# Rapport de Projet : Fine-Tuning de Depth Anything V2 avec LoRA

Membres du binôme : [Ton Prénom NOM] & [Prénom NOM du binôme]

Master : [Nom de ton Master]

Date : [Date de rendu]

## 1. Introduction et Objectifs

L'estimation de la profondeur à partir d'une seule image (Monocular Depth Estimation) est un défi majeur en vision par ordinateur. Si des modèles généralistes comme **Depth Anything V2** offrent d'excellentes performances "zero-shot", ils manquent souvent de précision sur des données issues de capteurs industriels spécifiques ou pour des scènes aux caractéristiques particulières.

L'objectif de ce projet est d'adapter (fine-tune) le modèle **Depth Anything V2 Small** sur un jeu de données spécifique acquis avec une caméra **Zivid** (images RGB et nuages de points).

Pour répondre aux contraintes de calcul (GPU limité et temps restreint), nous avons opté pour la technique **LoRA (Low-Rank Adaptation)**. Cette approche permet de spécialiser le modèle sans réentraîner l'intégralité des paramètres, en injectant des matrices de rang faible dans les couches d'attention.

## 2. Données et Prétraitement

Le jeu de données fourni est constitué de paires :

* **Entrée :** Images RGB (.jpg).
* **Vérité Terrain (Ground Truth) :** Nuages de points bruts (.npy) contenant les coordonnées X, Y, Z.

### 2.1. Extraction et Conversion d'Échelle

La première étape a consisté à extraire le canal Z (profondeur) des fichiers .npy.

Un défi majeur a été identifié lors des premiers tests : les données Zivid sont exprimées en millimètres, alors que le modèle attend des valeurs normalisées (mètres). Une Loss initiale de $2.6 \times 10^6$ a mis en évidence ce problème.

**Solution implémentée :** Une conversion automatique (division par 1000) est appliquée pour tout pixel dont la valeur dépasse 100, ramenant les données dans une plage métrique standard ($~0.5m - 2.0m$).

### 2.2. Gestion des Données Manquantes (Masquage)

Les capteurs de profondeur présentent des artefacts (pixels invalides, reflets) représentés par des valeurs NaN ou infinies.

Solution : Nous générons un masque binaire $M$ pour chaque image :

$$M\_{i,j} = 1 \text{ si } Z\_{i,j} \text{ est valide, } 0 \text{ sinon}$$

Ce masque est utilisé lors du calcul de l'erreur pour empêcher le modèle d'apprendre sur du bruit.

## 3. Méthodologie : Implémentation de LoRA

Nous avons utilisé l'architecture Depth-Anything-V2-Small via la bibliothèque Transformers d'Hugging Face. Au lieu d'un "Full Fine-Tuning" coûteux en VRAM, nous avons utilisé la bibliothèque PEFT.

### 3.1. Configuration LoRA

Nous avons ciblé les modules d'attention du Vision Transformer :

* **Modules cibles :** query, value.
* **Rang (r) :** 16.
* **Alpha :** 16.
* **Dropout :** 0.1.

Cette configuration nous a permis de n'entraîner que **294 912 paramètres**, soit environ **1.18%** du total des 25 millions de paramètres du modèle, rendant l'entraînement possible sur une seule carte graphique grand public.

## 4. Protocole d'Entraînement

L'entraînement a été réalisé sur un environnement GPU (Cluster RTX 2080 Ti / RTX 5070 Local) avec les hyperparamètres suivants :

* Fonction de Coût (Loss) : MSE Masquée (Masked Mean Squared Error).  
  $$Loss = \frac{\sum (M \cdot (\hat{Y} - Y))^2}{\sum M + \epsilon}$$
* **Optimiseur :** AdamW ($Learning Rate = 1e^{-4}$).
* **Durée :** 10 Époques.
* **Batch Size :** 4.

## 5. Résultats et Analyse

Les performances ont été évaluées selon trois métriques de référence :

1. **Loss (MSE) :** Erreur quadratique moyenne.
2. **Précision (**$\delta < 1.25$**) :** Pourcentage de pixels où le ratio $\max(\frac{\text{pred}}{\text{target}}, \frac{\text{target}}{\text{pred}}) < 1.25$.

### 5.1. Analyse Quantitative

L'entraînement a montré une convergence rapide et stable.

|  |  |  |
| --- | --- | --- |
| **Époque** | **Loss Moyenne (MSE)** | **Précision (δ<1.25)** |
| **Epoch 1** | 2.0889 | 16.65% |
| **Epoch 5** | 0.2432 | 52.68% |
| **Epoch 10** | **0.0898** | **79.77%** |

On observe une division de l'erreur par un facteur 23 et une augmentation de la précision de **+63 points**. Le modèle est passé d'une prédiction quasi-aléatoire à une reconstruction fidèle de la scène.

### 5.2. Analyse Qualitative

*(Insérer ici l'image comparative 'resultat\_epoch\_10.png' générée par le script)*

**Figure 1 :** *Comparaison Visuelle. Gauche : Image RGB Input. Milieu : Vérité Terrain Zivid (Z). Droite : Prédiction du modèle fine-tuné.*

Visuellement, le modèle parvient à restaurer la géométrie 3D des objets présents. Les contours sont nets et l'échelle de profondeur correspond à la vérité terrain, validant la correction d'unités effectuée en amont.

## 6. Défis Techniques Rencontrés

1. **Compatibilité Matérielle (RTX 5070) :** L'architecture récente du GPU a causé des erreurs NoKernelImage avec les versions stables de PyTorch. Nous avons résolu ce problème en utilisant une version Nightly (CUDA 12.6) et en implémentant un mode de repli sur CPU pour le développement.
2. **Ambiguïté d'Échelle :** L'écart initial entre les prédictions (normalisées) et la vérité terrain (en mm) empêchait la convergence. La normalisation dynamique a été la clé pour débloquer l'apprentissage.

## 7. Conclusion

Ce projet a permis de démontrer la puissance de LoRA pour adapter des modèles de fondation à des tâches industrielles spécifiques.

En entraînant moins de 1.2% des paramètres, nous avons obtenu un modèle capable de prédire la profondeur sur des images Zivid avec une précision de 79.8%, tout en respectant les contraintes de temps et de matériel imposées.