

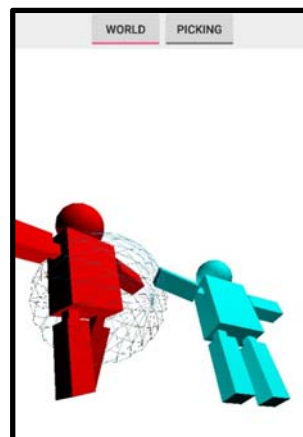
CS 482 Lab Session

Interactive Computer Graphics

2016.11.2

Goals

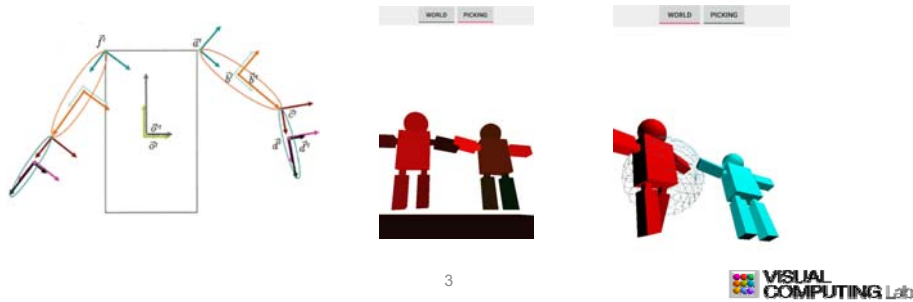
- Color picking with Scene graph
 - Implement user manipulation code



Preliminaries

KAIST

- You should be understand
 - How to work the SceneGraph
 - How to make a color-coded scene
 - How to draw scene on the FrameBuffer in OpenGL
 - How to pick color from the FrameBuffer



Pick OBJ with Color

KAIST

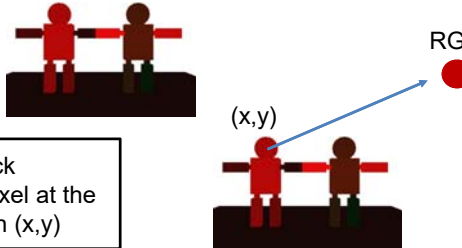
- Color mapping with ID
 - Make some color encoding method utilizing the ID of object: `colorTold()`
 - Store the color-ID pair
- Draw color-coded scene
 - Draw each object with coded color generated from object's ID
 - Color-coded scene is drawn on the FrameBuffer
- Pick object from the scene
 - Get the color on the position (x,y)
 - Find the object which is drawn with the picked color.

4

VISUAL
COMPUTING Lab


Pick OBJ with Color

Draw
color-coded scene
on the FrameBuffer



RGB: (24,2,2)

Pick
Color of pixel at the
position (x,y)





Pick
Object from the
picked color

(24,2,2) → id: 4 → Head_Joint Node

IMPORTANT:
Color-coded scene never
expose to a user

Draw
The scene which is
show to user

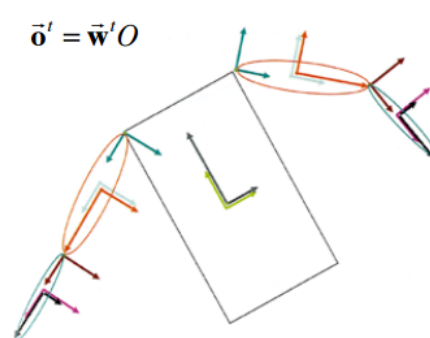




5


Scene graph

Draw
color-coded scene
on the FrameBuffer



RGB: (24,2,2)

Pick
Color of pixel at the
position (x,y)





Pick
Object from the
picked color

(24,2,2) → id: 4 → Head_Joint Node

IMPORTANT:
Color-coded scene never
expose to a user

Draw
The scene which is
show to user





6

Scene graph

KAIST

- Please remind the structure of SceneGraph

$$\vec{o}' = \vec{w}'O$$

$$\vec{o}' = \vec{w}'O$$

$$\vec{o}'' = \vec{o}'O'$$

$$\vec{a}' = \vec{o}'A$$

$$\vec{b}' = \vec{a}'B$$

$$\vec{b}'' = \vec{b}'B'$$

$$\vec{c}' = \vec{b}'C$$

$$\vec{d}' = \vec{c}'D$$

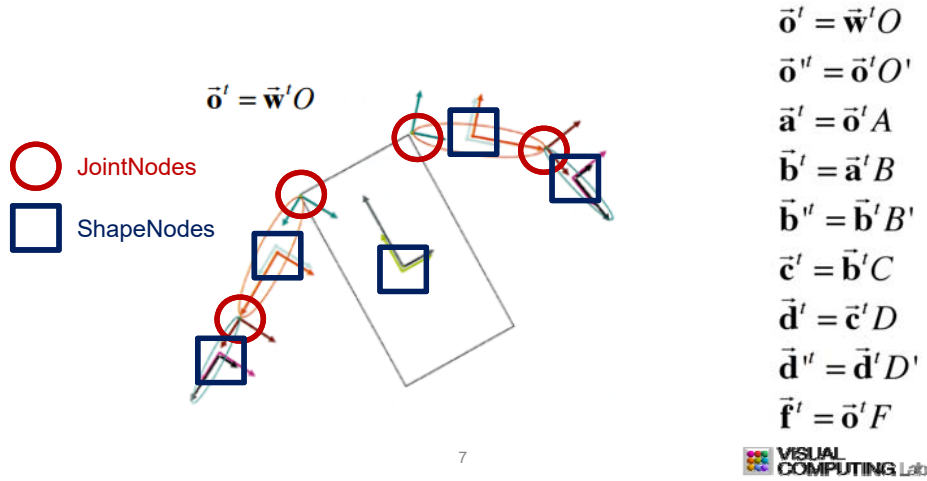
$$\vec{d}'' = \vec{d}'D'$$

$$\vec{f}' = \vec{o}'F$$



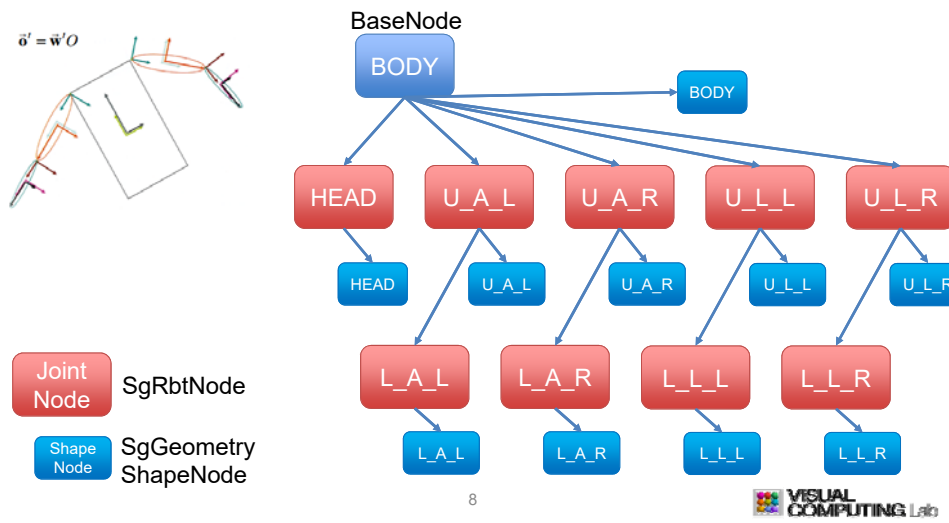
Scene graph

- Also, you should think about the different role of "JointNode" and "ShapeNode"



Scene graph

- Structure of MyRobot



Practice: result

KAIST

- Implement Color picker
 - Fill out the **Picker** class
 - Fill out the **MyGLRenderer** class
 - Replace the **blue robot** to the your own object (which is implemented at previous lab session)
- Motion events are already implemented
- World mode: is rendered with DEFAULT_SHADER
- Picking mode: is rendered with PICKING_SHADER (Color-coded scene)



9

VISUAL COMPUTING Lab

Practice: Task 1

KAIST

- Fill out the **Picker** class on the skeleton code

```
private void addMap(int id, SgRbtNode node) {
    //Task: add the pair of "id" and "node" into "idToRbtNodeMap".
}
```

4

```
private SgRbtNode find(int id){
    //Task: Find the JointNode utilizing the id from the idToRbtNodeMap
    return null;
}
```

5

```
@Override
public boolean visit(SgShapeNode node){
    boolean result = true;
    idCounter++;
    for(int i = nodeStack.size() - 1; i >= 0; --i){
        SgRbtNode asRbtNode = (SgRbtNode)nodeStack.get(i);
        if(asRbtNode != null){
            addToMap(idCounter, asRbtNode);
            break;
        }
    }
    float[] idColor = idToColor(idCounter);
    //Task: Draw the color-coded scene utilizing the generated idColor and drawer
    return result;
}
```

1

idToColor & colorToId methods are already implemented, but you should understand how to work those methods

```
public SgRbtNode getRbtNodeAt(int x, int y){
    //Task: Read the color at position (x,y) from the FrameBuffer in OpenGL.
    //After that, utilizing the color, find and return the JointNode which rendered at position (x,y)
    ByteBuffer query_color;
    int id = -1;
    return find(id);
}
```

6

The number is represented recommended fill-out order

10

VISUAL COMPUTING Lab

Practice: Task 2

- Fill out the **MyGLRenderer** class on the skeleton code

```

public void drawStuff(boolean picking) {
    ftime RigForm evsRt = RtAccumulator.getRtInCamera(s_world, s_currentCameraNode);
    ftime RigForm inEvRt = RigForm.in(evRt);

    if (picking) {
        ftime at11 = ftime.at11; at11.at11; at11.at11; 11;
        // ...
    }

    else {
        int sy = (int)(viewHeight - touchedY);
        //Task: utilizing PickingShader, pick object utilizing the Picker
        //Note: you should set the picked object to "g_currentPickedNode"
        //if you do that, picking will work.
        //Hint: you should use following: Picker class, g_world, g_currentPickedNode, glFlush method
        // ...

        if (g_currentPickedNode == g_groundNode) {
            s_currentPickedNode = null; // set to NULL
            mArcBall.setScale(1);
            // ...
        }
    }
}

public void picking(float x, float y) {
    touchedX = x;
    touchedY = y;
    pickingNode = true;
    //Task: draw solid-coded scene utilizing the PickingShader which is declared on initShader() method
    //Clear color, drawStuff with PickingShader, set back the clear color
}
    
```

11



Practice: Task3

- **MyGLRenderer**: Replace the “g_robotNode2” to the your own object which is implemented at previous lab session

```

public void initScene() {
    s_world = new SceneNode();
    s_world.name = "world";

    s_skyNode = new SceneNode(new RigForm(defaultCameraPos));
    s_skyNode.name = "sky";

    s_groundNode = new SceneNode();
    //define SphereToPlaneMeshGeometry g, float[] color, float[] translate, float[] scaling, float[] normal
    SphereToPlaneNode gnd = new SphereToPlaneNode(new Sphere(), new float[]{0.0, 0.0, 0.0}, new float[]{0.0, 0.0, 0.0}, new float[]{0.0, 0.0, 0.0}, new float[]{0.0, 0.0, 0.0});
    gnd.name = "gnd";

    s_groundNode.addChild(gnd);
    s_groundNode.name = "ground";

    s_robotNode = robotFactory.constructRobot(new float[]{10, 0, 0});
    s_robotNode.setName(new float[]{10, 0, 0});
    s_robot2Node = robotFactory.constructRobot(new float[]{0, 10, 0});
    s_robot2Node.setName(new float[]{0, 10, 0});
    s_world.addChild(s_robotNode);
    s_world.addChild(s_robot2Node);
    s_world.addChild(s_groundNode);

    s_currentCameraNode = s_skyNode;
}
    
```

12



If your object do not work as a MyRobot object, you should modify the center position of nodes on your own object.

Also you should refer the pages about SceneGraph structure.

Practice: Ref

KAIST

- Please check the **MyGLSurfaceView**
 - queueEvent
 - queueEvent is provided class in openGL ES
 - It is run on the rendering thread.
 - it is need for Event handling with your renderer

```
//queueEvent is run on the rendering thread.
//it is need for communicate with your renderer
queueEvent(new Runnable() {
    @Override
    public void run() {
        mRenderer.picking(mPreviousX, mPreviousY);
    }
});
```

13

 VISUAL
COMPUTING Lab

Advance

KAIST

- Let's trying to use embedded sensors on Android phone!
(for participators who already implement three tasks at the previous lab session)
- We can use three kind of IMU (Inertia Measurement Unit) measurements from an android phone.
 - Linear Accelerometer
 - Angular Accelerometer (Gyroscope)
 - Orientation (roll, pitch, yaw)

14

 VISUAL
COMPUTING Lab

Advance



- Let's trying to use embedded sensors on Android phone!

Orientation code is already implemented

```

public class MainActivity extends Activity implements SensorEventListener {
    public static Context context;
    private RelativeLayout rootView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // Initialize sensors
        SensorManager sensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);
        sensorBaro = sensorManager.getDefaultSensor(Sensor.TYPE_GYROSCOPE);
        sensorLinearAcc = sensorManager.getDefaultSensor(Sensor.TYPE_LINEAR_ACCELERATION);
        sensorDrift = sensorManager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD);
        sensorHwa = sensorManager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD);
        sensorAcc = sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
        sensorGyro = sensorManager.getDefaultSensor(Sensor.TYPE_GYROSCOPE);
        sensorGrav = sensorManager.getDefaultSensor(Sensor.TYPE_GRAVITY);

        // Register listeners
        sensorManager.registerListener(this, sensorLinearAcc, SensorManager.SENSOR_DELAY_GAME);
        sensorManager.registerListener(this, sensorDrift, SensorManager.SENSOR_DELAY_GAME);
        sensorManager.registerListener(this, sensorHwa, SensorManager.SENSOR_DELAY_GAME);
        sensorManager.registerListener(this, sensorGrav, SensorManager.SENSOR_DELAY_GAME);
    }

    @Override
    public void onSensorChanged(SensorEvent event) {
        switch (event.sensor.getType()) {
            case Sensor.TYPE_ORIENTATION:
                // Handle orientation change
                float[] values = new float[3];
                boolean success = SensorManager.getOrientation(event.values, values);
                if (success) {
                    float[] temp = new float[3];
                    SensorManager.getOrientation(R, 1, values);
                    // Handle orientation change
                    values[0] = temp[0];
                    values[1] = temp[1];
                    values[2] = temp[2];
                    temp = null;
                }
                // Handle orientation change
                values[6] = LPF(values[6], 6);
                values[7] = LPF(values[7], 7);
            // ... other cases
        }
    }
}
    
```

15



Advance



- Let's trying to use embedded sensors on Android phone!

```

public void updateSensorValues(float[] values, int type) {
    // Handle sensor data
    switch (type) {
        case Sensor.TYPE_LINEAR_ACCELERATION:
            // Handle linear acceleration
            break;
        case Sensor.TYPE_GYROSCOPE:
            // Handle gyroscope
            break;
        case Sensor.TYPE_ORIENTATION:
            // Handle orientation
            // Note: received ORIENTATION measurements are same to the following: values[0]: Yaw, values[1]: Pitch, values[2]: Roll, in Radian (ruler angle)
            // Note 2: Set the rotation component of red robot's rigid body transform to ORIENTATION value.
            break;
        case Sensor.TYPE_MAGNETIC_FIELD:
            // Handle magnetic field
            break;
        case Sensor.TYPE_ACCELEROMETER:
            // Handle accelerometer
            break;
    }
}
    
```

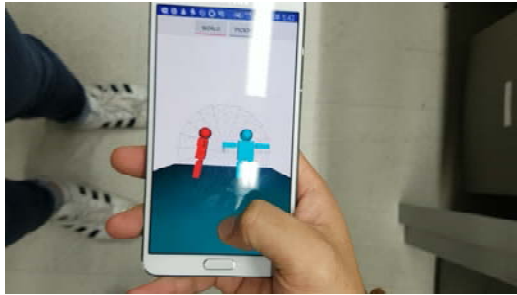
16



Advance

KAIST

- Let's trying to use embedded sensors on Android phone!
 - Utilizing the measured Orientation values (roll, pitch, yaw of your android phone),
 - Rotate red robot when you touched a finger on screen



17

 VISUAL
COMPUTING Lab