**Appendix**

**1 Development Environment & Tools**

This project was developed in a Google Colaboratory ("Colab") environment using **Python 3**. Key libraries included **PyTorch** and **PyTorch3D** for 3D rendering and tensor operations, **OpenCV** for core computer vision tasks, and **NumPy** with **Matplotlib** for numerical computation and visualization.

**2 Key Algorithms**

* **Camera Pose Estimation**: The camera's position and orientation were estimated using OpenCV's solvePnP algorithm. This process involved mapping manually identified 2D pixel coordinates of a planar object's corners to their known 3D real-world measurements. A simplified intrinsic camera matrix was assumed to solve for the camera's rotation (R) and translation (T) vectors, which form the extrinsic parameters.
* **3D Rendering & Compositing**: A synthetic 3D teapot was rendered using PyTorch3D. The estimated camera pose, after being converted from OpenCV's coordinate system to PyTorch3D's, was used to configure a virtual camera. The 3D mesh was rendered with a simple lighting model (SoftPhongShader), and the resulting RGBA image was composited onto the original photograph using its alpha channel as a mask to create the final augmented reality effect.

**Role of AI in Assignment Development**

Google's AI assistant, Gemini, was utilized as a development partner to accelerate learning and overcome technical challenges. The AI's role was not to generate the final solution, but to assist in the following ways:

* **Code Scaffolding**: Generated boilerplate code for tasks like setting up the PyTorch3D renderer and loading data, which was then adapted for the project's specific requirements.
* **Debugging & Error Resolution**: Diagnosed and provided solutions for runtime errors, such as NameError from out-of-order notebook execution and OpenCV assertion failures from incorrect function inputs.
* **Conceptual Clarification**: Explained core concepts, including the critical differences between OpenCV and PyTorch3D coordinate systems, which was essential for correct implementation.