Step-by-Step Transformation Explanation

1. **From Clip Space to Camera Space**: The quad's vertices are initially defined in clip space with coordinates ranging from $(-1, -1, 0.999, 1)$ to $(1, 1, 0.999, 1)$. To convert these clip coordinates to camera space, we need to apply the inverse of the projection matrix, M_{proj}^{-1}. This step effectively undoes the projection transformation, moving us from the normalized clip space back to camera space, where directions relative to the camera can be calculated.
2. **From Camera Space to World Space**: After obtaining the coordinates in camera space, we need to orient them in world space, which involves removing any translation effects so that only the rotational component of the view matrix is applied. This allows for the transformation of these coordinates to match the world directions. Specifically, we use the inverse of the rotational part of the view matrix (often denoted as R_{view}^{-1}), disregarding the translation part since we are working with directions rather than positions.
3. **Resulting Matrix M_{tex}**: Combining these two transformations, M_{tex} is constructed as:

   $$M_{tex} = R_{view}^{-1} \cdot M_{proj}^{-1}$$

   Applying M_{tex} to each vertex of the background quad in the vertex shader effectively transforms its clip space coordinates directly into world space directions.

Purpose of M_{tex}

For the sphere, M_{tex} is simply the identity matrix since texture coordinates directly correlate to the object's surface in world space. However, for the background quad, this transformation enables us to treat its vertices as world directions rather than specific locations, allowing the cube map to be sampled as if it were an infinitely distant background. This approach simulates a skybox effect, giving the illusion of an environment surrounding the entire scene.