ISO/IEC JTC 1/SC 24
Computer graphics, image processing and environmental data representation
Secretariat: BSI (United Kingdom)

Document type: Meeting Report

Title: N 3873-WG9-2016-MAR LAE InfoModel 20160822 KYoo

Status:

Date of document: 2016-09-05

Expected action: INFO

No. of pages: 31

Email of secretary: charles.whitlock@bsigroup.com

Committee URL: http://isotc.iso.org/livelink/livelink/open/jtc1sc24

Information Model for Live Actor and Entity in MAR

Kwan-Hee Yoo

ISO/IEC JTC 1 SC24 WG9 Meeting

2016. 8. 22

Chungbuk National University

Augmented Reality Continuum

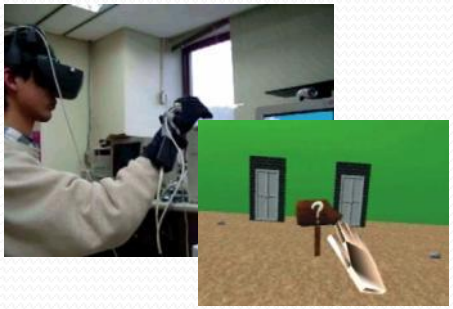
Mixed Reality

Real Environment

Augmented Reality

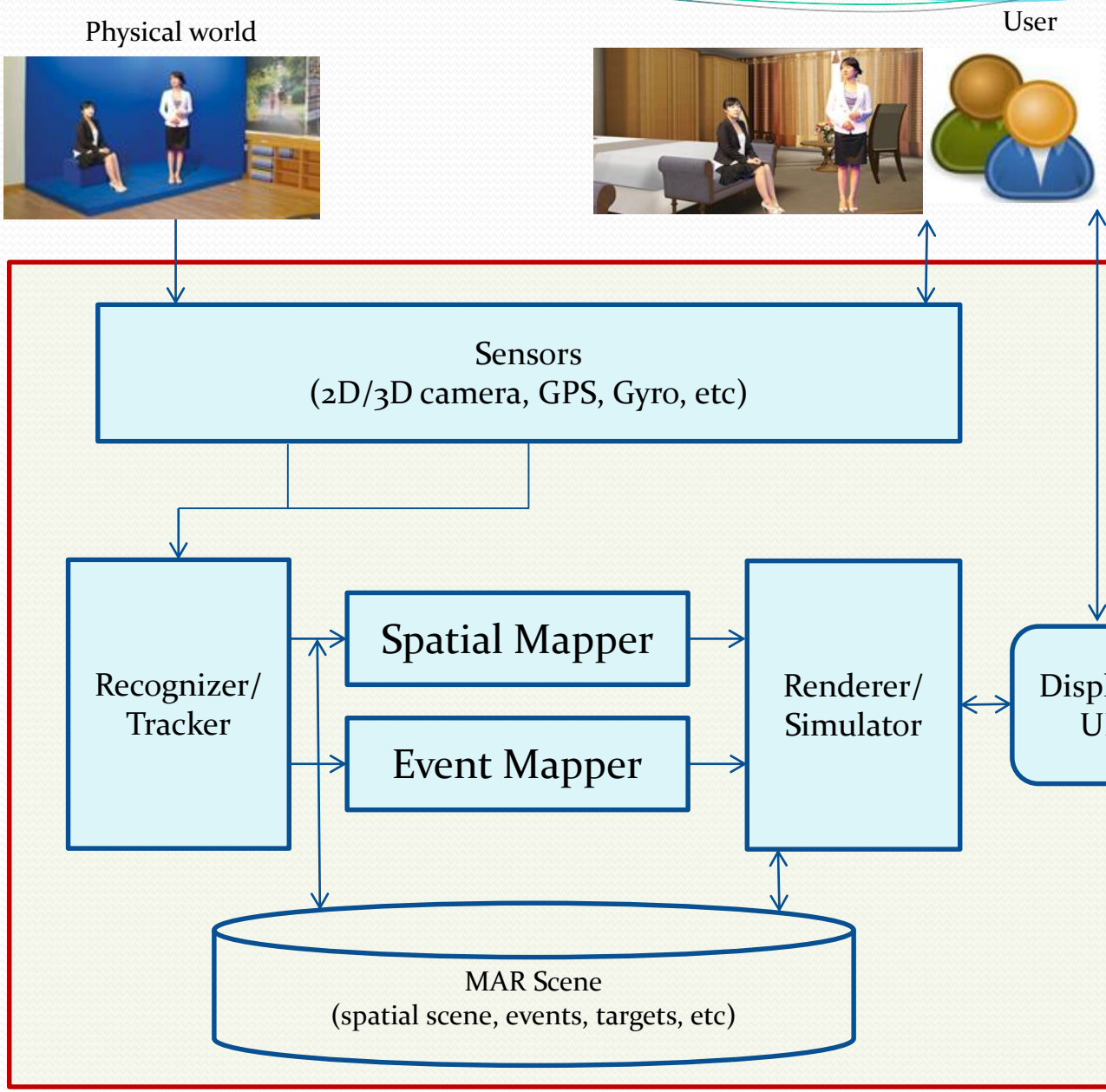
Augmented Virtuality

Virtual Environment



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

System Framework for representing LAEs in a MAR world



Live Actor and Entity in a MAR world

In a Real World

In a MAR World

+3D virtual world

+2D virtual world

-Sense LAEs and their pure sensing information in a real world

-Need to model 2D/3D objects for representing LAEs

-Spatial Mapping of LAEs into a MAR scene

-Event Mapping between LAEs and a MAR scene

-Display mixed information

Characteristics of Live Actor and Entity in a MAR world

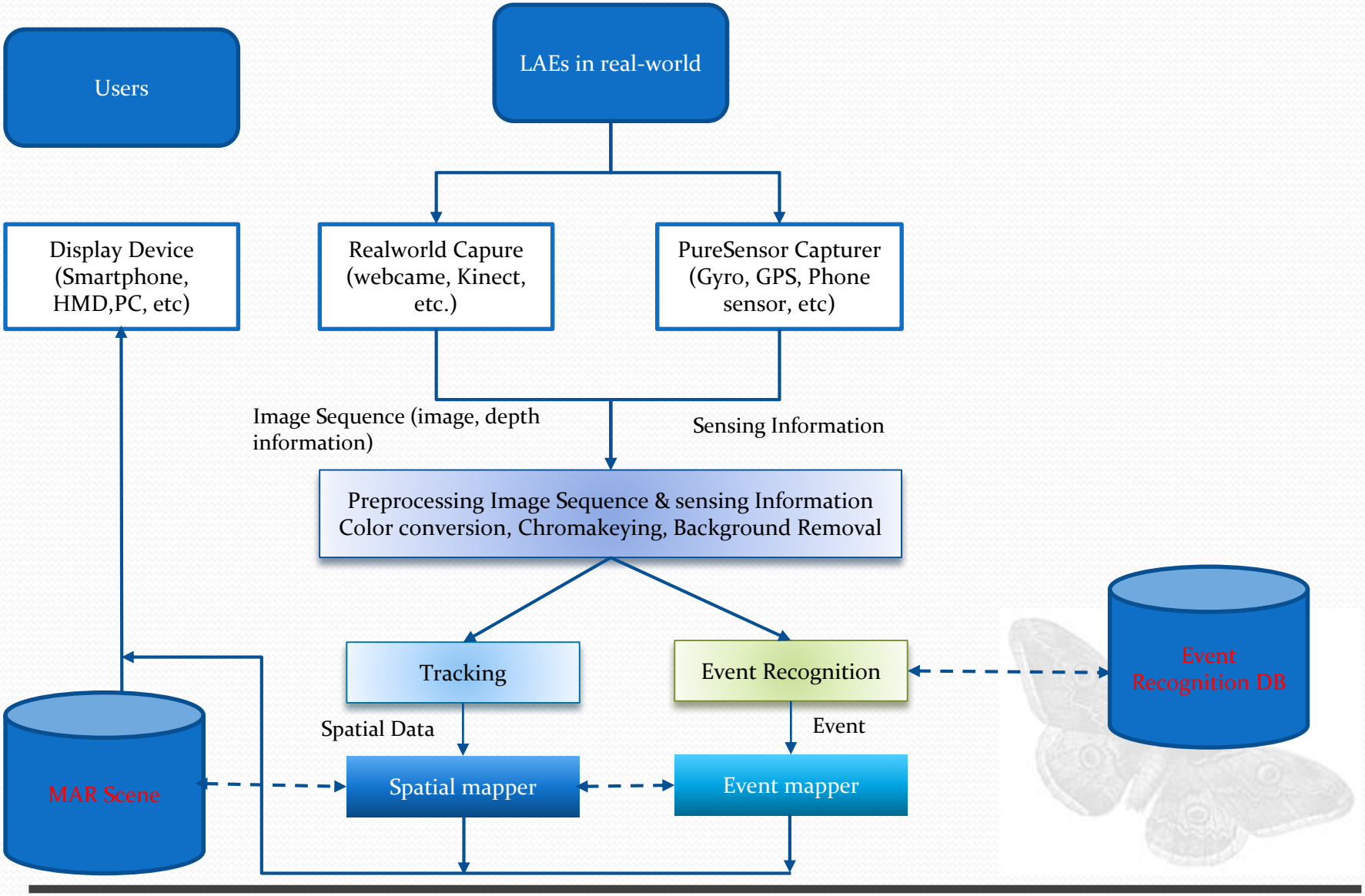
-Sensing LAEs
-Tracking LAEs
-Recognizing LAEs
-Spatial Mapping of LAEs into MAR world
-Event Mapping between LAEs and MAR world

Object Model for the live actor and entity
File Format for representing the live actor and entity

2015 London Meeting

Information Model for the live actor and entity

Module Configuration



Information Model for Live Actor and Entity

-define MAR scene
(virtual world – image, 3D virtual world)
(LAE model – 2D object, 3D object, digital hologram)
(event scene (mapping) information)

:a virtual world such as an image or X3D file

:object (2D, 3D, hologram) representation for modeling LAEs

-Sensor (real world capturer, pure sensor)
(real world capturer– camera, etc)
(pure sensor– HMD device, etc)

-Spatial Mapper

-define Movable Volume of Live Actors and Entities in a real world

-define Movable Volume of Live Actors and Entities in a virtual world

-calibrate spatial position to embed LAE in a MAR scene

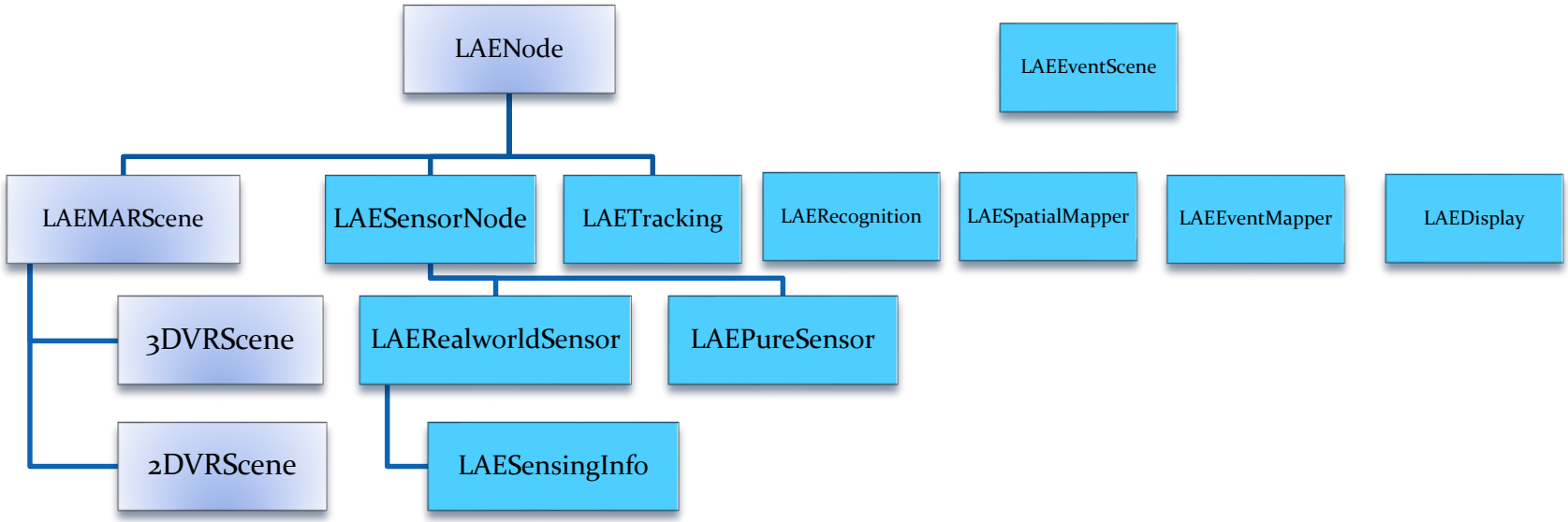
-Event Mapper

Information Model for LAE in MAR

Dimension	Types	
	Input	**Output**
Sensor	Real world Signal	Sensor data related to representation of live actor and entity
Recognizer	Raw or processed signals representing the LAE (provided by sensors) and target object specification data (reference target to be recognized).	At least one event acknowledging the recognition.
Tracker	Sensing data related to representation of live actor and entity.	Instantaneous values of the characteristics (pose, orientation, volume, etc.) of the recognized target signals.
Spatial mapper	Sensor identifier and sensed spatial information.	Calibrated spatial information for the given MAR scene.
Event mapper	Event identifier and event information.	Translated event identifier for the given MAR scene.
Renderer	MAR scene graph data	Synchronized rendering output (e.g., visual frame, stereo sound signal, motor commands, etc.).

Nodes for LAE

LAENode



Information Model for MAR Scene

```
LAEMARScene {
    String                  id
    String                  description
    Enum                    type
    MFString [in,out]       url         “”
}

BackdropBackground  {
            MFString [in,out]   url         “”
            SFColor    [in,out] color       (0,0,0,0)
            SFString [in,out] scaleType “FILL”
}
```

Information Model for Real world Capturer

Dimension	Types	
	Input	Output
Visual	WebCam RGB Camera	Image, Video, frame name, FOV Camera Information
	Depth Camera (KINECT)	Depth Image, Skeleton, Video, FOV Camera Information
	360º Camera	360º Image /Video
	Stereo Camera	Stereo Image /Video
Auditory	Microphones	Audio Information (audio signal, audio file…)

Information Model for Pure Sensor

Dimension	Types	
	Input	Output
GPS sensor	Location, geostationary satellite information	Electro magnetic waves, Coordinate value, Latitude, longitude, height
Gyro sensor	Rotation motion, changes of orientation	Electro magnetic waves, Coordinate value, Moving directions
Phone sensor	Accelerometer, Gyroscope, Magnetometer, Barometer, proximity, light sensor, touch sensor	Electro magnetic waves, Coordinate value, Camera Touch, Acceleration, Access, Motion
Wii Remote/Joystick	User input button function	Electro magnetic waves, Coordinate value, distance depth, force
HMD(Head Mounted Display)	Head tracking, Position tracking, user input button	Electro magnetic waves, Coordinate value, Motion Gyro, Access

Sensor Nodes for LAE

LAESensorNode

LAESensorNode is the abstract base type for all LAESensor interfaces.

Type	accessType	Name	Default	Range	Profile	Description
SFBool	InputOuput	enabled	true		Interactive	Specifics whether this sensor is enabled. A disabled sensor doesn't produce any output.
SFNode	inputOutput	metadata	MetadataObject		Interactive	Field to add metadata information.

Real World Sensor Nodes for LAE

RealworldSensor

Type	accessType	Name	Default	Range	Profile	Description
SFBool	InputOuput	enabled	true		Interactive	Specifics whether this sensor is enabled. A disabled sensor doesn't produce any output.
SFNode	inputOutput	metadata	X3DMetadata Object		Interactive	Field to add metadata information.
SFBool	Output	isActive			Interactive	When all preconditions for a given sensor are met.
SFString	inputOutput	decription	""		Interactive	Field to add description for node's work.
SFString	input	sensorType	"visual"		Interactive	Define type of sensor like visual or audio.
SFString	input	senseData	"image"		Interactive	Define sense data type like image, skeleton or depth image.
SFString	input	sensingDevice	"kinect"		Interactive	Device for capture real actor by using Kinect or general camera.

RealworldSensor

```
RealworldSensor  : LAESensorNode {
    SFBool          [in,out]   enabled              TRUE
    SFNode          [in,out]   metadata         NULL [X3DMetadataObject]
    SFBool          [out]      isActive
    SFString     [in,out]   description          “”
    SFString        [in,out]   sensorType         “visual”
    SFString        [in,out]   senseData          “image”
    SFString        [in,out]   sensingDevice      “kinect”
}
```

RealworldSensor

```
x3dom.registerNodeType(
    "LAESensor",
    "X3DLAESensorNode",
    defineClass(x3dom.nodeTypes.X3DNode,
        function (ctx) {
            x3dom.nodeTypes.LAESensor.superClass.call(this, ctx);

            /**
                Enabled field specifics whether this sensor is enabled. A disabled sensor doesn't produce any output.
            */
            this.addField_SFBool(ctx, 'enabled', true);

            /**
                isActive: When all preconditions for a given sensor are met.
            */
            this.addField_SFBool(ctx, 'isActive', true);

            /**
                description: Describe about the node's work
            */
            this.addField_SFString(ctx, 'description', '');

            /**
                sensorType: Define type of sensor like visual or audio.
            */
            this.addField_SFString(ctx, 'sensorType', 'visual');

            /**
                senseData: Define sense data type like image, skeleton or depth image.
            */
            this.addField_SFString(ctx, 'senseData', 'image');
            /**
                senseData: Device for capture real actor by using Kinect or general camera.
            */
            this.addField_SFString(ctx, 'sensingDevice', 'kinect');
        }, {
            nodeChanged: function() {
                x3dom.debug.logInfo('LAESensor created');
                }
            }
    )
);
```

RealworldSensor

```
{
    processImageData : function(imageBuffer, width, height){
        if (colorRenderer.isProcessing || (width <= 0) || (height <= 0)) {
        return;
    }

    colorRenderer.isProcessing = true;
    colorRenderer.processImageData(imageBuffer, width, height);
    colorRenderer.isProcessing = false;
    }
},
{
    getClosestBodyIndex : function(bodies){
        var closestZ = Number.MAX_VALUE;
        var closestBodyIndex = -1;
        for(var i = 0; i < bodies.length; i++) {
            if(bodies[i].tracked && bodies[i].joints[Kinect2.JointType.spineMid].cameraZ < closestZ) {
                closestZ = bodies[i].joints[Kinect2.JointType.spineMid].cameraZ;
                closestBodyIndex = i;
            }
        }
        return closestBodyIndex;
    }
},
{
    multiSourceFrame : function(frame){
        var closestBodyIndex = getClosestBodyIndex(frame.body.bodies);
        if(closestBodyIndex !== trackedBodyIndex) {
            if(closestBodyIndex > -1) {
                kinect.trackPixelsForBodyIndices([closestBodyIndex]);
            } else {
                kinect.trackPixelsForBodyIndices(false);
                //clear canvas
                //processImageData(emptyPixels.buffer, colorCanvas.width, colorCanvas.height);
                processImageData(emptyPixels.buffer, 1920, 1080);
                colorCanvas.parentNode._x3domNode.invalidateGLObject();
            }
        }
        else {
            if(closestBodyIndex > -1) {
                if(frame.bodyIndexColor.bodies[closestBodyIndex].buffer) {
                    //processImageData(frame.bodyIndexColor.bodies[closestBodyIndex].buffer, colorCanvas.width, colorCanvas.height);
                    processImageData(frame.bodyIndexColor.bodies[closestBodyIndex].buffer, 1920, 1080);
                    colorCanvas.parentNode._x3domNode.invalidateGLObject();
                }
            }
        }
```

X3D Nodes related with LAE

LAESensingInfo

Type	accessType	Name	Default	Range	Profile	Description
SFBool	InputOuput	enabled	true		Interactive	Specifics whether this sensor is enabled. A disabled sensor doesn't produce any output.
SFNode	inputOutput	metadata	X3DMetadata Object		Interactive	Field to add metadata information.
SFString	inputOutput	decription	""		Interactive	Field to add description for node's work.
SFImage	outputOnly	image			Interactive	Output the image data captured by KinectCamera
SFFloat	outputOnly	fieldOfView			Interactive	File of view information
SFFloat	outputOnly	trackPointInfo			Interactive	Tracking point information
MFString	outputOnly	audioInfo			Interactive	Display the url of audio info
SFVec2f	outputOnly	position			Interactive	The position of real actor

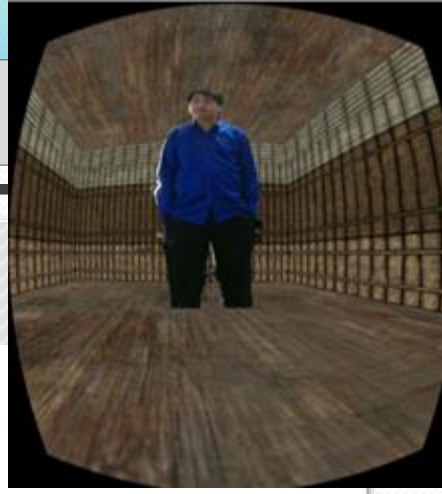
X3D Nodes related with LAE

LAESensingInfo

```
LAESensingInfo  : X3DLAESensorNode {
    SFBool      [in,out]   enabled          TRUE
    SFNode      [in,out]   metadata         NULL [X3DMetadataObject]
    SFBool      [out]      isActive
    SFString    [in,out]   description      “”
    SFImage     [out]      image
    SFFloat     [out]      fieldOfView
    SFFloat     [out]      trackingInfo
    MFString    [out]      audioInfo
    SFVec2f     [out]      position

}
```

X3D Nodes related with LAE



LAESensingInfo

```
x3dom.registerNodeType(
    "LAESensingInfo",
    "X3DLAESensorNode",
    defineClass(x3dom.nodeTypes.X3DNode,
        function (ctx) {
            x3dom.nodeTypes.LAESensingInfo.superClass.call(this, ctx);

            /**
                Enabled field specifics whether this sensor is enabled. A disabled sensor doesn't produce any output.
            */
            this.addField_SFBool(ctx, 'enabled', true);

            /**
                description: Describe about the node's work
            */
            this.addField_SFString(ctx, 'description', '');

            /**
                image: Output the image data captured by
            */
            this.addField_SFImage(ctx, 'image', '');

            /**
                Preferred minimum viewing angle from this viewpoint in radians.
                Small field of view roughly corresponds to a telephoto lens, large field of view roughly corresponds
                Hint: modifying Viewpoint distance to object may be better for zooming.
            */
            this.addField_SFFloat(ctx, 'fieldOfView', 0.785398);

            /**
                Tracking point information
            */
            this.addField_SFVec3f(ctx, 'trackPointInfo', 0, 0, 0);
            /**
                audioInfo Display the url of audio info
            */
            this.addField_MFString(ctx, 'audioInfo', 0, 0, 0);
            /**
                The position fields of the Viewpoint node specifies a relative location in the local coordinate syste
            */
            this.addField_SFVec3f(ctx, 'position', 0, 0, 10);
```

Pure Sensor for LAE

LAEPureSensor

Type	accessType	Name	Default	Range	Profile	Description
SFBool	InputOuput	enabled	true		Interactive	Specifics whether this sensor is enabled. A disabled sensor doesn't produce any output.
SFNode	inputOutput	metadata	X3DMetadata Object		Interactive	Field to add metadata information.
SFString	inputOutput	decription	""		Interactive	Field to add description for node's work.
SFString	inputOutput	sensingDevice			Interactive	Type of sensor device like GPS sensor, Gyro Sensor, Phone Sensor or HMD.
SFVec3f	outputOnly	coordinate			Interactive	Display the coordinate information
SFVec3f	outputOnly	translationInfo			Interactive	Translation Information
SFVec3f	outputOnly	position			Interactive	Display the position information
SFRotation	outputOnly	rotation			Interactive	The rotation information

LAEPureSensor

```
LAEPureSensor  : LAESensorNode {
    SFBool          [in,out]   enabled              TRUE
    SFNode          [in,out]   metadata         NULL [X3DMetadataObject]
    SFString     [in,out]   description        “”
    SFString        [in,out]   sensingDevice
    SFVec3f         [out]      coordinate
    SFVec3f         [out]      translationInfo
    SFVec2f         [out]      position
    SFRotation      [out]      rotation
}
```

LAEPureSensor

```
x3dom.registerNodeType(
    "LAEPureSensor",
    "X3DLAESensorNode",
    defineClass(x3dom.nodeTypes.X3DNode,
        function (ctx) {
            x3dom.nodeTypes.LAEPureSensor.superClass.call(this, ctx);
            /**
                Enabled field specifics whether this sensor is enabled. A disabled sensor doesn't produce any output.
            */
            this.addField_SFBool(ctx, 'enabled', true);
            /**
                description: Describe about the node's work
            */
            this.addField_SFString(ctx, 'description', '');
            /**
                sensingDevice : Type of sensor device like GPS sensor, Gyro Sensor, Phone Sensor or HMD.
            */
            this.addField_SFString(ctx, 'sensingDevice', '');
            /**
                coordinate: Display the coordinate information
            */
            this.addField_SFVec3f(ctx, 'coordinate', 0, 0, 0);
            /**
                translationInfo Information
            */
            this.addField_SFFloat(ctx, 'translationInfo', 0, 0, 0);
            /**
                Tracking point information
            */
            this.addField_SFVec3f(ctx, 'trackPointInfo', 0, 0, 0);
            /**
                The position fields of the Viewpoint node specifies a relative location in the local coordinate syster
            */
            this.addField_SFVec3f(ctx, 'position', 0, 0, 10);
            /**
                rotation : The rotation information
            */
            this.addField_SFVec3f(ctx, 'rotation', 0, 0, 0);
```

Information Model of Tracker

Dimension	Type	Types
Input	Real world	• Camera information • 2D Image/video of LAE • 2D Chromakeying image/video of LAE • Object specification data • 3D primitives (points, lines, polygons, shapes) • 3D Model
	Pure Sensor	• Sensing Information
Output	Real world	• Position, orientation, volume, location • Haptic (force, direction, …) • Aural (intensity, pitch, etc.)
	Pure sensor	Pure sensor information (Coordinate value, Latitude…)

TrackingNode for LAE

TrackingSensor

```
LAETracking: LAESensorNode {
        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
}
```

Information Model of Recognizer

Dimension	Type	Types
Input	Real world	• Camera information • 2D Chromakeying image • Audio • Gesture
	Pure Sensor	• Pure sensor Information
Output	Real world	Event indication recognized from LAE action such gesture and audio information
	Pure sensor	Event indication recognized from the sensing information

Information Model of Spatial mapper

Dimension	Types
Input	• LAE sensor identifier spatial data • LAE consecutive spatial data • Tracking spatial data • Spatial information
Output	• Calibrated spatial information • Audio (direction, amplitude, units, scale) • Haptics (direction, magnitudes, units, scale)

Information Model of Event mapper

Dimension	Types
Input	• Event Information • Event identifier (Virtual camera control, Virtual object control, AR content control)
Output	• Translated event identifier for the given MAR scene • Interaction of a virtual object event

Event Input & Output for LAE

Dimension	Types	
	Input(LAE)	**Output(MAR)**
Display	Coordinate info Image info Video info Depth info Tracking info	Moving Moving direction Location Gesture 3D Image & Video
UI Event **(User Action)**	Gesture info Distance info Depth info Force info Motion info Gyro info Access info	Drag Touch Moving Rotation

Submit as NWIP Dec . 2016

Thank you.