Mixed Augmented Reality Functions Extension in X3D

Gun Lee

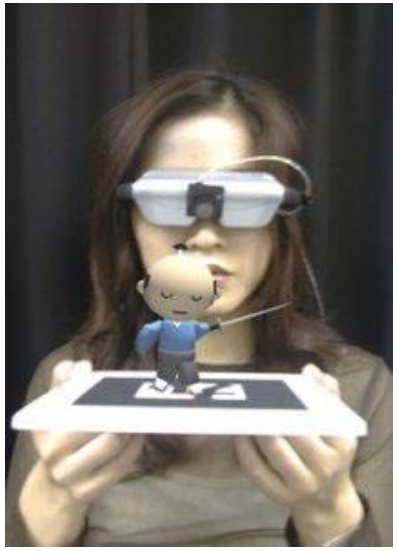
August 2014

MAR WG, Web3D Consortium
ar_chairs@web3d.org

Augmented Reality

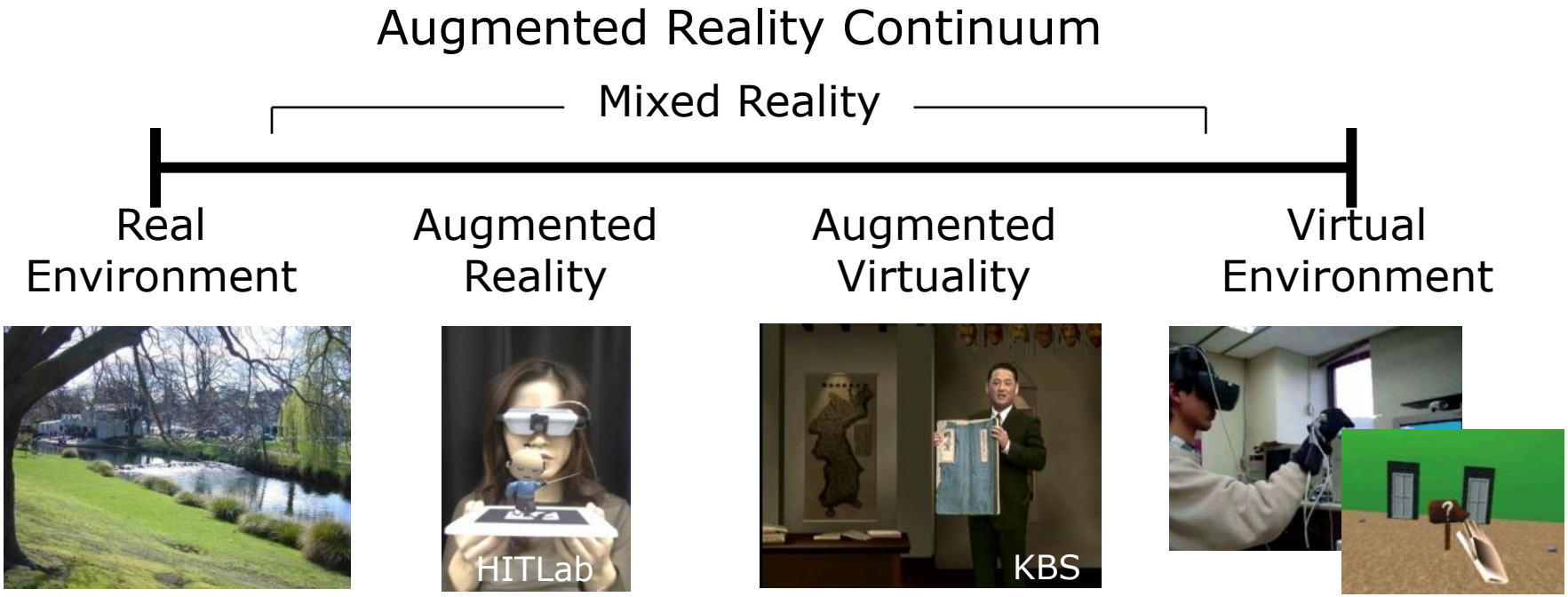
- What is AR (Augmented Reality) ?
  - "Augmented Reality (AR) is a field of computer research which deals with the combination of real-world and computer-generated data." – wikipedia.org
- Key Features of AR [R. Azuma 97]
  - Combines real and virtual images
  - Interactive in Real-Time
  - Registered in 3D Real World

STAR System
[HRL Laboratories, 1998]



ARToolkit
[HITLab, Univ. of Washington, 1999]

Mixed Reality Continuum



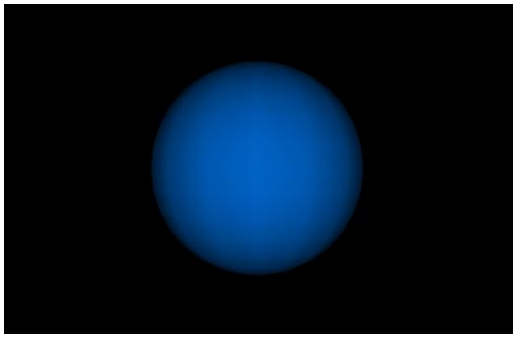
[Paul Milgram's Reality-Virtuality Continuum (1994)]

X3D

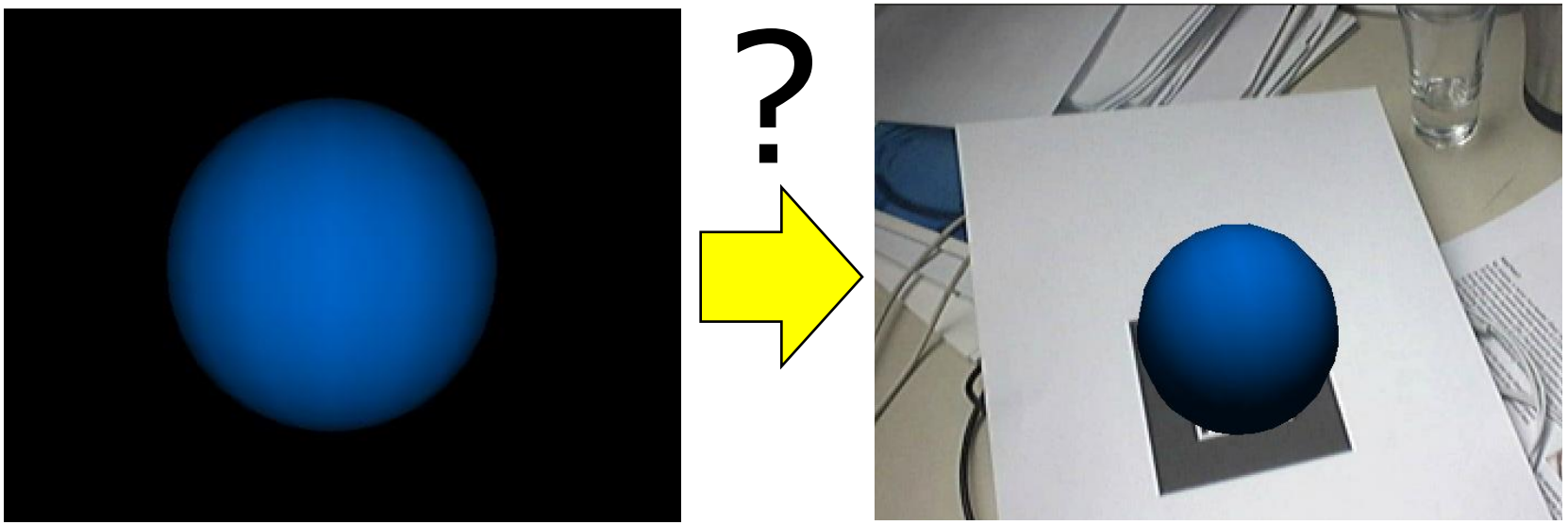
- Extensible 3D graphics
- ISO Certified Standard
- Royalty free open standard
- Developed by Web 3D Consortium – www.web3d.org
- Originated from VRML, now in XML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN"
"http://www.web3d.org/specifications/x3d-3.0.dtd">

<X3D version='3.0' profile='Interchange'>
<Scene>
    <Transform translation='-2.4 0.2 1.0' rotation='0.0 0.707 0.707 0.9'>
      <Shape>
        <Sphere radius=`10'/>
        <Appearance>
          <Material diffuseColor='0.0 0.5 1.0'/>
        </Appearance>
      </Shape>
    </Transform>
</Scene>
</X3D>
```



Extending X3D to be AR/MR capable!



MAR WG in Web3D Consortium

- Started up and running since June 2011
- Co-chairs:
  Gun Lee (HIT Lab NZ), Timo Engelke (Fraunhofer IGD)

- Final goal is to produce a unified proposal to extend X3D specification for AR and MR applications

- Tasks
  - Requirement and use cases – August 17, 2011 - Done
  - Comparison of X3D AR proposals - Final editing March, 2012 - Done
  - Developing plan to merge proposals – May, 2012 – Done
  - Non-functional requirements – June, 2012 - Done
  - Develop unified proposal – In progress
    - Unified proposal writing, public and internal reviews – 2014 - Done
    - Implementation based on the proposal – In progress
    - Revise proposal based on implementation results

http://www.web3d.org/x3d/wiki/index.php/X3D_and_Augmented_Reality

Compare & merge proposals

- Comparison of X3D AR proposals from Korea Chapter and Fraunhofer IGD
  - Trade off between simplicity and flexibility/extensibility
    - KC proposes on higher level AR/MR focused features, while Instant Reality has lower-level multi-purpose nodes

- Based on this comparison, MAR WG worked on merging the X3D AR proposals.

http://www.web3d.org/x3d/wiki/index.php/Comparison_of_X3D_AR_Proposals

Plan for Merging Proposals

1. Discuss general strategy/policy/guidelines
   - Revise non-functional requirement
2. Produce a merged proposal for each functional components
   - Investigate each functional features stepwise:
     - Camera video stream image into the scene (texture and background)
     - Tracking (including support for general tracking devices)
     - Camera calibration (viewpoints)
     - Others (color-keying, depth occlusion)
3. Check Integrity of the merged proposal
   - Resolve conflicts between individual functional components
   - Merge overlapping features
   - Public Review (March-Aug, 2013)
     - http://www.web3d.org/wiki/index.php?title=AR_Proposal_Public_Review
   - Revision and finalize internal review
4. Specification writing and development
   - Implementation
5. Review

Aug. 2012

Aug. 2013

Aug. 2014

MAR WG Meetings and Activities in 2013

- January
  - Teleconference meeting
  - Seoul Meeting (Jan. 29th)
- February
  - Teleconference meeting
- March
  - Merged Proposal Public review (~ August)
- August
  - Sydney SC24 Plenary & WG meetings
- October
  - Teleconference meeting resumed
    with new member from Fraunhofer IGD on board
- November
  - Teleconference meeting
- December
  - Teleconference meeting

MAR WG Meetings and Activities in 2014

- January
  - Seoul Meeting (Jan 20th) + Teleconference meeting
  - Timo Engelke, Fraunhofer IGD as a new Co-Chair of MAR WG
- February
  - Teleconference meeting – Merged proposal review
- March
  - Teleconference meeting – Merged proposal review
- April
  - Teleconference meeting – Finalized reviewing the merged proposal
- May
  - Teleconference meeting – Implementation plan

- Implementation – in progress
  - X3DOM based implementation (Gun Lee)
  - Instant AR or another X3D platform from Fraunhofer IGD (Timo Engelke)

Summary

- Web3D MAR WG AR proposal (technical details in appendix)
  - CalibratedCameraSensor and TrackingSensor nodes
    - Device independent on end-user side
    - Delegating specific device setup to browser/user
  - Reusing PixelTexture node
  - Backdrop node for background
    - Independent from viewpoint orientation
    - Naming following Fraunhofer's proposal
  - Minimal extension to Viewpoint node
    - Subset of Fraunhofer's proposal
    - Camera calibration information to come from sensor nodes
- Merged proposal finalized with reviewing process (April 2014)
- Implementation based on the proposal in progress
- Next Steps
  - Finish implementation & develop specification by late 2014
  - X3D 3.4

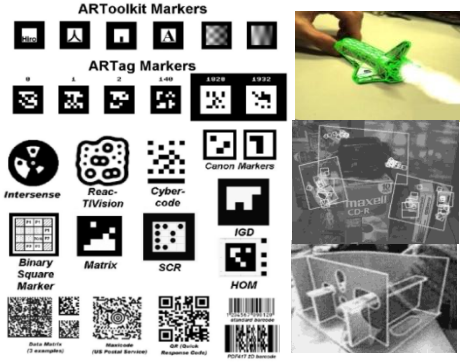
Thank you!

- http://www.web3d.org/wiki/index.php/X3D_and_Augmented_Reality
- AR WG Contact
  - Co-Chairs: Gun Lee, Timo Engelke
  - e-mail: ar_chairs@web3d.org
- Appendix
  - Technical Overview of the MAR WG Proposal

Appendix:
Overview of the MAR WG Proposal

Common fundamental features

- Camera Sensor
  - Providing video stream into X3D from physical camera
- Tracking Sensors
  - Position and Orientation of physical objects (relative to the camera under AR context)
  - Too many technologies in use
    - Magnetic, ultrasound, computer vision, etc.



- Camera Image/Video Background and Texture
  - Showing static (view registered) background that doesn't update according to Viewpoint orientation.
- Calibrated view frustum
  - Detailed control of projection parameters in Viewpoint

Camera and Tracking Sensors

- Browser and device independence
  - X3D scene writer has no knowledge on what browser or tracking device/sensor/algorithm would the end user be using.
  - The scene writer should be able to describe intention of use (or purpose) of the sensor to provide hint to the browser/user to which actual sensor the node should match
    - Head tracking, pointing device, user facing camera, world facing camera, etc.
  - The browser (end-user) maps sensor nodes to appropriate hardware/software sensors
    - Choose one based on the purpose/hint description
    - Users can setup the preference on the browser to map specific purposed nodes to sensors/trackers available on the user's hardware/software setup

Camera and Tracking Sensors

```
CalibratedCameraSensor : X3DSensorNode {
        SFBool   [in,out]     enabled       TRUE
        SFNode   [in,out]     metadata      NULL [X3DMetadataObject]
        SFBool   [out]        isActive

        SFString [in,out]     description   ""
        SFImage  [out]        image
        SFVec2f  [out]        focalPoint
        SFFloat  [out]        fieldOfView
        SFString [out]        fovMode
        SFFloat  [out]        aspectRatio
}
```

- Values for the *description* field
  - USER_FACING A camera facing towards the user.
  - WORLD_FACING A camera facing towards the user's view direction.

Camera and Tracking Sensors

```
TrackingSensor : X3DSensorNode {
        SFBool   [in,out]     enabled  TRUE
        SFNode   [in,out]     metadata NULL [X3DMetadataObject]
        SFBool   [out]        isActive

        SFString [in,out]     description      ""
        SFVec3f  [out]        position
        SFRotation [out]      rotation
        SFBool   [out]        isPositionAvailable      FALSE
        SFBool   [out]        isRotationAvailable      FALSE
}
```

- Values for the *description* field
  - VIEWPOINT_FROM_WORLD (e.g., 3D Visualization, CAVE)
  - OBJECT_FROM_WORLD (e.g. Interaction)
  - OBJECT_FROM_VIEWPOINT (e.g. CV based AR)

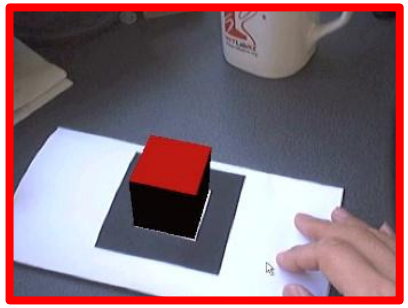
Camera Image in Texture

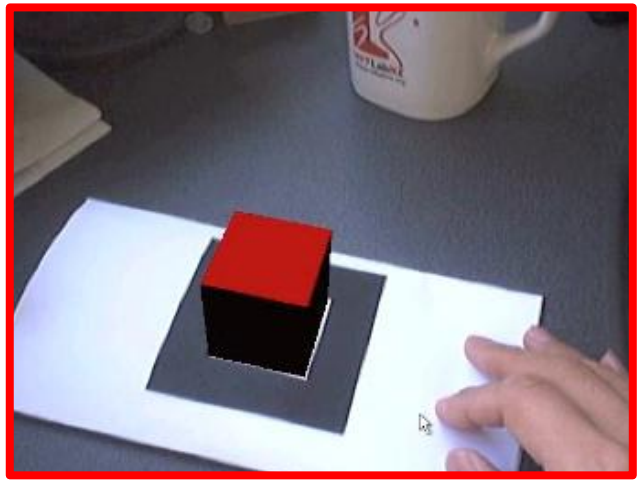
- Reuse PixelTexture node
  - Route image field from CalibratedCameraSensor

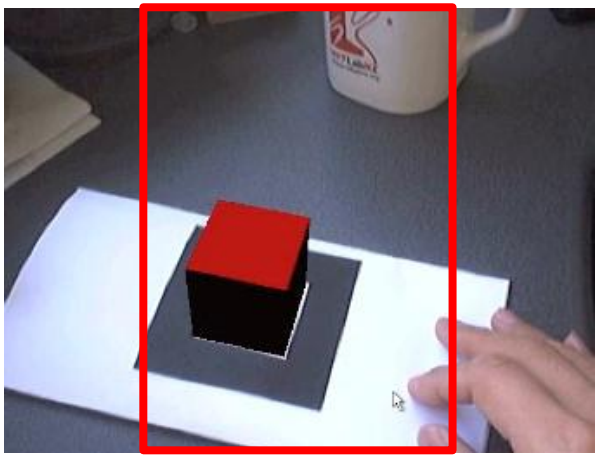
```
...
<CalibratedCameraSensor DEF=”camera” />
...
<PixelTexture DEF=”tex” />
...
<ROUTE fromNode='camera' fromField='image'
toNode='tex' toField='image'/>
...
```

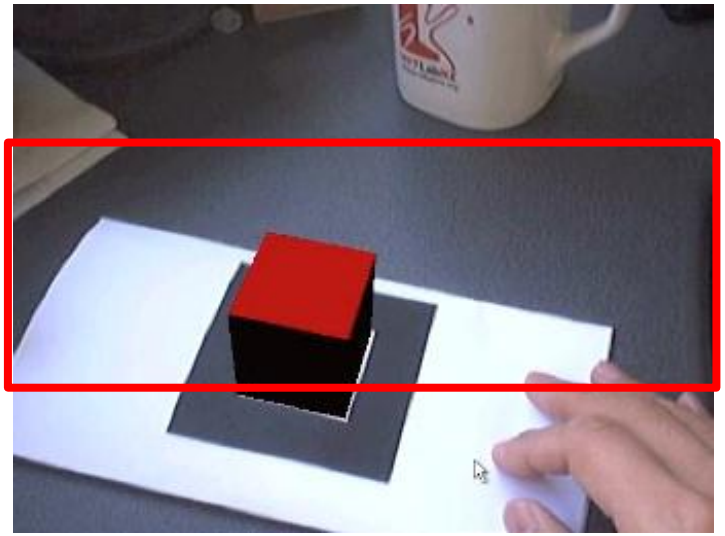
AR scene adaptive to viewport aspect ratio

* Viewport in the X3D browser depends on the screen size and browser window
(X3D scene writer/developer cannot know user's viewport size and aspect ratio)









Camera Image in Background

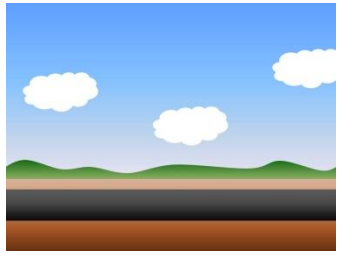
- Backdrop (new node)
  - Static background that doesn't changes to viewpoint orientation

```
BackdropBackground: X3DBackgroundNode {
        SFColor  [in,out]  color
        MFString [in,out] url
}

ImageBackdropBackground: X3DBackgroundNode {
        SFColor  [in,out]  color
        SFImage[in,out]  image
}
```

```
<CalibratedCameraSensor DEF="camera" />
...
<ImageBackdropBackground DEF="bg" />
...
<ROUTE fromNode='camera' fromField='image'
        toNode=bg toField='image'/>
```

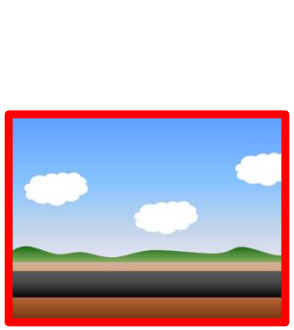
Behavior of ImageBackdropBackground

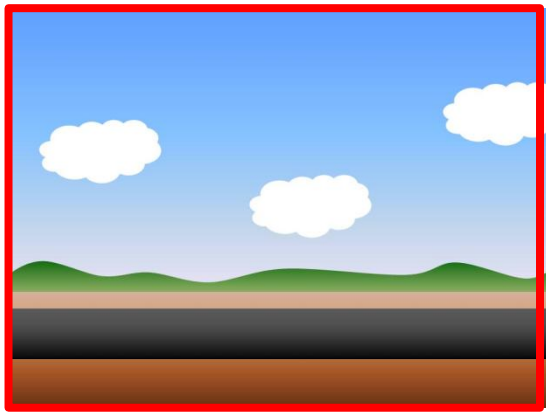


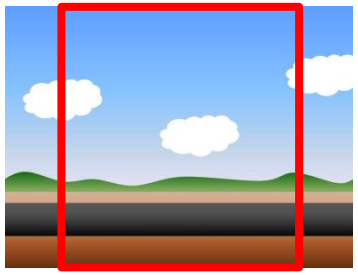
Backdrop image

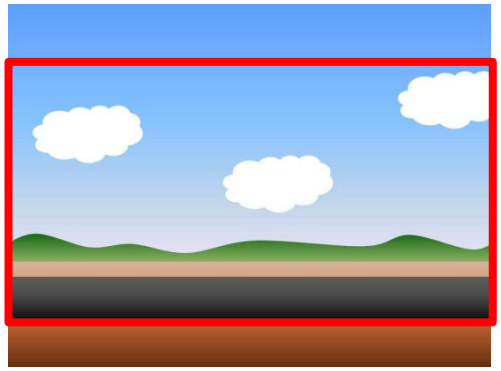
Viewport in the X3D browser
(depends on the screen size and browser window)

The backdrop image is automatically scaled and centered
(while retaining the aspect ratio) to fit either width or height,
so that there is no empty region on the viewport.









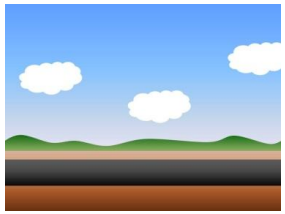
* Could add a enum field to define different filling/fitting behaviors if needed.

Calibrated View Frustum (Viewpoint Node)

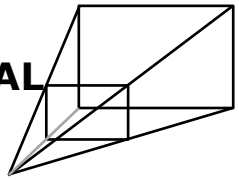
- Extend Viewpoint node with additional fields
  - Minimal addition to current spec
  - fieldOfView, fovMode, aspectRatio to be routed from CalibratedCamera sensor node

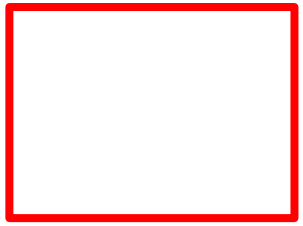
```
Viewpoint: X3DViewpointNode {
  SFVec3f    [in,out] centerOfRotation
  SFFloat    [in,out] fieldOfView
  SFRotation [in,out] orientation
  SFVec3f    [in,out] position
  SFString   [in,out] fovMode  MINIMUM (VERTICAL, HORIZONTAL, DIAGONAL)
  SFFloat    [in,out] aspectRatio
  SFVec2f    [in,out] focalPoint
}
```

Behavior of the Viewpoint node



fieldOfView: 60
fovMode: HORIZONTAL
aspectRatio: 0.75
focalPoint: 0,0

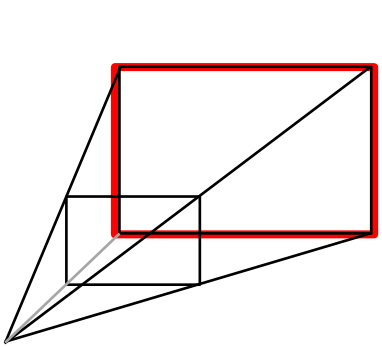


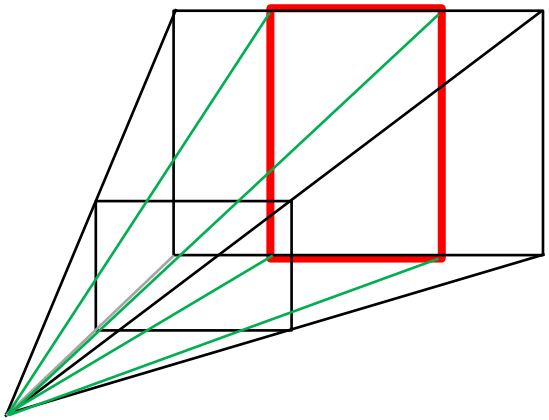


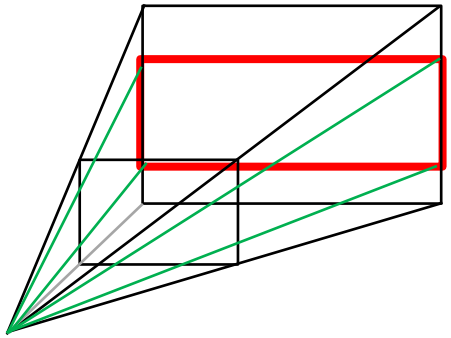
Internal parameter routed from CalibratedCameraSensor to Viewpoint node

Viewport in the X3D browser (depends on the screen size and browser window)

The Viewpoint node will generate view frustum (projection matrix) adaptively based on the difference between the expected aspect ratio of the viewpoint (streamed from CalibratedCameraSensor) and the aspect ratio of the viewport.

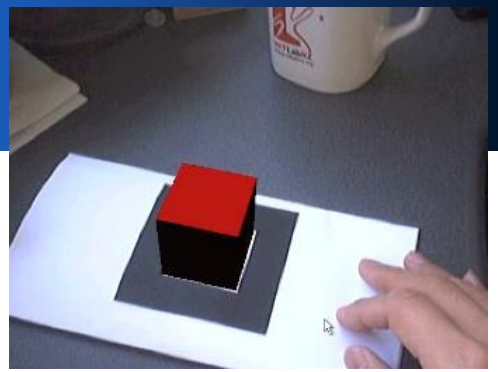






* Black represents original view frustum, green is adjusted view frustum.

Example AR scene



```
...
<CalibratedCameraSensor DEF=”camera” />

<ImageBackdropBackground DEF=”bg” />
<ROUTE fromNode=”camera” fromField=”value” toNode=”bg” toField=”image”/>

<Viewpoint DEF=”arview” position=”0 0 0” />
<ROUTE fromNode=”camera” fromField=”fieldOfView” toNode=”arview” toField=”fieldOfView”/>
<ROUTE fromNode=”camera” fromField=”fovMode” toNode=”arview” toField=”fovMode”/>
<ROUTE fromNode=”camera” fromField=”aspectRatio” toNode=”arview” toField=”aspectRatio”/>


<TrackingSensor DEF=”tracker1” description=”OBJECT_FROM_VIEWPOINT” />

<Transform DEF=”tracked_object”>
        <Shape>
                <Appearance><Material diffuseColor="1 0 0" /></Appearance>
                <Box />
        </Shape>
</Transform>

<ROUTE fromNode=”tracker1” fromField=”position” toNode=”tracked_object” toField=”position”/>
<ROUTE fromNode=”tracker1” fromField=”rotation” toNode=”tracked_object” toField=”rotation”/>
…
```