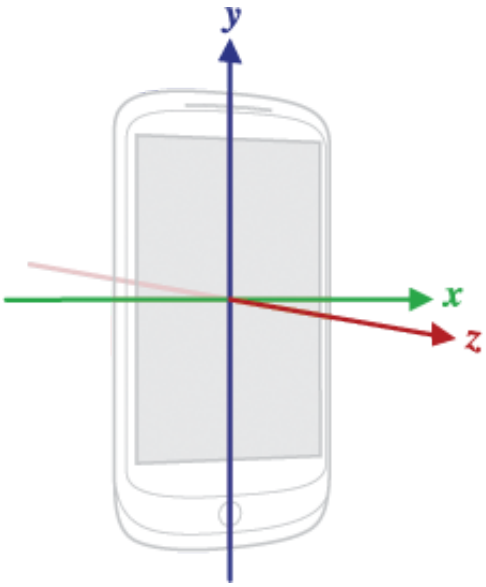
IMU方向说明

1. 手机的IMU坐标系统

1.1 手机IMU坐标轴定义

Android官网对手机IMU坐标轴的规定如下



需要注意的是：**当设备的屏幕方向更改时，坐标轴不会发生交换**，也就说，不管手机如何转动和摆放，Y轴始终是平行屏幕指向手机上方的，Z轴始终垂直屏幕指向上方。

1.2 手机上如何获取欧拉角信息

要通过 `ROTATION_VECTOR` 或者 `GAME_ROTATION_VECTOR` 获取欧拉角，需要用到系统两个方法：

- SensorManager#getRotationMatrixFromVector

getRotationMatrixFromVector

Added in API level 9

```
public static void getRotationMatrixFromVector (float[] R,
               float[] rotationVector)
```

Helper function to convert a rotation vector to a rotation matrix. Given a rotation vector (presumably from a ROTATION_VECTOR sensor), returns a 9 or 16 element rotation matrix in the array R. R must have length 9 or 16. If R.length == 9, the following matrix is returned:

```
  /  R[ 0]   R[ 1]   R[ 2]   \
  |  R[ 3]   R[ 4]   R[ 5]   |
  \  R[ 6]   R[ 7]   R[ 8]   /
```

If R.length == 16, the following matrix is returned:

```
  /  R[ 0]   R[ 1]   R[ 2]   0  \
  |  R[ 4]   R[ 5]   R[ 6]   0  |
  |  R[ 8]   R[ 9]   R[10]   0  |
  \  0       0       0       1  /
```

Parameters	
`R`	**`float`**: an array of floats in which to store the rotation matrix
`rotationVector`	**`float`**: the rotation vector to convert

- SensorManager#getOrientation

getOrientation

Added in API level 3

```
public static float[] getOrientation (float[] R,
                float[] values)
```

Computes the device's orientation based on the rotation matrix.

When it returns, the array values are as follows:

- values[0]: *Azimuth*, angle of rotation about the -z axis. This value represents the angle between the device's y axis and the magnetic north pole. When facing north, this angle is 0, when facing south, this angle is π. Likewise, when facing east, this angle is π/2, and when facing west, this angle is -π/2. The range of values is -π to π.
- values[1]: *Pitch*, angle of rotation about the x axis. This value represents the angle between a plane parallel to the device's screen and a plane parallel to the ground. Assuming that the bottom edge of the device faces the user and that the screen is face-up, tilting the top edge of the device toward the ground creates a positive pitch angle. The range of values is -π/2 to π/2.
- values[2]: *Roll*, angle of rotation about the y axis. This value represents the angle between a plane perpendicular to the device's screen and a plane perpendicular to the ground. Assuming that the bottom edge of the device faces the user and that the screen is face-up, tilting the left edge of the device toward the ground creates a positive roll angle. The range of values is -π to π.

Applying these three rotations in the azimuth, pitch, roll order transforms an identity matrix to the rotation matrix passed into this method. Also, note that all three orientation angles are expressed in **radians**.

Parameters	
`R`	`float`: rotation matrix see `getRotationMatrix(float[], float[], float[], float[])`.
`values`	`float`: an array of 3 floats to hold the result.

Returns	
`float[]`	The array values passed as argument.

通过以上方法和说明我们得知，通过getRotationMatrixFromVector可以从四元数得到旋转矩阵R，getOrientation方法，可以从旋转矩阵R得到欧拉角。通过getOrientation方法的说明，我们可以得知**欧拉角的方向定义和取值范围**。

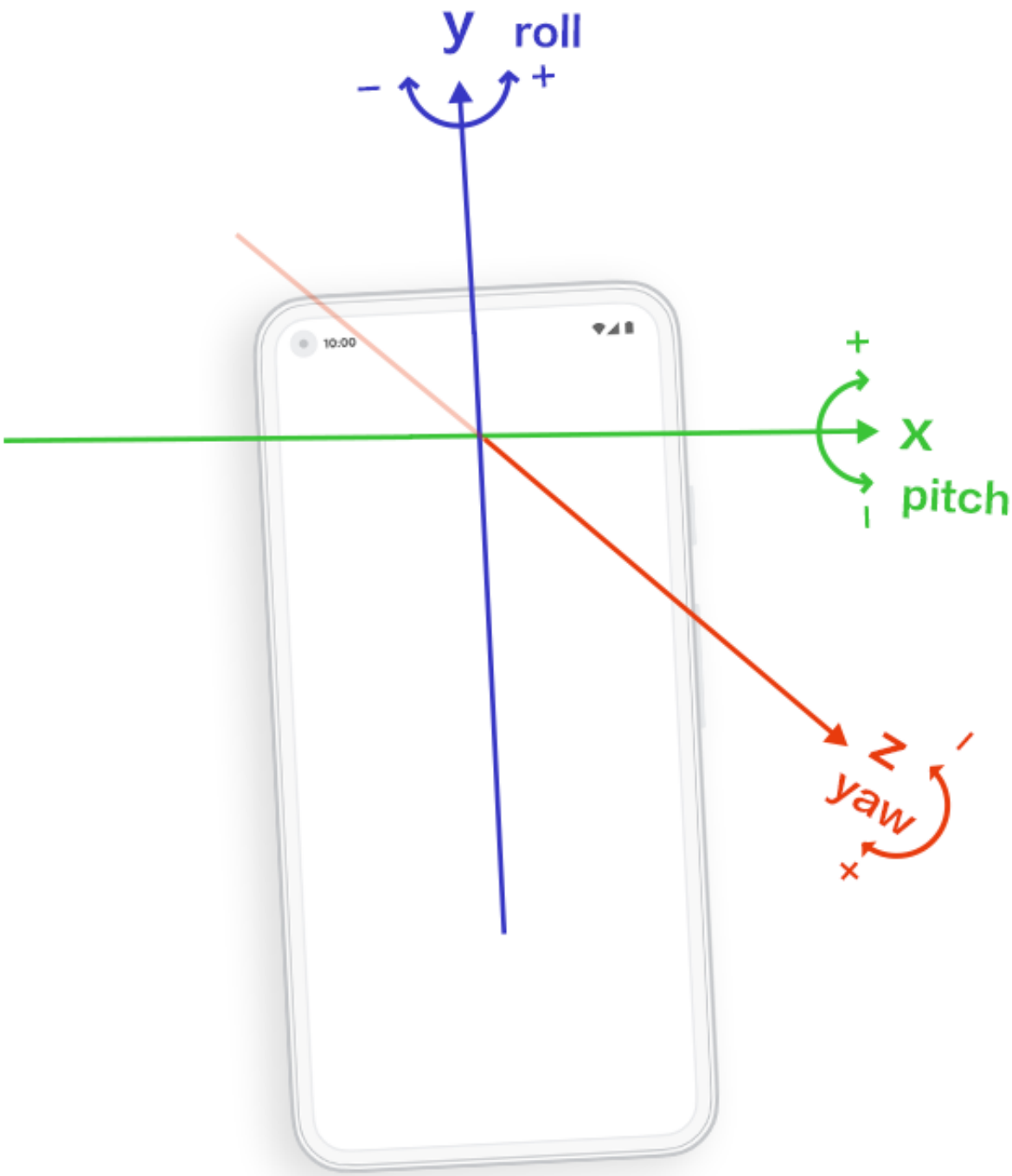
通过解释我们可以得知：**整个坐标体系的定义是Y指向北，-Z指向地面**

- 绕-Z轴旋转，定义为Azimuth，也就是Yaw，假设Y轴指向北为0，那么指向西是-π/2，指向东是π/2,指向南为π(-π)，左西右东，也就是**向左偏为负，向右偏为正**；
- 绕X轴旋转定义为Pitch，手机平行地面放置，尾端朝向用户时，Pitch为0，手机头朝地面转为正，朝上

转为负，也就是**低头为正，抬头为负**，范围为-π/2到π/2；

- 绕Y轴旋转定义为Roll，手机平行地面放置，尾端朝向用户时，抬起左边为正，也就是**顺时针翻转为正，逆时针翻转为负**，范围为-π到π；注：原文档里写，“左侧边缘向地面翻转为正”，与上面结论不符，但是实测了华为、三星、OPPO、Pixel等多款手机，都是按照上面的结论定义的正负，因此怀疑文档有误，可能是我理解有误，待考证

方向如下图所示：


y
roll
−
+
10:00
+
x
pitch
−
z
yaw
−
+

可以把手机屏幕向上放置，尾端朝向自己，想象成一架飞机，能比较直观的理解。

1.3 坐标轴旋转

假设用户把手机竖在面前使用，按照通常的自然理解，绕Z轴应该为roll，绕y轴为yaw，绕x轴仍为pitch，这时候，就需要用到坐标轴转换方法：

- SensorManager#remapCoordinateSystem

remapCoordinateSystem

Added in API level 3

```
public static boolean remapCoordinateSystem (float[] inR,
                int X,
                int Y,
                float[] outR)
```

Rotates the supplied rotation matrix so it is expressed in a different coordinate system. This is typically used when an application needs to compute the three orientation angles of the device (see `getOrientation(float[], float[])`) in a different coordinate system.

When the rotation matrix is used for drawing (for instance with OpenGL ES), it usually **doesn't need** to be transformed by this function, unless the screen is physically rotated, in which case you can use `Display.getRotation()` to retrieve the current rotation of the screen. Note that because the user is generally free to rotate their screen, you often should consider the rotation in deciding the parameters to use here.

Examples:

- Using the camera (Y axis along the camera's axis) for an augmented reality application where the rotation angles are needed:

  ```
  remapCoordinateSystem(inR, AXIS_X, AXIS_Z, outR);
  ```

- Using the device as a mechanical compass when rotation is `Surface.ROTATION_90`:

  ```
  remapCoordinateSystem(inR, AXIS_Y, AXIS_MINUS_X, outR);
  ```

  Beware of the above example. This call is needed only to account for a rotation from its natural orientation when calculating the rotation angles (see `getOrientation(float[], float[])`). If the rotation matrix is also used for rendering, it may not need to be transformed, for instance if your `Activity` is running in landscape mode.

Since the resulting coordinate system is orthonormal, only two axes need to be specified.

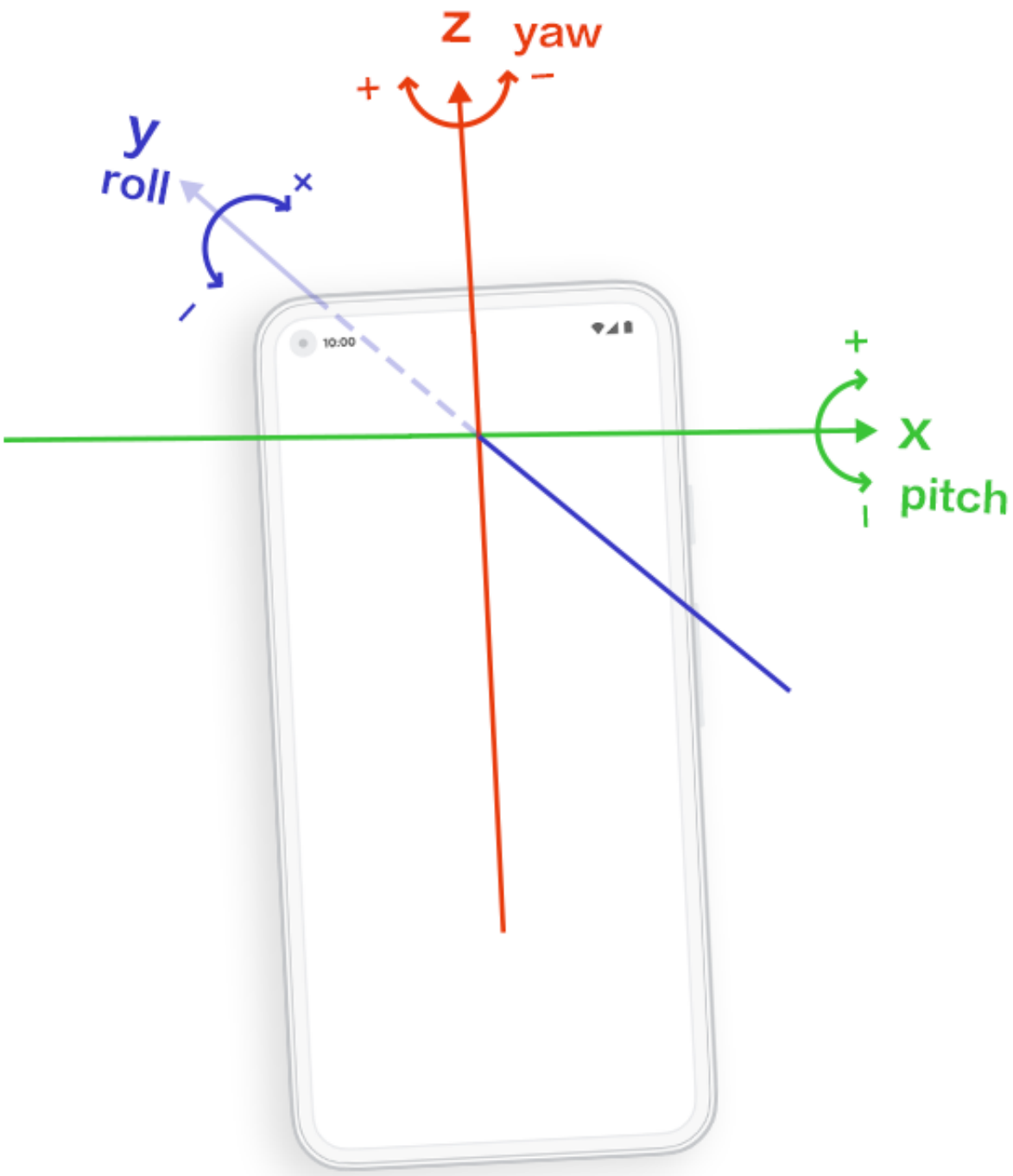
Parameters	
inR	**float**: the rotation matrix to be transformed. Usually it is the matrix returned by **getRotationMatrix(float[], float[], float[], float[])**.
X	**int**: defines the axis of the new cooridinate system that coincide with the X axis of the original coordinate system.
Y	**int**: defines the axis of the new cooridinate system that coincide with the Y axis of the original coordinate system.
outR	**float**: the transformed rotation matrix. inR and outR should not be the same array.

根据方法的说明，手机竖直放置，要把Z轴转到Y轴的方向，原Y轴转到-Z轴，才能实现绕Z轴为roll，绕y轴为

yaw的效果，因此，需要调用以下方法旋转坐标轴：

```
    remapCoordinateSystem(inR, AXIS_X, AXIS_Z, outR);
```

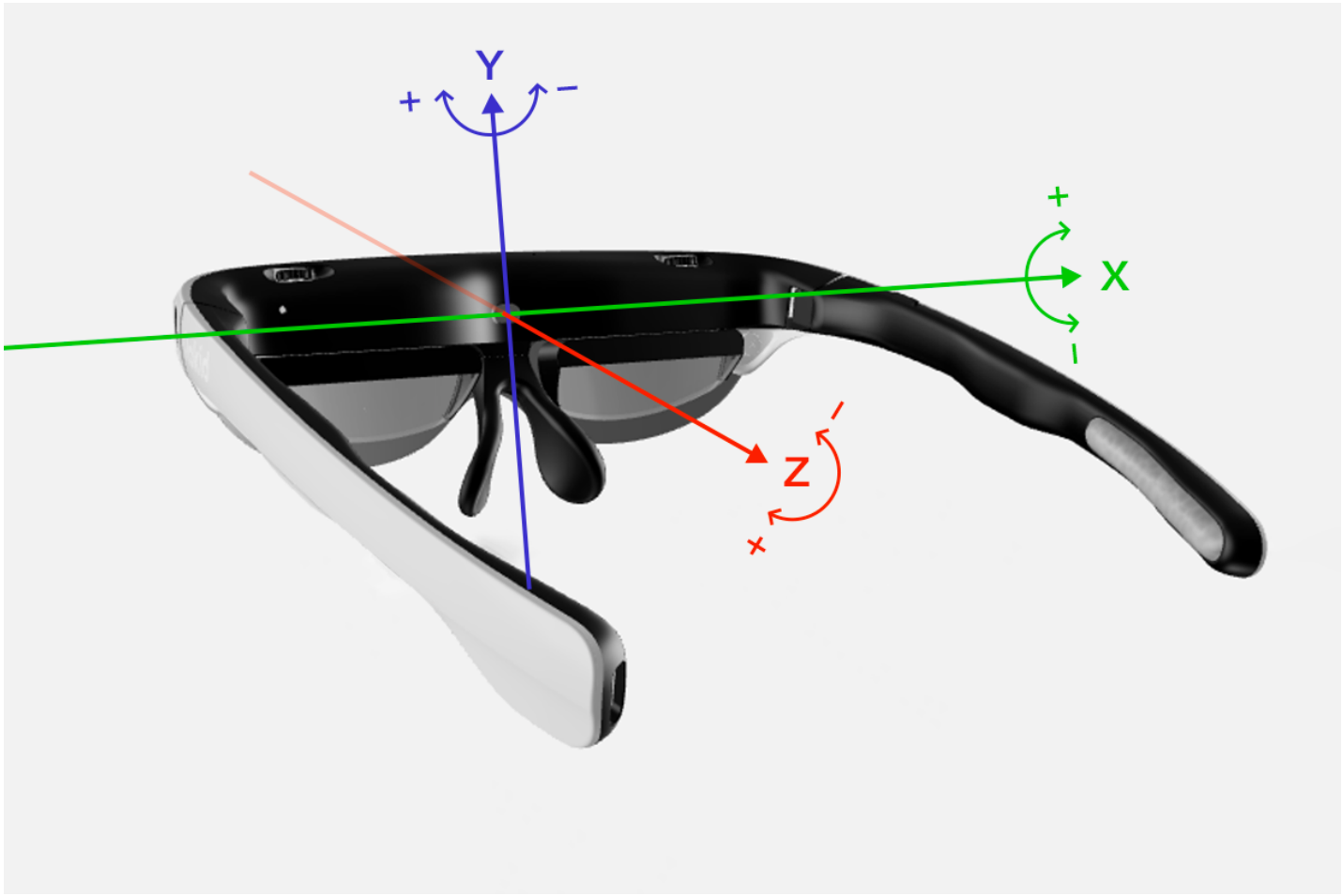
旋转后的坐标系和yaw pitch roll如下图：


z
yaw
+
−
y
roll
x
−
+
x
pitch
−
10:00

2. Rokid Glass的IMU坐标系统

2.1 Glass的IMU坐标轴定义

Rokid的Glass产品（Rokid glass2/Xcraft/Rokid Air）的坐标轴定义，参考了手机的坐标轴，**垂直屏幕指向用户的方向为Z**。因此，默认的Yaw，Pitch和Roll的定义，也与手机一致，如下：



2.2 Glass上如何获取欧拉角信息

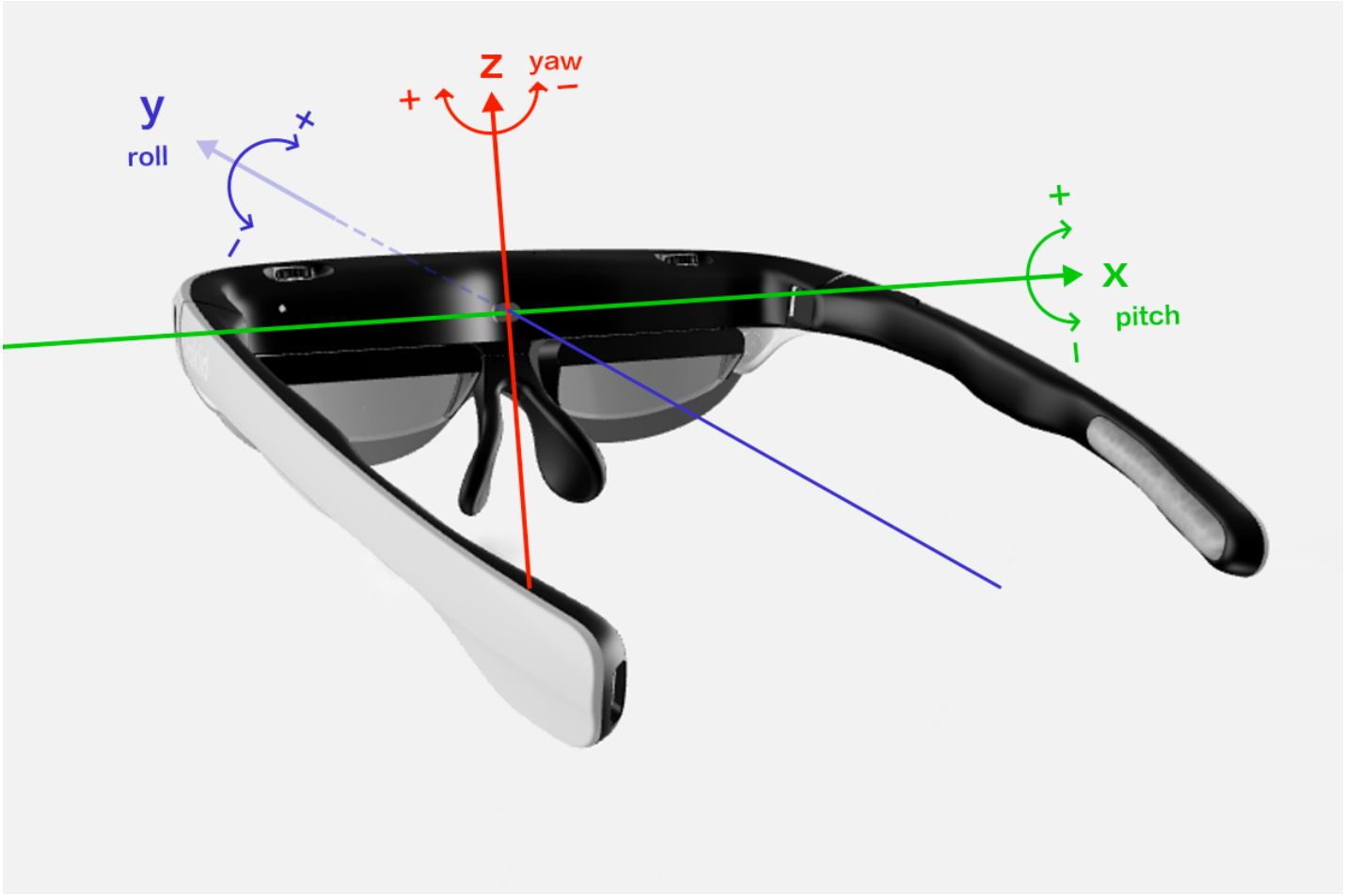
通常用户使用眼镜时，正常的佩戴姿势，都是**屏幕垂直于地面使用，也就是跟手机竖在面前使用的场景一样**，因此要在glass上由 `ROTATION_VECTOR` 和 `GAME_ROTATION_VECTOR` 获取到符合自然理解的Yaw Pitch和Roll，需要按照手机竖直放置使用的场景，进行坐标系的旋转，因此要获取欧拉角，需要依次调用如下三个方法：

```
SensorManager.getRotationMatrixFromVector(rotationMatrix, rotationVector);

SensorManager.remapCoordinateSystem(rotationMatrix, SensorManager.AXIS_X,
                SensorManager.AXIS_Z, adjustedRotationMatrix);

SensorManager.getOrientation(adjustedRotationMatrix, orientation);
```

通过这样的方法得到的orientation的[0]、[1]、[2]分别是Yaw，Pitch，Roll，方向分别为：

- Yaw：绕Z轴向左偏为负，向右偏为正
- Pitch：抬头为负，低头为正
- Roll：向左扭头为负，向右扭头为正

如图：



以上的欧拉角及方向的定义，与**手机屏幕垂直地面，屏幕朝向用户**使用时的定义一致。

3. UXR及AXR SDK输出的欧拉角定义

UXR及AXR SDK输出的欧拉角坐标轴及方向的定义，与上述2.2中通过Android方法得到的Glass的欧拉角坐标轴和方向定义一致。

其他说明

之前提供的Native方法获取欧拉角的示例，最后通过getOrientation方法获取的Yaw，Pitch，Roll都乘了负值，是因为2D绘制的时候，头动实际上是根据头动角度反方向移动画布，因此这里得到的欧拉角全部取了反，**并不表示系统标准定义的欧拉角方向是反的**。