ChatGPT

Immersive Mobile Video: Optimal Screen Layout & Geometry

Magic Window Immersion vs. Comfort

A *“magic window”* mode lets users view an immersive 3D scene on a phone without a headset – the phone acts like a window into a virtual world, updating the view as you rotate or tilt the device [1]. To keep this experience comfortable, it’s important not to overwhelm the viewer with a full 360° environment. In a full sphere video, the viewer can feel disoriented or fear missing action behind them [2]. Instead, most non-headset XR experiences focus the content in front of the user. For example, **180° VR videos** (a front-facing hemisphere) are popular because they still feel immersive while the back half is left blank [3] – the viewer isn’t forced to spin around. In short, limiting the field of view of the immersive content is key to maintaining comfort in magic-window mode.

Choosing a Curved Display Geometry

To preserve a sense of depth and immersion beyond a flat screen, use a **curved, concave display** in front of the viewer rather than a completely flat plane. Research confirms that curved “wrap-around” displays create a strong feeling of immersion [4]. In particular, a concave screen (curved toward the user) was rated highly for both **immersiveness and perceptibility (clarity)**, whereas a full spherical or hemispherical enclosure scored lower on clarity [4]. This suggests the optimal geometry is a **partial curved surface** – essentially a section of a sphere or cylinder that spans the user’s forward view. A **cylindrical segment** (curved only horizontally) or a **sphere segment** (curved both horizontally and vertically) in front of the user can provide the wrap-around effect without encircling them entirely. Both approaches keep the main action in a concave “screen” in front of the viewer, which users tend to find comfortable and natural.

- *Partial Cylinder vs. Sphere:* A partial **cylinder** is often ideal for video playback because it curves the image horizontally around the viewer but remains vertically upright. This means it **preserves vertical lines and proportions** (important for text or UI elements) [5]. A **sphere segment** (like a section of a dome) also works and can curve vertically as well, but large vertical curvature (e.g. a full dome above/below) isn’t usually necessary and could distort the image. In practice, many implementations use a sphere or dome geometry but restrict its angular size. For example, you can use a Three.js `SphereGeometry` with a limited angle (phi/theta length) to make a curved screen [6], effectively creating a “curved plane” in front of the camera.

Field of View and Curvature Range

For an immersive yet comfortable result, **limit the field of view of the curved screen to a moderate range** – roughly **90° to 120°** of horizontal coverage is a good target. This means the curved screen spans perhaps a quarter to one-third of a full circle around the viewer. Within this range, the user gets a wide peripheral image (far more immersive than a flat video), but does not feel like they must spin around excessively. In comparison, a full 360° sphere or even a 180° hemisphere is more likely to induce discomfort or dizziness, especially if the user is moving the phone around rapidly. Studies of VR sickness note that **reducing the FOV can reduce motion sickness** – extremely wide fields stimulate more peripheral vision which can aggravate wooziness [7]. Empirically, many VR “comfort modes” cap the view to narrower angles to keep users comfortable. By using, say, a 100° curved screen, you still engage the viewer’s peripheral vision for immersion, but you avoid the extremes that might cause strain.

In terms of **curvature**, a span in the 90–120° range (horizontal) typically corresponds to a gentle curvature that doesn’t overly distort the image. For instance, a 120° cylindrical screen would wrap a bit past the viewer’s left and right shoulders. This provides a wrap-around feel, but since it’s not a full half-circle, the edges of the content are still within a comfortable glancing angle. Many practitioners indeed consider **120°** a “wide FOV” experience that is still manageable without a headset – notably, Apple’s VisionOS even defines a “Wide FOV” video format for roughly GoPro-style wide-angle content (as distinct from full 180° or 360°) [8]. In summary, aiming for on the order of ~100° horizontal by ~60–90° vertical coverage will keep the immersive video front-facing and avoid the need for the user to physically turn all the way around or look straight up/down, thereby preventing disorientation.

Maintaining Clarity and Avoiding Warping

A crucial design goal is to preserve clarity – the curved screen shouldn’t turn the video into a warped, unwatchable image. To achieve this, **match the content’s projection to the screen geometry.** Immersive videos are typically captured with special wide-FOV projections (e.g. equirectangular 360, fisheye 180, etc.), and they need to be displayed on the correct curved surface to look right [8]. For example, a 360° equirectangular video should be mapped onto an actual sphere (or sphere segment) in the scene, not onto a flat plane, or else it will appear distorted. A 180° VR video (often stored as half-equirectangular) should be mapped onto a hemispherical surface. If you have a **“wide FOV”** normal video (say, a GoPro clip with a 120° fisheye lens), consider using a partial sphere or cylinder and the appropriate UV mapping so that straight lines appear correctly curved as they would in reality [8]. In other words, use the same projection in playback that was used in capture (or an equivalent) to maintain geometric fidelity. This will ensure that the image isn’t fun-house warped and that any text or on-screen graphics remain legible.

Avoid extreme curvature that would make parts of the image hard to see. If the screen’s curvature is too tight (covering too wide an angle in too small a radius), the edges of the video will be viewed at a sharp angle. That can cause stretching or make text near the periphery appear rotated or thin. Using a larger radius for the curved screen (i.e. placing it a bit further from the camera) makes the curve gentler for a given FOV, which helps keep the image natural. A cylindrical projection inherently helps with this by keeping vertical features undistorted [5]. In practice, creators often keep important content (faces, dialogue, subtitles, etc.) near the center of the view (“title safe area”), where the curvature is minimal, and let only background scenery reach into the far peripheral portions. This way, the immersive curvature enhances depth but **doesn’t compromise readability or detail** where it matters. Research in ergonomics has found that extremely wide or spherical displays can reduce “perceptibility” (the ease of visually interpreting details) compared to moderately curved ones [4]. Thus, balancing the curvature is key – enough to wrap the viewer, but not so much that the image bends beyond comfortable reading angles.

Interaction Considerations (Tilting and Dragging)

The magic-window experience relies on the user’s input (device orientation or mouse drag) to look around the video. You should design the viewing geometry to **work with these inputs naturally**. For mobile devices, the gyroscope/accelerometer can drive the camera rotation so that tilting the phone up/down or turning left/right lets the user inspect the curved video window. Keep the motion one-to-one and intuitive – when they rotate the phone 30° to the right, for example, the virtual camera pans

~30° into the panoramic content. If you've limited the content to, say, 120°, you may want to **clamp the camera rotation** at the edges so the user doesn't see empty space beyond the screen. For instance, allow at most ±60° of yaw rotation from center, so that the view stops at the edges of your curved screen. Hitting an edge could simply show a black background or faded falloff, but ideally the user shouldn't frequently hit the stops if the content naturally keeps their focus in the middle. Similarly, limit the vertical look range if the content doesn't cover a full vertical span (e.g. don't let the user flip the view completely upside-down if there's no video above or below).

On desktop (mouse drag), use an intuitive click-and-drag or swipe control that mimics a gentle camera orbit around the center. It's wise to keep the rotations **3-DoF (three degrees of freedom: yaw and pitch, but no roll)** in a video player – i.e. the horizon stays level. Tilting the phone naturally doesn't roll the horizon in most magic-window apps, and that avoids confusion. In short, the viewing should feel like a stable window into a world: the user *peeks* around by moving the device or cursor, within comfortable bounds, without unintended movement. By constraining the camera to the content's bounds and using the device's sensors properly, you maintain the illusion of a stable, immersive scene that the user is in control of [1].

Implementation Tips (Three.js Configuration)

To achieve the above in Three.js or WebXR frameworks, here are some practical recommendations:

- **Curved Screen Geometry:** Use a large, curved mesh for the video. A convenient choice is a section of a sphere. For example, you can create a `SphereGeometry` with a radius of a few units and specify `phiLength` (horizontal angle) and `thetaLength` (vertical angle) to, say, 100°–120° and 60°–90° respectively [6]. This generates a sphere segment – essentially a domed "screen" curving around the camera. If you prefer a purely cylindrical curve, Three.js's `CylinderGeometry` can be used (you would create a tall cylinder and perhaps use only the inner face). Ensure the geometry has sufficient segmentation (subdivisions) so that it appears smooth and doesn't pixelate the video texture.

- **Camera Setup:** Place the camera at the center of curvature. In the sphere segment approach, the camera should be at the sphere's center (e.g. position `(0,0,0)` if the sphere is centered there). This way, when the camera rotates, it's like the viewer's head rotating inside the curved screen. Set a reasonable camera field-of-view – often around 60–75° (Three.js camera `fov` parameter) – which is a typical perspective that maps well to how we view a phone at arm's length. The camera's FOV, combined with the screen's radius and size, will determine how much of the curved video is visible at once. You might experiment with slightly wider camera FOV (up to ~90°) if you want the phone to show more of the panorama without requiring as much physical turning, but note that too large a FOV can introduce distortion at the very edges of the phone's view.

- **Material and Texture:** Apply the video as a texture on the curved mesh using an unlit material. In Three.js, a `THREE.VideoTexture` for the video and a `THREE.MeshBasicMaterial` (or `MeshStandardMaterial` with emissive) works well. This ensures the video isn't affected by lighting or shading – it should appear exactly as shot. If the video source is equirectangular (common for 360/180 videos), Three.js has loaders/shaders (like using `scene.background` with a `THREE.TextureLoader` for equirect panoramas) which essentially map it onto a sphere correctly. For a custom geometry like a cylinder, make sure the UV mapping corresponds to how the video should be projected. Typically, equirectangular video can also map onto a cylinder by using UVs where *u* covers the horizontal angle and *v* covers the vertical (you may only use the middle portion of the equirect texture if you're ignoring top/bottom). The key is that straight lines in the video (e.g. a horizon) appear as straight or gently curving lines on the display, not wavy. If you see warping, adjust the projection or use a different geometry (for example, a common trick is to use a very large sphere and just rotate the camera – the large radius means the portion of sphere you see is almost planar, reducing distortion).

- **Clipping and Distance:** Set the camera's near clipping plane low enough (e.g. 0.1) so that it doesn't clip the nearest parts of the curved screen. If your curved screen's radius is small, the edges might come close to the camera when rotated – ensure they don't get sliced by the frustum. Often, developers will use a fairly **large radius for the curved screen** (e.g. radius 5 or 10 in arbitrary units) which makes it easier to keep the entire thing in front of the camera's near plane. As noted, the radius doesn't change the perspective from the center, but a larger radius + correspondingly larger screen ensures the content is always a bit "farther" in virtual terms, which can minimize perspective distortion. Essentially, it simulates a big curved cinema screen a few meters away rather than a tiny curve right in your face.

- **Controls:** Implement device orientation controls for mobile and orbit controls (with only yaw/pitch) for desktop. Three.js doesn't have built-in magic window by default, but you can use the DeviceOrientationControls to tie the camera's rotation to the phone's gyroscope. When doing so, remember to **lock the orientation** to a single axis if needed (for example, if you want to restrict looking behind, you can clamp the yaw rotation value between –60° and +60°). You might also implement a touch-drag fallback (so users can drag the view if they don't want to physically turn). For desktop, libraries like `OrbitControls` can be configured to disable zoom and roll, giving a smooth click-and-drag to look around. These controls should be tuned to be responsive but not too sensitive – a slight movement of the phone should correspond to a proportional slight pan in the video, to avoid any lag or overshoot that could cause discomfort.

By following these guidelines – using a **partial curved screen (~90–120° concave segment)** placed at a comfortable distance, with the correct projection mapping and sensible camera settings – you can deliver an immersive "magic window" video experience that pulls the viewer in while **avoiding nausea or confusion**. The user will feel as if they are looking into a window of a 3D world with real depth and wrap-around scenery, yet they'll remain comfortable and oriented, without the need for a headset. This approach has been validated both by industry practice (e.g. VR180 videos for front-facing immersion [3]) and by human-factors research (users prefer wide concave screens over fully spherical ones for clarity [4]). The result should be a compelling immersive mode for your OTT platform that enhances video content with spatial depth, all while remaining easy and enjoyable to watch on a mobile device.

Sources: The recommendations above are informed by VR/XR best practices and research. For instance, Meta's creator guidelines note that 180° hemispherical video can be "convincingly immersive" without imagery behind the viewer [3]. Human-factors studies on curved vs. spherical displays show higher user preference for partial concave screens due to better clarity [4]. It's also known that excessively large fields of view can increase motion sickness, so limiting the FOV can improve comfort [7]. Additionally, proper projection mapping (e.g. equirectangular to sphere) is crucial to maintain image fidelity in immersive displays [8]. A cylindrical projection preserves vertical object proportions (helpful for text legibility on a curved screen) [5]. These insights, combined with practical Three.js techniques [6], should help in building a well-balanced immersive video mode. Enjoy creating your magic window!

---

[1] Implementing "magic window" in Unity | Google VR | Google for Developers
https://developers.google.com/vr/develop/unity/guides/magic-window

2 The equirectangular projection of a sphere to a flat plane and the... | Download Scientific Diagram
https://www.researchgate.net/figure/The-equirectangular-projection-of-a-sphere-to-a-flat-plane-and-the-distortion-of-a-cell_fig5_329494073

3 Immersive Media Formats | Getting Started | Meta Quest for Creators
https://creator.oculus.com/getting-started/immersive-media-formats/

4 Comparing immersiveness and perceptibility of spherical and curved displays - PubMed
https://pubmed.ncbi.nlm.nih.gov/32961464/

5 (PDF) Evaluation of 360° Image Projection Formats; Comparing Format Conversion Distortion Using Objective Quality Metrics
https://www.researchgate.net/publication/353715626_Evaluation_of_360_Image_Projection_Formats_Comparing_Format_Conversion_Distortion_Using_Objective_Quality_Metrics

6 Simple curved plane - Questions - three.js forum
https://discourse.threejs.org/t/simple-curved-plane/26647

7 Towards benchmarking VR sickness: A novel methodological ...
https://www.sciencedirect.com/science/article/pii/S0141938224001719

8 WWDC 25 Deep Dive: Immersive Video & OpenImmersive 1.4 | by Anthony Maës | Jun, 2025 | Medium
https://medium.com/@portemantho/wwdc-25-deep-dive-immersive-video-openimmersive-1-4-3f7fe6054e96