Handling Virtual Contact in Virtual Reality: Issues and Solutions

Dr. Robert W. Lindeman

The George Washington University

Institute for Computer Graphics

gogo@seas.gwu.edu

Overview

- *Motivation*
- *Vibrotactile feedback*
- *Our prototype*
- *Sensory substitution*
- *Motor substitution*
- *Unifying multimodal feedback*
- *Applications*

Problem Statement

- ***Virtual environments are typically limited to visual and audio feedback***
  - Do not faithfully recreate reality
  - Sensorially-deprived environments
  - Only receive feedback produced by the system
  - Do not take advantage of human bandwidth capacity

Problem Statement (cont.)

- ***Purely Virtual Objects (PVOs)***
  - Objects in the VE that have no physical component
  - Difficult to manipulate objects effectively
  - Feedback for understanding the nature of contact with objects is over-simplified
    - *Sound events*
    - *Change of color*

Problem Statement (cont.)

- ***Virtual contact***
  - What should we do when we know that contact has been made with a PVO?
  - The output of collision detection is the input to virtual contact
- ***It is difficult to perform interaction tasks from within a VR environment***
  - Configure things from "outside"
  - Enter the environment

The Nature of Contact

- ***Object properties***
  - Surface (texture)
  - Compliance
  - Physical makeup
- ***Contact properties***
  - Velocity
  - Location(s) on the object
  - Location(s) on the person

Active- vs. Passive-Haptic Feedback

- ***Active-haptic feedback***
  - Typically, force-reflecting devices under computer control
  - Expensive
  - Cumbersome
- ***Passive-haptic feedback***
  - Inherent properties of objects
  - Cheap
  - High fidelity
  - Limited amount and type of feedback

A Holistic Approach

- ***Stimulate multiple senses in concert***
  - Add support for vibrotactile (VT) feedback
  - Devise methods for combining feedback channels into a unified system
- ***Benefits***
  - Provide a deeper feeling of immersion
  - Allow us to use "lower-resolution" displays
  - Convey more information to users

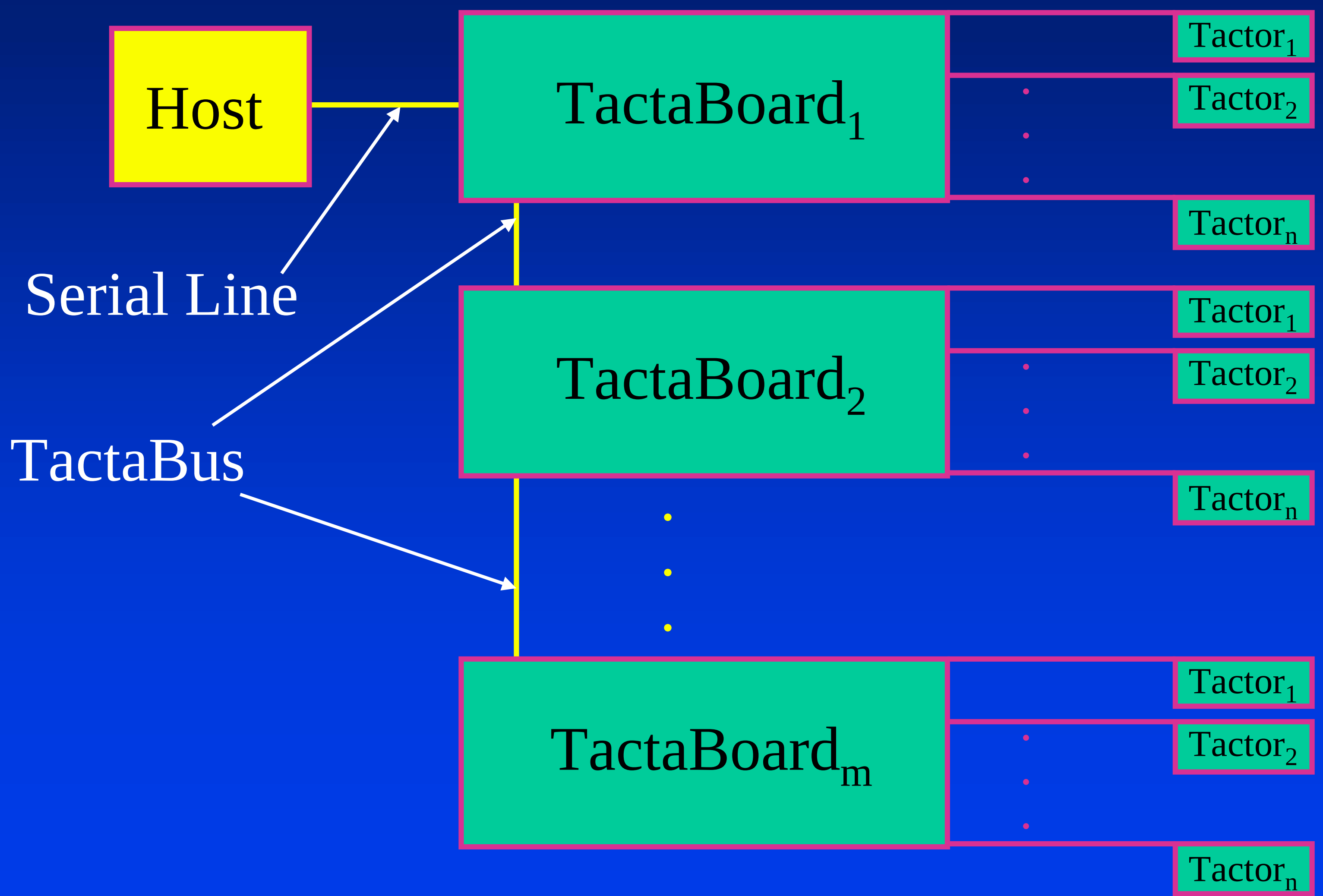
VT Feedback Devices

- ***VT feedback has been incorporated into many devices***
  - Used for decades for the hearing impaired
  - Widely used in cell phones and pagers
    - *"Manner" button*
  - Console controllers from Sony, MS, Nintendo
  - PC joysticks from MS, Logitech, etc.
  - Research devices from Immersion Corp., Virtual Technologies, etc.

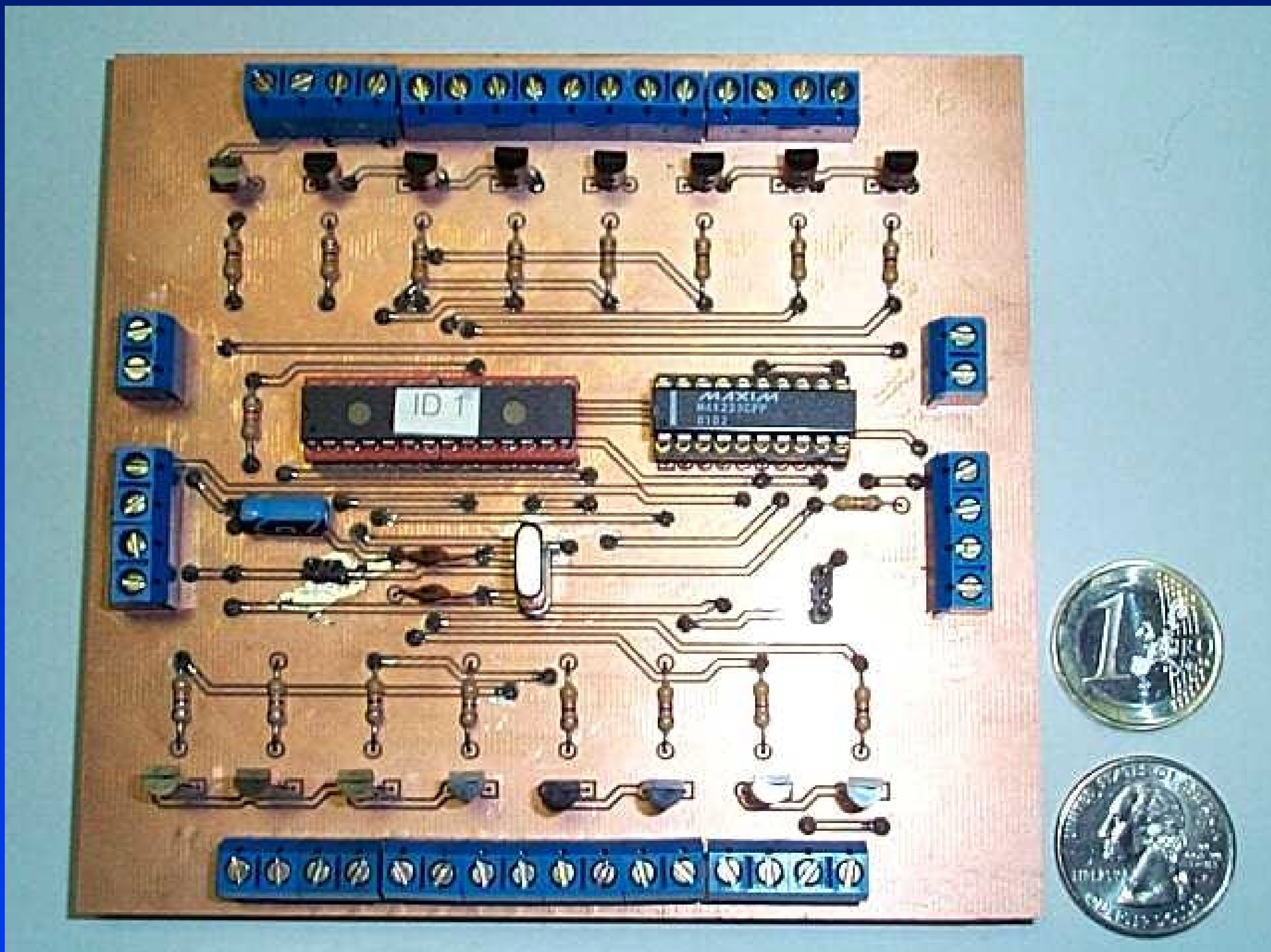
Prototype VT System: The TactaBoard

- ***Design goals***
  - Low cost
  - Low power
  - High update rate
  - Many form factors
  - Scalable
  - Different tactors
  - Individual control
  - Simple Interface
  - Wearable
- ***Design decisions***
  - Use COTS
  - Use PWM
  - Small number of tactors
  - Flexible design
  - Communication bus
  - External power supply
  - Multiple PWM signals
  - ASCII command set
  - Small footprint

The TactaBoard System (cont.)

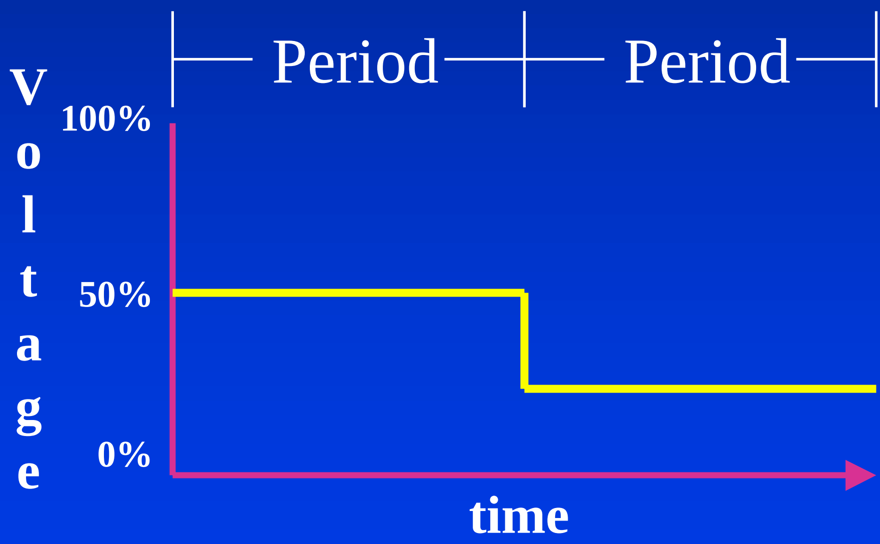


The TactaBoard System (cont.)

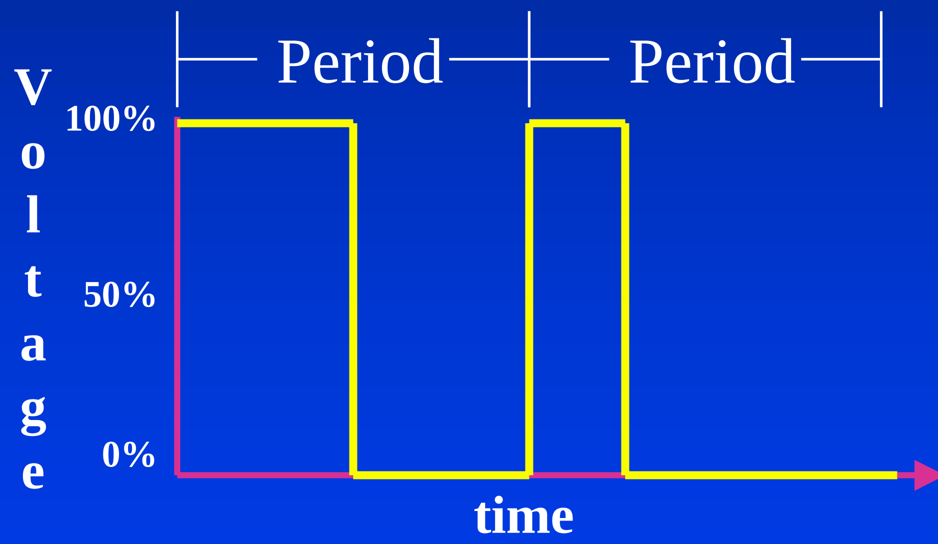


Pulse-Width Modulation (PWM)

- ***Shortening the duty cycle reduces the output voltage***



(a)



(b)

The TactaBus

- ***Serial connection from host to first TactaBoard***
  - RS-232
- ***TactaBus connection between TactaBoards***
  - CAN protocol*
  - Power for motors and PIC
  - Cat-5 cable and connectors

* planned

The TactaBoard Protocol

- ***ASCII command packets***
  - Set motor *m* on board *b* to value *v*
    - *0 <= m <= 254, 255 = broadcast*
    - *0 <= b <= 254, 255 = broadcast*
    - *0 <= v <= 255*
  - Enable/disable motor
  - Reset PIC
  - Query motor level*
  - EPROM curve definition*

* planned

The TactaBoard API

- *C/C++ API*

```
int main( int argc, char* argv[] )  {
  TactaBoard tb;
  double x = 4.71;
  unsigned char v;
  int i;

  tb.open( "COM1" );

  for( i = 0; i < 1000; i++ )  {
    v = ( 1.0 + sin( x ) ) * 127.5;
    tb.SetOutputValue( BOARD_ID, OUTPUT_ID, v );
    printf( "Value: %u\n", v );
    x = x + 0.02;
  }

  tb.SetOutputValue( BOARD_ID, OUTPUT_ID, 0 );
  return( 0 );
}
```

Varying the Feedback

- ***Individual tactors***
  - Frequency
  - Amplitude
  - Temporal delay
  - Pulses
- ***Groups of tactors***
  - Waveform
  - Tactor placement
  - Interpolation method

Sensory Substitution vs. Motor Substitution

- ***Sensory Substitution***
  - Replacing feedback to one channel with feedback to another channel
  - Creates a disjoin between real and virtual experience
- ***Motor Substitution***
  - Restricting the motion of one's body parts (limb segments, *etc.*) so that they no longer match one-to-one to their real-world counterparts

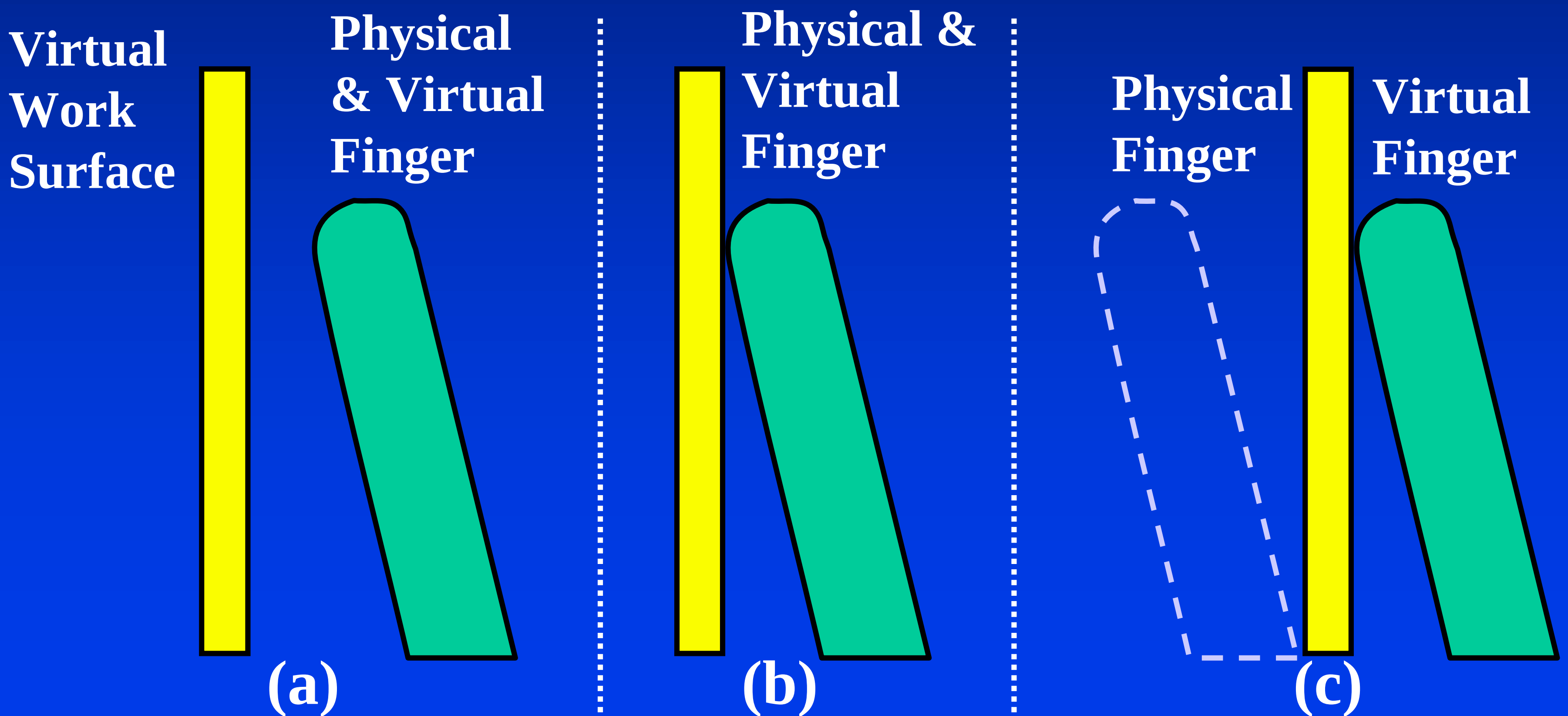
Sensory Substitution

- ***Examples:***

Feedback Technique	Modality	Mapped to…
Color change	Visual	Location/depth of penetration
Vector glyphs	Visual	Force and direction of contact
Texture distortion	Visual	Location/depth of penetration
Shape distortion	Visual	Location/depth of penetration
Contact illumination	Visual	Location of collision
Pitch change	Auditory	Depth of penetration
Amplitude change	Auditory	Force of collision
Spatialization	Auditory	Location of collision
Vibrotactile amplitude	Haptic/Tactile	Location/velocity/depth of penetration

Motor Substitution

- ***Clamping***
  - Imposition of surface constraints

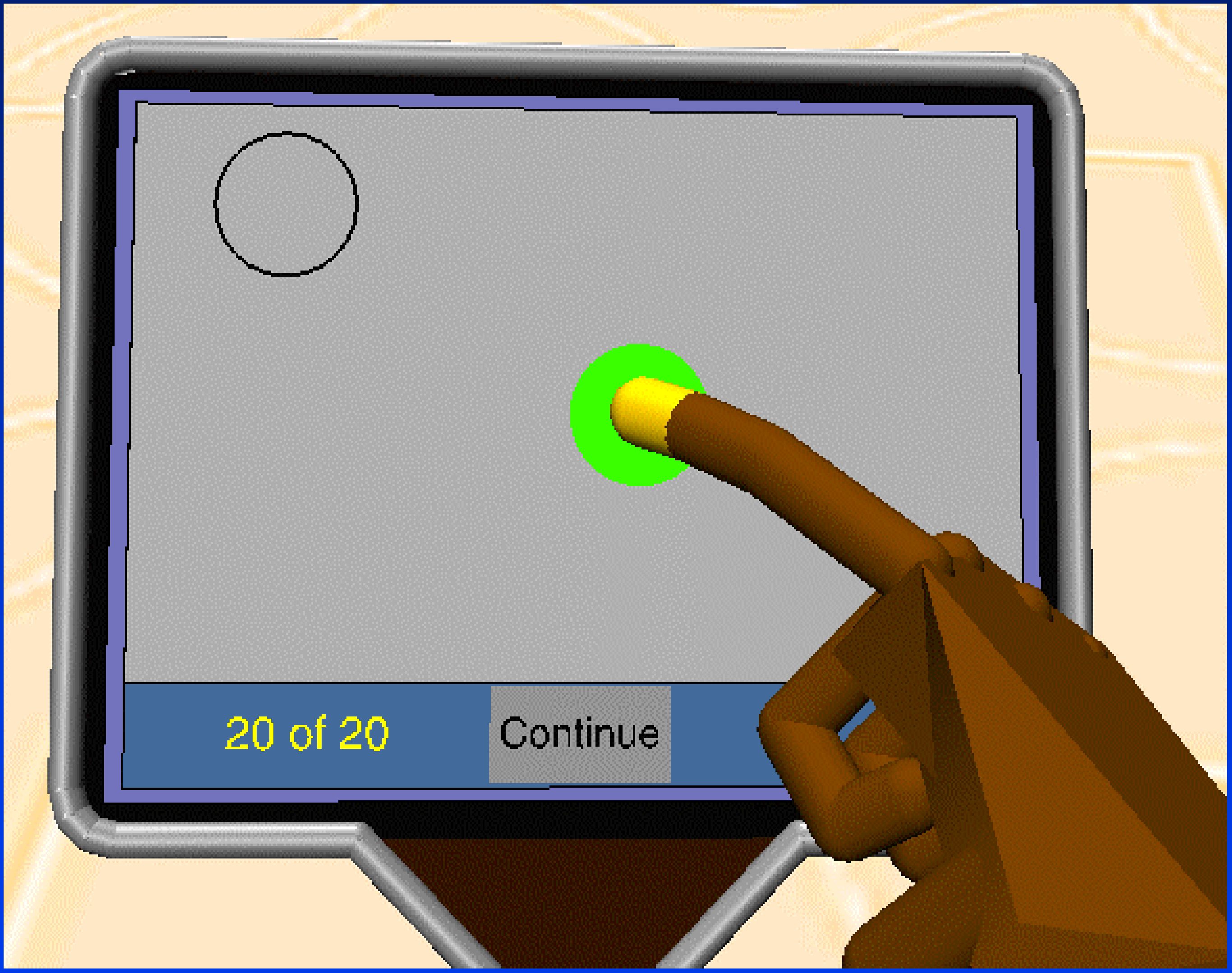


Simulated Surface Constraints

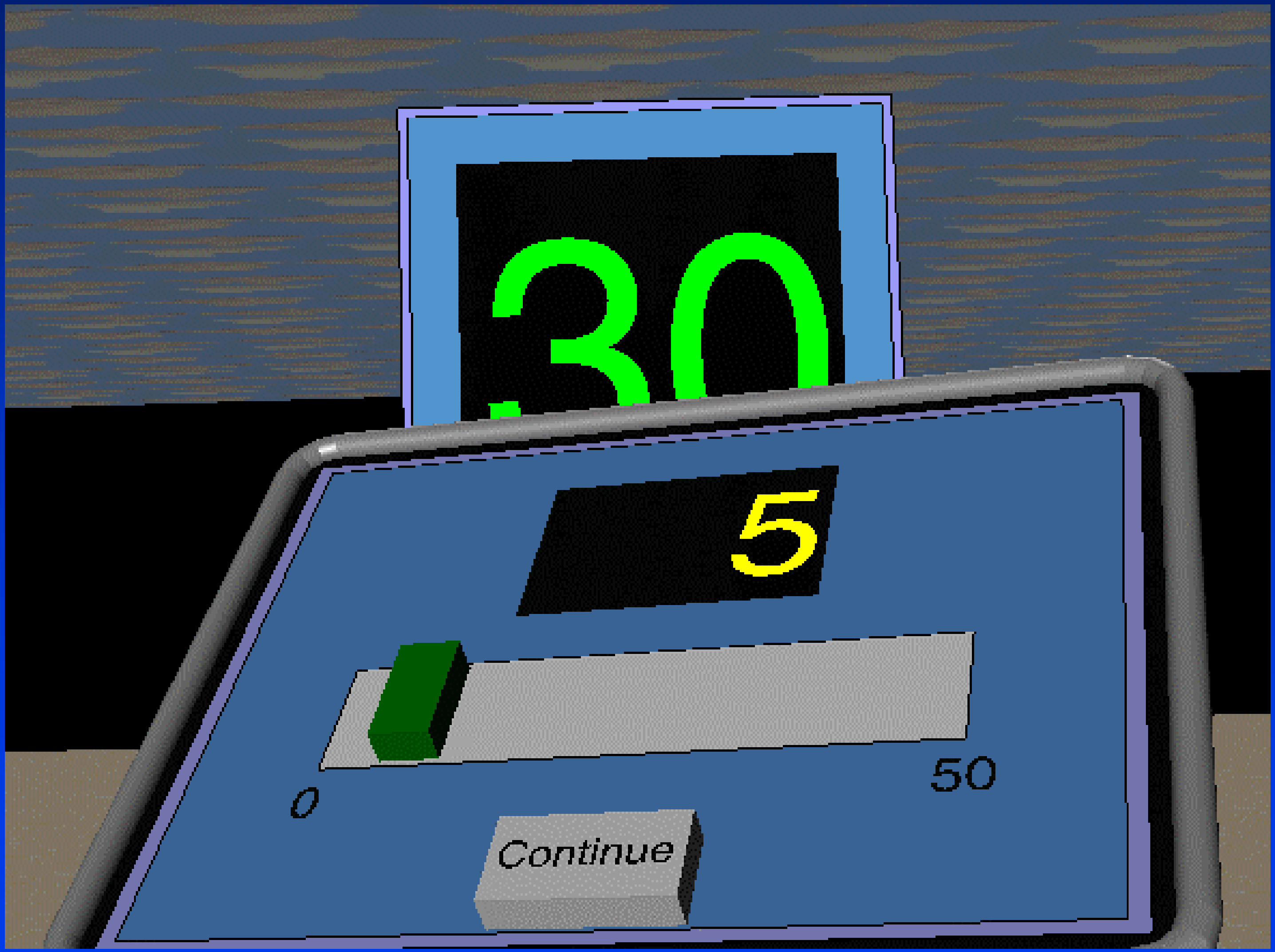
- ***Physical surfaces constrain the movement of the finger***
  - Reduce the degrees of freedom, improving control
  - Our previous work showed that having a physical surface is superior to not having one
  - We can improve performance by simulating the presence of physical surfaces

Physical Setup

The Docking Task (2D Widget Representation)

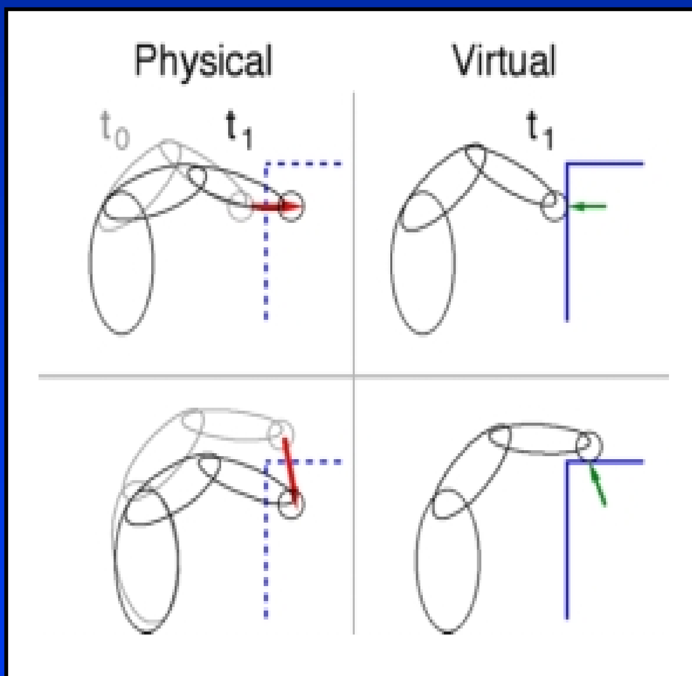


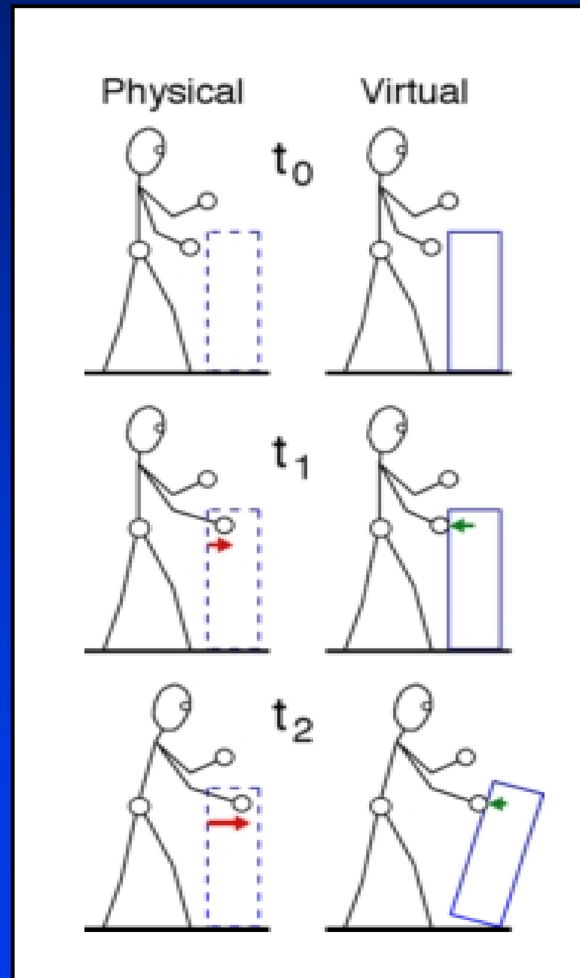
The Sliding Task (3D Widget Representation)



Motor Substitution

- ***Difficult problems***
  - Trajectory-based computation
  - Object properties





Multimodal Feedback

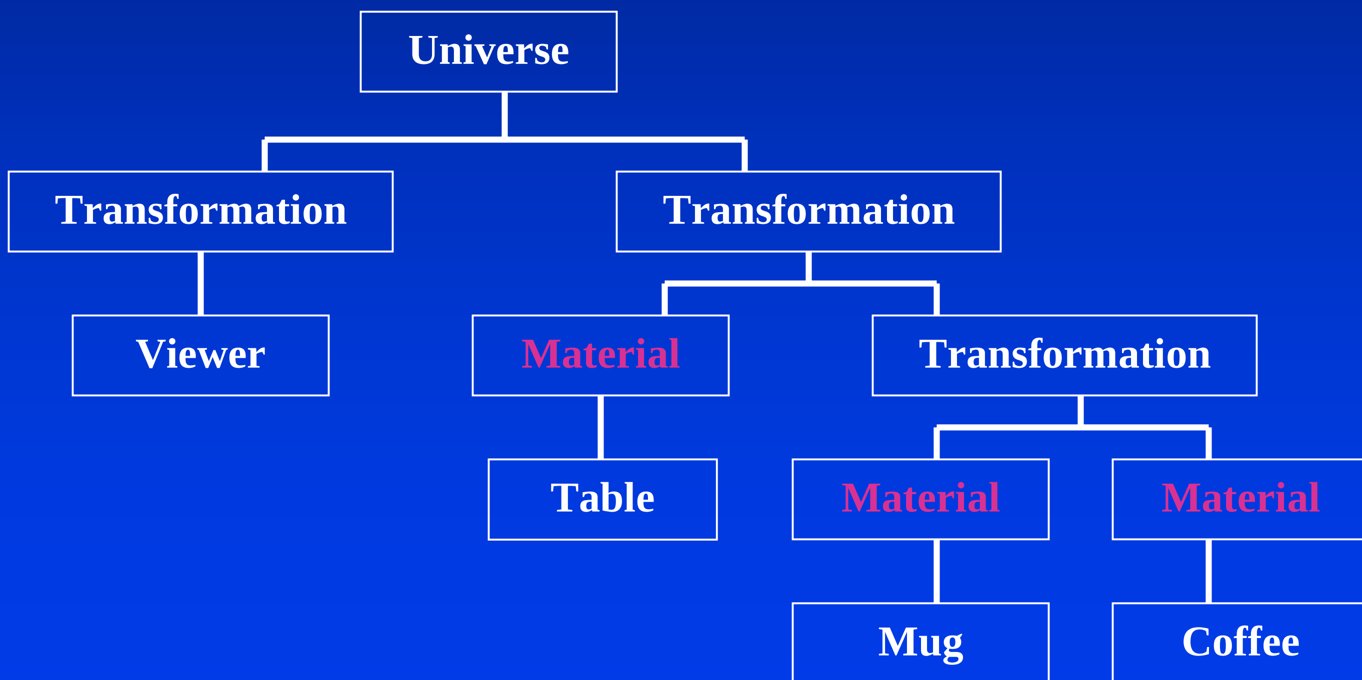
- ***Display device makers each have their own APIs***
  - Visuals: Performer, Java3D, OpenInventor
  - Audio: RSX, DirectX, A3D, VAS
  - Haptics: Ghost, ReachIn
  - Smell? Taste?
- ***Motivation***
  - Different hardware
  - Need for speed
- ***Programmer must update each modality when the scene changes***

The Unified Scene Graph (USG)

- ***Design goals***
  - Provide programmer with a single scene graph
  - Need to retain speed
  - Need to support different "renderers"
- ***Two main concepts***
  - Material node on steroids
  - Compile/flatten routines

Object-Oriented Model

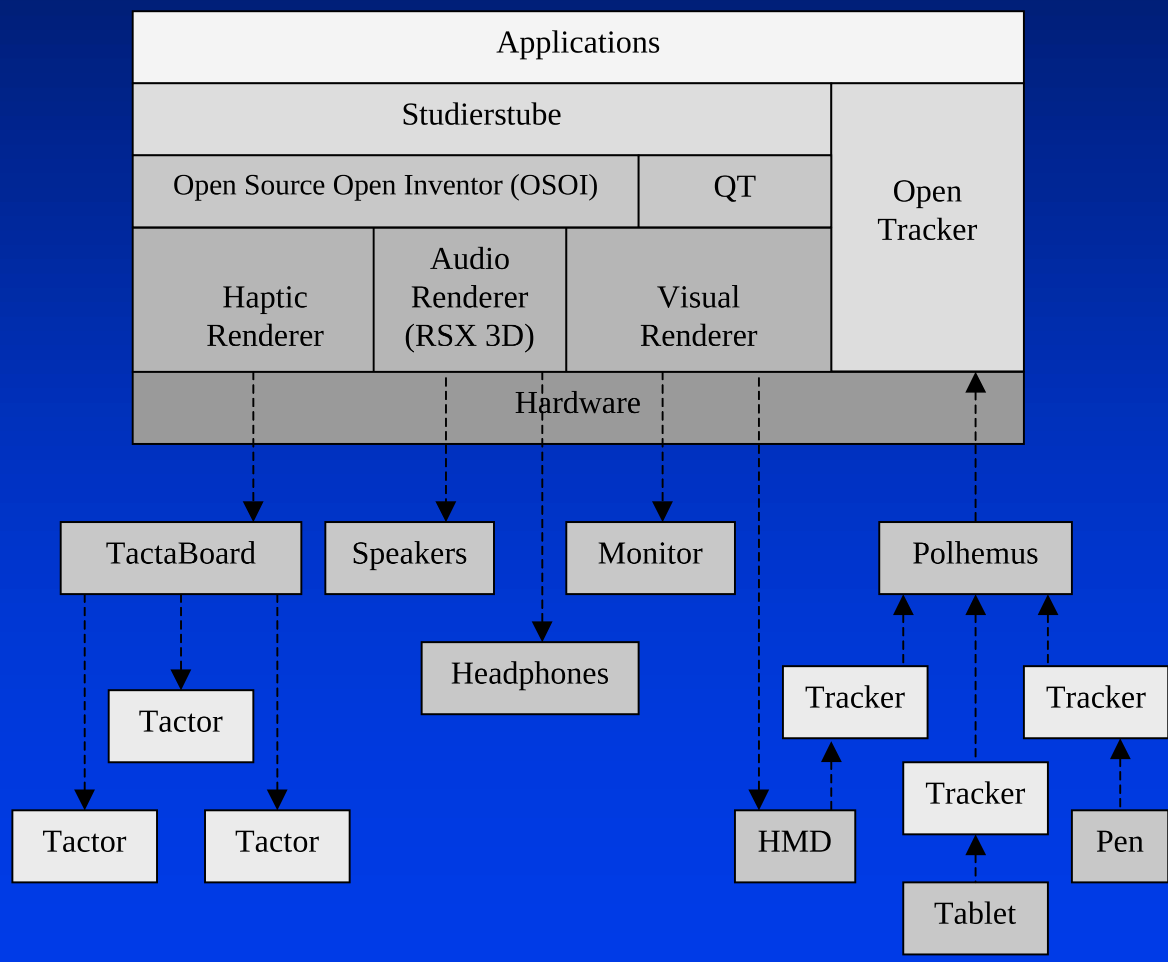
- ***Associate all properties in Material Node***



Modality-Specific Compile Methods

- ***Renderers can access state from scene graph traversals***
  - Culling
  - Transformation stack
  - etc.
- ***Need code to translate from USG to proprietary renderers***
  - Might need to communicate between renderers

USG Prototype System

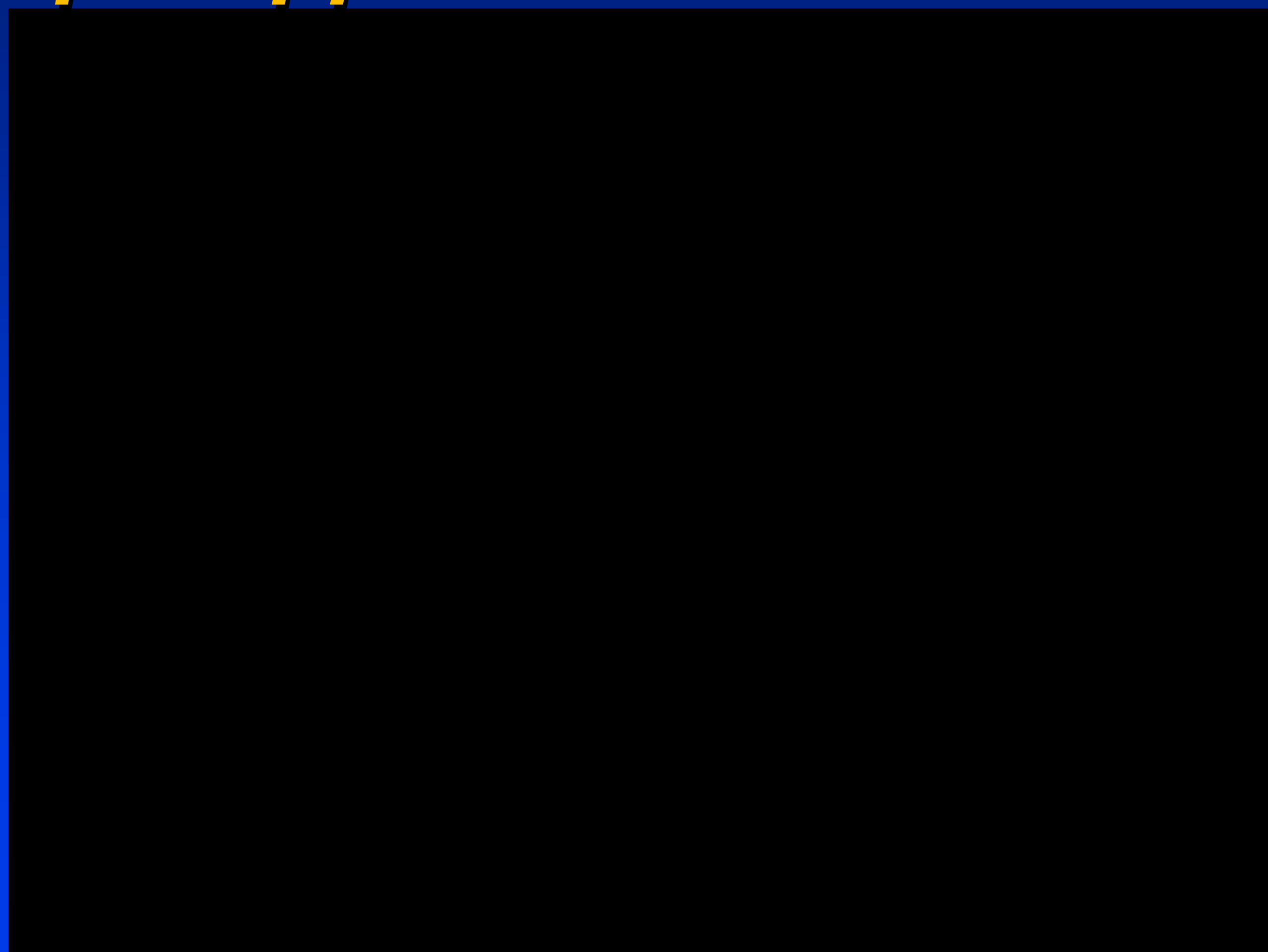


USG Prototype System Videos

- ***System setup***

USG Prototype System Videos

- ***Sample application***



Sample Applications

- ***Non-verbal communication***
  - SWAT teams
  - Firefighters
- ***Improved HCI***
  - Low cost = increased accessibility
  - Standard interface = easier integration
- ***Data perceptualization***
  - Map additional variables in Viz applications