# A Virtual Reality Cookbook

Jean-Francis Balaguer
Enrico Gobbetti

Computer Graphics Laboratory
Swiss Federal Institute of Technology
CH-1015 Lausanne, Switzerland

E-MAIL: {balaguerlgobbetti}@di.epfl.ch





# A Virtual Reality Cookbook

Introduction

Temporal-spatial realism

Sensory feedback

The VB2 system

# Virtual Reality

## Goal

- Convince the participants that they are in another place





# Virtual Reality

## Technique

- Participants' sensory inputs are replaced with synthetic information
- Participants are put in the loop of a real-time simulation

# Virtual Reality

## Motivation

- Participants' tasks are shifted from cognitive to perceptual activities
- The correlation between manipulation and effect on manipulated information is increased

## Participants can concentrate on application tasks



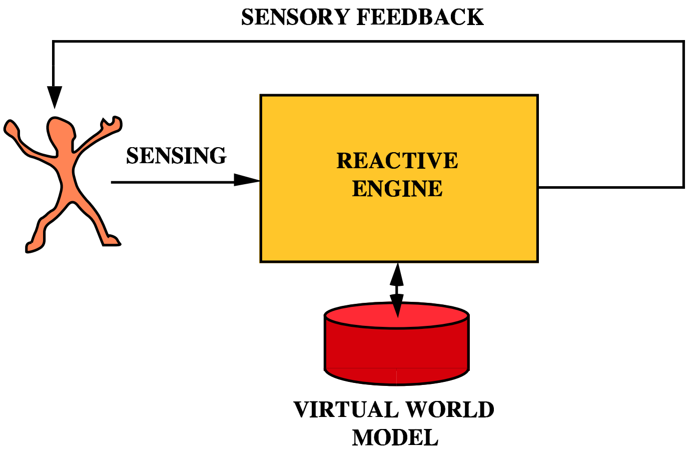


# Virtual Reality

## Needs

- Input devices
  - Sense participants motion
- Output devices
  - Replace participants' sensory input
- Reactive applications
  - Immediate response to participants' action
  - Simulation of the virtual world

# The Reactive Engine








# Reactive Applications

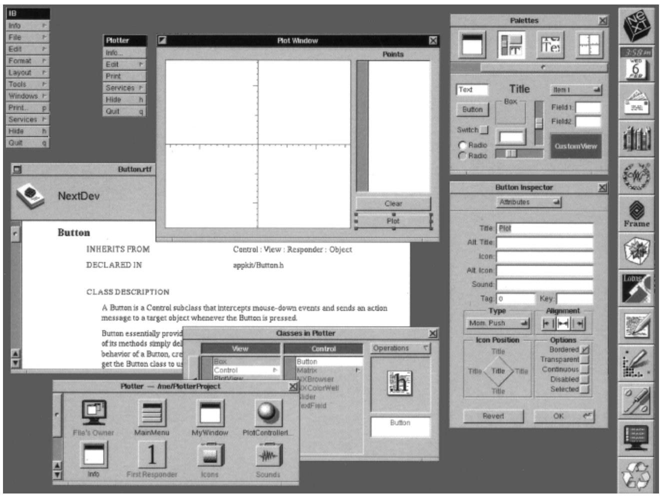
Reactive applications are not a new idea:

- Ivan Sutherland, 1965
- Alan Kay, 1969

Modern interactive systems are based on these concepts

- The DeskTop is a virtual environment!

# The DeskTop Virtual Environment








# The Desktop Virtual Environment

## Input device (mouse)

- 2D motion sensing

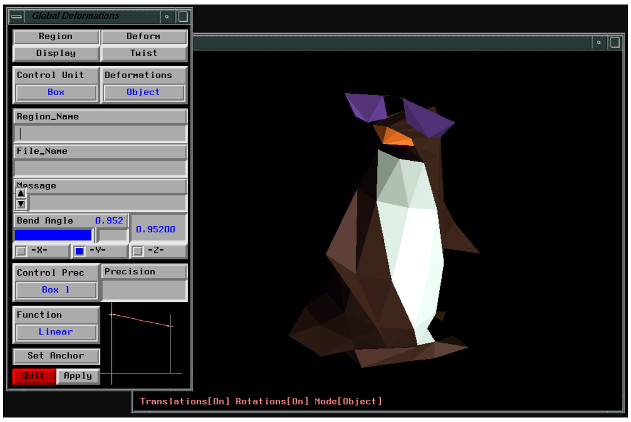
## Output device (screen)

- 2D image, visual feedback

## Direct manipulation

- Continuous two-way man-machine communication
- Physical metaphors, visibility of operations

# DeskTop: A 2D world!

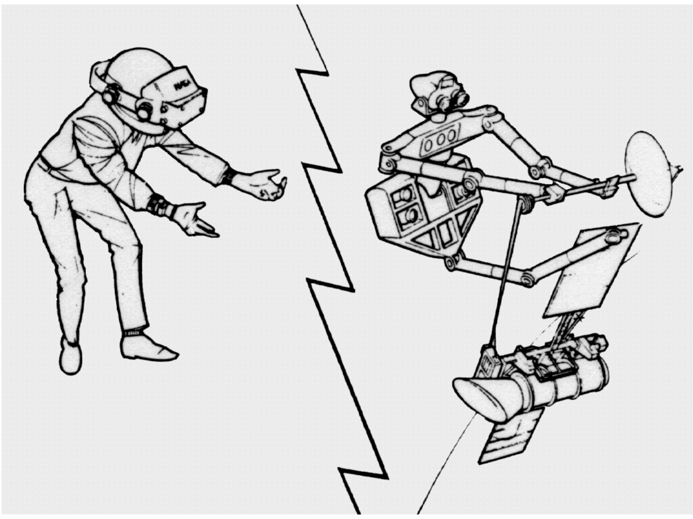



The manipulation of 3D information
requires a 3D interface!





# Teleoperation



# Synthetic Environment






# Synthetic Environment

# 3D Virtual Environments

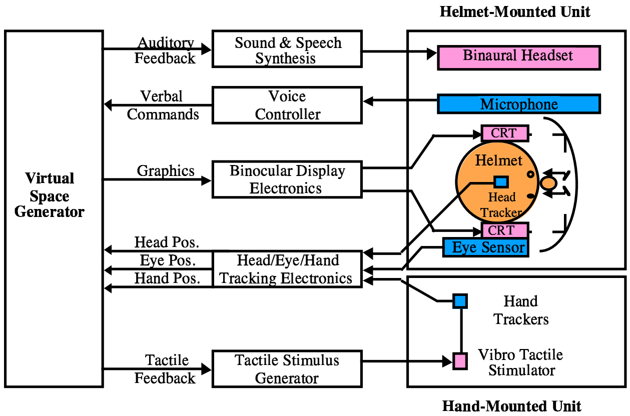
Same basic reactive model as for 2D applications, but:

- Need for temporal-spatial realism
- Need for new input devices
  - 3D motion sensing
- Need for new output devices
  - Depth cues to help 3D perception
- Need for new interaction metaphors
  - Manipulation of 3D information





# 3D Virtual Environment

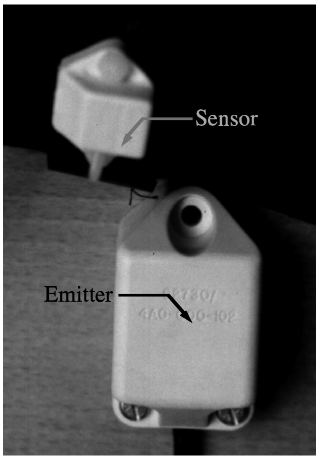



- Head tracking for viewpoint specification
- Hand tracking for manipulation

# Tracking

## Polhemus IsoTrak

- Absolute position and orientation tracking
- Magnetic tracker
- Update rate
  - 60 Hz with one sensor
- Accuracy
  - decreases with sensor-emitter distance
  - at 75 cm from source
    - 0.63 cm in translation
    - 0.85° in orientation



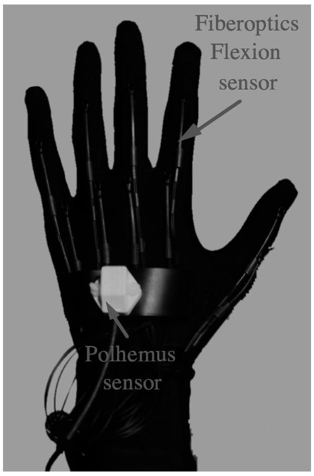




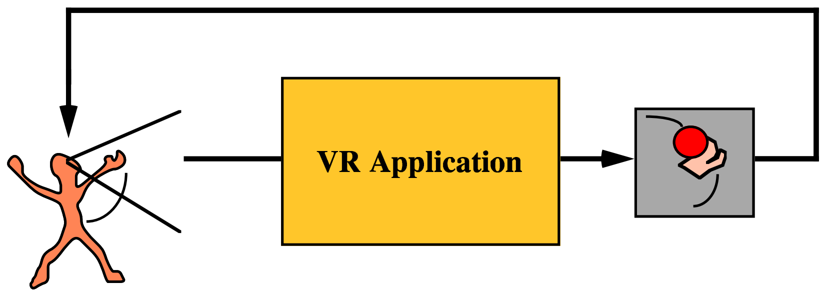

# Hand Measurement

## DataGlove

- Polhemus tracker
- Finger flexion sensors
  - 5° resolution
- Update rate
  - Finger data: 60 Hz
  - Finger and transform data: 30 Hz



# Temporal-Spatial Realism

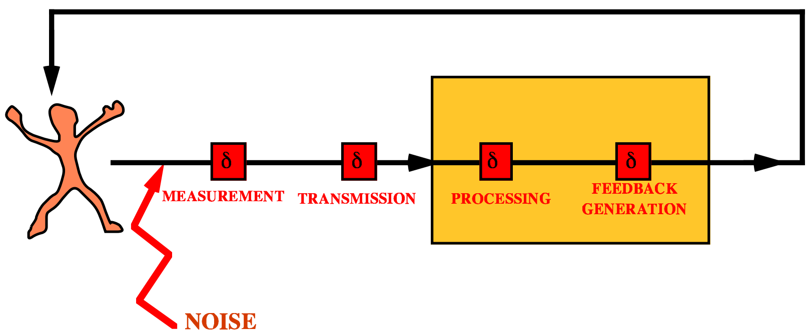



VR requires correspondance between participants' motion and sensory feedback



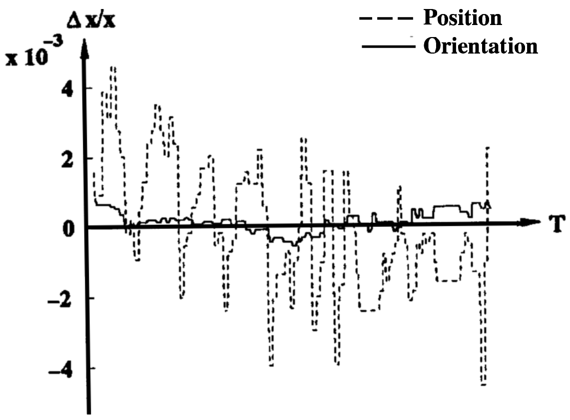


# Noise and delay




Noise and delays cause temporal-spatial distorsion

# Polhemus Isotrak Noise








# Polhemus Isotrak Delays

- Measurement: 80-150 ms
- Transmission (IPC): 10-50 ms
- Processing and feedback generation delays are dependent on the application

# Temporal-Spatial Distortion

Noise in sensor data

- Jittering of images

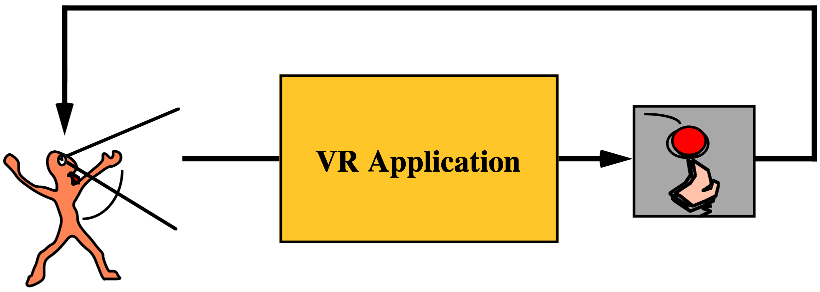
Accumulation of delays

- Lag between participants' motion and sensory feedback





# Temporal-Spatial Distortion




Motion Sickness

Difficulty in performing participants' tasks

# Temporal-Spatial Realism

## Noise reduction techniques

- Use better trackers
- Smoothing

## Delay reduction techniques

- Speed-up applications
  - Faster trackers
  - Faster machines
  - Parallel processing (pipe-line)
- Prediction





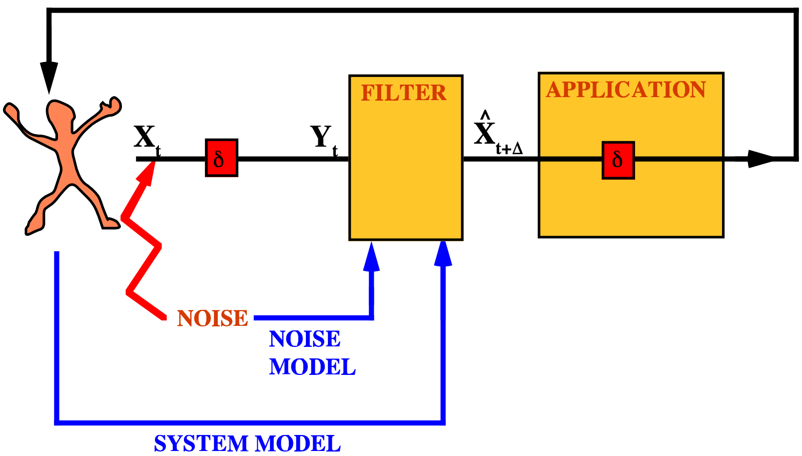
# Temporal-Spatial Realism

## Noise and delay reduction

- Simple low-pass filters increase delays
- Simple predictors increase noise

## Noise and distortion have to be handled together

# Optimal Estimation








# Kalman Filter

Optimal linear estimates of the state of dynamic models

- Maximum likelihood estimates for Gaussian noises
- Weigthed least square estimate for non-Gaussian noises

Can be used to predict future values of the state vector

# System model

## System model

$$X_{t+\Delta t} = f(X_t, \Delta t) + \xi(t)$$

***f*** models the dynamic evolution of the state vector ***X***

$\xi$ is a white noise process

## Observation model

$$Y_t = h(X_t, \Delta t) + \eta(t)$$

***h*** models the measurement process

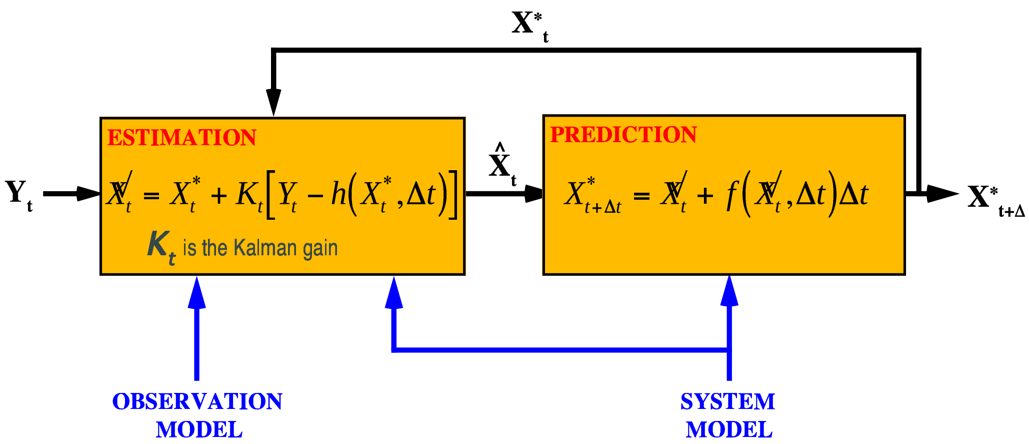
$\eta$ is a white noise process

***Y*** consists of the sensor readings





# Kalman Filter




- The Kalman gain determines the influence of the residual in updating the estimate

# Kalman Filter

## Filter design steps

- Choose state variables
- Define the random process models
  - System process
  - Observation process
- Find optimal values of the model parameters
  - Measure device noise
  - Minimize filter's error on training data





# Kalman Filter: Problems

## Humans are not simple dynamic processes

- Their behavior is variable and depends on the tasks

## Modeling human behavior with a state equation is difficult

# Modeling Complex Behavior

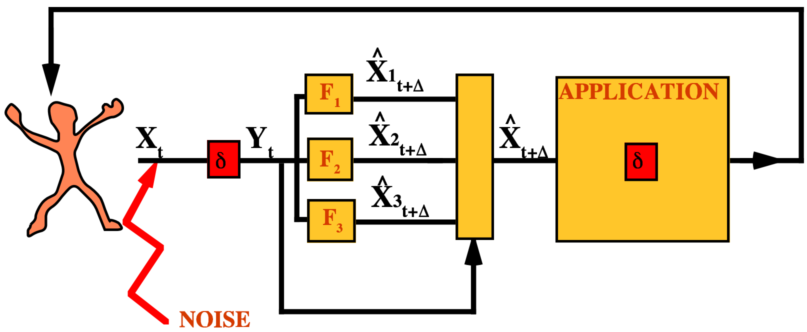
## Participant's behavior is modeled by a state machine

- Each state represent a particular well-defined behavior
  - looking-around, observing, manipulating, etc.
- Each state has its own filter
- The correct filter is chosen based on the prediction error

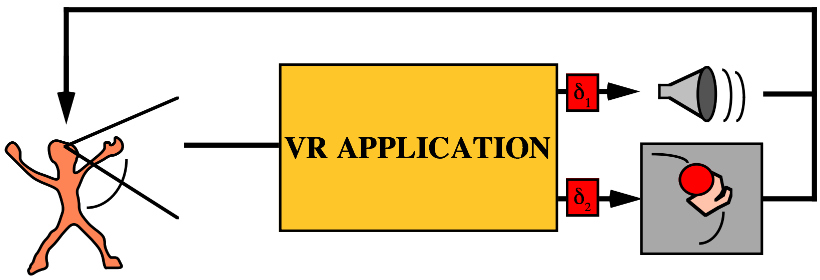




# Multiple Filters



# Feedback Synchronicity




VR requires feedback synchronization



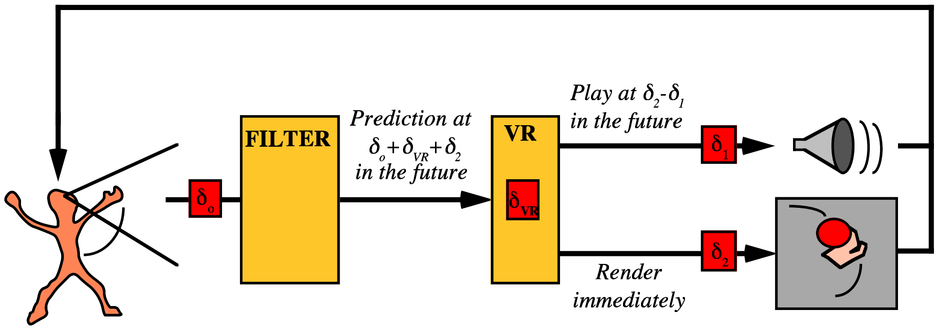


# Feedback Synchronicity

To obtain feedback synchronicity

- Compute longer output generation delay
- Use longest delay for input prediction
- Compensate difference in output delays by using lead times

# VR Application








# Sensory Feedback

VR systems must provide information on the virtual world through sensory feedback

- Visual feedback
- Sonic feedback
- Force feedback
- Tactile feedback

# Visual Feedback

## Feedback must match human vision capabilities

- Should provide cues for depth perception





# Depth Perception

## Shading and Shadows

- Help reconstruction of shapes

## Parallax/Relative Motion

- Perceived image changes when the viewer changes position

## Binocular disparity

- Images perceived by left and right eyes for a given point differ in their horizontal position

# Depth Perception

## Perspective

- Far objects appear smaller than close objects

## Occlusion

- Far objects are hidden by closer objects on the line of sight





# Depth Perception

## Convergence

- Coordinated rotation of eyes when focusing on an object in space

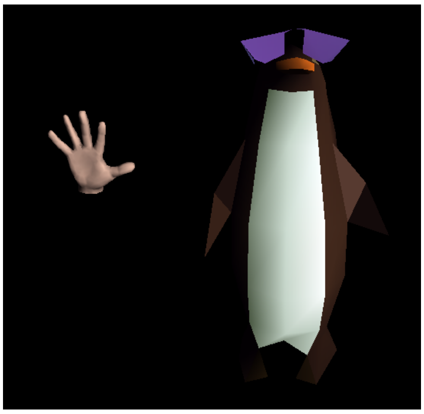
## Accommodation

- Muscular tension needed to adjust the focal length of the crystalline lens to focus on an object in space

# Standard Interactive Displays

## Limited Depth Cues

- Perspective
- Hidden surface removal
- Basic shading







# Stereo Graphics Displays

## Binocular disparity

- Compute the left and right images from different positions

## Head tracking

- Allows the simulation of relative motion

## Fixed accommodation

- Decoupled accommodation and convergence

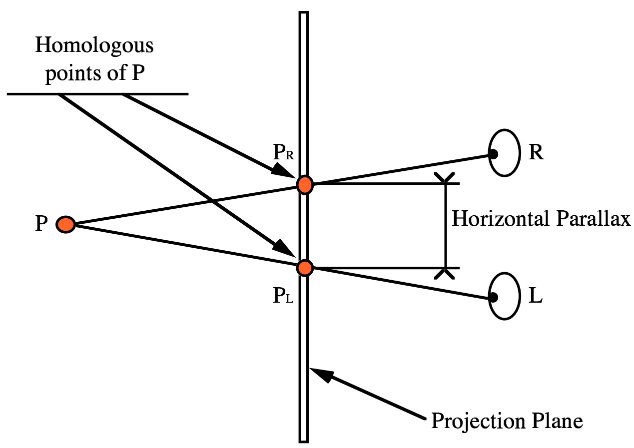
# Computing Stereo Pairs: Generalities

- Horizontal parallax
- Vertical parallax
- Rotations and Perspective
- Translations and perspective

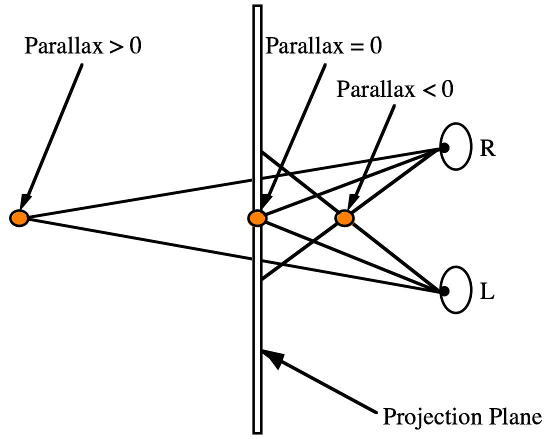




# Horizontal Parallax



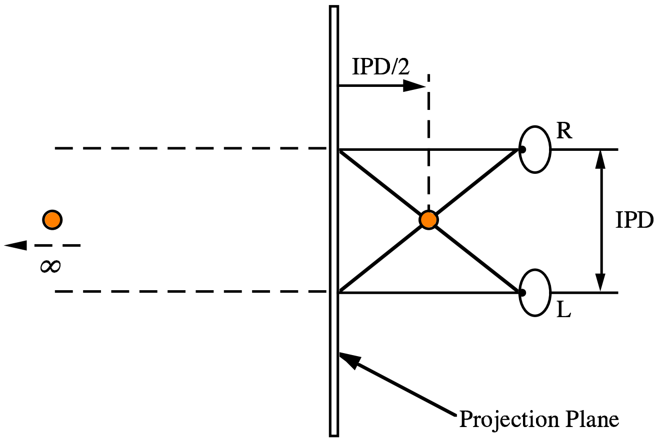
# Horizontal Parallax








# Horizontal Parallax Properties



# Vertical Parallax

## Definition

- Difference between the vertical coordinates of homologous points

## Causes difficulties for image fusion

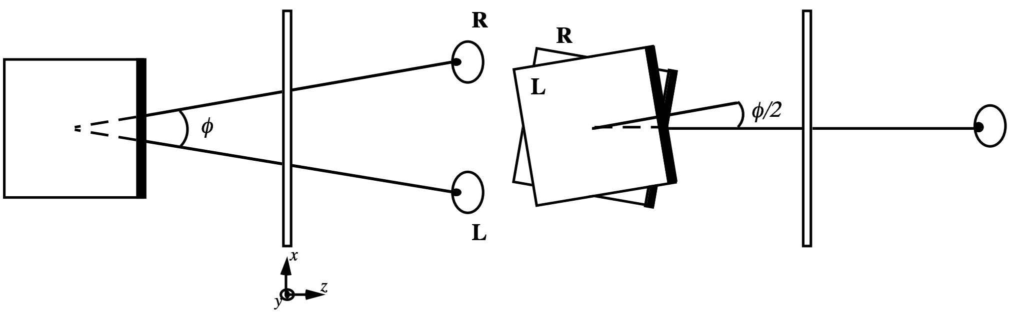




# Rotations and Perspective

## Motivation

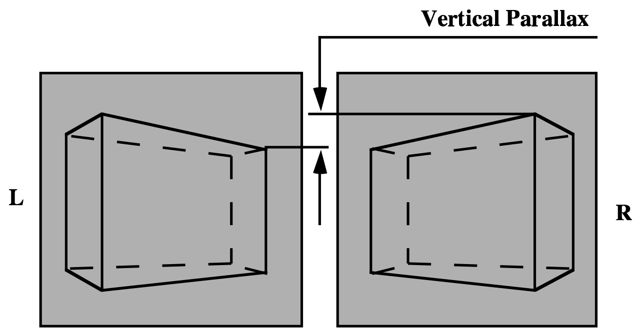
- Rotate scene around Y axis to simulate eye convergence



# Rotations and Perspective

## Problem

- Introduces non constant vertical parallax



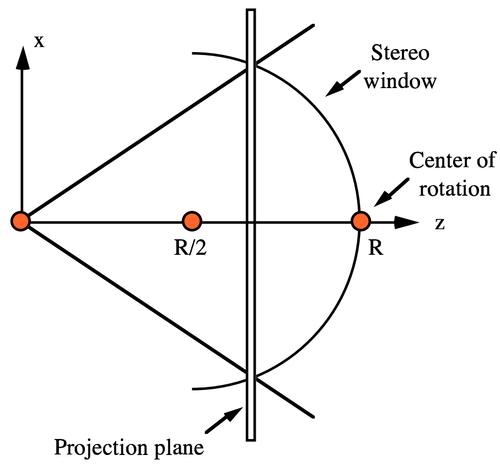





# Rotations and Perspective

## Problem

- Non planar stereo window introduces distortion

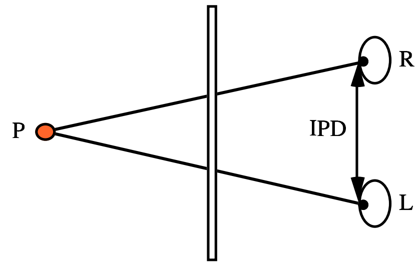


# Translations and Perspective

Use two centers of projection

Properties

- No vertical parallax
- Stereo window is planar and coincides with projection plane








# Hardware configuration

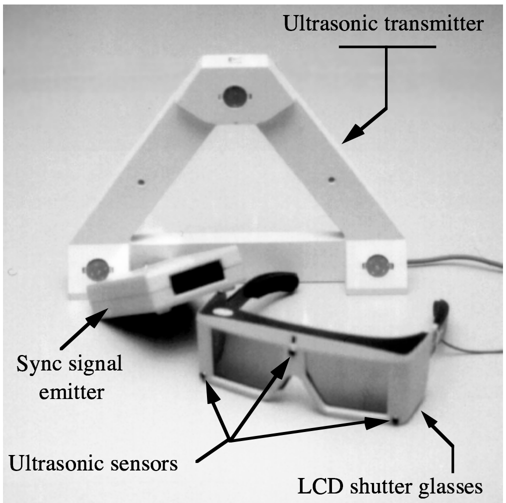
Fixed displays

- CrystalEyes

Head mounted displays

- EyePhones

# Cristal Eyes/Cristal Eyes VR








# Fixed Displays - Head Tracking

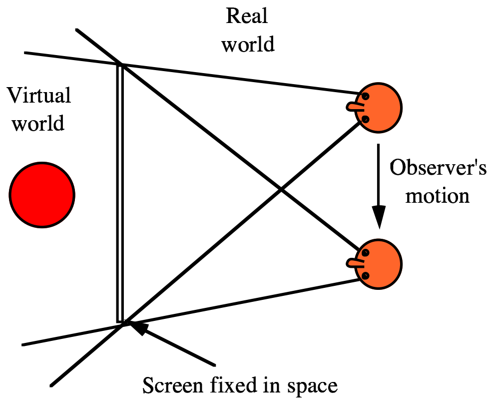
The screen is a window on the virtual world

- Window is fixed in space
- Software must precisely model hardware configuration
- Software must take into account user's IPD

# Fixed Displays - Head Tracking

## Tracking

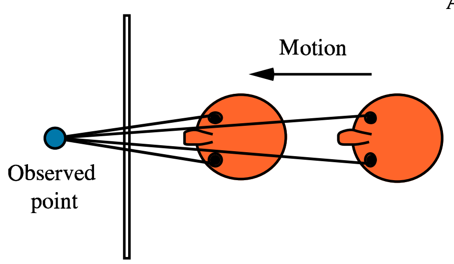
- Software must take into account observer's motion
- Viewing volumes must be recomputed continuously



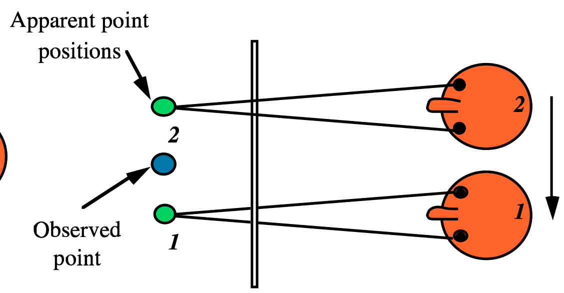





# Fixed Displays - No Head Tracking




Parallax changes with head motion




Distortion caused by head motion and fixed horizontal parallax

# Head Mounted Display

## EyePhones

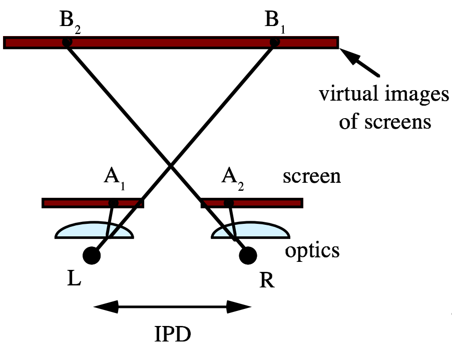
- Non see-through HMD
- Polhemus magnetic tracker
- LCD screens
  - 210x140 color pixels
- LEEP optics
  - Horizontal field of view (FOV)

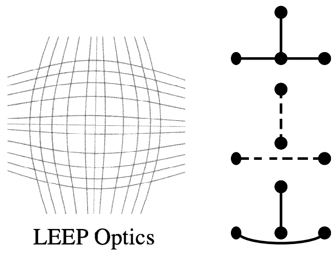
    Single eye: 75°, Overlapped: 60°, Binocular 90°
  - Vertical field of view: 58.4°

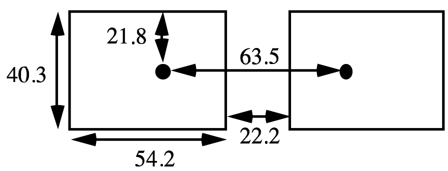




# EyePhones



# VB2



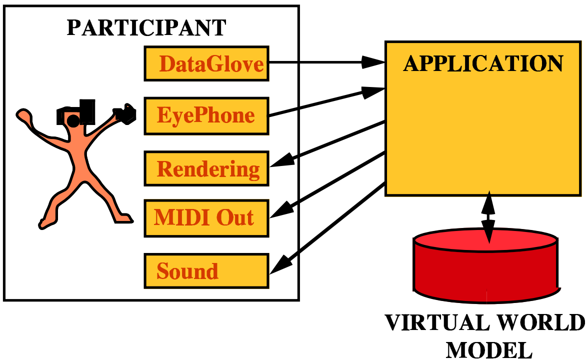



# The VB2 System

## Goals

- Provide a basis for constructing VR applications
- Allow experimentation of 3D interaction techniques

# System Structure

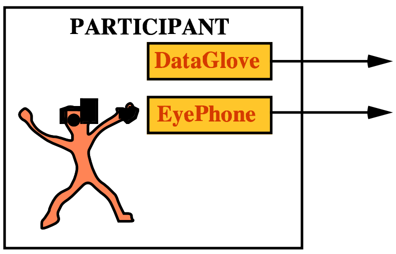



Group of continuously running processes
producing and consuming IPC messages



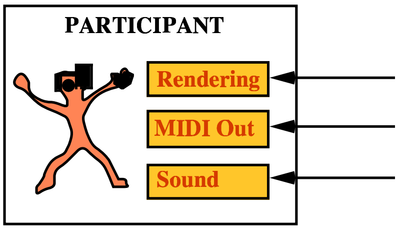


# Input Processes




- Encapsulation of input devices
  - DataGlove, SpaceBall, Head Tracker
- Filtering of device data
- Generation of event messages at specified rates

# Output Processes

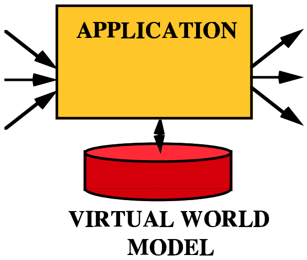



- Encapsulation of output devices
  - Rendering on graphics workstations, MIDI output, playback of prerecorded sound
- Output is triggered by messages from the application process



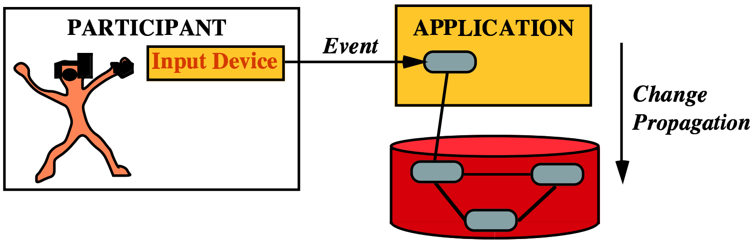


# Application Process




- Simulation of the evolution of the synthetic world
  - Immediate response to events from input processes
  - Must ensure world's model coherence
- Providing of interaction metaphors
- Generation of appropriate sensory feedback

# Interactive Behavior




- The virtual world model is updated in response to events coming from input processes
- A change propagation mechanism is responsible of obtaining a coherent evolution





# Change Propagation

## Expressiveness

- should allow the specification of general dependencies between objects
  - multi-way relationships
- should permit triggering of output operations

## Efficiency

- should ensure the responsiveness of the interface

# Change Propagation

## Constraint Imperative Programming

- Constraints are used to maintain relationships between objects
  - Declarative
- All computation is performed by constraints
  - Assignment is also a constraint
- The evolution of the model is obtained by adding or removing constraints
  - Imperative





# Dynamic Model

## Components

- Active Variables
  - store the state of the system
- Hierarchical Constraints
  - declaratively represent multi-way relationships between active variables (introduced in ThingLab II, 1987)
- Daemons
  - react to variable changes for imperatively sequencing between system states

# Active Variable

## Primitive element storing the system state

- Maintains its value
- Maintains a lists of dependents
- Keeps track of its state changes
- Can maintain the history of its past values

## Only constraints can modify a variable's value





# Hierarchical Constraint

## Specifies a multi-way relation between active variables

- Declarative part
  - the set of constrained variables
- Imperative part
  - the set of methods that could be used to enforce the constraint
- Priority
  - defines the order in which constraints need to be satisfied in case of conflict

# Daemon

Allows the execution of imperative code when a variable changes value

- Declarative part
  - the set of variables that trigger the daemon's execution
- Imperative part
  - the code that has to be executed when a variable change



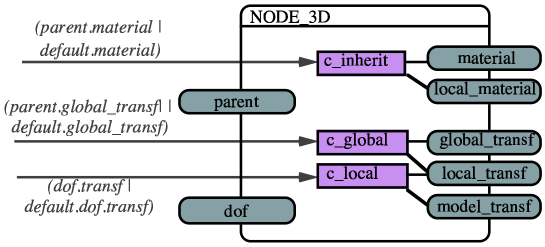


# Variable Path

Symbolic expression of an active variable's location as a function of other variables

- Example
  - Upper_global_transf:= (parent.global_transf or else Identity_transf)
- Allows the definition of indirect constraints and daemons

# Dynamic Model




- State: active variables
- Behavior: constraints





# State Manager

## Tasks

- Keep the constraint network up-to-date
- Trigger daemon execution
- Maintain indirect paths and variables' history

# State Manager

## Primitive Operations

- Activation and deactivation of constraints
  - Updating of the constraint graph
- Activation and deactivation of daemons
  - Registering of dependencies





# State Manager

## Constraint Satisfaction

- SkyBlue algorithm (Sannella, 1993)
  - Local propagation

    heuristic search of the best constraint graph
  - Method selection purely based on constraint priorities
  - Limited to acyclical constraint graphs
  - Separate planning and evaluation phases
- Lazy evaluation

# Model Evolution

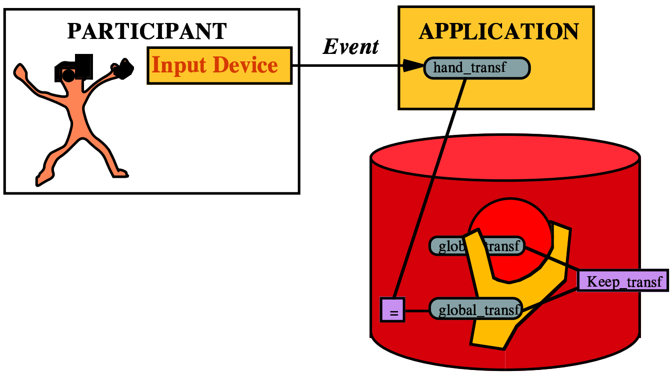
## Interaction

- Mapping between sensor measurements and actions in the virtual world
- Defined using constraints and daemons





# Direct Model Manipulation




- Interaction constraints relate sensors' active variables to variables in the model

# Adaptive Pattern Recognition

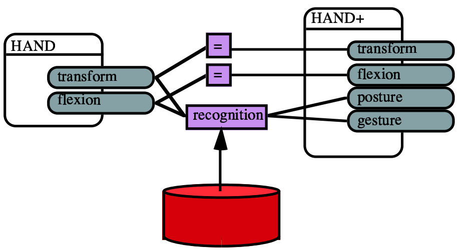
## Enhances data coming from sensors with classification information

- The mapping is learned from examples
- Increases device expressiveness
- The mapping can be adapted to preferences of the user





# Augmented Device Interface




- The adaptive pattern recognizer is a constraint

# Recognition Techniques

## Recognition steps

- Preprocessing
  - Filtering, accumulation
- Extraction of a feature vector
  - Features reduce variability within a class and enhance separation between classes
- Classification
  - Comparison with examples: parametric and non-parametric statistical classification, neural networks



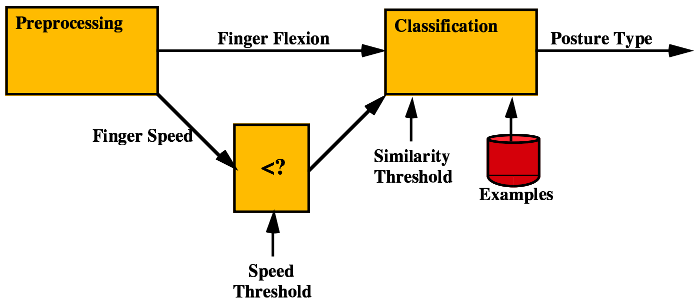


# Hand Gestures

## Hand posture and gesture recognition

- Important mean of non-verbal communication
- Allows the simultaneous specification of categorical and quantitative information

# Hand Posture Classification

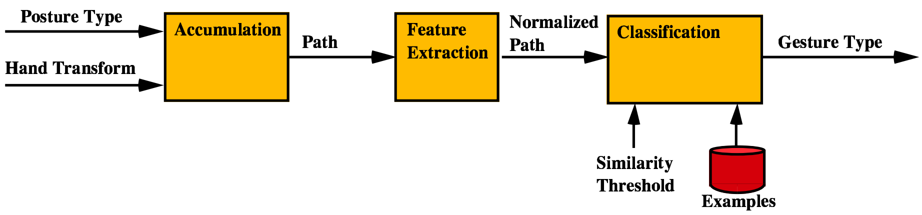



- A posture is a stable hand configuration



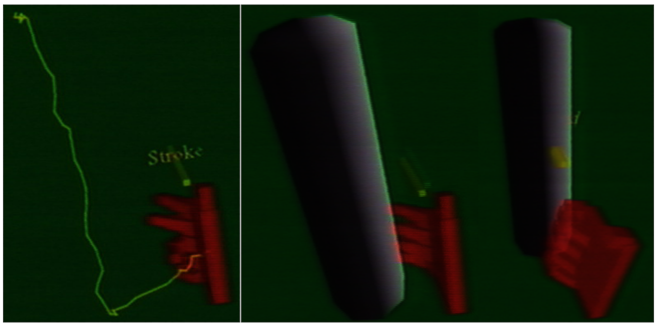


# Hand Gesture Classification




- A gesture is a path of the hand done while maintaining the fingers in the same posture

# Hand Gesture Examples



**Creating a cylinder by gestural input** **Grabbing the cylinder through posture recognition**



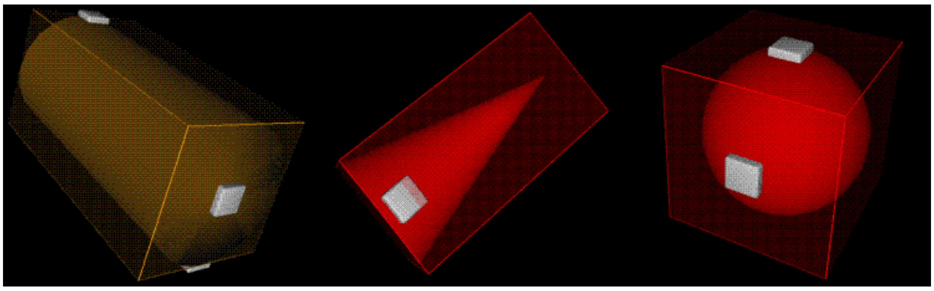


# Virtual Tools

## Motivation

- Gestural input and direct manipulation
  - Partial solutions to the interaction problem
  - Participant must know what can be manipulated and how to manipulate it
- Mediator objects
  - Help understand a model's behavior and interaction metaphors
  - Present a selective view of model's information
  - Offer the interaction metaphor to control it

# Virtual Tools



First class objects

Encapsulation of visual appearance and behavior





# Visual Appearance

Goals

- Provide information about the tool's behavior
- Offer visual semantic feedback during manipulation

Representation

- Articulated structure

# Behavior

## Goals

- Maintain consistency between visual appearance and manipulated information
- Allow information editing through a physical metaphor

## Representation

- Internal constraint network





# Virtual tools

## Multiple tools

- Manipulation of different parts of model's information
- Manipulation of same parts of model's information with different interaction metaphors

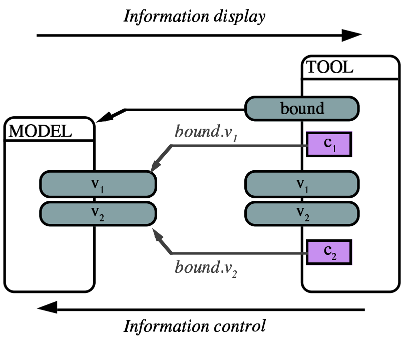
# Virtual Tool Protocol

- Binding
  - Bound active variable

    References the manipulated model
  - Binding constraints

    Multi-way relations

    Use indirect path to reference model's variables
- Second order control
  - ensure simultaneous activation/deactivation of binding constraints








# Manipulation

## Physical metaphor

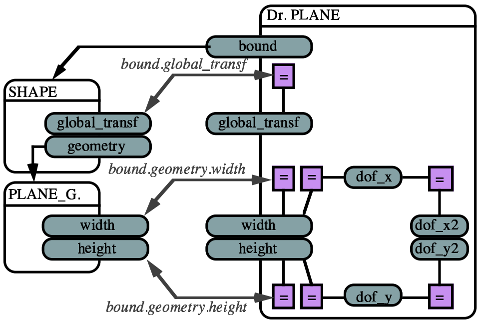
## Elementary manipulations

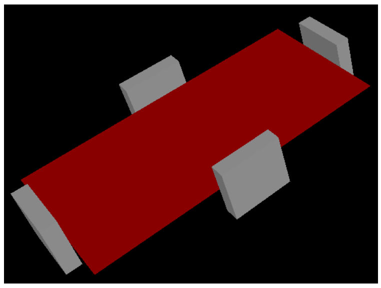
- Gestural input
  - initiate and terminate manipulation

    selection/deselection of tool's parts

    activation/deactivation of a motion control constraint
- Information transformation
  - Device sensor values propagate through the tool's constraint network
  - Participant's motion results in model's information changes

# Example of Tool: Dr. Plane







# Composite Tools

## Tool's composition

- Definition of more complex tools
  - reuse of abstraction
- Enforce interface consistency
  - rapid perception of possible actions

# Examples of Composite Tools






# VB2

## Implementation and results

- Object-Oriented, written in Eiffel, composed of more than 500 classes
- Runs on Silicon Graphics workstations
- Complex applications with thousands of variables and constraints can run at interactive speeds

# Back To The Future

"The ultimate display would, of course, be a room within which a computer can control the existence of matter. (...) With appropriate programming such a display could literally be the Wonderland into which Alice walked".

- Ivan Sutherland, The Ultimate Display, 1965.