ChatGPT

Resolving Magic Window Immersive and Comfort Mode Rendering Issues

Issue 1: Cropped View in Immersive Mode (Partial Screen Visible)

In Immersive mode, only a portion of the curved screen is visible (as seen above, where the 360° video screen is cut off on one side). This indicates the camera is not properly aligned with the screen, causing a cropped or off-center viewport.

Cause – Camera Alignment: In the immersive "desktop" mode, the camera was placed at the center of the curved screen but not oriented toward it. The code positions the camera at the same point as the screen's center (0, 1.6, 0) [1] and even sets the OrbitControls target to that same point [2]. This means initially the camera has no offset from the screen and likely retained its default orientation (facing the negative Z-axis). However, the curved screen geometry was generated spanning ~100° in front of the camera but oriented around the **+Z axis** (the code uses `thetaStart = Math.PI/2 - arc/2`, centering the cylinder segment at +Z). In other words, the screen was effectively **behind the camera's view**. As a result, only an edge of the screen fell into the camera's frustum (appearing cropped) instead of the screen filling the view. In summary, a misalignment between the camera's gaze direction and the curved screen's position caused most of the screen to be out of view.

Fix – Reorient the Screen or Camera: We need to ensure the curved screen is directly in front of the camera. The simplest solution is to **rotate the screen geometry 180° around the Y-axis** so that the portion of the cylinder with the video texture faces the camera. For example, after creating the curved screen mesh, do:

```
if (currentMode === 'desktop') {
    screen.rotation.y = Math.PI;  // rotate 180 degrees to face camera
}
```

This flips the curved screen to the opposite side, aligning it with the camera's forward direction. Now the camera (even if still at the center) will look **into** the curved screen rather than away, making the entire screen visible. An alternative is to adjust the cylinder's creation parameters (e.g. set `thetaStart` so the arc is centered at –Z) or explicitly set the camera to look at a point on the screen. For instance, one could set `controls.target` to a point slightly **in front** of the camera on the screen's surface instead of the exact center. However, rotating the screen 180° is straightforward and ensures the screen's content is front-facing. After this adjustment, the immersive view should show the full curved video screen without cropping.

Issue 2: Upside-Down Video in Comfort Mode (Flipped Screen)

In Comfort mode, the video on the flat screen appears upside down (as shown above, the content is inverted vertically). This points to a texture orientation issue on the flat screen.

Cause – Texture UV/Orientation Mismatch: The video texture is being applied with the wrong vertical orientation on the flat screen. By default, Three.js **flips textures vertically** (the `Texture.flipY` property is true by default) when uploading to the GPU, because the WebGL UV coordinate system's origin differs from the image pixel origin. In our code, a "critical fix" explicitly set `videoTexture.flipY = false` for the video texture [3]. This *unflipped* the video on upload – a step that was likely done to correct the inside of the curved screen or a model's UV mapping. While that helped the immersive screen, it caused the flat screen to render the video upside down. Essentially, the flat **PlaneGeometry**'s UV coordinates were not adjusted, so with `flipY=false` the image ended up inverted (head at bottom, sky at top, as seen in the screenshot).

Notably, the code already handles this for the curved screen: it manually flips the V (vertical) texture coordinates of the curved cylinder geometry [4] to correct the video's orientation on the inside of the curve. No such flip is applied for the flat screen, so with the texture no longer auto-flipped, the flat screen's image is upside down.

Fix – Correct the Texture's Vertical Orientation: There are two ways to fix the upside-down video on the flat screen:

- **Option 1: Flip the UV coordinates for the flat screen geometry.** After creating the PlaneGeometry for comfort mode, iterate over its UVs and invert the V coordinate, just as done for the cylinder. For example:

```
// After creating PlaneGeometry for flat screen (16:9 aspect)
const uvAttr = screenGeo.attributes.uv;
for (let i = 0; i < uvAttr.array.length; i += 2) {
  uvAttr.array[i+1] = 1 - uvAttr.array[i+1];  // flip V coordinate
}
uvAttr.needsUpdate = true;
```

This will flip the video texture vertically on the plane. Since we set `flipY=false` on the texture globally, adjusting the geometry's UV to compensate ensures the video appears right-side up on the flat screen (the top of the video maps to the top of the plane).

- **Option 2: Use the texture's flipY for flat mode.** If we only need the flip for the flat screen, we could re-enable vertical flip on the texture in comfort mode. For instance, when switching to comfort mode (screenCurve = 0), do: `videoTex.flipY = true; videoTex.needsUpdate = true;`. This lets Three.js flip the video texture upright for the flat screen. (In immersive mode you would set it back to false or create a separate texture if needed.) This approach is slightly less elegant because the code currently reuses one VideoTexture for all modes, but it is a viable quick fix if managing separate textures.

Both approaches achieve the same result: the video will no longer render upside down. The first approach (flipping the plane's UVs) aligns with how the curved screen was handled and keeps the texture settings consistent (leaving `flipY=false` globally). The second approach leverages Three.js's built-in flipping for that mode. After applying one of these fixes, the **Comfort mode** flat screen will display the video in the correct orientation (no longer mirrored vertically).

Additional Notes

By implementing the above adjustments, the immersive video player should render correctly in Chrome's Magic Window mode on the ASUS ZenBook Duo (or any device):

- **Immersive mode** will have the camera properly facing the curved screen, eliminating the cropped-view issue.
- **Comfort mode** will show an upright (non-inverted) video on the flat screen.

It's also wise to double-check horizontal orientation if any text or logos are mirrored. In our case, using `THREE.BackSide` for the curved screen means we're viewing the inside of the cylinder. We corrected the vertical flipping; if any left-right mirroring is observed, we could address it by flipping the U coordinates or rotating the screen 180° on the Z-axis. However, assuming the content was appearing correct aside from the upside-down issue, no further horizontal fixes are needed.

With these code-level fixes (camera/screen alignment and texture orientation adjustments), the immersive video should display as intended in both modes, providing a stable Magic Window experience in Chrome without a headset. All changes are confined to the rendering logic – no headset or WebXR session is required, so testing in a regular browser window should now produce the correct full-view, correctly oriented output in each mode.

Sources: The Three.js documentation and community forums confirm the importance of `flipY` for texture orientation and show that video textures can appear inverted if not handled properly [5]. The provided code snippets (from the application code) illustrate where the issues arose and how to fix them – flipping UVs for the curved screen [4], the texture settings [3], and the camera positioning logic [1] [2] that needed adjustment. These changes directly address the root causes, ensuring the video mapping and camera view are correct in both Comfort and Immersive modes.

---

[1] [2] [3] [4] app.js
file://file-2VsAWVesHE962QhDF7UbDS

[5] The video is getting played inside out/ inverted - Questions - three.js forum
https://discourse.threejs.org/t/the-video-is-getting-played-inside-out-inverted/60494