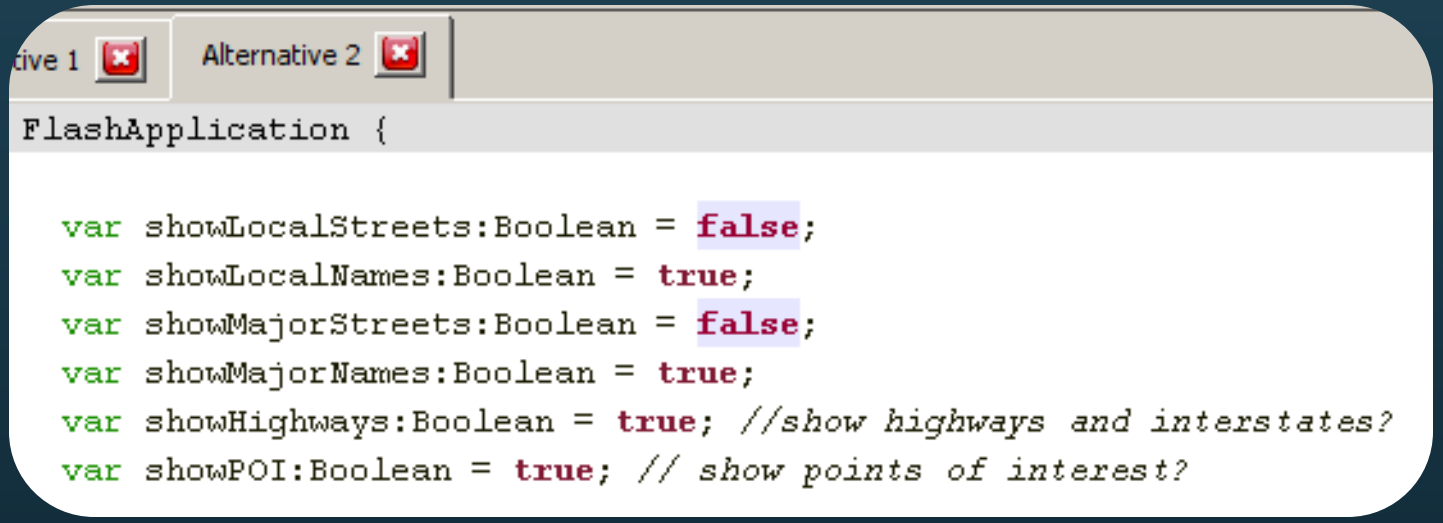


# Parallel Editing




*Linked Editing: Toomim 2004*

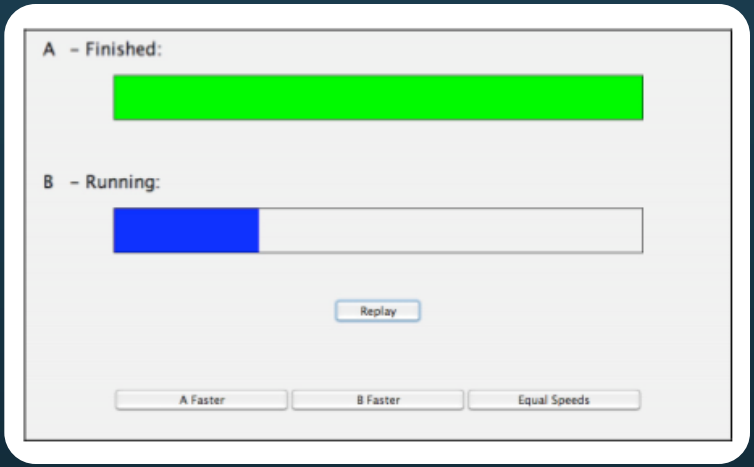
# Parallel Editing



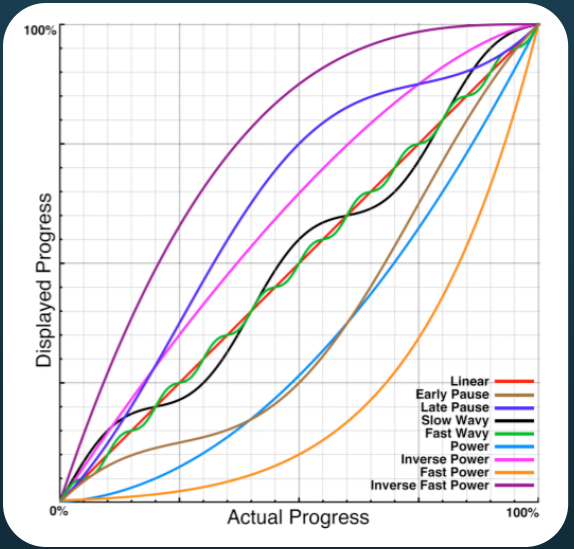

*Linked Editing: Toomim 2004*

# Example: Rethinking the Progress Bar

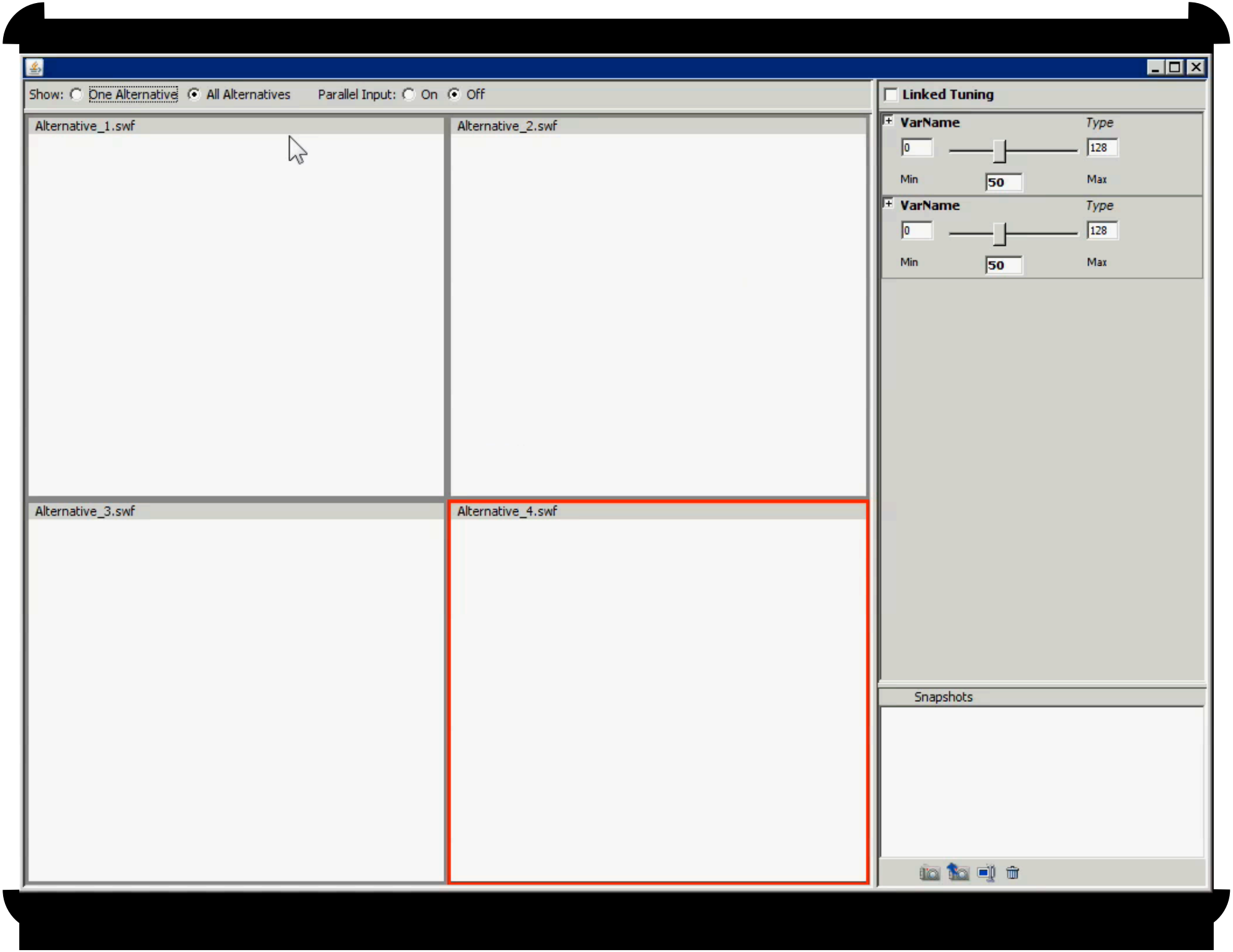
Harrison et al., UIST 2007

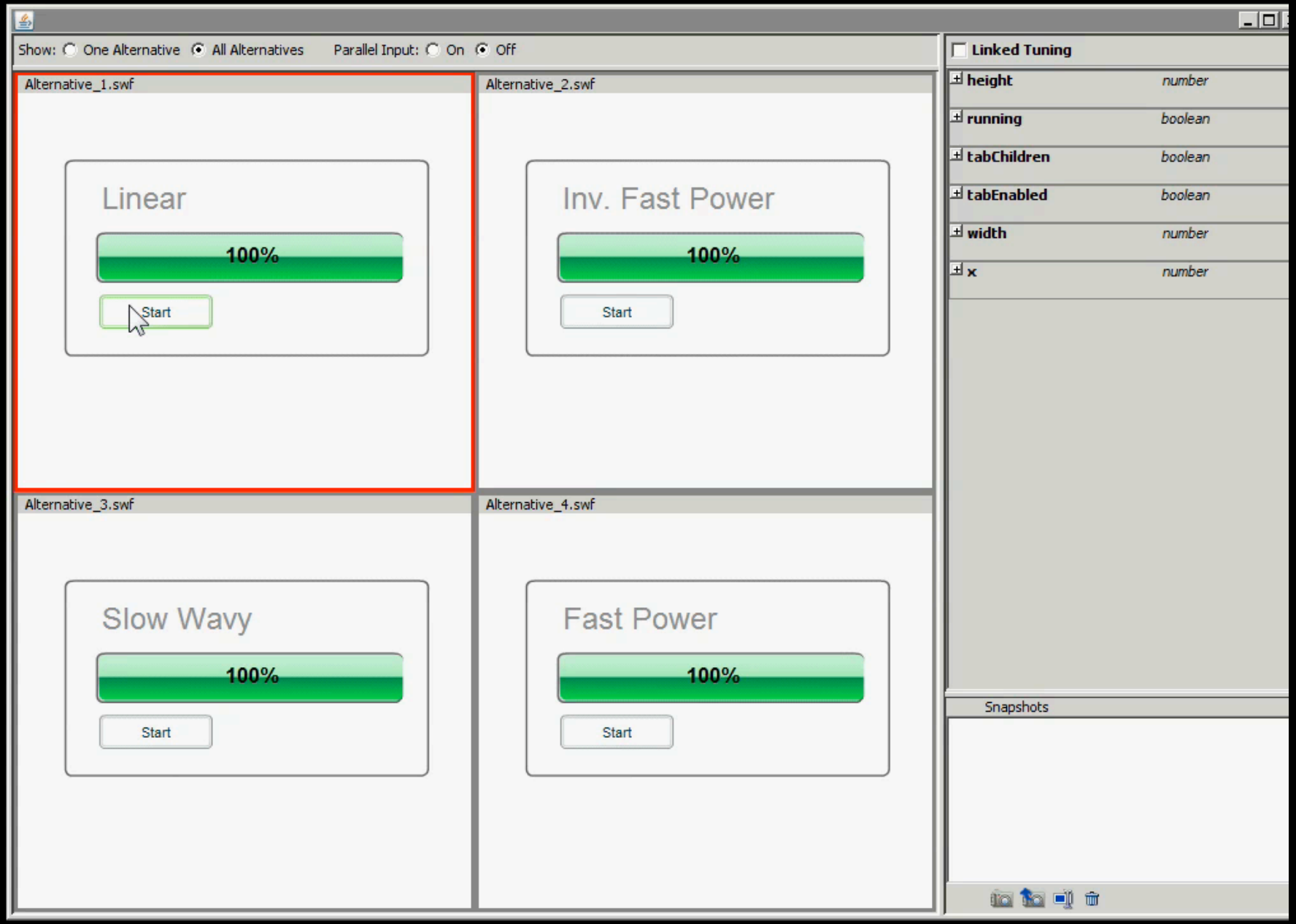



Progress bar test application



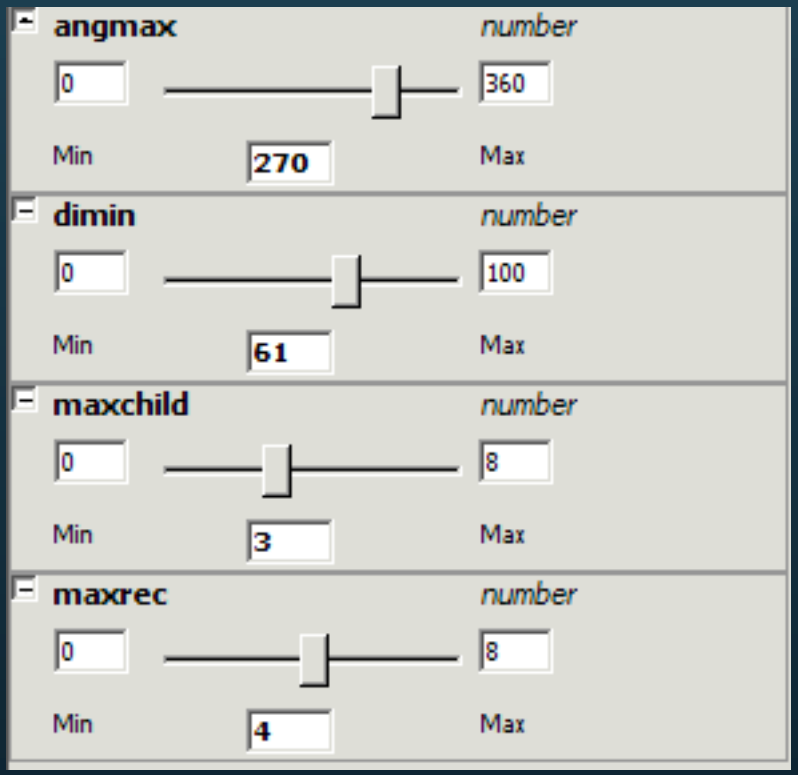

Progress profiles


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
Alternative_1.swf
Linear
100%
Start
Alternative_2.swf
Inv. Fast Power
100%
Start
Alternative_3.swf
Slow Wavy
100%
Start
Alternative_4.swf
Fast Power
100%
Start
Linked Tuning
height number
running boolean
tabChildren boolean
tabEnabled boolean
width number
x number
Snapshots

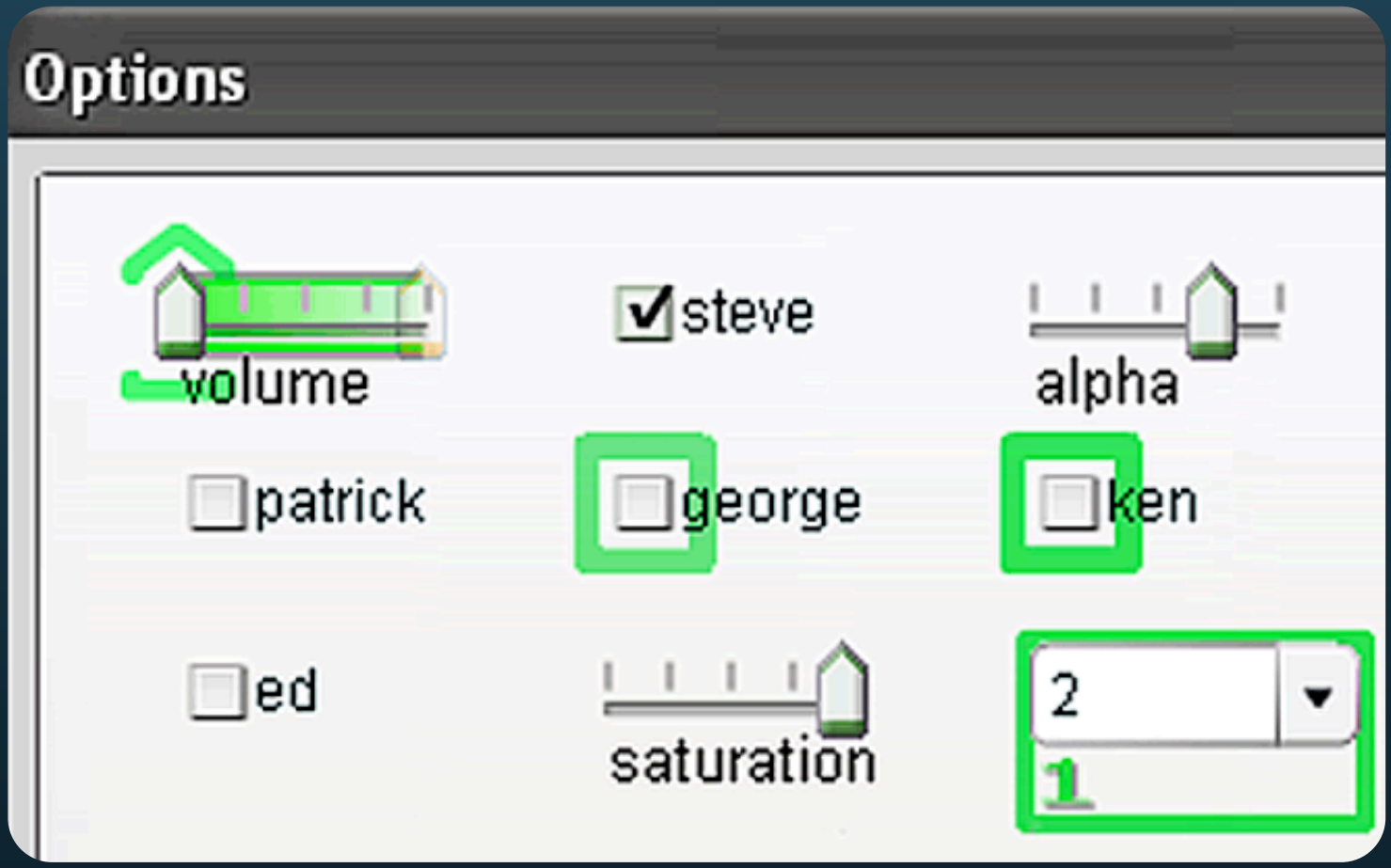
# Parameter Tuning

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



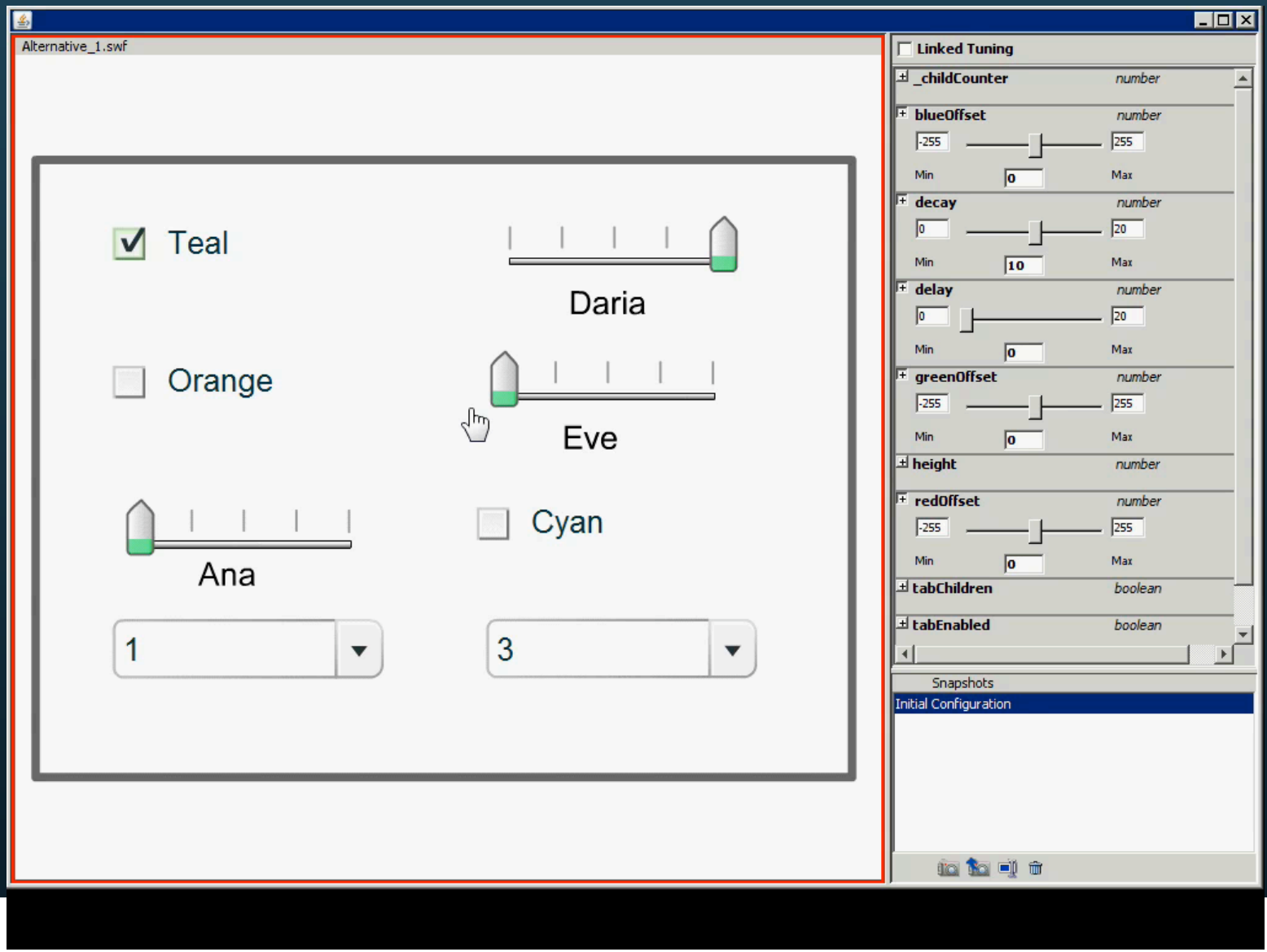
# Example: Tuning Phosphor

Baudisch, UIST 2006

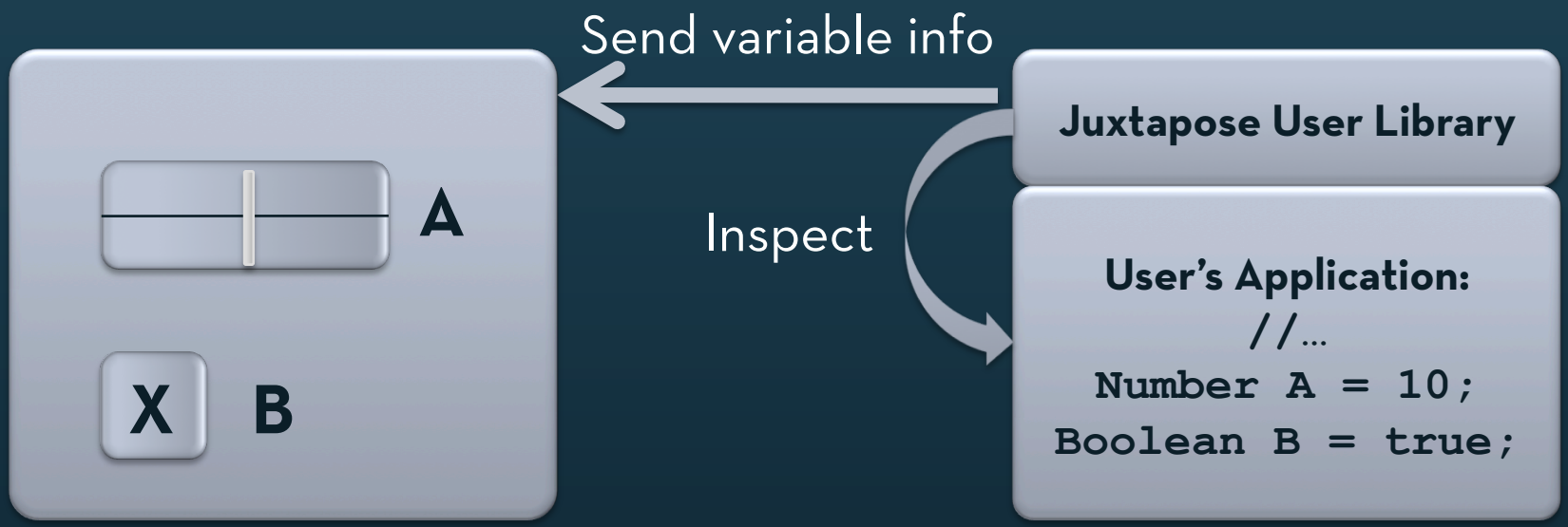


# Example: Tuning Phosphor

Baudisch, UIST 2006



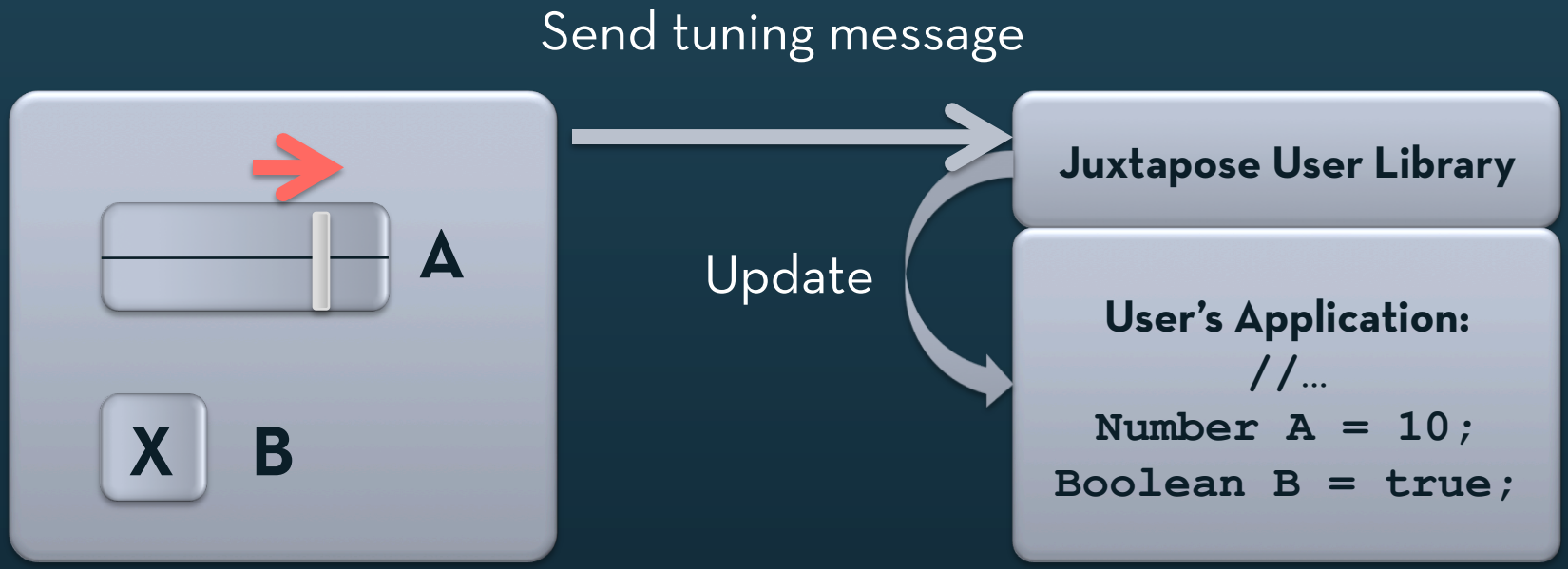
# Generating Tuning Interfaces




*Juxtapose Process*

*User Process*

# Generating Tuning Interfaces

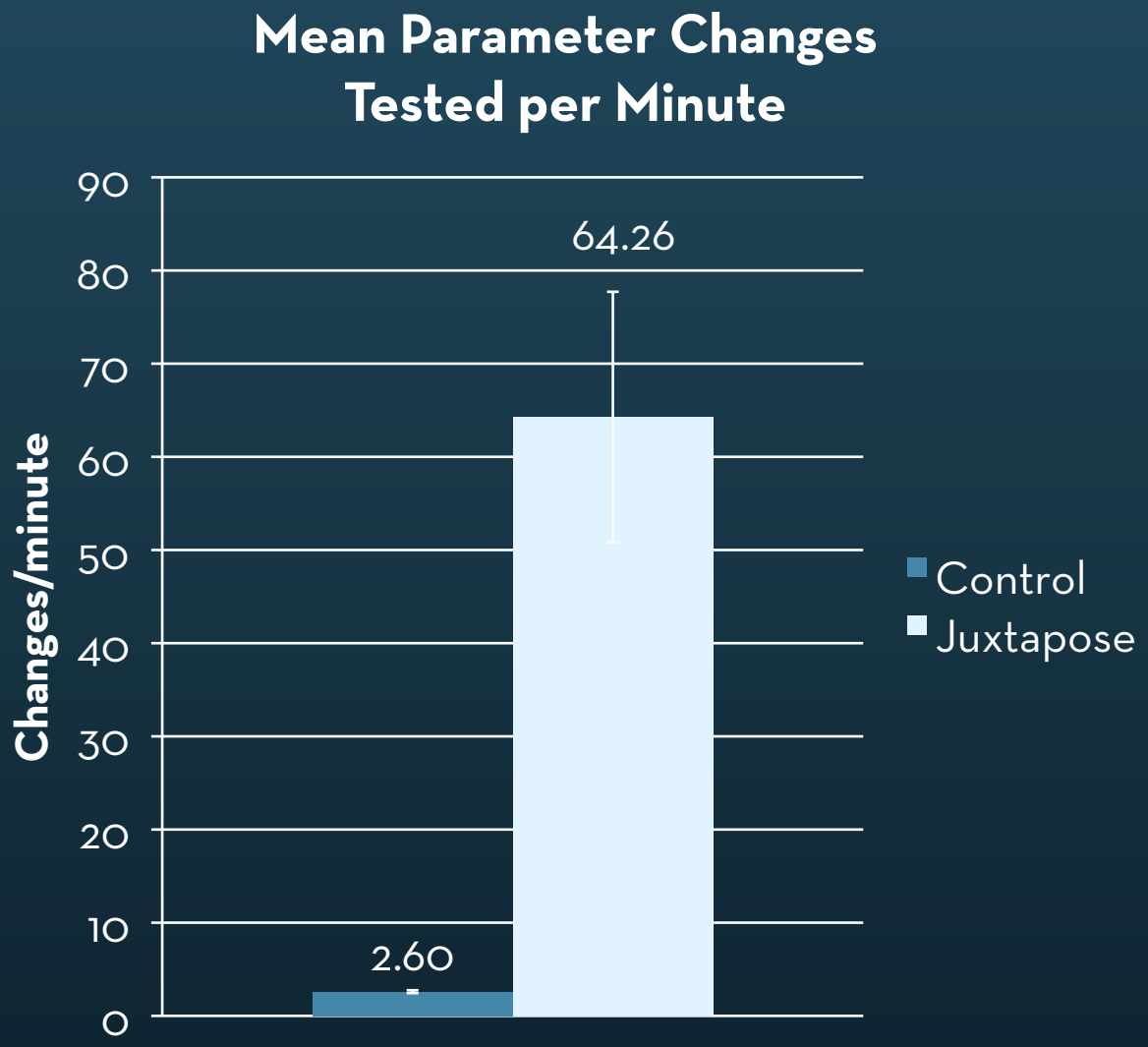


# When is tuning most useful?

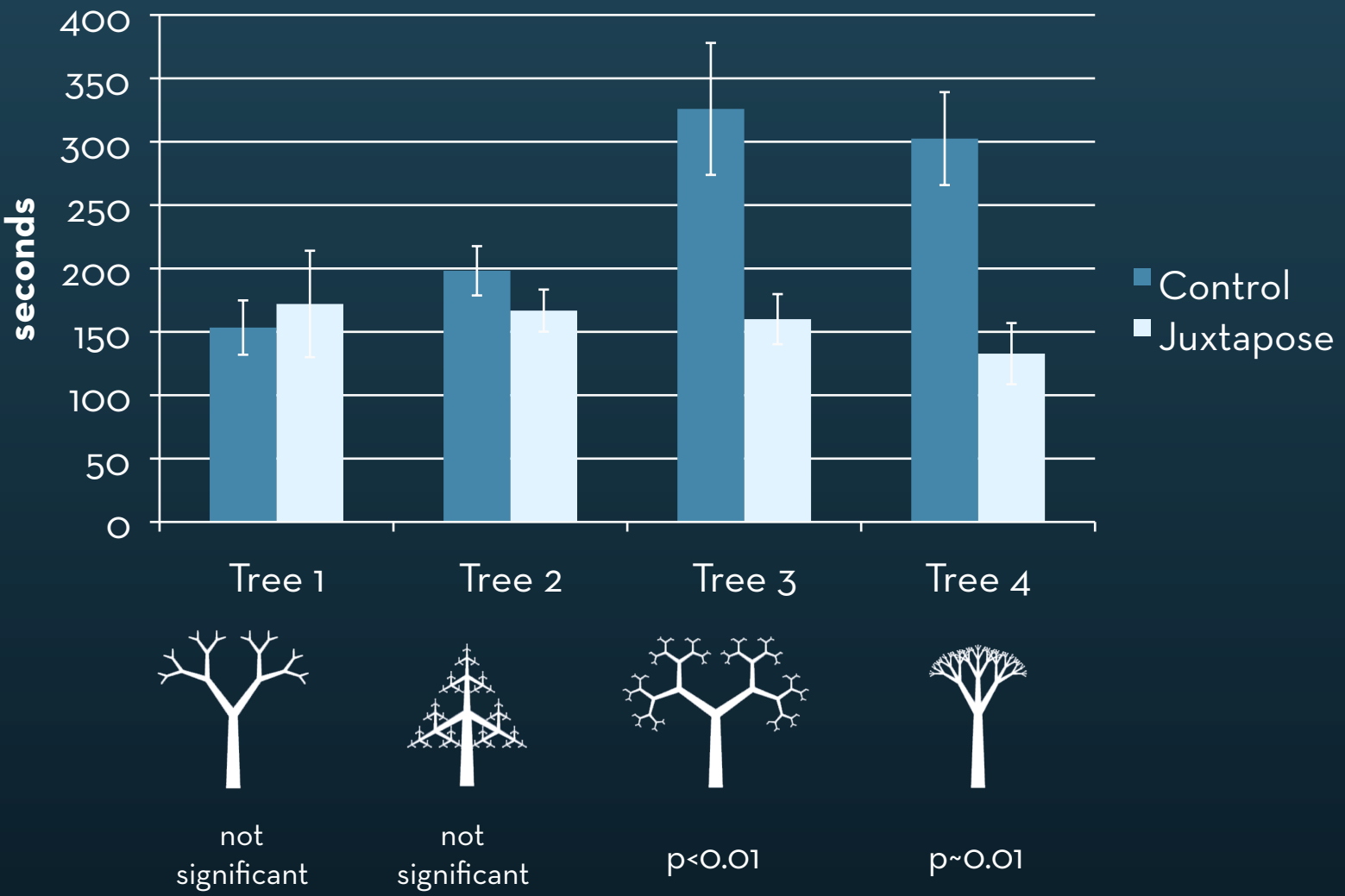
Search in 4D parameter space. Given recursive drawing code for this:

Produce these:

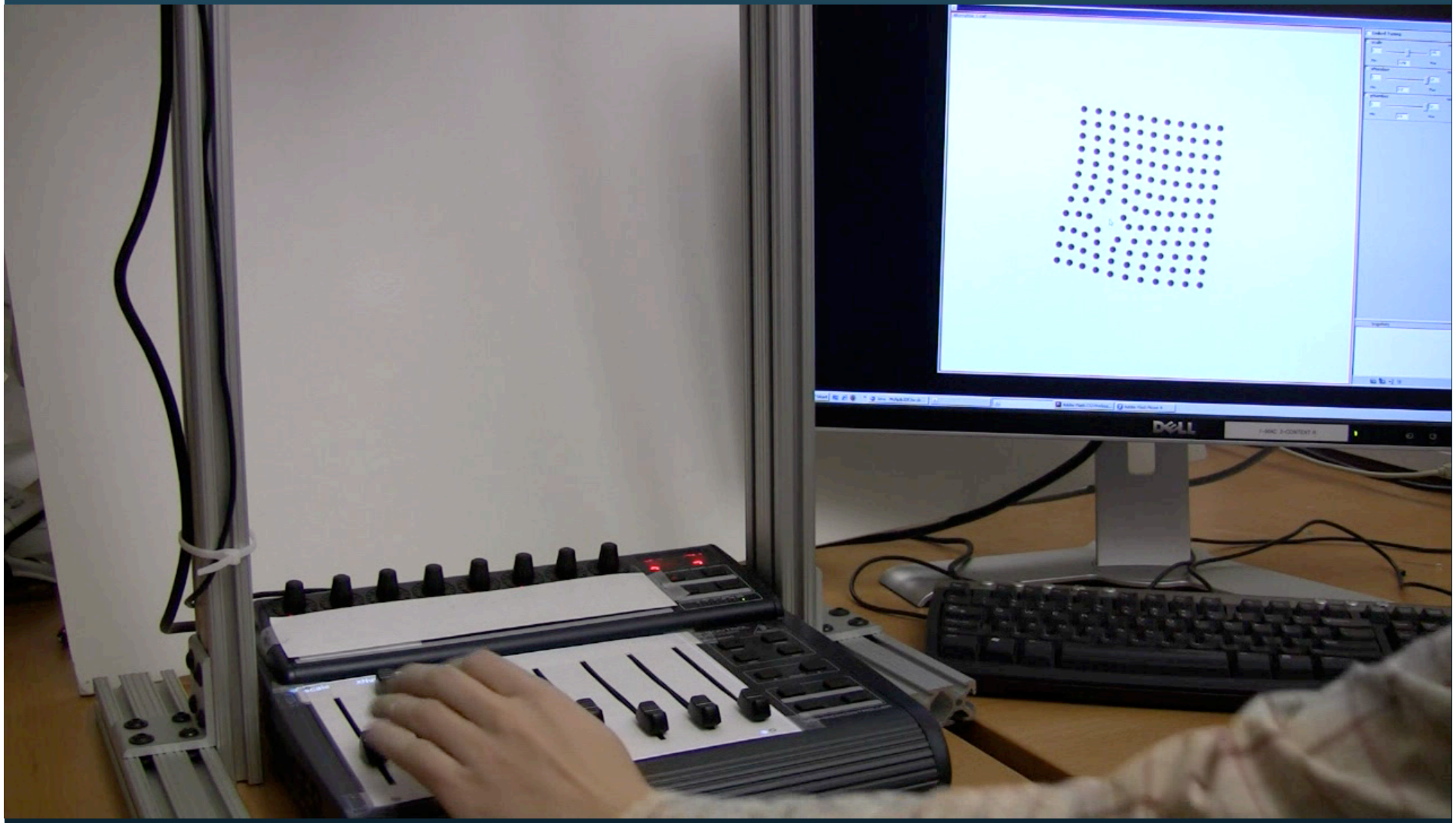




Mean Parameter Changes Tested per Minute
Changes/minute
90
80
70
60
50
40
30
20
10
0
64.26
2.60
Control
Juxtapose

## Tree Matching Task:
## Mean Completion Times by Tree

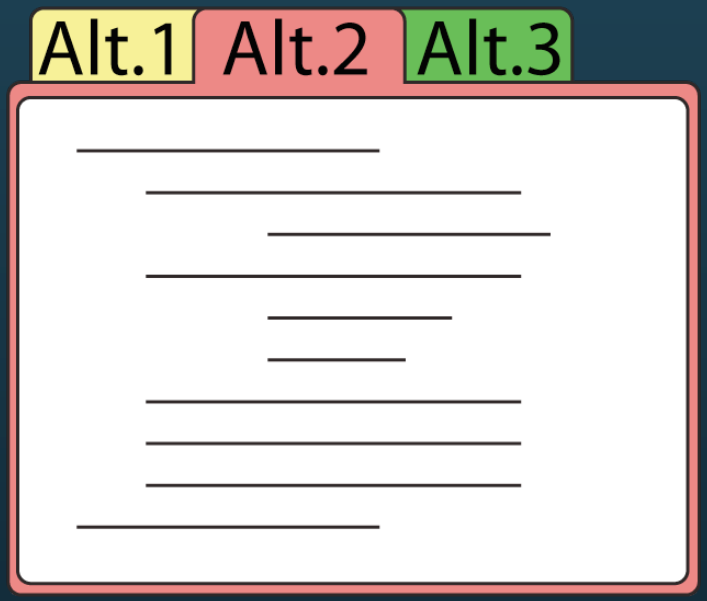


# Tangible Control

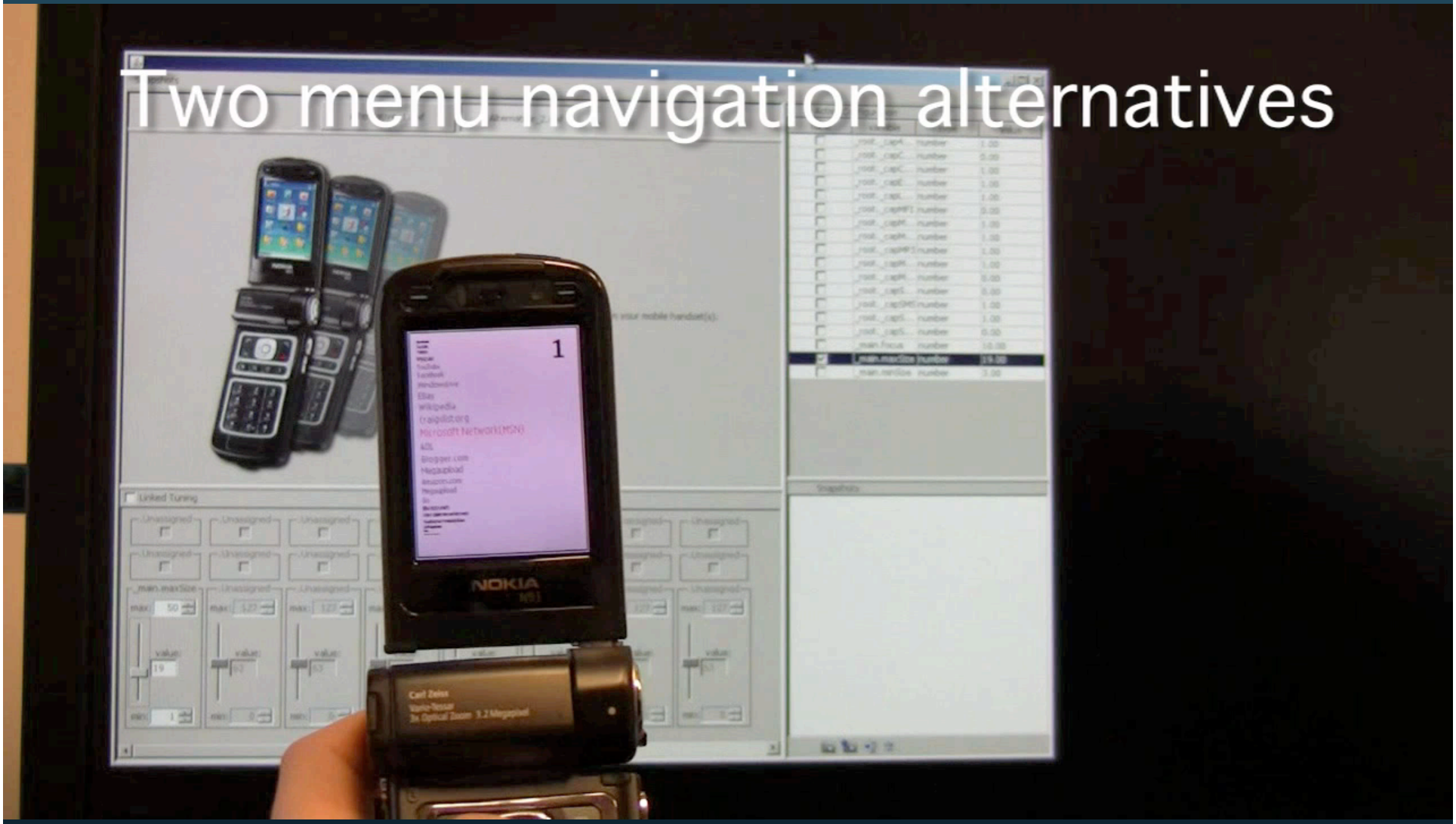


# Do these principles hold beyond desktop GUI applications?

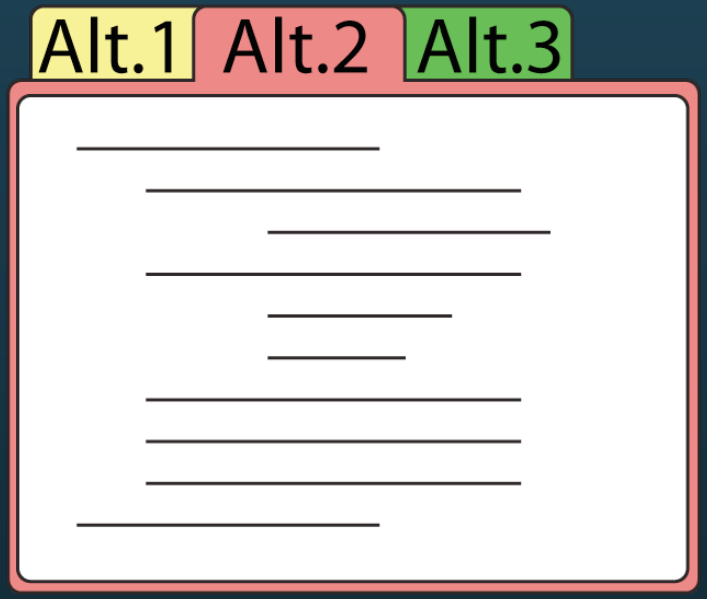
# Juxtapose Mobile

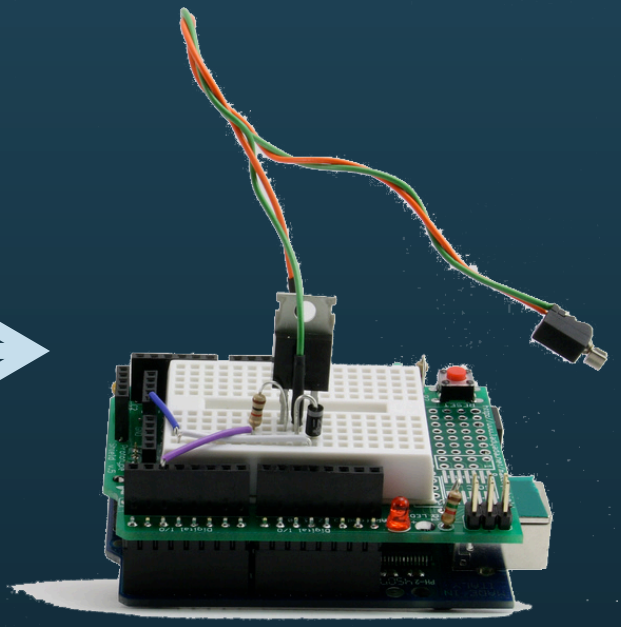



# Two menu navigation alternatives



# Juxtapose for Microcontrollers



**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 |

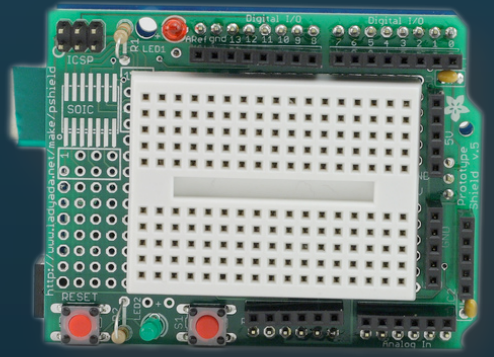
# Emit table into code & compile

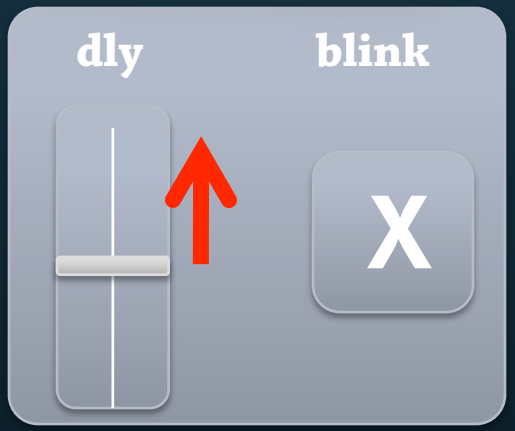

```
void initVarTable(void) {
  varTable[1].varName = PSTR("dly");
  varTable[1].varType = VAR_TYPE_INT;
  varTable[1].varPtr = &dly;

  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 cruft and enables more exploration.

Tool support requires connecting authoring and execution environments.

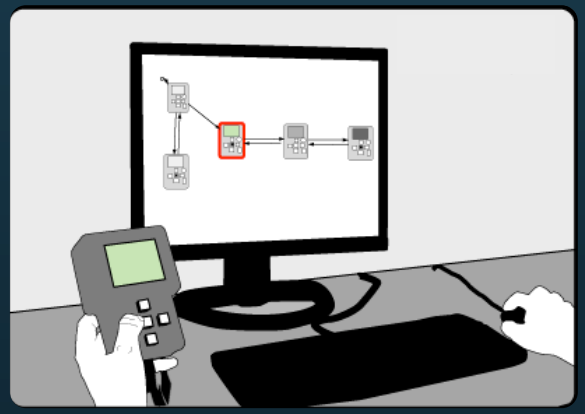
3 Techniques:

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

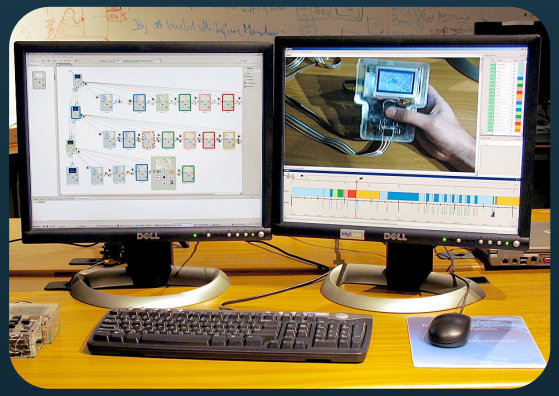
# SUMMARY & FUTURE AGENDA

# Prototyping Tools

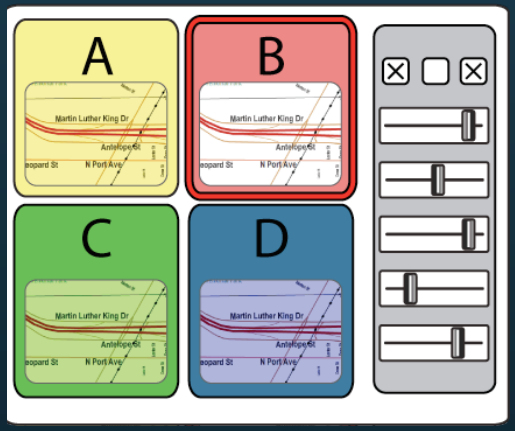
**1** Enable **Rapid Authoring** Off the Desktop



**2** Aid **Iteration** by Managing Feedback



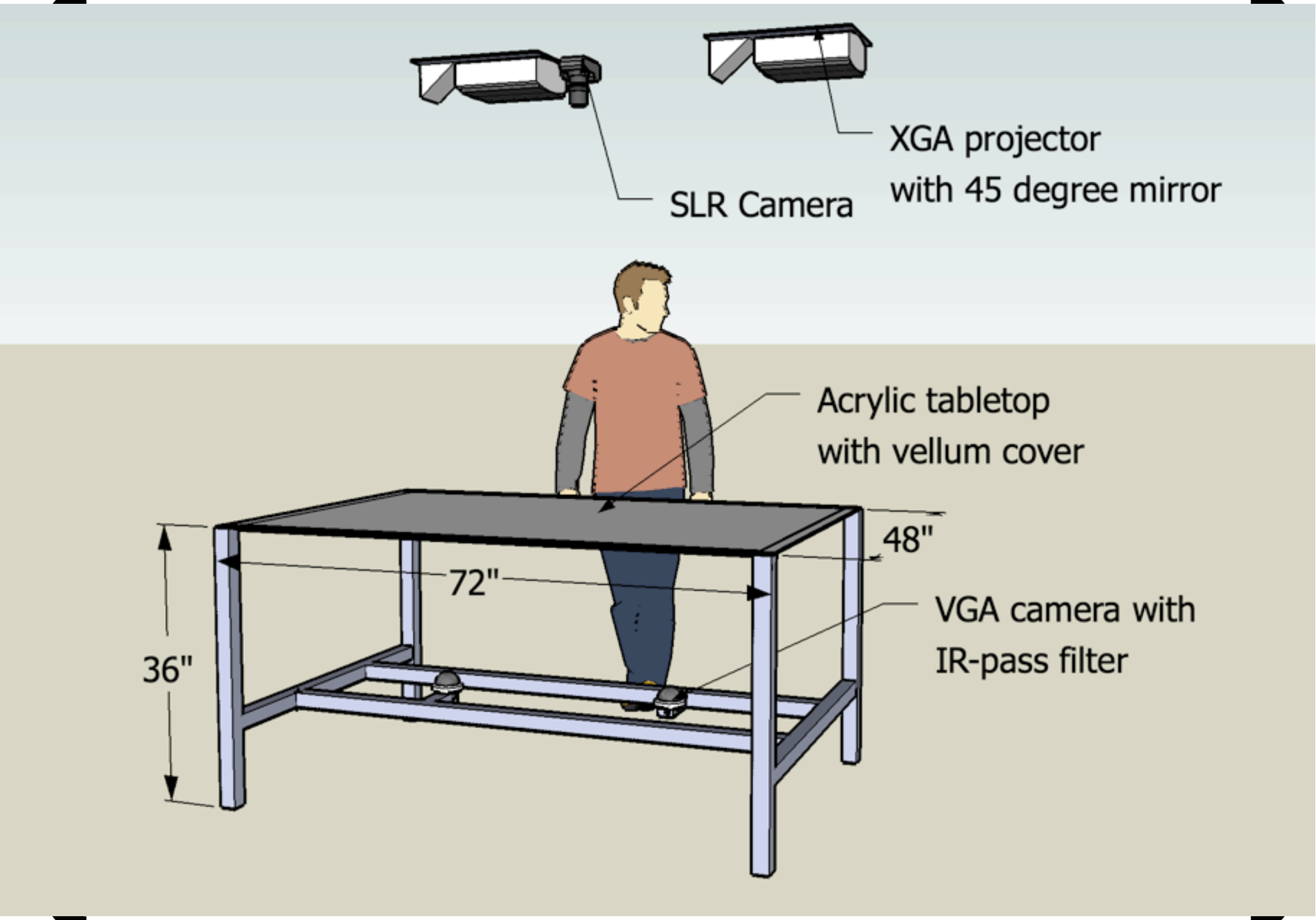
**3** Aid **Exploration** of Multiple Alternatives

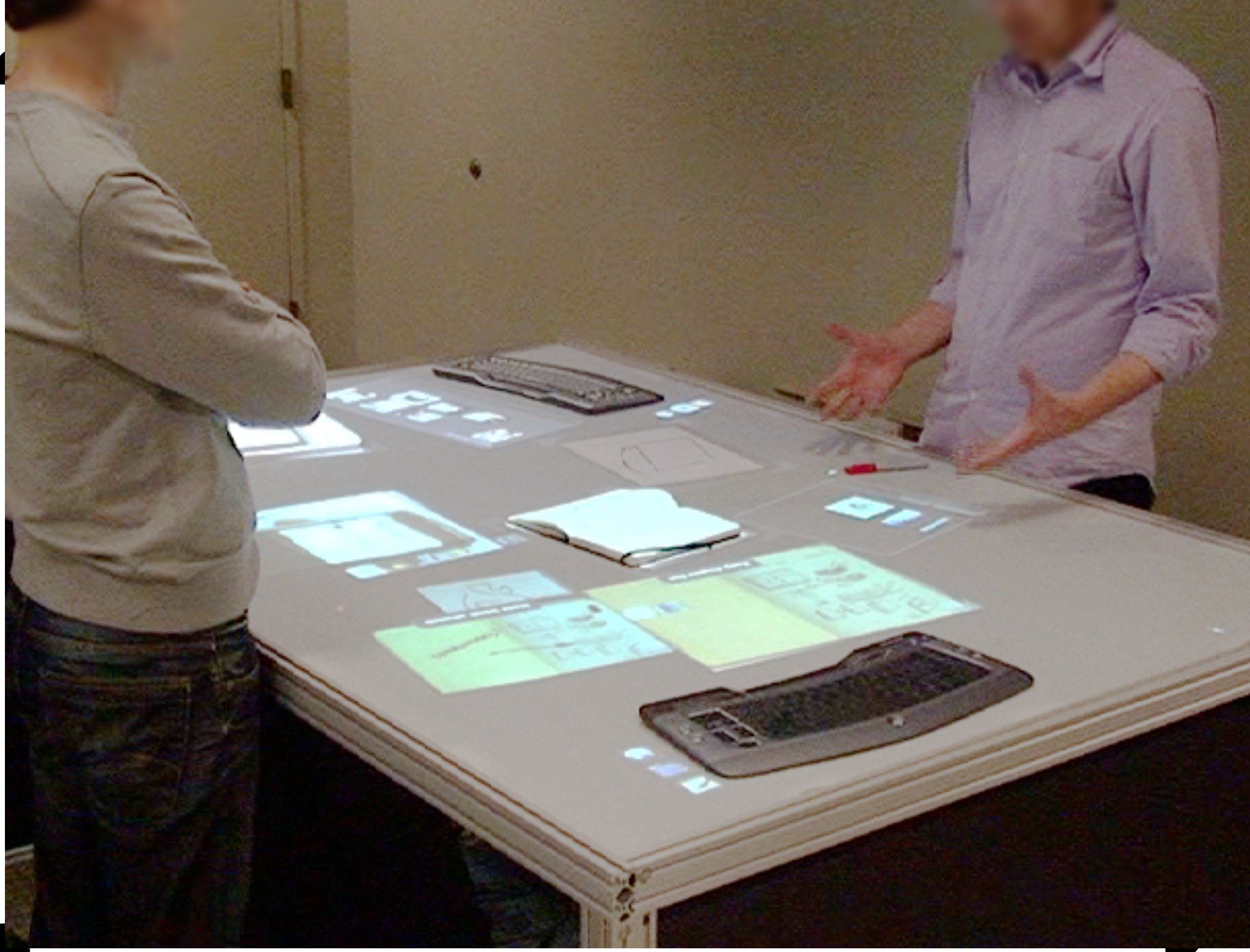


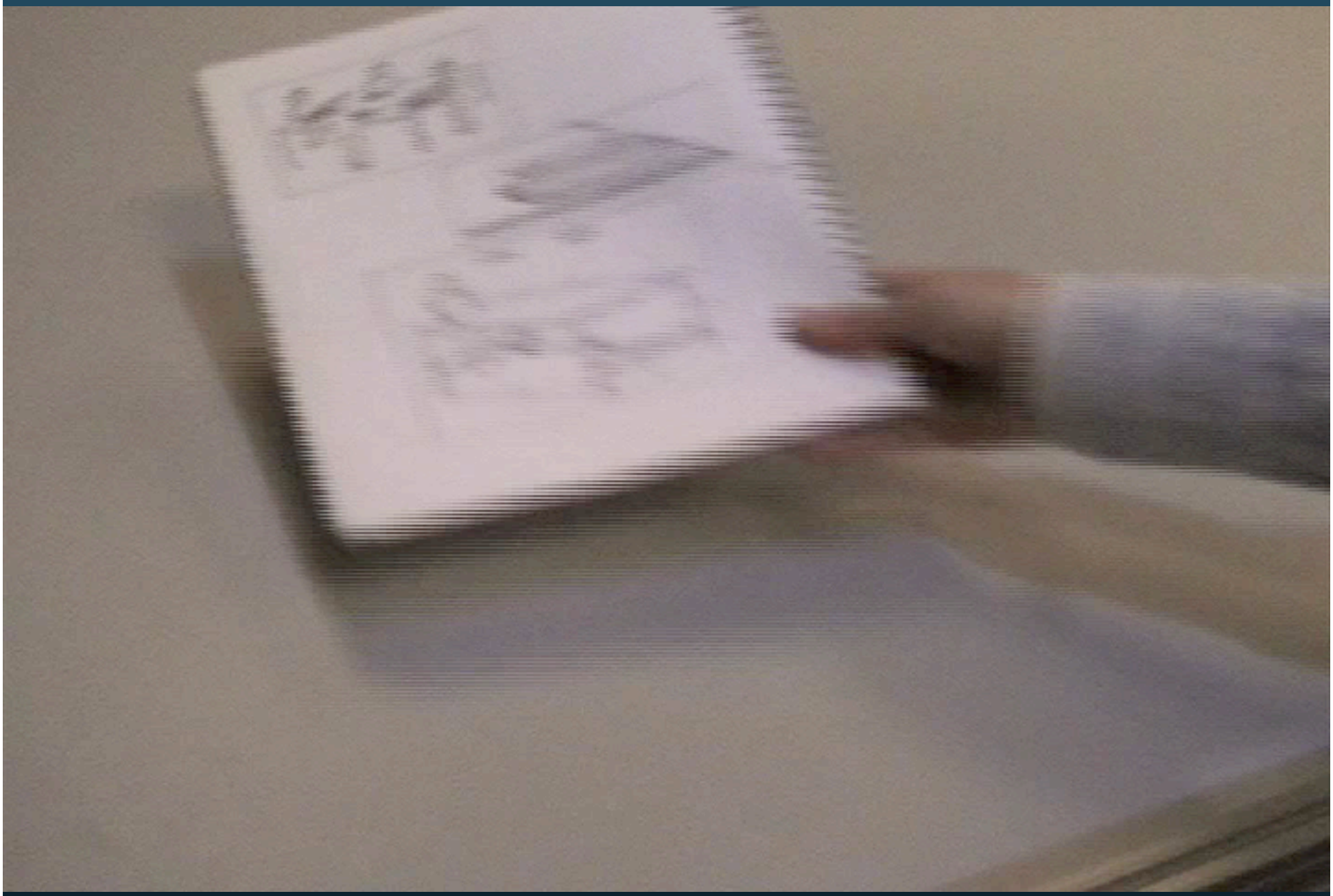
# Community Impact

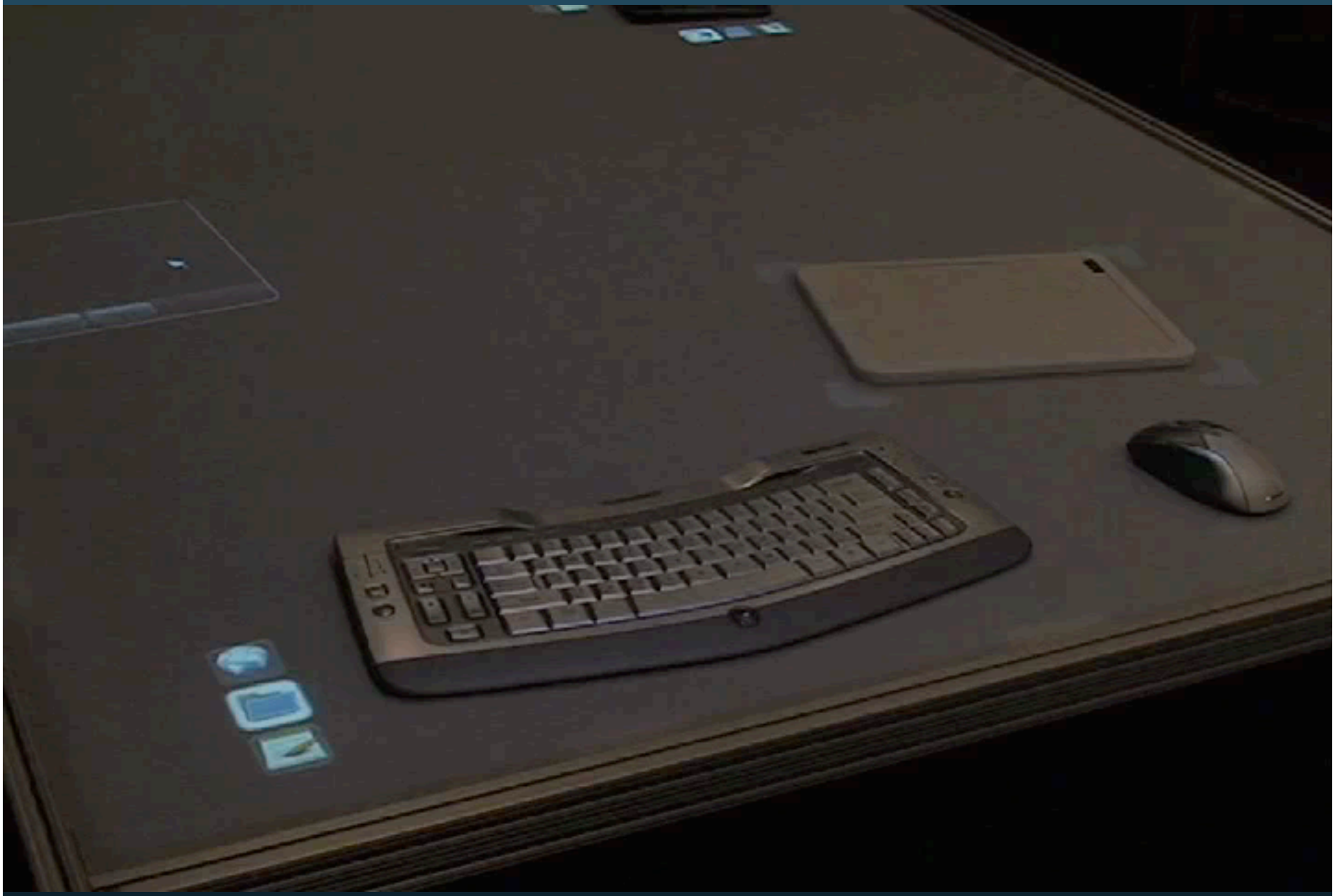
# Research Agenda

- 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
48"
72"
36"
VGA camera with
IR-pass filter








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)**

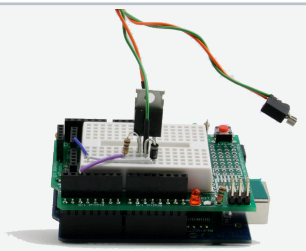
Hartmann, Björn, Scott R. Klemmer, Michael Bernstein, Leith Abdulla, Brandon Burr, Avi Robinson-Mosher, and Jennifer Gee. Reflective physical prototyping through integrated design, test, and analysis. In Proceedings of uist 2006: ACM Symposium on User Interface Software and Technology. Montreux, Switzerland,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