TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PASHCHIMANCHAL CAMPUS

LAMACHAUR, POKHARA

[Subject Code: EX 707]

A FINAL REPORT

ON

ADVANCEMENTS IN TEXTURE EXTRACTION AND DEPTH ESTIMATION :TECHNIQUES AND APPLICATION

SUBMITTED BY:

AVISHEK POUDEL	[WRC077BEI009]
BINAYAK SHRESTHA	[WRC077BEI012]
PRAKANDA BHANDARI	[WRC077BEI030]
PRATHAM ADHIAKRI	[WRC077BEI032]

SUBMITTED TO:

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

May, 2025

COPYRIGHT

LETTER OF APPROVAL

The undersigned certify that they have read, recommended to the Institute of Engineering for acceptance, a project report entitled **"ADVANCEMENTS IN TEXTURE EXTRACTION AND DEPTH ESTIMATION: TECHNIQUES AND APPLICATION"** submitted by **Avishek Poudel, Binayak Shrestha, Prakanda Bhandari and Pratham Adhikari** in partial fulfillment of the requirements for the Bachelor's degree in Electronics, Communication and Information Engineering.

Supervisor, Asst. Prof. **Er. Khem Raj Koirala**
Department of Electronics and Computer Engineering
Institute of Engineering, Pashchimanchal Campus

External Examiner: **Er. Ram Chandra Pokhrel**
Senior Engineer
Nepal Telecom
Provincial Office Pokhara
Gandaki Province, Kaski

Coordinator, Asst. Prof. **Er. Khem Raj Koirala**
Head of Department
Department of Electronics and Computer Engineering
Institute of Engineering, Pashchimanchal Campus

DATE OF APPROVAL: 21 May, 2025

DECLARATION

We hereby declare that this final year project entitled "AI-Driven Texture and Depth Estimation from PBR Images" is the result of our original research and study. We conducted this project under the guidance of our supervisor **Asst. Prof. Khem Raj Koirala**, which is being submitted to the Department of Electronics and Computer Engineering, IOE, Pashchimanchal Campus. The materials contained in this report have not been submitted to any University or Institution for the award of any degree and we are the only author of this complete work and no sources other than those listed here have been used in this work. We assert that this project represents our own independent work and that we have not used any unauthorized sources or materials in its completion. Any references, quotations, or ideas borrowed from the work of others have been properly cited and acknowledged within the project.

AVISHEK POUDEL	[WRC077BEI009]
BINAYAK SHRESTHA	[WRC077BEI012]
PRAKANDA BHANDARI	[WRC077BEI030]
PRATHAM ADHIAKRI	[WRC077BEI032]

ACKNOWLEDGEMENT

Firstly, we would like to thank the Institute of Engineering (IOE) for the inclusion of this Major Project on the syllabus for the course Bachelor in Electronics, Communication and Information Engineering. Also, we would like to thank the Department of Electronics and Computer Engineering, Pashchimanchal Campus for providing us with the proper guidance and a wonderful learning atmosphere. We would also like to thank our supervisor Asst. Prof. Khem Raj Koirala for providing us required guidance and his precious time for us to guide and make our project better. We are extremely thankful to Asst. Prof. Khem Raj Koirala, Head of Department of Electronics and Computer Engineering for all kinds of possible help and advice during the course of this work. We would also like to express our gratitude towards our parents and our friends for their kind cooperation and encouragement which help us a lot in completing this project. The experience of doing this project will surely enrich our technical and teamwork skills to a great extent.

Finally, we would also like to thank our friends who have helped us with their valuable suggestions and guidance which has been very helpful in various phases of the completion of the project.

With warm regards,

AVISHEK POUDEL (WRC077BEI009)
BINAYAK SHRESTHA (WRC077BEI012)
PRAKANDA BHANDARI (WRC077BEI030)
PRATHAM ADHIKARI (WRC077BEI032)

ABSTRACT

Accurate depth estimation is a fundamental challenge in computer vision with significant implications for autonomous navigation, augmented reality, and robotics. Traditional depth sensing methods using stereo cameras or LIDAR systems, although effective, are often cost-prohibitive and complex. This proposal outlines the development of a cost-effective, real-time depth estimation system using RGB images.The core of the system involves capturing RGB images through a camera module interfaced with the Raspberry Pi. Advanced deep learning algorithms will be employed to process these images and estimate depth, utilizing the Raspberry Pi's computational power. The Arduino Nano will control a Visual cue generator and auxiliary components such as a flashlight for illumination in low-light conditions and an IR light to enhance depth estimation accuracy by providing additional visual cues.By combining low-cost hardware with sophisticated depth estimation algorithms, this project aims to provide an accessible alternative to traditional depth sensing technologies. The proposed system's ability to deliver accurate, real-time depth information using a single RGB camera makes it a valuable tool for various applications, from simple obstacle avoidance in robotics to complex interactive environments in augmented reality.

Keywords: depth, normal, roughness, transformer, visual cues, blender

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

PBR	*Physically Based Rendering*
CRT	*Convolutional Regression Tree*
CNN	*Convolutional Neural Network*
CRF	*Conditional Random Field*
DC	*Direct Current*
DORN	*Deep Ordinal Regression Network*
DPT	*Dense Prediction Transformer*
GHz	*Giga Hertz*
GPIO	*General Purpose Input/Output*
HDMI	*High-Definition Multimedia Interface*
HTML	*Hypertext Markup Language*
IDE	*Integrated Development Environment*
I2C	*Inter-Integrated Circuit*
JSX	*JavaScript XML*
KB	*Kilo Byte*
LIDAR	*Light Detection and Ranging*
NRF	*Neural Regression Forest*
RGB	*Red Green Blue*
SID	*Spacing Increasing Discretization*
SPI	*Serial Peripheral Interface*
TOF	*Time of Flight*
UART	*Universal Asynchronous Receiver/Transmitter*
UI	*User Interface*
USB	*Universal Serial Bus*
URL	*Universal Resource Locator*

CHAPTER 1

INTRODUCTION

The increasing demand for realistic three-dimensional modeling and immersive digital environments has spurred significant advancements in computer vision and rendering techniques.The critical component for achieving photo-realistic graphics is texture extraction and in texture extraction, Depth estimation is the most critical problem.
Depth maps are typically single-channel images in which the depth (distance) from the camera is represented by the intensity of each pixel. They usually provide roughly 0.3% precision in comparison to the entire depth range because they are stored in a single channel and are constrained by sensor precision.

Normal maps use three channels (usually in a 24 bit RGB format) to encode the orientation of each surface element (the "normal" vector) rather than just providing distance. They can depict extremely subtle changes in surface orientation, including details that a depth map might overlook, thanks to their approximately 16.7 million possible color combinations. This makes a 3D model's shading and light interaction appear far more detailed without adding to the geometric complexity of the model itself.

Micro-scale surface details that are too fine for the camera's resolution are captured by roughness maps. A roughness map shows the microscopic smoothness or roughness of a surface, which affects how light is reflected or scattered. Since it determines the specular response or how "shiny" or "matte" a surface appears—this information is essential for rendering realistic materials.

These maps complements each other: the roughness map records subtle, unresolved surface variations that impact the material's appearance in light, the normal map adds high-frequency directional detail, and the depth map provides the coarse geometry.

1.1 Background

Human with two eyes can effortlessly find depth of object.But implementing the capability in a computer is a challenging task.Traditional approach used time-of-flight

system or active sensors like LIDAR. LIDAR and TOF sensors use light or radiation to strike environment object and calculates total time to return to the sensor, yeilding precise distance measurement. Despite their preciseness and effectiveness, these systems are complex, costly and high power consumption, which have restricted them in many practical scenarios. Texture is generally defined as the feel, appearance, or consistency of a surface or substance. Color, reflectance, and microstructure are three major contributor of texture to any materials.Extracting such microstructure and reflectance has possibility to highly enhance visual fidelity of rendered images and also more effective material analysis and simulation.

The history of depth estimation goes back to 1830's when Charles Wheatshone introduced the stereoscope.By presenting images at slightly different angle, he was able to trick brain to combine them into 3d perception. Later Sir David Brewster was able to refine the concept to stereo photography and it was foundational framework for photogrammetry. In the mid 20th century,Random dot stereograms emerged with key contributions from Béla Julesz , demonstrating that depth can be perceived even when the scene lacks recognizable objects. When computing capacity of computer exploded in 2000's ,various algorithms were developed that could automatically match corresponding points between stereo images and find depth with high accuracy. At the same time, Structure form motion (SFM) techniques were also introduced to further refine depth estimation. With the introduction of ML and AI and presence of large amount of dataset for machine learning task, researchers were able to train model with unsupervised learning (without explicit ground truth) between stereo images. In the past few years, Hybrid VIT architectures were emerged which shows significant capability where model were able to generalize enough where depth can be accurately estimated from just one camera and the emerging field was known as Monocular Depth estimation.

1.2 Problem Statement

Accurate depth estimation is a fundamental requirement for a wide range of applications in computer vision and robotics. Traditional depth estimation methods predominantly rely on TOF or LIDAR sensors, which provide reliable depth measurements but come with significant drawbacks including high cost, substantial power consumption, and

increased system complexity. Depth estimation, which infers depth from RGB image, presents a more accessible and cost-effective alternative. The approach must infer from visual cues such as texture, shading, perspective, and motion, which can be ambiguous and complex to interpret.

1.3 Objectives

The objectives of this project is:

- To design and develop an accurate depth estimation system using RGB images.
- To use depth estimation to accurately extract texture from images.

1.4 Application

The major application of the project are:

- Enhances the ability of robots to understand and navigate human environments safely and effectively.
- Allows systems to detect and avoid obstacles in real-time, preventing collisions and enhancing navigation.
- Provides accurate 3D maps of the environment, crucial for navigation and situational awareness.
- Improved depth perception in AR/VR system and Realistic object interaction and manipulation and enhances immersive experiences by providing accurate spatial information and enabling precise tracking of virtual objects.
- Enables vehicles to accurately perceive and navigate their surroundings, improving safety and efficiency.
- For 3D reconstructions and modeling, it facilitates the creation of detailed and accurate 3D models for various applications.

- Enhances medical imaging by surgical planning, navigation and diagnosis by providing detailed 3D views of anatomical structures, aiding in precise surgical planning and diagnosis.
- Enables the creation of customized and well-fitted prosthetic and orthotic devices through accurate 3D modeling.
- Enhances the realism and interactivity of games by providing accurate depth information for virtual environments.
- In photography and cinematography, depth maps can improve the quality of images and videos by enabling advanced effects and accurate focus.
- Security and surveillance with enhanced facial recognition and biometrics
- Enhances the efficiency and precision of automated systems in manufacturing, industrial automation and other industrial applications.
- In telepresence and teleconferencing, it can provide a more immersive and interactive experience by accurately capturing and rendering 3D environments.
- Assists in creating accurate 3D reconstructions of crime scenes, aiding in investigations and analysis.

CHAPTER 2

LITERATURE REVIEW

Depth estimation from images has been a central problem in computer vision after it's inception, driving the need to understand three-dimensional scene structure from two-dimensional data [1]. Early research mainly focused on traditional methods such as shape-from-shading, which exploits variations in image intensity in order to infer surface geometry under assumed illumination models [1, 2]. Complementary techniques such as shape-from-texture drew on variations in surface texture to infer depth cues and provide additional ground for later advancements[3]. In addition, structure from motion and SFM methods exploited multi-view geometry to recover depth by tracking correspondences on features across image sequences, thus laying the foundations for 3D reconstruction in dynamic scenes[4, 5].

The development of deep learning has changed the face of depth estimation in that now models can learn complex, hierarchical features directly from raw images[6]. Eigen et al. initiated this change with a two-stage deep convolutional network accordingly: coarse global prediction and then having a finer resolution in the local context in order to give a high-resolution depth map[6]. Then, inspired by this work came several other binocular depth estimations using Siamese network architectures with 3D convolutional layers to jointly learn the global context along with parallax information to arrive at a low-resolution depth estimate which is subsequently upsampled to full resolution[7, 8, 9]. Roy et al. proposed a hybrid architecture which combines CNN with regression forests, in which several small CNNs compute the local depth map predictions that have their own loss function, which when combined together, yield smoother and more robust depth maps. Further, some of the unsupervised learning methods arise as a means of a solution for challenges related to the collection of ground-truth depth data, converting depth estimation to a problem related to image reconstruction [10].

Unsupervised learning methods have also emerged to overcome the challenges associated with collecting ground-truth depth data, reframing depth estimation as an image reconstruction problem [11, 12]. Godard et al. showed that enforcing left-right con-

sistency between stereo views in a fully differentiable framework allows monocular depth estimation with competitive performances without any explicit supervision [11]. Further down the line, some improved unsupervised methods encompassed the use of minimum reprojection losses and auto-masking techniques, which have greatly aided in bolstering these models against occlusions and dynamic scenes [13]. At the same time, researchers have started looking at synthetic-data-based training of models for durability to environmental variations such as rainy, nighttime, and fog conditions that open up the subsequent depth estimation discussed in the more challenging real-world scenarios [14].

Recent advancements include repackaging depth estimation into an ordinal regression framework. Continuous depth values are mapped into ordered bins levelwise to encourage more stable training and faster convergence [15]. In survey studies, current developments are compared with supervised, weakly-supervised, and unsupervised methods, featuring improvements in network architectures, loss functions, and data augmentation methods propelling state-of-the-art technologies in autodyne [16]. Lightweight encoder–decoder architectures and network pruning techniques have achieved accurate depth estimation even on computations constrained embedded platforms in commercial applications [17]. Other strategies for depth sensing have been investigated too; mechanisms using structured illumination apply optimized projection patterns together with the analysis of motion blur, known as light flow, to derive depth measures and offer additional solutions in cases where passive imaging may not suffice [18, 19, 20].

The progress of quality datasets has been of utter importance in the development of depth estimation research. For instance, datasets like MatSynth offer large collections of high-resolution, physically based rendered materials so that researchers can build up synthetic data and look at the problem of depth estimation under well-controlled variations of lighting and texture [21]. Through the Metric3D v2 and other similar works, canonical camera transformations integrated with joint depth-normal optimization have achieved state-of-the-art results across NYUv2, KITTI, and other benchmarks, thereby providing further evidences of depth estimation applicability for a variety of domains ranging from monocular SLAM to robotics [22]. In rather more recent developments, DreamMat builds upon already extraordinary expectations by exploiting environmental lights and inverse rendering via Monte Carlo sampling to produce photorealistic ma-

terial images, thus ensuring some form of continuity between material synthesis and precise depth reconstruction [23].

To sum up, the history of depth estimation research unfolds on the transition between handcrafted feature-based approaches-such as shape-from-shading and structure from motion-to datadriven, deep-learning techniques learning how to extract and fuse multi-scale cues straight from pictures [6, 10, 14]. Such deep learning models make use of extremely huge training sets combined with advanced architecture to yield dense and accurate depth over both supervised and unsupervised settings [11, 15]. Yet the remaining challenges-generalizing over different environments, occlusions, and real-time performance-are not far behind or too despised, with the rapid strides towards network designs, loss function formulation, or curation of datasets further developing depth estimation that remains quite robust and more adaptable [16, 17]. Many more may benefit directly from very accurate 3D reconstructions in autonomous driving, robotics, and augmented reality-as well as inspire future research that integrates other modalities and tackle issues of enhancing efficiency and accuracy beyond just that[18, 22, 23].

CHAPTER 3

RELATED THEORY

3.1 Hardware

3.1.1 Raspberry Pi 4 Model B

The Raspberry Pi is a credit card-sized single-board computer that provides a powerful processing platform in a compact form factor. It is equipped with a range of input/output interfaces, including USB ports, HDMI, Ethernet, GPIO, and a camera interface. The processor speed of Raspberry Pi 4 Model B is 1.5 GHz. It comes with onboard wireless networking and Bluetooth .

3.1.2 Polarizer

It is an optical filter that which allows specific orientation of light to pass through.Polarization have been used in photography and display devices such as LCD technology. In photography, Polarizing filter is used to filter out reflections which gives a crisp,sharp and beautiful images.There are mainly two type of polarizers, linear and circular polarizers.[24]

3.1.3 Circular Light

Lighting systems that have light sources positioned in a circle around the camera lens or target object are commonly referred to as circular light systems. These systems are frequently implemented as ring lights. This design reduces the appearance of shadows by achieving a very uniform and diffused illumination. Any possible shadows cast by one part of the object are filled in by light from the other side because the light is evenly distributed from all directions. Because the subject is uniformly lit from the same angle that the camera views it when the ring light is positioned coaxially with the camera, this type of lighting is frequently referred to as "shadowless". This is particularly valuable in machine vision, where even illumination is essential for capturing fine details without harsh contrasts.

3.2 Software

3.2.1 PyTorch

PyTorch is an open source machine learning framework based on the Python programming language and the Torch library. Torch is an open source machine learning library used for creating deep neural networks and is written in the Lua scripting language. It's one of the preferred platforms for deep learning research. The framework is built to speed up the process between research prototyping and deployment.

3.2.2 Flask

Flask is a lightweight and easy-to-use micro web framework for Python, designed to facilitate quick web development while being flexible enough for complex applications. As a micro framework, Flask provides the essential tools needed to build web applications without mandating specific libraries or components, focusing on simplicity and ease of use. Its flexibility allows developers to choose and integrate various libraries and tools, making it highly customizable. Key features of Flask include a built-in development server and debugger, which streamline testing and debugging processes. It uses Jinja2 as its templating engine, enabling the creation of dynamic HTML pages with powerful features. Flask's intuitive URL routing system maps URLs to Python functions, simplifying endpoint management. [25]

3.2.3 React

React is a popular JavaScript library developed by Facebook for building user interfaces, particularly single-page applications. It allows developers to create large web applications that can update and render efficiently in response to data changes. React is component-based, meaning the UI is divided into reusable components, each managing its own state and rendering logic. This modular approach promotes code reusability and easier maintenance.

3.2.4 Blender

Blender is the free and open source 3D creation suite. It supports the entirety of the 3D pipeline—modeling, rigging, animation, simulation, rendering, compositing and motion tracking, even video editing and game creation.Blender has developed Blender's API for Python scripting to customize the application and write specialized tools. Blender is cross-platform and runs equally well on Linux, Windows, and Macintosh computers. Its interface uses OpenGL to provide a consistent experience.

3.3 Visual Cues

3.3.1 Stereo Vision (Binocular Disparity)

Using two images captured from slightly different viewpoints (like human eyes), depth can be estimated by calculating the disparity between corresponding points in the two images. Disparity refers to the difference in the relative position of an object in the two images, which is inversely proportional to the object's distance [26] .

3.3.2 Motion Parallax

Motion parallax is the optical change of the visual field of an observer which results from a change of the observer's viewing position [27]. When an observer or the camera moves, nearby objects appear to move faster across the field of view than distant objects. This relative motion provides cues about the depth and distance of objects [28].

3.3.3 Structured Light

Structured lighting for depth estimation involves projecting a known pattern (like grids or stripes) onto a scene. The pattern's deformation when viewed from a different angle is analyzed to determine the depth and shape of objects.Thus, structured lighting technique is based on projecting a light pattern and viewing the illuminated scene from one or more points of view. By comparing the projected and observed patterns, a depth map is generated [29]. This technique is widely used in 3D scanning, robotics, and computer vision for accurate and detailed depth measurements.

3.3.4 Active Illumination

Active illumination for depth estimation involves projecting light, such as laser or infrared, onto a scene and measuring the reflected light to determine depth [30]. This method provides precise depth information and is commonly used in applications like autonomous vehicles, 3D mapping, and augmented reality.

3.3.5 Texture Gradient

The density of texture patterns changes with distance, textures appear denser and smaller as they get further away. This gradient helps infer the relative depth of surfaces and objects in the image.

3.3.6 Occlusion (Interposition)

When one object overlaps or covers another, the occluded object is perceived to be farther away. This cue provides relative depth information between overlapping objects. Shading and Shadows:

Variations in light and shadow on objects provide information about their shape and depth. The position and length of shadows also help in estimating the spatial relationship and distance of objects.

3.3.7 Aerial Perspective (Atmospheric Perspective)

Distant objects often appear hazier and less distinct due to atmospheric scattering of light. This effect can be used to gauge the depth of objects based on their clarity and color saturation.

3.3.8 Relative Size

When the size of familiar objects is known, their relative sizes in the image can be used to infer their distances. Larger objects are perceived as closer, while smaller ones are seen as farther away.

3.3.9 Known Object Size (Size Constancy)

Using the actual size of known objects to infer depth. If an object appears smaller than its known size, it is inferred to be further away.

3.3.10 Linear Perspective

Parallel lines appear to converge as they recede into the distance, meeting at a vanishing point. This convergence provides cues about the depth and distance of objects along those lines.

3.3.11 Defocus Blur (Depth from Defocus)

Objects at different distances from the camera have different amounts of blur due to the depth of field effect. The amount of defocus blur can be used to estimate the distance of objects from the camera.

3.4 PBR Maps

3.4.1 Base Color (Albedo) Map:

The base color map is the fundamental layer of your material's appearance—it's like the natural skin of your object. This map contains the raw color information without any lighting or shading baked in, so it tells you what the surface would look like under neutral, evenly distributed light. Think of it as the canvas on which all the other visual details are built.

3.4.2 Depth map

A depth map is basically a grayscale picture in which the distance between a specific point on the surface and the observer is indicated by each shade. Darker tones depict regions that are farther away, while lighter shades show regions that are closer.This map enables digital systems to decipher and replicate the changes in distance in more coarse

form, across the surface. It gives object a sense of volume and dimension that improves the scene's overall realism, much like a topographical map would.

3.4.3 Metallic Map

The metallic map is what tells the renderer which parts of material should behave like metal. Whiter the map is the object reflects light sharply and often carrying the characteristic bright, mirror-like highlights, while black areas behave like non-metals, diffusing light more gently. In essence, it's a simple guide to differentiate between shiny, reflective surfaces and dull, matte ones.

3.4.4 Roughness Map

Let's imagine, running our hand over a surface,the roughness map captures that tactile feel visually. It controls how smooth or rough the material appears by dictating the spread and clarity of reflections and how microstructre governs the surface. Low roughness values yield smooth, glossy surfaces with crisp reflections, whereas high roughness values create a more matte, diffused look, mimicking surfaces like brushed metal or rough stone.

3.4.5 Normal Map

The normal map is a kind of shortcut to adding depth and detail without increasing model's complexity. It encodes tiny bumps, dents, and textures into a colorful image, tricking light into interacting with the surface as if those details were physically present. It's like giving model a finely textured skin that comes alive under varying lighting conditions.

3.4.6 Ambient Occlusion (AO) Map

The ambient occlusion map adds that extra bit of realism by simulating how ambient light naturally finds it hard to reach crevices and corners. It subtly darkens those areas, enhancing the sense of depth and grounding object in its environment. This map is

like the gentle shadow that emphasizes the intricacies of material, making it look more lifelike and three-dimensional.

CHAPTER 4

METHODOLOGY

The project follows an Iterative Development Process, allowing for continuous refinement and adaptation based on learning from each phase of the project. Different statistical and mathematical tools are used to establish clear decision points throughout the project lifecycle and define key artifacts (both input and output) to guide the project's progression and ensure alignment with project goals.

4.1 Preliminary study system Design

4.1.1 Hardware design

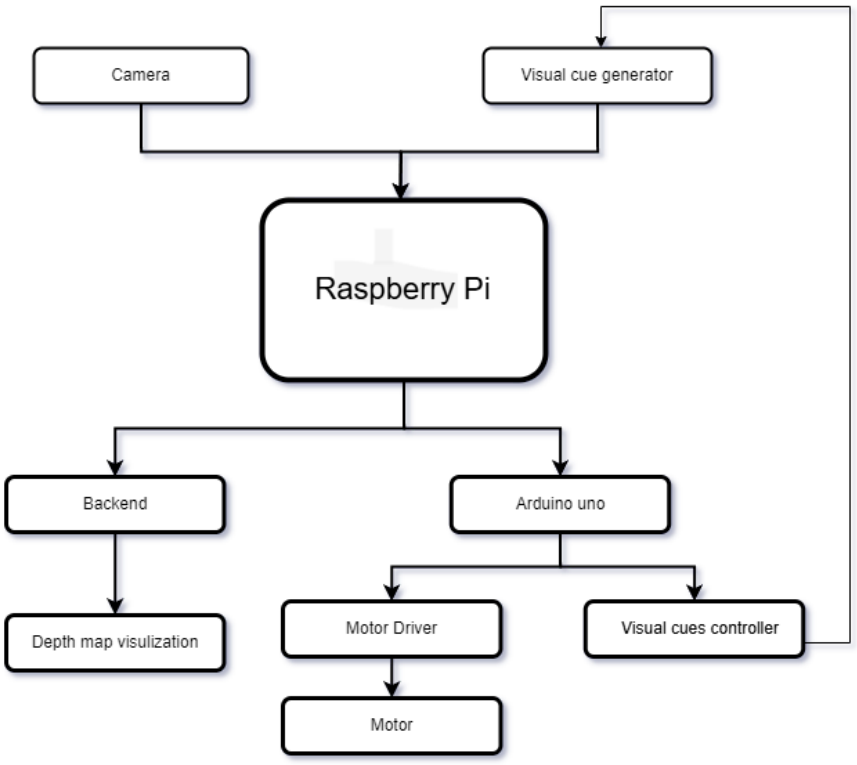

Figure 4.1: Hardware block diagram

The project uses a Raspberry Pi as the brain.Visual Cue generator generates cues that help in understanding how far away objects are. An Arduino Uno microcontroller steps in to control the Visual cue generator, and motor driver, which in turn powers a mo-

tor.The Raspberry Pi processes images captured by camera and sends them to the back-end through Wi-Fi for visualization. The software for this project tackles the video

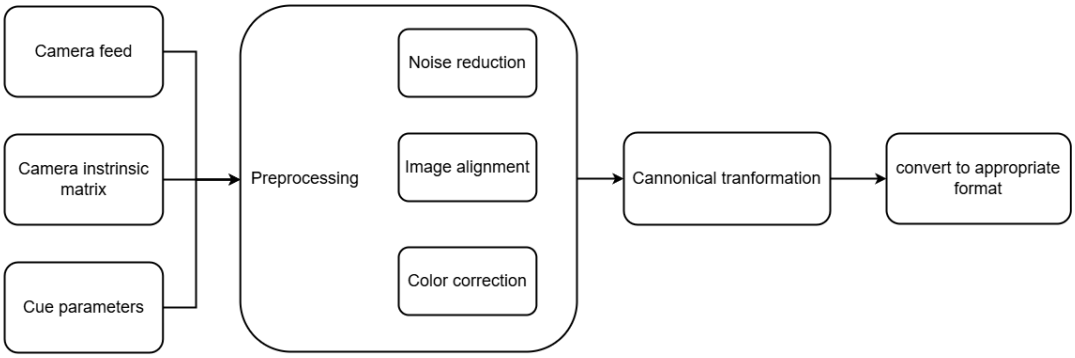

Figure 4.2: Data pipeline

feed from the camera and turns it into a depth map showing how far away things are. First, the program cleans up the video, getting rid of any noise. If multiple pictures are needed, the software align them up perfectly. Then the model figures out depth from the video.This might involve comparing slightly different pictures. Once it has depth information, the program combines it back with the original color picture to create a special image with both color and depth data. There might be some extra steps to fine-tune the picture and account for the camera's quirks.

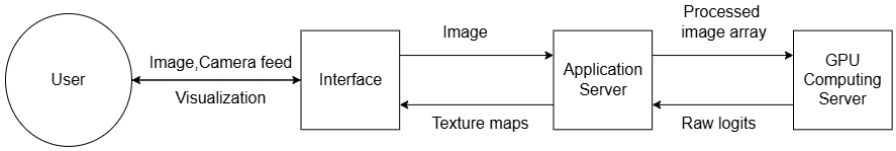

Figure 4.3: Software block diagram

4.2 Dataset

4.2.1 Available dataset

All available dataset were throughly searched and listed.EDA was performed in short-listed dataset where we choose the best dataset for training.Hypersim[31] and ARK-itScenes dataset[32] were found to have best quality of depth information.For texture extraction, Only Matsynth[21] dataset was available with all required texture maps for our purpose.

4.2.2 Synthetic Dataset

We found lack of dataset for normal,roughness and metallic maps.To solve the problem we opted for a method where we first pretrain our model with synthetic data and then finetune on high quality data. For generation of roughness, we searched for existing method for texture extraction from images where we chosed 2 methods according to its quality and computation required and three methods are compared to select the best method for generation of synthetic roughness dataset.

Wavelet analysis

Wavelet analysis involves breaking an image into smaller components, called wavelets, which are localized in both time (or space) and frequency. This method allows us to analyze an image at multiple scales and resolutions. Wavelets are particularly useful for capturing fine details, such as edges and textures. By decomposing an image into wavelet coefficients, we were able to highlight important features while reducing noise.

Local Binary Pattern (LBP) Transform

The Local Binary Pattern (LBP) transform is a widely used method for texture analysis and feature extraction in image processing. It is simple, computationally efficient, and highly effective for capturing local texture information. The LBP transform works by comparing the intensity of a central pixel in a local neighborhood with its surrounding pixels. For each surrounding pixel:

◦ If the intensity of the neighboring pixel is greater than or equal to the central pixel, it is assigned a value of 1.

◦ Otherwise, 0 is assigned.

These binary values form a binary pattern (usually an 8-bit binary number for a 3×3 neighborhood), which is then converted into a decimal value to represent the texture feature.

4.2.3 Custom Dataset

The combination of synthetic, Matsynth, and custom datasets ensured the model was trained on a comprehensive and diverse data pool. This holistic approach enhanced the model's ability to perform depth estimation accurately in real-world applications. We created our own dataset for further finetuning of our model.From the preliminary study, we found methods to make depth maps,metallic and roughness map.

Depth map

Camera triangulation was used to create depth map. It is a fundamental technique in stereo vision where depth of object is estimated from image of slightly different viewpoint.This method relies on two or more camera and calculates depth using geometry.
In a stereo camera setup, two camera are seperated by a known distance defined as baseline. Any object in the image appears in slightly different position when compared to two images due to parallax.Now, the difference can be measured in the position (disparity), from which we can calculate depth using triangulation.

Roughness map

Roughness map can be extracted form cross polarization of image. when light is reflected from the surface, the reflected light gets partially cross polarized.Two polarizing filters are taken, one is used in light source and another in front of the camera. When the angle between two polarization is perpendicular all glossy reflection are removed allowing diffuse reflection which are depolarized by roughness surface. Rough surfaces cause multiple scattering events, randomizing light polarization, the intensity of the light passing through cross polarized filter increases with surface roughness.so, High intensity in cross polarized images correlates with greater roughness. Smooth areas appear darker while rough areas appear brighter.
Roughness is extracted by normalizing cross polarized intensity with co-polarized image. This method also extracts microstructure details without any color dependency when cross-polarized image and co-polarized image are subtracted from each other. Also spatial variation of pixel intensities indicate rougher surface on local level.

Metallic map

Metallic map can also be generated through cross polarization.Metal produce high amount of specular reflection such reflection are filtered by polarizers.By finding high dip in intensity,metallic objects can be segmented from the rest using a binary mask to outline the object , in this way metallic object in the images can be mapped out in form of binary image.

Albedo map

A cross-polarized uniformly lighted image is taken as an albedo map.Such map has no shadows or specular reflection and is composed of just color of the object.

4.2.4 Blender-created dataset

3d modelers and 3d scanners have created millions of 3d objects. Such objects are collected in a single repository for research purpose.[33] something et al have created a repository of over 10 million 3d objects.From the repository, we separated 2.5 million objects usable for model training. Every object was then taken from 12 different viewpoints and RGB, albedo, depth, normal and metallic images are obtained by rendering graphics from blender.

4.3 Model Architecture

Architecture was designed so that single model can be used to generate all maps so it will be one step inference for all maps. A fully covolutional hybrid vision transformer was used as framework for our model. For encoder we used DINOv2 as encoder as it was state-of-the-art encoder[34].For decoder, we used RaftDepthNormalDPT5 for creation of depth and normal map. features form divov2 encoder was tested to create roughness map, but high details features was not preserved due to dimension loss so we opted for extracted high details feature form grayscale RGB images and concatenate with encoder to create roughness map.We found that, if both depth and ambient occlusion map was generated independently , the information collided so we complemented

these two map where generated depth map is concatenated to features with grayscale image to generate AO map. for loss function we tried ,L1 loss and L2 loss which didn't

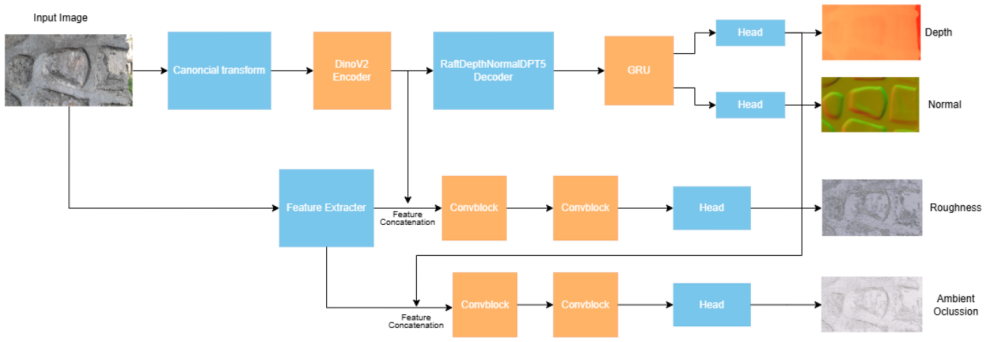

Figure 4.4: Model Architecture

give promising result.Balanced Huber loss was used for this purpose. The Huber loss is more robust loss function which behaves quaratically for small errors and linearly for large errors. The standard formula for Huber loss is

$$L_\delta(a) = \begin{cases} \frac{1}{2}a^2, & \text{if } |a| \leq \delta, \\ \delta\left(|a| - \frac{1}{2}\delta\right), & \text{if } |a| > \delta, \end{cases}$$

where $a = y - f(x)$ is the difference between ground truth y and prediction f(x) and δ is a threshold which controls when to transition between quadratic and linear response. Balanced Huber loss is modified form to handle imbalanced data by introducing extra weighting factor, which is often denoted by α to adjust the loss contribution from different errors ranges.Below is formulation of Balanced Huber loss.

$$L_{\delta,\alpha}(a) = \begin{cases} \alpha\,\frac{1}{2}a^2, & \text{if } |a| \leq \delta, \\ \alpha\,\delta\left(|a| - \frac{1}{2}\delta\right), & \text{if } |a| > \delta. \end{cases}$$

Here, α typically ranges from values between 0 and 1 and can be tuned based on imbalance in the data. We also applied gradient loss and made new loss function form combining different loss function. Gradient loss function was used to minimize errors from edges of object present in the image where camera triangulation was not accurate. AdamW optimizer was used as optimizer to train our model with a learning rate of 10^{-3} to 10^{-4} for pretaining and 10^{-6} for fine-tuning. Linear polynomial learning rate scheduler was used with 500 warming steps to train the transformer model.

4.4 Model training Pipeline

All the images on the input to the network should undergo a transformation into a canonical camera space first. With the same input size for all images for a constant focal length, this transformation is able to normalize the images so that any change of camera intrinsics (e.g., differences in focal lengths) would not confuse the network. The ground-truth depth maps correspondingly will have to be transformed into this canonical space as well. Then canonical-space image is fed into a depth-normal estimation network. For canonical transform we define the depth scaling ratio as

$$\omega_d = \frac{f_c}{f},$$

where f is the original focal length and f_c is the canonical focal length. The ground-truth depth D^* is then rescaled to obtain the canonical depth label:

$$D_c = \omega_d \, D^*.$$

Meanwhile, the input image remains unchanged:

$$I_c = I.$$

The image is then normalized using standard image normalization with mean = [0.485, 0.456, 0.406] and standard deviation of [0.229, 0.224, 0.225] Image augmentation was applied to prevent model for overfitting.Below are some of image augmentation used for training.

- Random resize
- Random crop
- Random blur
- Random brightness
- Random noise

The image is passed through Hybrid-VIT model and to recover the metric depth from the prediction D_c (which is in canonical space), a de-canonical transformation is applied:

$$D = \frac{1}{\omega_d} D_c.$$

while other images are just extracted and clipped to range of (0,1) and then scaled to 0-255 as uint8 format for standard image.

CHAPTER 5

RESULT

5.1 Accuracy and loss metrics

5.1.1 Depth Map

The performance of the depth-estimation form model is quantified in terms of standard accuracy and loss metrics, which help understand both the predicted depth values' precision and the error distribution with respect to the ground truth.

1. Accuracy Metric (Delta Thresholds)

Delta metrics quantify the accuracy of the model by measuring the percentage of predictions falling within a particular margin of error from the actual depth values. These thresholds are defined as:

$$\delta_n = \text{percentage of } y_i \text{ such that } \max\left(\frac{\hat{y}_i}{y_i}, \frac{y_i}{\hat{y}_i}\right) < 1.25^n \tag{5.1}$$

where:

- $\hat{y}_i$ is the predicted depth,
- y_i is the ground truth depth,
- n determines the tolerance level for errors.

The results obtained for the delta metrics are as follows:

Metric	Value
δ_1	88.44%
δ_2	98.30%
δ_3	98.31%

Table 5.1: Delta accuracy metrics for depth estimation.

These values indicate that the model achieves **high accuracy**, with nearly all predictions

falling within a reasonable margin of error. The δ_1 score of 88.44% suggests that a majority of predictions are well-aligned with the true depth, while the δ_2 and δ_3 scores exceeding 98% confirm the robustness of the model in maintaining depth consistency across different regions of the image.

2.Loss Metrics

To further analyze the model's performance, we consider three loss functions:

Logarithmic Loss ($\log_{10}$)

$$\frac{1}{N}\sum_{i=1}^{N}|\log_{10}\hat{y}_i - \log_{10} y_i| \tag{5.2}$$

The **log10 loss** was recorded at **0.1812**, indicating a moderate deviation between the predicted and actual depth values when considering a logarithmic scale. This metric is crucial for assessing the model's ability to maintain **global depth consistency**.

3. Root Mean Squared (RMS) Loss

$$\sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2} \tag{5.3}$$

The **RMS loss** value of **0.1407** suggests that, on average, the predicted depth deviates by approximately **14% of the total depth range**. Since RMS loss gives greater weight to large deviations, this value indicates that while most predictions are close to the actual depth, there are some instances where larger discrepancies exist.

4. RMS Logarithmic Loss (RMS_{log})

$$\sqrt{\frac{1}{N}\sum_{i=1}^{N}(\log\hat{y}_i - \log y_i)^2} \tag{5.4}$$

The **RMS_{log} loss** was **0.0776**, which is relatively low, indicating that the model effectively maintains the **relative depth relationships** across the image. Since this loss function operates in log-space, it is less sensitive to absolute depth errors and focuses on preserving the proportional depth structure.

5. Interpretation and Discussion

The evaluation results demonstrate that the model achieves **high accuracy and low error rates**, making it suitable for depth estimation tasks. The δ_1 score (88.44%) suggests that a portion of the predictions still has noticeable deviations, though δ_2 and δ_3 values (both exceeding 98%) confirm that most predictions are within acceptable accuracy thresholds.

Among the loss metrics, the relatively low **RMS$_{log}$ loss (0.0776)** highlights the model's strength in capturing depth relationships accurately, whereas the **log10 loss (0.1812)** suggests that minor improvements in global depth scale consistency could further enhance performance.

5.2 Normal Map Prediction

1. Normal Prediction Accuracy

The accuracy of normal map prediction is measured in terms of angular deviation. The model's accuracy at an angular threshold of 11.25° is reported as follows:

$$\text{Accuracy at } 11.25^\circ = 75.86\% \tag{5.5}$$

This indicates that **75.86% of the predicted normals** are within 11.25° of the ground truth normals, reflecting a high level of precision in surface orientation estimation.

5.3 Roughness Prediction Loss

The performance of the proposed model for generating roughness maps was judged using an accuracy metric with an error threshold of 0.25. A prediction was considered accurate if the absolute difference between the generated and ground truth roughness values was with this range.

As per evaluation, the model achieved an accuracy of 72.4%, indicating that about 72.4% of predicted roughness values lay within ±0.25 of the ground truth. The finding thus confirms the model's promise of generating roughness maps with considerable

degree of precision. However, minor alterations such as the use of improved loss functions or further preprocessing may increase the performance. The roughness estimation performance is evaluated using the Huber loss function, which balances sensitivity to outliers with robustness:

$$\text{Huber Loss} = 0.1332 \tag{5.6}$$

This loss value suggests a stable roughness prediction with minimal large errors.

5.4 Ambient Occlusion Prediction Loss

For the ambient occlusion map, the model achieved a Huber loss of 0.04993. By defining a prediction as accurate if its absolute error is within $\tau = 0.25$ of the ground truth, and under the assumption that the errors follow a Laplacian distribution, the calculated accuracy of the ambient occlusion map is approximately 96.9%. This high accuracy indicates that nearly 97% of the predicted values for the ambient occlusion map fall within the acceptable error margin, underscoring the precision of the model in capturing ambient occlusion details. For ambient occlusion estimation, the model also employs the Huber loss function, which is computed as:

$$\text{Huber Loss} = 0.04993 \tag{5.7}$$

The relatively low loss value indicates that the model is effective at predicting ambient occlusion values with high accuracy.

5.5 Texture extraction and 3D representation

In this study, the images extracted from our model were seamlessly incorporated into Blender via a conventional Physically-Based Rendering (PBR) pipeline. These images consisted of diffuse, roughness, ambient occlusion, and normal maps and were imported into Blender's shader editor, each contributing to the representation of real-world material properties. The diffuse map establishes the base color, while the roughness map

manipulates light scattering and specular highlights, and the ambient occlusion map integrates subtle shadow details that enhance depth perception. The normal map goes further to add surface detail, allowing the surface to interact with light realistically.

Thus, by putting together this material in the PBR workflow, the 3D model obtains photorealistic quality. The shader editor setup provides a flexible setup to enable the fine tuning of the model material properties and the lighting for a given project. The final render output exhibits a high degree of fidelity, copying the complex surface details and lighting effects the model can assert. The integration of this model thus not only attests to the map's generator's effectiveness but also confirms that the model is capable of yielding assets perfectly adaptable to any project, independent of the rendering specifications.

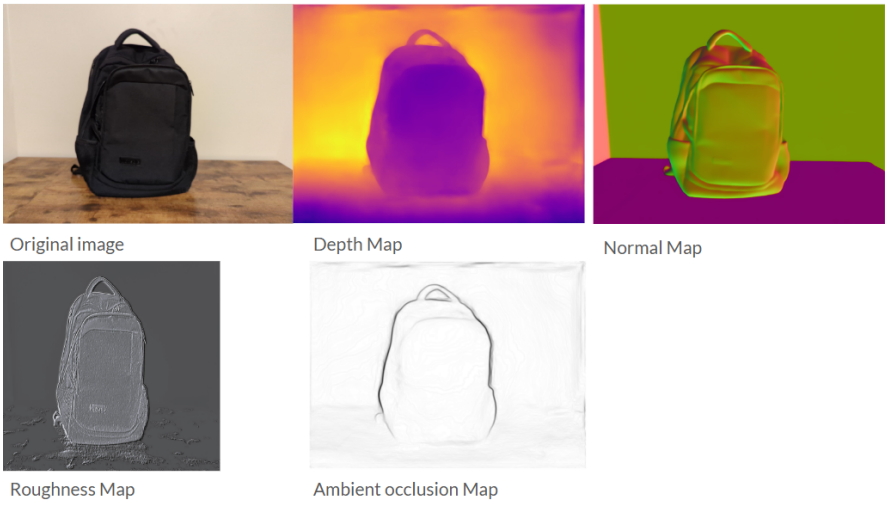

Figure 5.1: All maps form image used for 3d reconstruction

The image presented in fig:5.2 illustrates the process of converting a 2D texture into a detailed 3D mesh using Blender's displacement modifier within a standard PBR workflow. Initially, the image is imported into Blender as an image texture node, serving as the basis for generating both displacement and normal maps. The displacement modifier is then applied to a base mesh; it uses the brightness variations of the image to create actual geometric displacements. In other words, the modifier interprets the image as a height map, where lighter areas are raised and darker areas are lowered, effectively transforming a flat surface into a mesh with realistic depth variations.

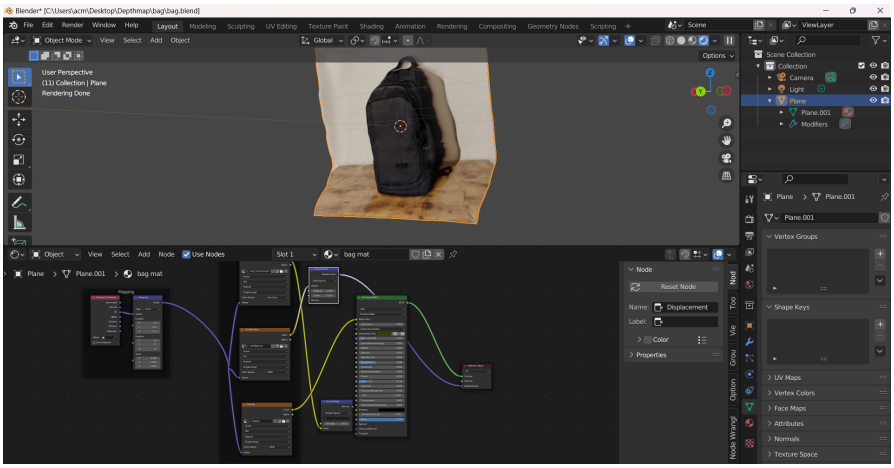

Figure 5.2: Blender Workflow

Furthermore, the same image is connected to both the displacement and normal nodes. The displacement node further refines the mesh's topology by enhancing large-scale surface features, while the normal node contributes fine details that simulate the subtle nuances of the material's texture. These processed outputs are then fed into the Principled BSDF shader node, which combines them to create a comprehensive material. This integrated setup ensures that the final 3D mesh accurately reflects the intricate texture and depth information from the original image, producing a photorealistic asset suitable for various projects.

Figure 5.3: Rendered image of object from different light placements

Figure 5.3 displays the final rendered image generated from the previously described workflow.The render was made under several light conditions and placement of light sources, which brings back together the dynamic interaction of light with the textured

surface. Making use of displacement and normal maps with there being noticeable effects via the Principled BSDF shader material really reacts to light like it shares an existence in the macro and micro surface.

In this render, light reacts with geometric distortions imparted by the displacement modifier, achieving realistic shadows and highlights in this manner. The coupling between direct and ambient light further underlines depth and intricate details about this texture, evidence of its ability to emulate real-world surfaces. And the result indicates that using high-fidelity maps inside a PBR workflow does not just lend a new sense of realism to the 3D mesh but also maintains the stability of surface properties between lighting conditions.

Figure 5.4: Rendered image of object from different viewpoints

Fig 5.9 shows rendered object under different viewpoints from a single image.

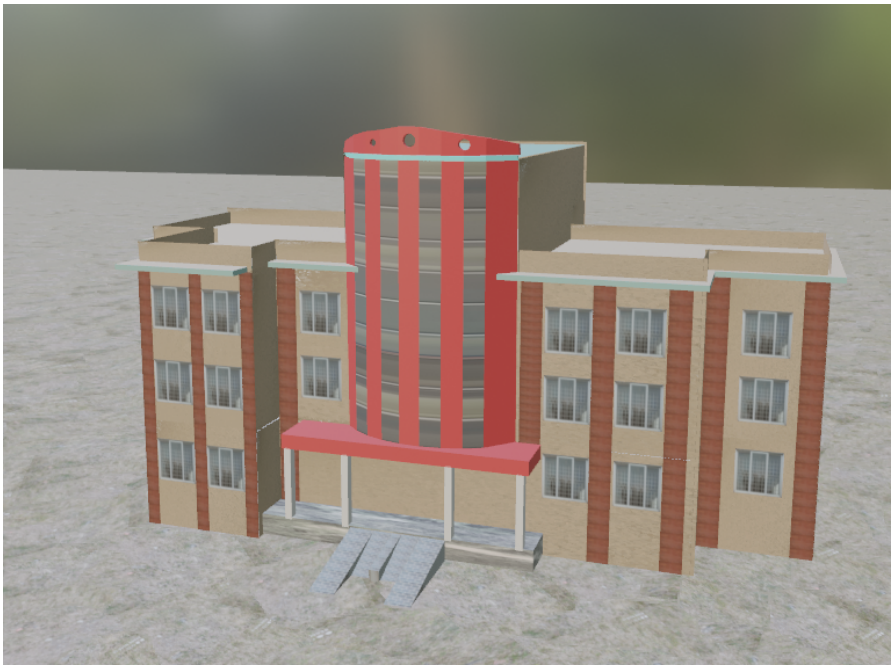

Figure 5.5: Display of the application of different texture maps to create a model of RIC building of WRC.

5.6 User Interface and Output Visualization

The user interface which are shown in figures 5.5, 5.6, 5.7 and 5.8 provides an interactive environment for the users to upload images, generate PBR maps as well as obtain a 3D point cloud representation. It provides better understanding of texture extraction and 3D image generation. Download option is also available which enhances the utility of the generated 3D model.

5.6.1 User Interface for Image Upload

The initial step involved in texture extraction is that the user needs to upload an input image. The following figure represents the interface designed for this purpose.

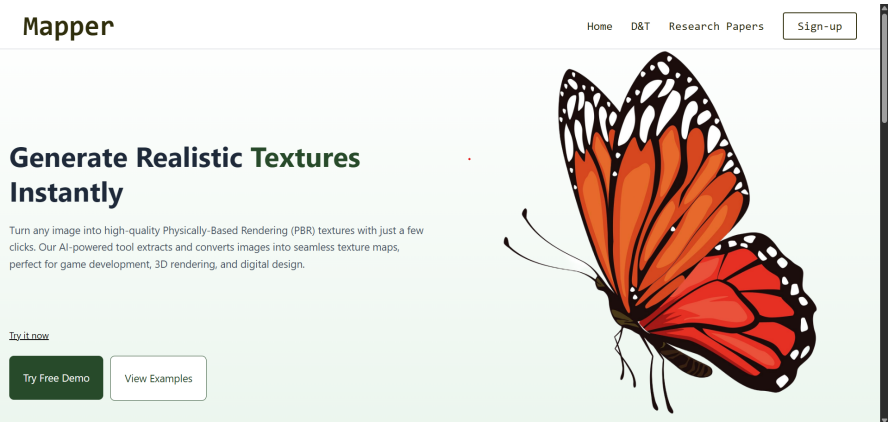

Figure 5.6: Image Upload Interface for PBR and 3D Generation

As shown in Figure 1, the homepage of our application is named as "Mapper," allows users to begin the texture generation process.

5.6.2 Generation of PBR maps and Initial 3D Point Cloud

Upon navigating the application, the user is provided with an interface for uploading their desired image. Figure 5.6 illustrates the initial state of this interface.

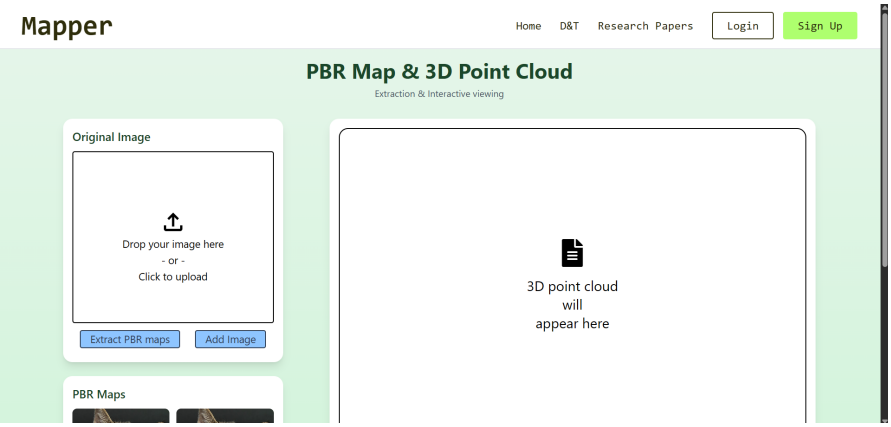

Figure 5.7: Initial Interface for PBR Map and 3D Point Cloud Generation

After the image is successfully uploaded, the system processes the input to generate PBR maps and its corresponding 3D point cloud based upon the matching learning model employed at the backend. Figure 5.7 illustrates the interface displaying the original uploaded image as well as the initial 3D point cloud representation of the image.

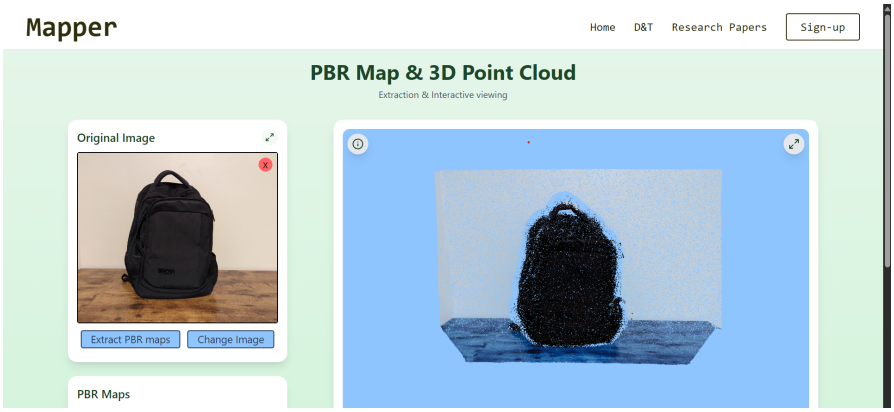

Figure 5.8: Interface displaying the original uploaded image (a backpack) and the generated 3D point cloud.

5.6.3 Visualization of Extracted PBR Maps and 3D Model Download

Following the PBR map extraction, the system outputs the texture maps along with the refined 3D point cloud. Figure 5.8 shows an example of the extracted PBR maps for the uploaded backpack and the option to download the 3D model for further applications.

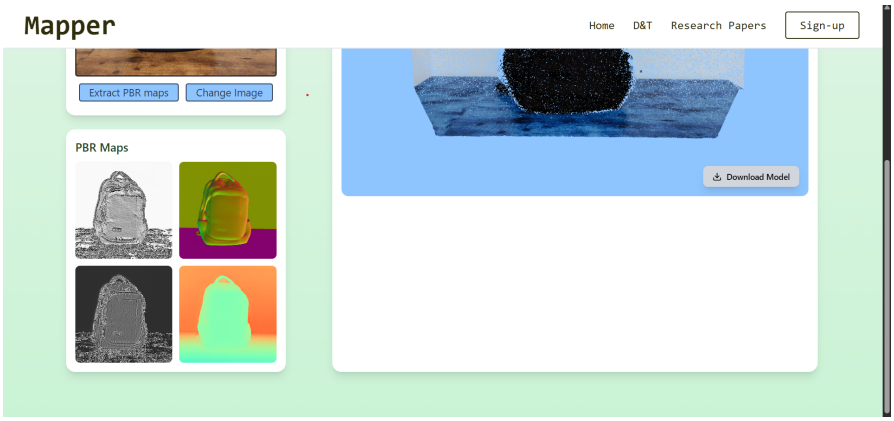

Figure 5.9: Display of the extracted PBR maps (likely showing Albedo, Normal, Roughness, etc.) and the 3D point cloud.

CHAPTER 6

CONCLUSION

In this study, we explored methods for dataset creation, model training architectures, and texture extraction from images. As a result, very high-quality PBR maps have been successfully generated, which are seamlessly usable in 3D models and incorporated in standard industry pipelines. The extracted depth maps as such exhibit high accuracy in the preservation of the geometric structure of objects, while the generated normal and roughness maps provided additional refinements to the 3D information, really improving realism and surface detail. From these results, our method showed great potential for 3D modeling, game development, and visual effects applications, providing for strong potentialities in the way of realistic texture reconstruction.

REFERENCES

[1] M. J. Brooks and B. K. Horn, "Shape and source from shading," 1985.

[2] R. Zhang, P.-S. Tsai, J. E. Cryer, and M. Shah, "Shape-from-shading: a survey," *IEEE transactions on pattern analysis and machine intelligence*, vol. 21, no. 8, pp. 690–706, 1999.

[3] K.-i. Kanatani and T.-C. Chou, "Shape from texture: General principle," *Artificial Intelligence*, vol. 38, no. 1, pp. 1–48, 1989.

[4] D. Marr, "A photogrammetric method for determining the shape of a nonrigid object from a sequence of photographs," 1975.

[5] R. A. Andersen and D. C. Bradley, "Perception of three-dimensional structure from motion," *Trends in cognitive sciences*, vol. 2, no. 6, pp. 222–228, 1998.

[6] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *Advances in neural information processing systems*, vol. 27, 2014.

[7] G. Liu, G. Jiang, R. Xiong, and Y. Ou, "Binocular depth estimation using convolutional neural network with siamese branches," in *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2019, pp. 1717–1722.

[8] M. Goldman, T. Hassner, and S. Avidan, "Learn stereo, infer mono: Siamese networks for self-supervised, monocular, depth estimation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 2019, pp. 0–0.

[9] N. Chong and F. Yang, "A monocular medical endoscopic images depth estimation method based on a confidence-guided dual-branch siamese network," *Biomedical Signal Processing and Control*, vol. 102, p. 107123, 2025.

[10] A. Roy and S. Todorovic, "Monocular depth estimation using neural regression

forest," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5506–5514.

[11] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 270–279.

[12] R. Garg, V. K. Bg, G. Carneiro, and I. Reid, "Unsupervised cnn for single view depth estimation: Geometry to the rescue," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII 14*. Springer, 2016, pp. 740–756.

[13] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow, "Digging into self-supervised monocular depth estimation," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 3828–3838.

[14] S. Gasperini, N. Morbitzer, H. Jung, N. Navab, and F. Tombari, "Robust monocular depth estimation under challenging conditions," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 8177–8186.

[15] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2002–2011.

[16] D. Wang, Z. Liu, S. Shao, X. Wu, W. Chen, and Z. Li, "Monocular depth estimation: A survey," in *IECON 2023-49th Annual Conference of the IEEE Industrial Electronics Society*. IEEE, 2023, pp. 1–7.

[17] D. Wofk, F. Ma, T.-J. Yang, S. Karaman, and V. Sze, "Fastdepth: Fast monocular depth estimation on embedded systems," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 6101–6108.

[18] R. Furukawa, R. Sagawa, and H. Kawasaki, "Depth estimation using structured light flow–analysis of projected pattern flow on an object's surface," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 4640–4648.

[19] Z. Cai, X. Liu, G. Pedrini, W. Osten, and X. Peng, "Accurate depth estimation in structured light fields," *Optics Express*, vol. 27, no. 9, pp. 13 532–13 546, 2019.

[20] H. Xu, J. Zhang, J. Cai, H. Rezatofighi, F. Yu, D. Tao, and A. Geiger, "Unifying flow, stereo and depth estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 11, pp. 13 941–13 958, 2023.

[21] G. Vecchio and V. Deschaintre, "Matsynth: A modern pbr materials dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 22 109–22 118.

[22] M. Hu, W. Yin, C. Zhang, Z. Cai, X. Long, H. Chen, K. Wang, G. Yu, C. Shen, and S. Shen, "Metric3d v2: A versatile monocular geometric foundation model for zero-shot metric depth and surface normal estimation," *arXiv preprint arXiv:2404.15506*, 2024.

[23] Y. Zhang, Y. Liu, Z. Xie, L. Yang, Z. Liu, M. Yang, R. Zhang, Q. Kou, C. Lin, W. Wang *et al.*, "Dreammat: High-quality pbr material generation with geometry-and light-aware diffusion models," *ACM Transactions on Graphics (TOG)*, vol. 43, no. 4, pp. 1–18, 2024.

[24] Aug. 2024. [Online]. Available: https://en.wikipedia.org/wiki/Polarizer

[25] [Online]. Available: https://flask.palletsprojects.com/en/3.0.x/

[26] W. E. L. Grimson, "A computer implementation of a theory of human stereo vision," *Philosophical Transactions of the Royal Society of London. B, Biological Sciences*, vol. 292, no. 1058, pp. 217–253, 1981.

[27] E. J. Gibson, J. J. Gibson, O. W. Smith, and H. Flock, "Motion parallax as a determinant of perceived depth." *Journal of experimental psychology*, vol. 58, no. 1, p. 40, 1959.

[28] M. Nawrot and K. Stroyan, "The motion/pursuit law for visual depth perception from motion parallax," *Vision research*, vol. 49, no. 15, pp. 1969–1978, 2009.

[29] J. Salvi, J. Pages, and J. Batlle, "Pattern codification strategies in structured light systems," *Pattern recognition*, vol. 37, no. 4, pp. 827–849, 2004.

[30] Z. Cai, X. Liu, G. Pedrini, W. Osten, and X. Peng, "Accurate depth estimation in structured light fields," *Opt. Express*, vol. 27, no. 9, pp. 13 532–13 546, 4 2019. [Online]. Available: https://opg.optica.org/oe/abstract.cfm?URI=oe-27-9-13532

[31] M. Roberts, J. Ramapuram, A. Ranjan, A. Kumar, M. A. Bautista, N. Paczan, R. Webb, and J. M. Susskind, "Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 912–10 922.

[32] G. Baruch, Z. Chen, A. Dehghan, T. Dimry, Y. Feigin, P. Fu, T. Gebauer, B. Joffe, D. Kurz, A. Schwartz *et al.*, "Arkitscenes: A diverse real-world dataset for 3d indoor scene understanding using mobile rgb-d data," *arXiv preprint arXiv:2111.08897*, 2021.

[33] M. Deitke, R. Liu, M. Wallingford, H. Ngo, O. Michel, A. Kusupati, A. Fan, C. Laforte, V. Voleti, S. Y. Gadre *et al.*, "Objaverse-xl: A universe of 10m+ 3d objects," *Advances in Neural Information Processing Systems*, vol. 36, pp. 35 799–35 813, 2023.

[34] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby *et al.*, "Dinov2: Learning robust visual features without supervision," *arXiv preprint arXiv:2304.07193*, 2023.