OSGART: A Pragmatic Approach to MR

Julian Looser, Raphaël Grasset, Hartmut Seichter, Mark Billinghurst
Motivations

- Augmented Reality Applications

Need Tools, Utilities, SDK
Motivations

- Design an AR Application ([Ponder05]):
  - Start from ‘scratch’
  - Extend VR/Game Solution
  - AR Librairies
  - AR ToolKit
  - AR Framework
  - AR Authoring Tools

Programmer

Designer/Artist
Existing Solutions

- **Libraries/Toolkit**: MR-Platform, MXToolKit, ARToolKit, MXRToolKit, ImageTclAR, ARTag
- **Framework**: COTERIE, Studierstube, DWARF, AMIRE, ARTHUR, VHD++, Shared-Reality, Metaio
- **Authoring Tools**: APRIL, CMIL++, DART

Do we need a new tool?
Emerging Trends

- More spaces, more mixing (along Milgram’s Continuum)
- More tangible, more electronic smart devices
- More alternative collaborative system
Objective 1: Supporting New Issues

- No Real Dedicated and Flexible Solution
- Intrinsically supporting:
  - TANGIBILITY
  - TRANSITIONALITY
  - COLLABORATION AWARENESS
Objective 2: Supporting Different Capabilities

- From Programmer to Designer
- From Learning to Prototyping to Developing
- Be pragmatic: Multi-platform, Simple, Easy to use, All-in-one

ARToolKit?
ARToolKit

- Most used AR Library
- ~1000 download/month
- 100,000 google references
- Multiplatform, all-in-one, small package, wrapping other libraries, simple API, easy learning curve
- BUT limited rendering, lack robustness, alternative solutions (ARTPlus), etc.

⇒ Enhance it and Replicate the approach
**OSGART**

= *OpenSceneGraph* + *ARToolKit*

- ToolKit supporting easy development of AR/AV (MR), *MediatedR* Applications
- Multi-level programming interfaces
  - C++, Lua/Python, Built-in Authoring Tools
- Wrapping rather than redeveloping
- More features than OSGAR
- Base Layer for Objective 1
Features

- Functional Services
  - High Level Video Layer
  - Video Objects
  - Geometric/Spatial Registration
  - Photometric/Visual Registration

- Developer Services
  - C++/Scripting (RAD)/Simple Authoring Tools
  - Documentation/Samples/Tutorials
  - Utilities
  - Free for Academic, License for Industrial
OpenSceneGraph

- High Quality Rendering
- Import/Export Max/Maya + Utilities
- Community Plugins: OSGAL, ReplicantBody, osgVortex, etc ➔ Wrapping
Functionalities Overview..
Video Input

- **Multiple Video Support:**
  - Video Input: WebCam (USB, Firewire), PtGrey, DCAM, etc.
  - Video Files: AVI, Quicktime, etc.
  - Video Streaming: Gstream, RTSP

- **Integrating others video libraries:** ARVideo, VideoWrapper, VideoCapture, CVCam, etc.
  - Usual tool of developers
Video Objects

- Video Textures
- Video Shader
- Video Objects: VideoPlane, VideoBillboard, VideoLayers, etc.
AR Spatial Registration

- Integrated on the SceneGraph (*ARTTransform*)
- Multiple Markers, Multi-Marker support
- Tracker Plugins: ARToolKit, ARToolKit4, ARToolKit4+NFT, etc.
AR Visual Registration

- Photorealistic: Occlusion, Shadow
- Non Photorealistic: on video, on the content, on both (stylized AR)
Developer Overview..
ARToolKit Examples

- simpleVideo, simpleTest, multi, simpleShadow
- Collide, PaddleInteraction
Scripting Coverage

- Hand tuned SWIG wrapper scripts
- Wrapper interfaces use unmodified OpenSceneGraph header file
- Platforms: Windows, Linux, Unix (incl. Mac OS X)
- Simple integration into wxPython, Ruby GTK+ etc.
Ruby Version

```ruby
#!/usr/bin/env ruby
#
# Demonstration for Ruby
#
# extend the load path for vorg
LOAD_PATH << '../bin/ruby'
#
# include all necessary libs
require 'vorg'
require 'vorgDB'
require 'vorgProducer'

class Viewer
  def initialize()
    puts "Open a window ..."
    @viewer = VorgProducer::Viewer.new
    @viewer.setUpViewer(VorgProducer::Viewer::STANDARD_SETTINGS)
    puts "Reading Data ..."
    n = VorgDB::readNodeFile("cow.org")
    root = Vorg::Group.new()
    root.addChild(n)
    @viewer.setSceneBackup(root)
    puts "Show the Window ..."
    @viewer.realize()
  end
end
```

Same on Linux
def loadMarker(self, root, num, filename):
    # get the marker transform
    markertrans = self.armTrackerTransform(num)
    scaler = self.armAutoTransform()  
    scaler.setScale([0.0])              
    markertrans.addChild(scaler)        
    # add to root transform
    root.addChild(markertrans)          
    # get the marker
    marker = markertrans.getMarker()    
    # get the marker active
    marker.setActive(True)              
    # add a scene
    scaler.addChild(osgDB.readNodeFile(filename))  
    # set the bin
    markerTransNode.getUserData().setRenderBinDetails(5, "Render")
And the Upper Level..
Manipulation: Tangible

- Spatial+Physical Input: Prototyping Interface
  - ARToolKit+homemade microcontroller
  - Large choices of sensor/actuators
  - (switch, pressure, LEDs, Motor)
Navigation: Transitionality

- Transitional Framework:
  - Multispace support: VR, AR, AV
  - Multiscale, Multiviewpoint, Multirepresentation support
  - Navigation: Transitional Interface

- Transition Awareness
Collaboration Awareness

- **Awareness**: Head/Hand Position, Gaze Direction, Telepointers, etc.
- **Dedicated Objects** (VideoAvatar, GazeAwar..)
- **Approach**: ICE (Middleware)
Results
First Results

- Integration with OSG plugins
First Results

- Animalia (CONVERGE 05)
First Results

- Experimenta Vanishing Project (Australia)
MagicLens Research
TankWar
First Results

- Industrial Prototypes
  - NZ Army
  - Airways
  - W&M
- Commercial Project (MagicBook)
Contact, Information

- Website: http://www.hitlabnz.org
- Distribution, Availability:
  - Mark Billinghurst
  - Philip Lamb (Licensing)
- Development:
  - Julian Looser
  - Hartmut Seichter
  - Raphael Grasset
- Pre-Alpha Release: December 2006