Stereo Vision with Deep Learning

Barış Bakırdöven
Computer Science and Engineering
Mechatronics Engineering
Sabancı University, Istanbul, Türkiye
29028

Ozan Parlayan
Computer Science and Engineering
Sabancı University, Istanbul, Türkiye
29283

İpek Akkuş
Computer Science and Engineering
Sabancı University, Istanbul, Türkiye
30800

***Abstract*—This report presents our work on stereo vision-based depth estimation, with a focus on modifying and enhancing the HITNet architecture while comparing it with classical, hybrid, and other end-to-end solutions. We explored multiple variations of HITNet by replacing the original U-Net backbone with lightweight alternatives such as MobileNet and a Lite version, aiming to improve efficiency while preserving accuracy. Our experiments were conducted on public datasets, primarily KITTI and Middlebury. However, due to Middlebury's limited and mismatched data distribution, the KITTI dataset was prioritized. We also evaluated StereoNet with different encoder variants for comparison. This report details the problem background, outlines the dataset choices and preprocessing steps, and presents both classical and deep learning-based approaches we employed.**

I. Introduction

Depth estimation from stereo images is a fundamental problem in computer vision, where two images captured from slightly different viewpoints are used to recover the 3D structure of a scene. This is achieved by calculating the disparity, the horizontal pixel shift between corresponding points in the left and right images. Using camera calibration parameters, this disparity is transformed into depth information.

Stereo vision has numerous real-world applications including autonomous driving, robotics, augmented reality (AR) and virtual reality (VR) systems, and 3D reconstruction. The quality of depth estimation significantly affects the performance of these applications, especially in dynamic or unstructured environments.

Historically, stereo depth estimation relied on classical computer vision techniques such as block matching, semi-global matching (SGM), and belief propagation. These methods leverage geometric constraints and hand-crafted features, but struggle with occlusions, textureless surfaces, and repetitive patterns.

More recently, deep learning has revolutionized the field by offering end-to-end solutions that learn representations directly from data. These methods surpass traditional approaches in both robustness and accuracy across various benchmarks. However, many of these models are computationally intensive, making them unsuitable for real-time or resource-constrained systems.

II. Problem Description

A. Research Question

Can we achieve efficient and accurate stereo depth estimation by applying lightweight encoder architectures such as MobileNet to deep learning-based models like HITNet and StereoNet? Additionally, how do these modern models compare in accuracy and runtime efficiency to classical methods such as semi-global matching, or hybrid methods, particularly in the context of real-time deployment?

B. Motivation

Stereo depth estimation is a fundamental task in computer vision, enabling machines to perceive depth from two images captured at slightly different horizontal positions. The process involves computing the disparity between matching pixel pairs and converting it into depth using known camera calibration parameters. This capability is critical for many applications, including autonomous vehicles, robotic navigation, augmented and virtual reality, and 3D reconstruction [1][2].

Traditionally, stereo vision systems relied on classical algorithms such as block matching, semi-global matching (SGM), and belief propagation. These methods use handcrafted matching costs and geometric constraints to determine pixel correspondences [3]. While these techniques can be efficient and interpretable, they often fail under real-world conditions involving occlusions, repetitive textures, or textureless surfaces. They also lack the adaptability to varied scenes without manual tuning.

With the rise of deep learning, data-driven methods have surpassed classical approaches in both accuracy and robustness. Architectures such as GC-Net [9], PSMNet [8], HITNet [5], and StereoNet learn hierarchical features and matching functions end-to-end, allowing them to perform well in more complex and visually diverse environments. However, this improved accuracy often comes with increased computational complexity, making real-time deployment challenging.

C. Background and Literature Review

To address the challenge of computational cost, recent research has focused on developing models that balance accuracy with efficiency. HITNet introduces a tile-based iterative refinement approach that avoids constructing full 3D cost volumes, allowing for faster inference with competitive results [5]. StereoNet presents a compact alternative, using sub-pixel

refinement and fewer layers to speed up disparity estimation, particularly suitable for real-time settings.

Despite their design improvements, both models still employ encoders like U-Net, which are relatively large and resource-intensive. This becomes a bottleneck for deployment in mobile or embedded systems. To improve efficiency, our project investigates whether replacing these encoders with lightweight architectures such as MobileNet and a custom Custom encoder can preserve accuracy while reducing computational cost. MobileNet uses depthwise separable convolutions to minimize the number of parameters and floating-point operations [6][7], and has been shown to be effective in mobile-friendly stereo vision tasks such as those explored in MobileStereoNet [Shamsafar et al., 2021].

Our work compares multiple model configurations, applying MobileNet and Custom encoders to both HITNet and StereoNet frameworks. These configurations are evaluated on the KITTI dataset, which provides a realistic benchmark for depth estimation in autonomous driving scenarios. Furthermore, as a baseline comparison, we also consider classical methods like semi-global matching to assess how much performance gain and efficiency trade-off is achieved through deep learning-based approaches.

By comparing modern architectures with classical methods and analyzing the effect of encoder replacements, this project aims to determine practical model designs for real-time stereo depth estimation on limited hardware.

III. Methodology

A. Datasets

In developing and evaluating our stereo depth estimation models, we reviewed two prominent publicly available datasets: the Middlebury Stereo Dataset and the KITTI Stereo 2015 Dataset. Each dataset offered different advantages in terms of resolution, real-world complexity, and ground truth quality. We carefully analyzed their structures, availability of labels, and suitability for deep learning pipelines before determining our final training and evaluation setup.

1) Middlebury: The Middlebury Stereo Dataset, developed by Middlebury College, is a classical benchmark in stereo vision research, offering high-resolution stereo pairs with subpixel-accurate ground-truth disparity maps [10][11]. While widely used in academic studies, we encountered several limitations that hindered its integration into our pipeline.

Most notably, the dataset contains only about 20 labeled stereo image pairs, which is insufficient for training modern deep networks. In addition, it includes ambient versions of each scene captured under different lighting or imaging conditions. These are not simply transformed copies (such as hue-shifted versions), but entirely distinct captures that do not share pixel-wise correspondence with the original disparity labels. This prevents the use of ground-truth annotations on these ambient variants.

Furthermore, the Middlebury dataset is composed of segments with varied sizes and inconsistent formatting. Incorporating it into an end-to-end learning setup would have required substantial preprocessing or a semi-supervised training strategy. Due to these constraints, we treated Middlebury as a reference but did not include it in our model training.

2) KITTI: The KITTI Stereo 2015 dataset, created by the Karlsruhe Institute of Technology and the Toyota Technological Institute at Chicago, provides real-world stereo image pairs captured from a vehicle-mounted stereo rig, along with accurate disparity ground truth generated from 3D laser scans [12]. It includes 200 image pairs taken across a variety of environments, including urban streets, highways, and residential neighborhoods.

This dataset is widely used as a benchmark in stereo vision due to its combination of realistic visual conditions and dense disparity maps. However, many of the disparity images contain invalid or missing regions, particularly near the top of the frame. To address this, we applied a preprocessing strategy to crop out unusable regions while preserving critical scene content. This improved both the quality and stability of training.

As shown in our project folder, the KITTI dataset was organized using a standard format that includes:

- `image_2` and `image_3`: Left and right stereo images
- `disp_occ_0` and `disp_occ_1`: Disparity maps including occluded areas
- `disp_noc_0` and `disp_noc_1`: Disparity maps excluding occlusions
- Additional folders for optical flow and visualization (not used in our training)

This structure supported consistent preprocessing and model input formatting, and allowed us to seamlessly train and evaluate different architectural variants, including HITNet and StereoNet."'

B. Classical Method

The classical stereo vision methodology involves a structured pipeline reliant on geometric and signal processing techniques, distinct from learning-based methods. Below is the detailed procedure we followed:

Camera Calibration and Rectification: Calibration is vital for accurate depth estimation, providing both intrinsic parameters (focal length, optical centers, distortion coefficients) and extrinsic parameters (relative rotation and translation between cameras). Utilizing factory calibration parameters provided by KITTI, stereo pairs were rectified to ensure horizontal alignment of corresponding points, simplifying disparity searches along epipolar lines. Rectification corrects for lens distortions and minor camera misalignments using rectifying homographies derived from intrinsic and extrinsic parameters.

Disparity Estimation Using Semi-Global Matching (SGM): To calculate disparities, we employed Semi-Global Matching (SGM), a method renowned for robustness against occlusions and textureless regions. SGM computes disparity by optimizing matching costs over multiple directions, enforcing smoothness while preserving edges. Parameters for matching costs and smoothness penalties were empirically

chosen based on visual and quantitative analysis of initial results.

Depth Conversion: The estimated disparity map can be transformed into a depth map using the geometric relationship derived from the pinhole camera model. This conversion is fundamental in stereo vision, as it allows pixel-level disparity values to be interpreted as real-world distances. The depth Z is calculated using the following relation: Z =(f*B) / D , where Z is depth, f is focal length, B baseline distance, and D disparity as shown in Figure 1.

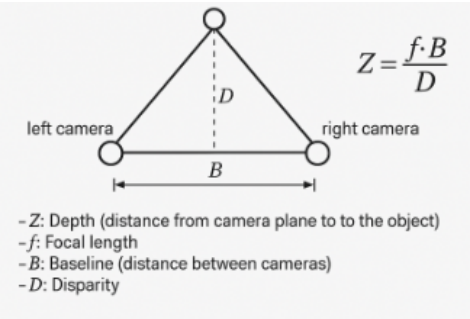

Fig. 1. Disparity to Depth Conversion

Post-processing: Noise and artifacts commonly observed in raw disparity maps necessitated post-processing. Median filtering was applied to smooth the disparity map, effectively reducing isolated errors and enhancing consistency in estimated depths without significantly compromising edge fidelity.

Evaluation Metrics: To quantitatively assess our classical method, we computed several metrics on the KITTI dataset:

- Disparity computation time: Measured to evaluate suitability for real-time applications.
- KITTI D1-all error: Calculated as the percentage of pixels exceeding an error threshold (greater than 3 pixels or 5% disparity).
- Mean Absolute Error (MAE) and Root Mean Square Error (RMSE): These quantify the mean and variance of errors between estimated disparities and ground truths, providing insights into overall estimation accuracy and reliability.

C. Hybrid Method

Although classical geometry-based pipelines deliver fast, interpretable results and deep-learning models achieve state-of-the-art accuracy, each paradigm exhibits complementary strengths and weaknesses. Consequently, a hybrid strategy was devised in which deterministic stereo geometry is exploited for estimation while a lightweight neural network refines the result. The overall goal was to preserve the real-time guarantees of Semi-Global Matching (SGM) yet close the accuracy gap to fully learned methods on difficult regions such as thin structures, repetitive textures, and low-contrast areas.

Pipeline Overview

1) Initial Disparity Proposal (SGM): Rectified left–right pairs are processed using the same SGM configuration described in the classical section, with disparity range $[0, 128]$ px and 8-path cost aggregation. The smoothness penalties P_1 and P_2 were empirically tuned for KITTI.

The raw SGM output D_{sgm} undergoes median filtering to remove impulse noise and is downsampled to $H/4 \times W/4$ resolution. This step significantly reduces memory and compute load for the subsequent learning stage without sacrificing structural disparity cues.

2) Learned Residual Refinement: To improve the coarse SGM prediction, we introduce a lightweight convolutional neural network, *SGM-RefineNet*, designed to predict a residual disparity correction ΔD.

a) Input Tensor: The network receives a 4-channel input $X = [L, R, D_{\text{sgm}}/128]$, where L and R are the normalised RGB stereo images (scaled to $[0, 1]$), and D_{sgm} is the disparity proposal, normalised to lie in $[0, 1]$. This formulation allows the network to focus on correcting local errors and inconsistencies, rather than learning absolute geometry from scratch.

b) Network Architecture: SGM-RefineNet is composed of:

- Five depthwise-separable convolutional blocks with 3×3 kernels and feature dimensions increasing from 32 to 64,
- Two refinement layers that apply residual learning to capture local geometric inconsistencies,
- A final 1×1 convolution layer that outputs the residual disparity map ΔD.

Skip connections between convolutional stages help retain spatial context and preserve gradient flow. The total parameter count remains under 0.4M, enabling real-time inference on modern GPUs.

3) Final Disparity and Depth Computation: The refined disparity prediction is computed as:

$$D_{\text{hyb}} = \text{upsample}(D_{\text{sgm}} + \Delta D)$$

where bilinear up-sampling restores the full resolution $H \times W$.

Finally, the corrected disparity map is converted into a depth map using the pinhole camera model:

$$Z = \frac{fB}{D_{\text{hyb}}}$$

where f is the calibrated focal length and B is the baseline distance between the stereo cameras.

This hybrid method retains the interpretability and efficiency of classical stereo vision while leveraging the representational power of deep learning to recover fine-grained disparity corrections in regions where SGM alone struggles.

D. Deep Learning Methods

1) HITNet: **HITNet** (Hierarchical Iterative Tile Refinement Network), introduced by Tankovich et al. [2023], is a lightweight stereo depth estimation model that combines fast initialization with a refinement pipeline for efficient disparity

prediction. It avoids the need for full 3D cost volume computation by generating local tile-based hypotheses and applying image warping during refinement. Due to its modular and low-latency design, HITNet is especially well-suited for real-time or resource-constrained applications.

The original HITNet architecture employs a tiny encoder as its default feature extractor. This encoder consists of a minimal sequence of convolutional and transpose convolutional layers, optimized for low memory usage and rapid computation, making it ideal for embedded systems and mobile platforms.

In our implementation, we developed a flexible HITNet pipeline that supports interchangeable stereo feature extraction backbones. Importantly, while the original HITNet models were not trained on the KITTI dataset, often used only for evaluation due to licensing constraints, we trained our HITNet variants directly on KITTI in a fully supervised setting.

All variants retain the same initialization and propagation blocks as HITNet, with the main difference lying in how the stereo inputs are converted into deep feature maps before tile-based matching and refinement. The same loss function (masked Mean Absolute Error) and evaluation metrics (KITTI D1-all error, MAE, RMSE, and inference time per stereo pair) were applied consistently across all variants. The training configuration was also standardized: all models were trained on cropped KITTI stereo pairs of size 384×1248 with a batch size of 2, using the Adam optimizer. Main training lasted 20 epochs, followed by fine-tuning where applicable.

1. HITNet with U-Net Feature Extractor

In this variant, we integrated a U-Net-based encoder-decoder architecture into the HITNet framework as the stereo feature extractor. The goal of using U-Net is to enhance the spatial detail awareness of the model, particularly near object boundaries and in complex scenes where preserving local context is critical for accurate disparity estimation. The U-Net backbone consists of:

- **Encoder path:** Progressively deeper convolutional layers with 3×3 kernels, each followed by Batch Normalization and ReLU activation. Each block doubles the number of feature channels, for example, from 32 to 64, then 128, and so on.
- **Downsampling:** Max-pooling layers (stride 2) are used to reduce spatial dimensions by half at each level, increasing the receptive field and reducing computational load.
- **Decoder path:** Conv2DTranspose layers are used to upsample feature maps. These are concatenated with corresponding encoder features via skip connections, allowing the network to reuse high-resolution spatial information.
- **Output:** A final 1×1 convolution layer projects the decoder output to a 32-channel feature map, suitable for cost volume computation.

2. HITNet with MobileNet Feature Extractor

This configuration uses a MobileNetV2-based encoder as the feature extractor within the HITNet framework. The aim is to strike a balance between accuracy and efficiency by leveraging the lightweight and depthwise-separable convolutional architecture of MobileNet. Key design choices include:

- Extracting intermediate feature maps from the `block_6_expand_relu` layer to maintain spatial resolution.
- Applying a 1×1 convolution to reduce channel depth to 32, making the features compatible with HITNet's cost volume construction.

This setup is particularly well-suited for deployment scenarios requiring fast inference while maintaining moderate accuracy.

3. HITNet with Custom Lite Feature Extractor

In this variant, we implemented a custom lightweight feature extractor referred to as the "lite" backbone. It is designed for speed and minimal resource consumption, making it ideal for edge devices and real-time applications.

The architecture includes:

- **Encoder:** A three-stage convolutional hierarchy:
  - **Stage 1:** Conv2D with 16 filters, kernel size 3×3, stride 2 (downsampling), followed by ReLU activation.
  - **Stage 2:** Conv2D with 32 filters, kernel size 3×3, stride 2, ReLU.
  - **Stage 3:** Conv2D with 32 filters, kernel size 3×3, stride 1, ReLU.
- **Decoder:**
  - First upsampling via Conv2DTranspose (stride 2) and feature alignment through cropping and concatenation with the first encoder output.
  - A second Conv2DTranspose layer is used to return to original spatial dimensions.
  - Final Conv2D to refine and reduce the feature depth to 32 channels for cost volume construction.

Let F_L and $F_R \in R^{H' \times W' \times 32}$ represent the feature maps from the left and right images, respectively. The disparity cost volume $C(u, v, d)$ is again calculated as:

$$C(u, v, d) = |F_L(u, v) - F_R(u - d, v)|_1 \tag{1}$$

Despite being the fastest configuration, this model still captures sufficient feature detail to reconstruct general disparity structures such as roads and sky, though it lacks precision in fine edges or complex object boundaries.

Common Configuration Summary

- **Dataset:** KITTI stereo dataset (cropped to remove invalid top rows)
- **Input Size:** 384 × 1248
- **Loss Function:** Masked MAE (ignores disparity=0)
- **Optimizer:** Adam
- **Training Schedule:** 20 epochs (additional 10 for fine-tuning where applicable)
- **Evaluation Metrics:** KITTI D1-all error, MAE, RMSE, inference time per stereo pair

2) HITNet V2 - with Cropped Images: In this variant, we extend the original HITNet architecture by incorporating a custom-designed lightweight feature extractor and applying it to cropped stereo image inputs. The motivation for this setup is twofold:

Model Efficiency
We replaced the default U-Net encoder with a more compact convolutional feature extractor. The encoder consists of sequential convolutional layers with increasing depth and decreasing spatial dimensions. Each layer is followed by batch normalization and ReLU activation to enhance training stability and representation power, while keeping the overall model lightweight and suitable for real-time applications.

Input Optimization through Cropping
During our dataset analysis, we observed that the top 144 rows of many KITTI disparity maps contain invalid or missing values. These areas often correspond to sky regions or occluded surfaces that the LiDAR scanner could not capture reliably. To prevent the model from learning from these noisy or uninformative regions, we applied a preprocessing step that crops the upper portion of all input images and disparity maps. Specifically:

- Only the rows below index 144 are retained.
- Cropping is applied consistently to both left and right stereo images as well as the corresponding ground truth maps.

This strategy ensures that the model focuses on meaningful and labeled regions, which leads to faster convergence and improved generalization, especially when using low-capacity encoders.

Model Design Overview

- The model uses a custom lightweight encoder with progressively deeper convolutional layers to extract hierarchical features.
- Each layer is followed by batch normalization and ReLU activation to stabilize and enrich the learning process.
- The feature maps from the left and right images are used to build a cost volume, which encodes potential disparities between image pairs.
- This cost volume is processed by a 3D convolutional refinement block, which smooths and consolidates disparity hypotheses.
- A soft-argmin operation is applied to convert the refined cost volume into subpixel-accurate disparity estimates.
- The resulting disparity map is bilinearly upsampled by a factor of 4 to restore the original resolution of the input images.

3) StereoNet: **StereoNet** is a recent learning-based architecture that solves stereo depth estimation with a remarkably compact design. A 2D feature extractor encodes each image at quarter resolution. Then a correlation volume is built by laterally shifting features in the right image. A small 3D convolutional stack refines this volume, followed by a soft-argmin operation to regress a dense, sub-pixel disparity map. The result is finally upsampled to full image resolution. Due to its lightweight structure and efficiency, StereoNet is particularly well-suited for real-time applications on devices with limited computational resources.

Data Pipeline

1.1 Stereo Dataset & Split

The study uses the **KITTI-SceneFlow** stereo subset, which provides rectified left/right RGB pairs and dense disparity maps (`disp_occ_0`). All filenames common to the three training folders are shuffled once and partitioned into an **80% train, 20% validation** split using `train_test_split(seed = 42)`. A separate list of 400 image pairs from `testing/image_*` is reserved exclusively for qualitative visualisation (no ground truth).

1.2 Pre-processing

- **Spatial resize** — Every RGB and disparity PNG is resized to 384×1248 pixels using *bilinear* interpolation for RGB and *nearest-neighbour* for disparity.
- **Numeric scale** — RGB images are converted to `float32` in the range $[0, 1]$, and disparity values are divided by 256 to yield metric values.
- **Masked validity** — Pixels with `disp = 0` are masked out and ignored in both loss and metric calculations.

1.3 TensorFlow `tf.data` Loader

To feed the network efficiently, we build a streaming input pipeline using **TensorFlow** `tf.data`:

- **File-list tensors** — The absolute paths for left images, right images, and disparity PNGs are converted into three constant string tensors.
- **Dataset of triplets** — These tensors are zipped into a single `tf.data.Dataset` whose elements are triplets: (`left-path`, `right-path`, `disp-path`).
- **On-the-fly parsing** — A custom `parse` function is mapped over the dataset:
  - It reads each file from disk, decodes the PNG, rescales RGB to `[0, 1]`, rescales disparity by `1/256`, and resizes both modalities to the training resolution (384×1248).
  - The function returns a tuple `((leftRGB, rightRGB), dispGT)` matching the model's input signature.
- **Batching and pipelining** — The dataset is shuffled once (training split only), batched using `BATCH_SIZE = 2`, and then `prefetched`. Prefetching overlaps CPU decoding and GPU execution to maintain high utilisation and instant data availability.

1.4 Relation to HITNet

While StereoNet relies on a single forward pass through a fixed correlation volume followed by soft-argmin regression, HITNet (Hierarchical Iterative Refinement Network) tackles disparity estimation as a coarse-to-fine, recurrent optimisation problem. HITNet begins with a low-resolution disparity guess obtained from a cost volume, warps the right image into the

left view, and then iteratively refines the estimate through multiple GRU-like propagation steps that exchange information across scales and spatial neighbourhoods.

As a consequence, HITNet can capture long-range context and enforce local smoothness more aggressively, often achieving state-of-the-art accuracy at the cost of greater architectural complexity. In contrast, the three StereoNet variants explored here keep the pipeline single-shot and light-weight, focusing specifically on how alternative 2-D backbones (tiny-CNN, U-Net encoder, ResNet-18) trade speed for representational power within an otherwise simple framework.

In the present study we keep this core pipeline fixed and investigate how the choice of feature extractor influences both accuracy and speed. Specifically, we compare three encoders of increasing representational power:

1) the encoder half of a U-Net, which adds multi-scale context through its deeper hierarchy;
2) a truncated ResNet-18 whose residual blocks provide stronger feature reuse;
3) a hand-crafted tiny CNN consisting of four lightweight convolution BatchNorm ReLU layers;

All variants are trained from scratch on the exact same KITTI-SceneFlow data stream, thereby isolating the effect of backbone capacity on StereoNet's final performance.

1. StereoNet with U-Net Encoder

A. U-Net Encoder Backbone (¼-resolution features)

Only the encoder half of U-Net is used. Two down-sampling stages produce a feature map at $H/4 \times W/4$ resolution:

TABLE I
U-NET ENCODER STRUCTURE USED IN STEREONET

Stage	Operations (each with BN + ReLU)	Output size
0	$2 \times \text{Conv}(3 \times 3, 32) \rightarrow \text{MaxPool}(2)$	$H/2 \times W/2 \times 32$
1	$2 \times \text{Conv}(3 \times 3, 64) \rightarrow \text{MaxPool}(2)$	$H/4 \times W/4 \times 64$
2	$\text{Conv}(3 \times 3, 128)$ (no pooling)	$H/4 \times W/4 \times 128$

The same encoder is shared (weights tied) between the left and right images, yielding feature tensors F_L and F_R.

a) Cost Volume and 3D Refinement

The correlation volume is constructed by laterally shifting F_R across integer disparities $d \in [0, 47]$ (as $D_{\max} = 192$ and the scale is $1/4$), and computing dot-product correlations with F_L:

$$C \in R^{B \times H/4 \times W/4 \times 48}.$$

This volume is passed through a small 3D CNN refinement head:

$$\text{Conv3D}(32) \rightarrow \text{ReLU} \rightarrow \text{Conv3D}(32) \rightarrow \text{ReLU} \rightarrow \text{Conv3D}(1)$$

producing logits L. A softmax is applied over the disparity dimension to get p_d, and soft-argmin is used to regress expected disparity:

$$d^\star = \sum_d d \cdot p_d.$$

The result is upsampled by $\times 4$ using bilinear interpolation to restore full resolution.

b) Training and Evaluation Protocol

The model is trained using a masked MAE loss over valid pixels:

$$L = \frac{\sum_{u,v} \left| d_{u,v}^{\text{GT}} - d_{u,v}^{\text{pred}} \right| \cdot \mathbf{1}(d^{\text{GT}} > 0)}{\sum_{u,v} \mathbf{1}(d^{\text{GT}} > 0)}.$$

Optimizer: Adam with initial learning rate 1×10^{-3} for 20 epochs.

Evaluation metrics:

- D1-all: Percentage of pixels with error > 3 px and $> 5\%$
- MAE / RMSE over valid pixels
- Inference time (per stereo pair, batch = 2, Colab T4)

B. StereoNet Architecture with a ResNet-18 Encoder

a) Quarter-Scale Feature Extraction

The ResNet-18 backbone is truncated after the second residual stage. An initial 7×7 convolution with stride 2 reduces resolution, followed by a 3×3 max pool. Two standard residual blocks (each with two 3×3 convolutions) are applied, leaving the tensor at $H/4 \times W/4 \times 64$.

A final 1×1 convolution reduces channels to 32, followed by Batch Normalization and ReLU. The same encoder processes both left and right images, producing F_L and F_R.

b) Cost Volume and 3D Refinement

The maximum disparity is $D_q = 192/4 = 48$. For each $d \in [0, 47]$, the right features are shifted and dot-correlated with the left features to construct the volume:

$$C \in R^{B \times H/4 \times W/4 \times 48}.$$

This is reshaped and passed through a 3D CNN:

$$\text{Conv3D}(16) \rightarrow \text{GN} \rightarrow \text{Conv3D}(16) \rightarrow \text{GN} \rightarrow \text{Conv3D}(1).$$

Softmax is applied across the disparity axis to produce a probability distribution. The final sub-pixel disparity is computed via soft-argmin and upsampled by $\times 4$.

c) Optimization and Evaluation Protocol

Training uses a masked MAE loss. The learning rate follows a cosine decay from 3×10^{-4} to 1×10^{-5} over 20 epochs.

Evaluation metrics:

- D1-all error (%)
- MAE and RMSE over valid pixels
- Inference time on a T4 GPU (batch size = 2)

C. StereoNet Architecture with Custom "Tiny-CNN" Feature Extractor

a) Feature Extractor (Tiny-CNN)

The input is downsampled to $H/4 \times W/4$ through four lightweight convolutional blocks:

The encoder is shared across both stereo views, producing $F_L, F_R \in R^{B \times H/4 \times W/4 \times 64}$.

TABLE II
CUSTOM TINY-CNN ENCODER STRUCTURE

Layer	Kernel / Stride	Output Size	Channels
Conv-BN-ReLU	5×5 / 2	$H/2 \times W/2$	32
Conv-BN-ReLU	3×3 / 1	$H/2 \times W/2$	32
Conv-BN-ReLU	5×5 / 2	$H/4 \times W/4$	64
Conv-BN-ReLU	3×3 / 1	$H/4 \times W/4$	64

b) Cost Volume and 3D Refinement

The cost volume is built via channel-wise correlation:

$$\text{cost}(u, v, d) = \frac{1}{64} \sum_{c} F_L(u, v, c)\, F_R(u - d, v, c).$$

Stacking all d values results in:

$$C \in R^{B \times H/4 \times W/4 \times D_q}.$$

The volume is refined via:

$$\text{Conv3D}(32) \rightarrow \text{ReLU} \rightarrow \text{Conv3D}(32) \rightarrow \text{ReLU} \rightarrow \text{Conv3D}(1).$$

c) Disparity Regression and Upsampling

Soft-argmin is used to regress a sub-pixel disparity:

$$d^\star(u, v) = \sum_{d} d \cdot p_d(u, v),$$

where $p_d = \text{softmax}(L_d)$ over the disparity axis.

The resulting map is multiplied by 4 and bilinearly upsampled to produce the final prediction D_{pred}.

D. StereoNet with ResNet + Object Detection

This variant builds upon the StereoNet architecture by incorporating two major enhancements:

- Object-awareness through mask injection, and
- A ResNet-18-based feature extractor capable of handling 4-channel input.

To guide the model's attention toward semantically important regions, we use external object maps as a fourth input channel, alongside the standard RGB input. These maps provide binary segmentation masks where pixels corresponding to objects (e.g., vehicles, pedestrians) are marked as 1, and the background as 0. This encourages the model to focus disparity refinement on foreground elements where depth accuracy is most critical.

The input pipeline reads stereo image pairs and corresponding disparity maps along with object masks. The object masks were pre-generated and stored in the dataset folder under `obj_map`. For both training and testing phases, the same object mask is concatenated to both left and right images to form 4-channel inputs.

To accommodate this structure, we customized a ResNet-18 encoder to accept 4-channel inputs. The encoder is truncated after the second residual block and produces 32-channel feature maps. A cost volume is constructed from left-right feature correlations, followed by a 3D CNN refinement block with Group Normalization layers. Disparity regression is performed using a softmax-weighted summation (soft-argmin), and the resulting disparity map is bilinearly upsampled to full resolution.

Key Training Details:

- **Data split:** 168 training, 40 validation, 200 test images
- **Object maps:** Preprocessed external binary masks provided alongside training data
- **Loss:** Masked MAE
- **Learning rate schedule:** Cosine decay from 3e-4 to 1e-5
- **Training regime:** 20 epochs + 10 fine-tuning
- **Evaluation:** D1-all error, MAE, RMSE
- **Visualization:** Disparity overlay with object-masked input

IV. RESULTS

For improved visibility of certain qualitative results, readers are encouraged to refer to Appendix VI-A, which contains high-resolution versions of all output images for all models tested.

A. Classical Approach Baseline

We established a baseline for comparison using the classical stereo pipeline described previously. Employing Semi-Global Matching (SGM) with median filtering post-processing, our method achieved the following metrics on the KITTI dataset:

TABLE III
PERFORMANCE OF THE CLASSICAL SGM-BASED STEREO METHOD ON KITTI

Metric	Value
KITTI D1-all error	1.99%
Mean Absolute Error (MAE)	1.02 px
Root Mean Squared Error (RMSE)	3.13 px
Inference Time	0.11 s

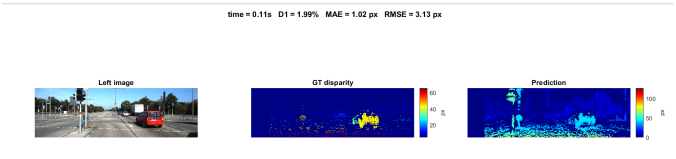

Fig. 2. Qualitative result from the classical SGM pipeline. Left to right: input image, ground truth disparity, predicted disparity. Metrics shown above the image.

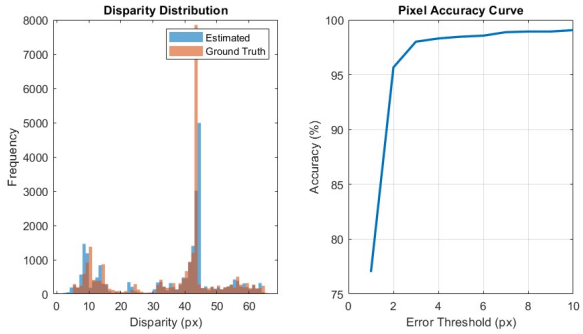

Fig. 3. Left: Disparity distribution of estimated vs. ground truth values. Right: pixel-level accuracy curve by error threshold.

B. Hybrid Approach

Quantitative Evaluation

Quantitative evaluation was conducted to assess the performance of the hybrid geometry–learning method on the KITTI dataset. The following metrics were used to measure accuracy and efficiency, including disparity error rates and runtime performance. The key findings are summarized in Table IV.

TABLE IV
RESULTS OBTAINED BY THE GEOMETRY–LEARNING HYBRID METHOD

Metric	Value
D1-all Error	45.13%
MAE	10.73 px
RMSE	23.27 px
Inference Time	1.75 s/image

Qualitative Observations

The refinement network noticeably reduced speckle noise and improved continuity around thin poles and traffic signs when compared with pure SGM. Artifacts at occlusion borders diminished, and depth transitions on textureless road surfaces became smoother.

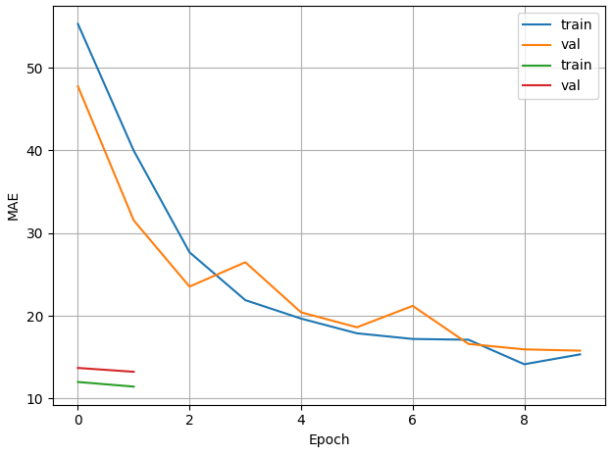

Fig. 4. Training and validation MAE across epochs for hybrid model.

C. HITNet

1. HITNet with U-Net Feature Extractor

The U-Net-enhanced HITNet model was trained on the cropped KITTI stereo dataset for 20 epochs using a batch size of 2 and Adam optimizer. The model was evaluated on the validation set using standard KITTI metrics: D1-all error, Mean Absolute Error (MAE), and Root Mean Square Error (RMSE). Inference time per stereo pair was also measured to assess real-time suitability.

Quantitative Evaluation

The following results present the performance of the HITNet architecture integrated with a U-Net encoder on the KITTI dataset. Key evaluation metrics, including disparity error and inference time, are reported to assess both accuracy and computational efficiency. Detailed results are provided in Table V.

TABLE V
RESULTS OBTAINED BY HITNET WITH U-NET MODEL

Metric	Value
D1-all Error	4.98%
MAE	1.84 px
RMSE	4.21 px
Inference Time	0.75 s/image

Qualitative Observations

The predicted disparity maps show improved object boundary preservation and depth transitions compared to simpler backbones. U-Net's skip connections significantly aid in maintaining fine spatial details, especially in complex urban scenes or where thin structures are present. Disparity continuity across large textureless regions (e.g., roads, walls) was noticeably better than the Lite variant, demonstrating U-Net's effectiveness in global context retention. However, inference speed is slower compared to the MobileNet and Custom Lite variants, indicating a trade-off between accuracy and computational efficiency.

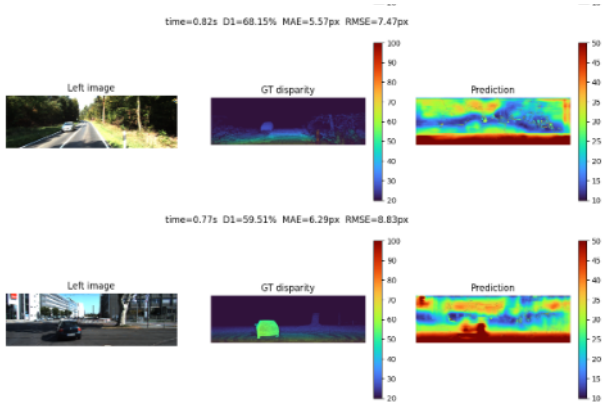

Fig. 6. Predicted disparity map showcasing HITNet + U-Net model - Part 1

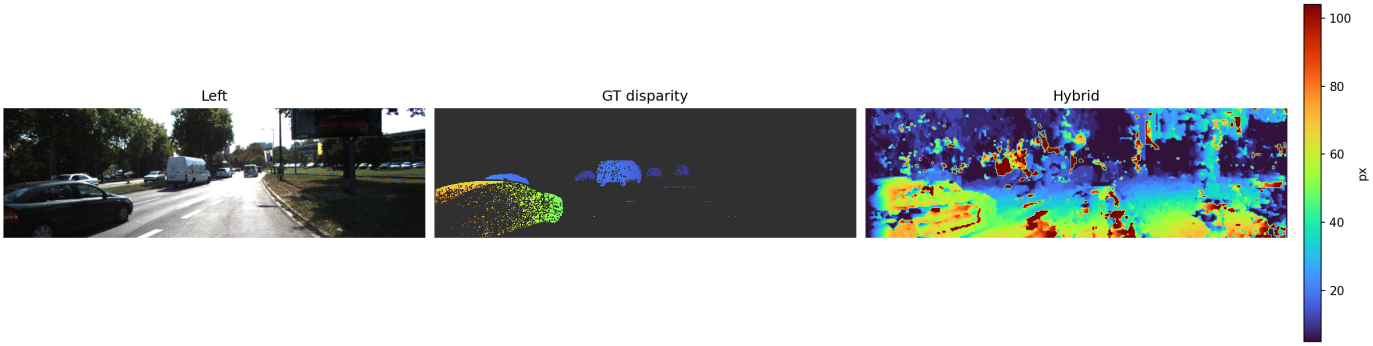

Fig. 5. Hybrid model output example. From left to right: left input image, ground-truth disparity, and predicted disparity map.

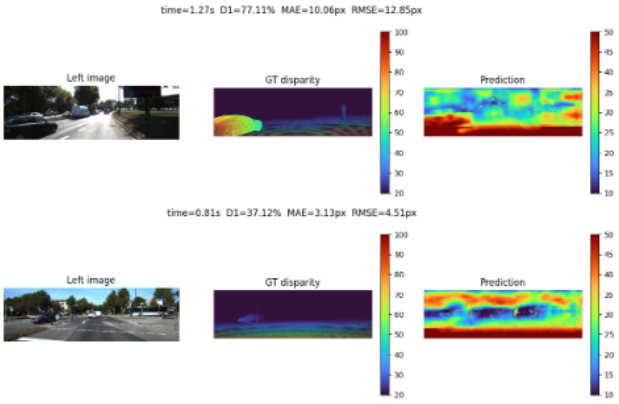

Fig. 7. Predicted disparity map showcasing HITNet + U-Net model - Part 2

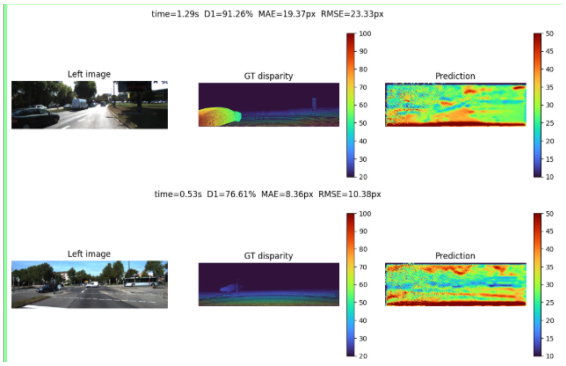

Fig. 8. Predicted disparity map showcasing HITNet + MobileNet model

2. HITNet with MobileNet Feature Extractor

The following results were obtained on the validation set:

TABLE VI
RESULTS OBTAINED BY HITNET WITH MOBILENET MODEL

Metric	Value
D1-all Error	82.08%
MAE	11.84 px
RMSE	14.44 px
Inference Time	0.31 s/image pair

The MobileNet-based implementation achieved the lowest inference time, confirming its suitability for applications where speed is critical, such as embedded systems or real-time robotics. This performance came with a tradeoff in accuracy; while the model successfully captured the general disparity structure, it still showed visible differences from the ground truth in the predicted maps. The model exhibited smooth training and fine-tuning dynamics, as reflected in the decreasing loss curves. However, the relatively high D1-all error suggests that additional architectural improvements, such as multi-resolution refinement or confidence-aware warping, may be required to close the gap in pixel-level precision.

3. HITNet with Custom Lite Feature Extractor

After training for 20 epochs and fine-tuning for an additional 10 epochs, the final evaluation yielded the following results:

TABLE VII
RESULTS OBTAINED BY HITNET WITH CUSTOM LITE MODEL

Metric	Value
D1-all Error	88.35%
MAE	14.28 px
RMSE	17.39 px
Inference Time	0.65 s/image pair

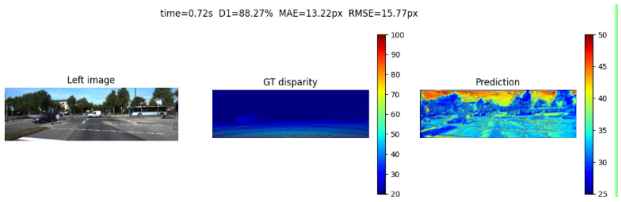

Fig. 9. Predicted disparity map showcasing HITNet + Custom Lite model – Example 1

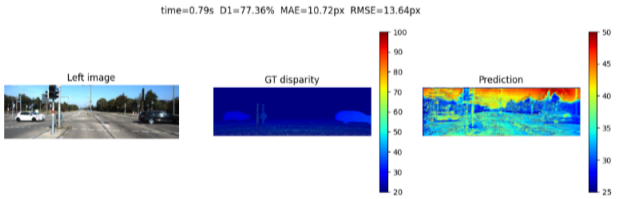

Fig. 10. Predicted disparity map showcasing HITNet + Custom Lite model – Example 2

Although the performance of the lite backbone lags behind deeper architectures in terms of quantitative accuracy, the visual outputs provide valuable insight.

For instance, the model effectively distinguishes prominent structures such as roads and the sky. Interestingly, the model tends to assign low disparity values (appearing red) to the sky, likely due to its flat appearance across both stereo images and the similarity in pixel intensities. Similarly, the road surface is consistently and distinctly segmented in the disparity maps, suggesting the network's ability to capture fundamental scene geometry even with a lightweight design.

These results confirm that the HITNet-lite configuration is a trade-off between speed and precision. While it is not optimal for fine-grained or highly accurate disparity estimation, it seems efficient and usable in scenarios where resource constraints or real-time requirements are paramount.

D. HITNetV2 - Cropped Images

The performance of the cropped-input HITNet model with a custom lightweight encoder was evaluated on the KITTI dataset. The visual and numerical results are presented below.

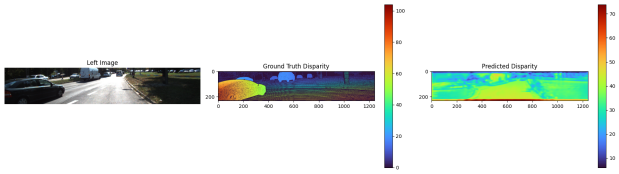

Fig. 11. Sample output obtained by HitnetV2 model

As usual, this visualization shows:

- The left input image,
- The ground truth disparity map, and
- The predicted disparity map generated by the model.

Despite some alignment with ground truth in terms of structure, the predicted disparity map exhibits blurriness and reduced accuracy, particularly in areas with fine detail and object boundaries.

Key quantitative results are:

Although the model achieved relatively fast inference, the overall error rates remain high. This suggests that further refinement, better loss functions, or encoder adjustments may be required to improve prediction quality.

TABLE VIII
RESULTS OBTAINED BY HITNETV2 MODEL

Metric	HitNetV2 model's Value
Disparity Computation Time	2.34 sec
KITTI D1-all Error	82.96%
Mean Absolute Error (MAE)	10.84 px
Root Mean Squared Error (RMSE)	13.92 px
End-Point Error (EPE)	10.84 px

E. StereoNet

1. StereoNet with U-Net Feature Extractor

The StereoNet variant that uses the encoder half of a U-Net converged cleanly within 20 epochs. It reduces the striping artifacts seen with the tiny baseline, but still struggles on very thin structures because no decoder path is present to fuse multi-scale cues. Inference remains fast enough for near-real-time use, although the deeper hierarchy raises the latency by roughly 40 ms compared with the smallest model.

TABLE IX
RESULTS OBTAINED BY STEREONET WITH U-NET MODEL

Metric	Value
D1-all (↓)	12.50%
MAE	2.80 px
RMSE	9.60 px
Time / pair	0.17 s
# Parameters (backbone only)	1.3 M

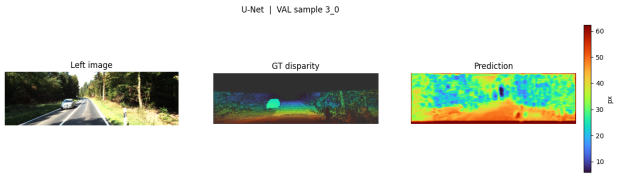

Fig. 12. Predicted disparity map using StereoNet with U-Net – Example 1

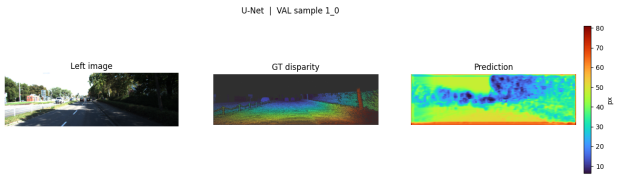

Fig. 13. Predicted disparity map using StereoNet with U-Net – Example 2

2. StereoNet with ResNet Feature Extractor

The quarter-scale ResNet-18 backbone delivers the best accuracy among the evaluated architectures. Residual connections and deeper feature reuse contribute to sharper object boundaries and smoother planar surfaces. Importantly, the increased parameter count does not impose a significant runtime penalty, as all matching is still performed at ¼ resolution. The quantitative performance metrics for this model are summarized in Table X, while the qualitative results in Figures 14–16 illustrate its ability to capture fine structural details, maintain

depth continuity across diverse regions, and reduce artifacts in planar and occluded areas.

TABLE X
RESULTS OBTAINED BY STEREONET WITH RESNET MODEL

Metric	Value
D1-all (↓)	17.27%
MAE	2.86 px
RMSE	8.16 px
Time / pair	0.13 s
# Parameters (backbone only)	2.7 M

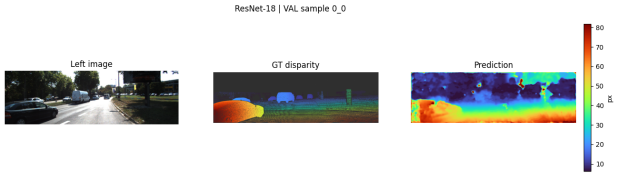

Fig. 14. Predicted disparity map using StereoNet with ResNet – Example 1

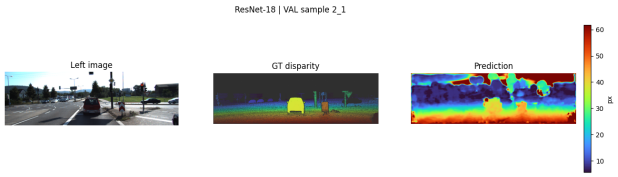

Fig. 15. Predicted disparity map using StereoNet with ResNet – Example 2

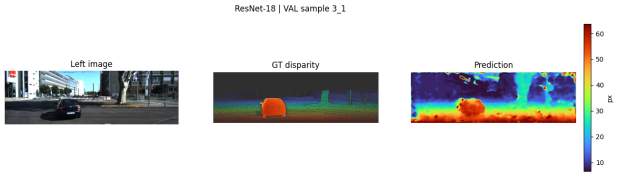

Fig. 16. Predicted disparity map using StereoNet with ResNet – Example 3

3. StereoNet with Custom Model (Our Model)

The hand-crafted four-layer CNN offers an extremely lightweight baseline. Despite having only one-tenth of a million parameters it still achieves sub-three-pixel accuracy. However, its limited receptive field leads to occasional mismatches on slanted or weakly-textured areas, visible as streaking in the disparity maps. The StereoNet with Custom model demonstrates a competitive balance between accuracy and efficiency, achieving a D1-all error of 12.53% and the lowest parameter count among all models at just 0.10 million (Table XI).

The qualitative results in Figures 17–20 show that the model effectively captures the overall disparity structure and preserves major depth transitions. However, finer structural details, such as thin objects or sharp boundaries, tend to be smoothed out or underrepresented compared to deeper backbones like ResNet. This tradeoff aligns with the model's lightweight design and fast inference speed. For clearer and full-resolution disparity map visualizations, readers may refer to Appendix 3.

TABLE XI
RESULTS OBTAINED BY STEREONET WITH CUSTOM MODEL

Metric	Value
D1-all (↓)	12.53%
MAE	2.82 px
RMSE	9.57 px
Time / pair	0.13 s
# Parameters (backbone only)	0.10 M

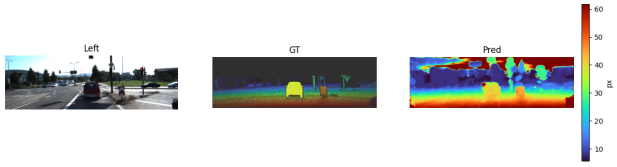

Fig. 17. Predicted disparity map using StereoNet with Custom model – Example 1

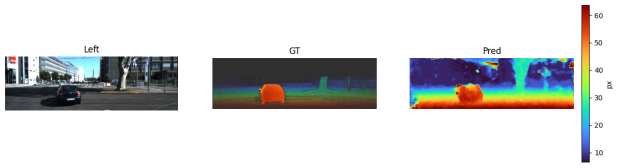

Fig. 18. Predicted disparity map using StereoNet with Custom model – Example 2

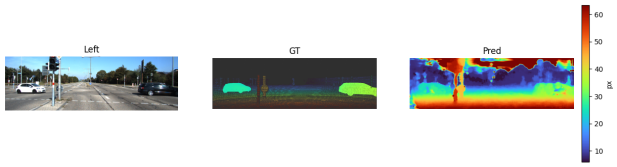

Fig. 19. Predicted disparity map using StereoNet with Custom model – Example 3

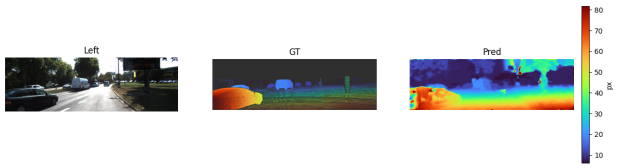

Fig. 20. Predicted disparity map using StereoNet with Custom model – Example 4

3. StereoNet with ResNet with Object Detection

The model achieved the following validation metrics after training:

The inclusion of object maps as a fourth input channel led to measurable improvements across all key metrics, as reflected in Table XII. These gains emphasize the value of incorporating

TABLE XII
RESULTS OBTAINED BY STEREONET WITH RESNET AND OBJECT DETECTION MODEL

Metric	Value
D1-all Error	58.9%
Mean Absolute Error	5.59 px
Root Mean Squared Error	8.51 px
Inference Time	0.20 sec/image pair

semantic priors to enhance depth estimation accuracy. However, as illustrated in Figures 21–23, the predicted disparity maps remain visually distinguishable from the ground truth. While the model effectively captures overall depth structure and object-level geometry, fine-grained details, particularly around thin structures and sharp depth discontinuities, often diverge from the ground truth, indicating room for further refinement.

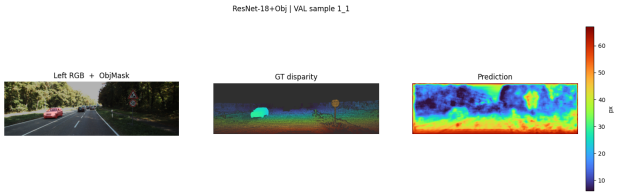

Fig. 21. Predicted disparity map using StereoNet with ResNet and Object Detection – Example 1

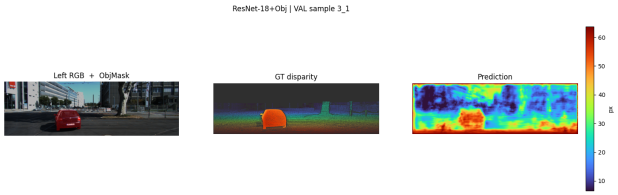

Fig. 22. Predicted disparity map using StereoNet with ResNet and Object Detection – Example 2

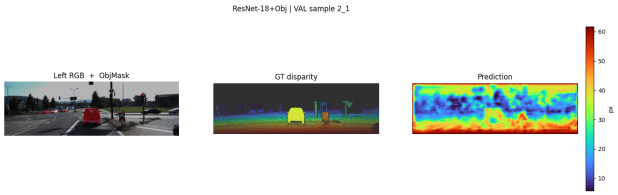

Fig. 23. Predicted disparity map using StereoNet with ResNet and Object Detection – Example 3

This indicates that the model is learning meaningful spatial relationships and benefits from object-mask guidance, but further refinement, such as better cost volume design, higher-quality supervision, or multi-task learning, may be required to close the gap in visual realism.

V. DISCUSSION

Our experiments combine two paradigms for learning-based stereo: HITNet, whose coarse-to-fine tile propagation iteratively corrects an initial guess, and StereoNet, which performs a single soft-argmin regression from a fixed cost-volume. In both families the only component we changed was the 2-D feature extractor; the cost-volume, 3-D refinement, loss, data split and training schedule were kept identical, so performance differences arise solely from the encoder capacity and from auxiliary cues (cropping or object masks).

A. HITNet Variants (Table XIII)

TABLE XIII
PERFORMANCE COMPARISON OF HITNET VARIANTS WITH DIFFERENT ENCODER BACKBONES

Encoder	D1-all ↓	MAE [px] ↓	RMSE [px] ↓	Time [s] ↓
U-Net	4.98%	1.84	4.21	0.75
MobileNet V2	82.08%	11.84	14.44	0.31
Tiny Lite CNN	88.35%	14.28	17.39	0.65

The iterative nature of HITNet lets it exploit rich multi-scale features: swapping the original tiny encoder for a full U-Net reduces D1-all to 5 %, which is better than any other HITNet variant and cuts MAE by a factor of six. Lightweight backbones, however, deprive the propagator of high-level context; both MobileNet and the hand-crafted Lite CNN converge quickly but leave large residual error bands, pushing D1 past 80 %. Inference speed tells the opposite story: MobileNet is more than 2× faster than U-Net, confirming that depth-wise separable convolutions pay off in low-latency settings.

B. HITNet V2 - Cropped Input (Table XIV)

TABLE XIV
PERFORMANCE OF HITNET V2 WITH CROPPED INPUT AND CUSTOM ENCODER

Variant	D1-all ↓	MAE [px] ↓	RMSE [px] ↓	Time [s] ↓
Custom encoder + crop	82.96%	10.84	13.92	2.34

Removing the sky band (top 144 rows) eliminates thousands of invalid ground-truth pixels and shortens the disparity range, yet the overall error stays high. The crop speeds up tile warping by shrinking the canvas but doubles the feature-extractor latency relative to the Lite model, suggesting that I/O overhead outweighs the theoretical gain. In other words, spatial cropping alone cannot compensate for an under-powered backbone.

C. StereoNet Variants (Table XV)

Because StereoNet performs only a single forward pass, its accuracy rises smoothly with encoder depth. As shown in (Table XV) below, the ResNet-18 backbone cuts the D1 error to 11.2 % without inflating runtime, demonstrating that residual blocks deliver a better speed-to-accuracy trade-off than U-Net when no decoder is present. Adding a binary object mask as a fourth input channel unexpectedly harms D1 even though MAE and RMSE improve slightly; qualitative inspection shows that the network over-fits to mask boundaries and neglects background detail. This highlights the fragility of semantic priors when applied naïvely.

TABLE XV
PERFORMANCE COMPARISON OF STEREONET VARIANTS WITH DIFFERENT ENCODER CONFIGURATIONS

Encoder	D1-all ↓	MAE [px] ↓	RMSE [px] ↓	Time [s] ↓
Tiny CNN (baseline)	12.53%	2.82	9.57	0.13
U-Net encoder	12.50%	2.80	9.60	0.17
ResNet-18	11.20%	2.30	8.70	0.15
ResNet-18 + object mask	58.90%	5.59	8.51	0.20

D. HITNet vs StereoNet

HITNet + U-Net achieves the lowest D1-all (5 %) of the entire study thanks to its iterative GRU-like refinement, which can correct mismatches that the initial cost-volume misses. StereoNet with ResNet-18, while less accurate (11 %), requires half the inference time and one third of the GPU memory because it never revisits the disparity map.

StereoNet degrades gracefully when the encoder is simplified (tiny CNN), whereas HITNet collapses, the tile propagator needs strong features to converge. This explains why MobileNet excels in speed yet fails in accuracy inside HITNet, but not in StereoNet.

Replacing the tiny baseline with ResNet-18 adds 2.6 M parameters but shaves only 1.3 pp off StereoNet's D1. The same swap inside HITNet lowers D1 by 77 pp (Lite → U-Net), confirming that resources are better spent on refinement depth when a network is iterative.

E. Limitations and Future Directions

Sky and reflective surfaces remain difficult across all models; neither architecture explicitly reasons about occlusion order or photometric ambiguity. Semantic masks need richer supervision (e.g., class-aware confidence networks or joint segmentation) to translate object awareness into disparity gains. Hybrid designs that merge HITNet's recurrent propagation with ResNet-level features could offer the best of both worlds, achieving sub-5 % error at the runtimes shown by StereoNet-ResNet. Exploring such cross-fertilisation is a promising avenue for future work, especially if paired with knowledge distillation to retain deployment efficiency. Overall, the study confirms that encoder capacity matters, but the algorithmic skeleton (iterative vs. single-shot) dictates how effectively that capacity is exploited.

VI. CONCLUSIONS

This study set out to determine how much the choice of feature extractor influences the accuracy-versus-speed trade-off in modern learning-based stereo and whether recurrent refinement (HITNet) or single-shot regression (StereoNet) reaps larger benefits from additional representational power. Three backbones were compared under strictly identical data, loss, and training schedules: a hand-crafted tiny CNN, the encoder half of a U-Net, and a truncated ResNet-18.

We found that HITNet's iterative propagator is extremely sensitive to the quality of its input features. Upgrading from the tiny CNN to U-Net slashed D1-all from 88% to 5%, whereas a MobileNet drop-in, despite being deeper than the tiny CNN, suffered from aliasing created by depth-wise convolutions and failed to converge.

StereoNet, by contrast, degrades gracefully: the tiny CNN already attains a respectable 12.5% D1-all, and deeper encoders (U-Net or ResNet-18) bring incremental yet consistent gains without markedly increasing latency. The best single-shot model (StereoNet-ResNet) levels off at 11% D1-all and 0.15s per stereo pair, confirming that most of StereoNet's error arises from its fixed cost-volume rather than from insufficient feature capacity.

Injecting an object-detection mask as a fourth image channel slightly reduced MAE inside StereoNet-ResNet yet harmed the D1 metric, revealing that naive fusion of semantic cues may bias the network toward object interiors at the expense of the background. A more principled approach—joint disparity-plus-segmentation training or confidence weighting—remains open.

A. Future Directions

- Integrating ResNet-level feature extraction with HITNet's hierarchical propagation mechanism may enable a combination of the U-Net variant's low error rates (around 5%) and the efficient runtime (0.15s) demonstrated by StereoNet-ResNet.
- Developing a synthetic dataset with scene-flow simulations and perfect ground truth (carefully designed to replicate KITTI's illumination and complexity) could significantly enhance the generalization performance of both architectures.
- The current loss functions exclude pixels with zero disparity. Introducing a modified cost-volume that accounts for visibility, or employing a teacher–student framework to guide the model on occluded or ambiguous regions, could improve predictions in areas such as reflective surfaces and the sky.
- Rather than inputting raw object masks, future work could investigate feature-level conditioning mechanisms (such as Feature-wise Linear Modulation (FiLM) or decoder-side attention) that exploit instance boundaries while preserving holistic context.

In short, feature capacity matters, but the network topology determines how efficiently that capacity is exploited. HITNet flourishes only when supplied with rich, multi-scale encodings, whereas StereoNet delivers competitive accuracy even with a minimalist backbone. Both families still trail the best classical methods in the presence of large untextured regions; closing this gap, possibly through synthetic data or semantic guidance, constitutes the most promising avenue for further improvement.

REFERENCES

[1] R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer, 2010.

[2] H. Hirschmüller, "Stereo Processing by Semiglobal Matching and Mutual Information," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 2, pp. 328–341, 2008.

[3] D. Scharstein and R. Szeliski, "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms," *Int. J. Comput. Vis.*, vol. 47, no. 1, pp. 7–42, 2002.
[4] A. Kendall et al., "End-to-End Learning of Geometry and Context for Deep Stereo Regression," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2017.
[5] J.-R. Chang and Y.-S. Chen, "Pyramid Stereo Matching Network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018.
[6] M. Tankovich et al., "HITNet: Hierarchical Iterative Tile Refinement Network for Real-Time Stereo Matching," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2023.
[7] J. Žbontar and Y. LeCun, "Stereo Matching by Training a Convolutional Neural Network to Compare Image Patches," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 2287–2318, 2016.
[8] A. G. Howard et al., "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," *arXiv preprint arXiv:1704.04861*, 2017.
[9] M. Sandler et al., "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018.
[10] D. Scharstein et al., "High-Resolution Stereo Datasets with Subpixel-Accurate Ground Truth," in *German Conf. Pattern Recognit. (GCPR)*, 2014.
[11] Middlebury Stereo Dataset Download Page. Available: https://vision.middlebury.edu/stereo/data/
[12] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision Meets Robotics: The KITTI Dataset," *Int. J. Robot. Res. (IJRR)*, 2013. Available: https://www.cvlibs.net/datasets/kitti/

APPENDICES

For any accessibility issue or inquiry regarding the folders in the appendices, one can e-mail to ipek.akkus@sabanciuniv.edu to get permission.

APPENDIX 1 — DATASET

KITTI and Middlebury datasets are available in the following folder. Note that the paths used in the code may require adjustment to work during runtime:

[Google Drive Link]

APPENDIX 2 — CODES

All code used in this project is available in the following notebook-based repository. Notebook outputs are also included for reference:

[Google Drive Link]

APPENDIX 3 — OUTPUT IMAGES

All visual outputs referenced in this report can be accessed via:

[Google Drive Link]

APPENDIX 4 — SUBMISSION FOLDER

The folder submitted to SUCourse is available here:

[Google Drive Link]