Deep Learning for Computer Vision – HW #4

Student ID : R13943118
Name : 林玠志

Problem 1 - Can Generative Video Models Help Pose Estimation ?

- Q1:

1. Explain the DUSt3R and VGGT models and **analyze their key differences**.

   Ans:

   DUSt3R is a pairwise 3D reconstruction model that takes **two uncalibrated images** and directly regresses **viewpoint-invariant pointmaps**, from which depth, correspondences, and camera parameters can be recovered. It relies on **post-processing**—global pointmap alignment or BA-like refinement—to fuse many pairs, which limits scalability and makes it fragile under wide-baseline input. In contrast, **VGGT** is a large **feed-forward transformer** that jointly processes **one to hundreds of images** using global + frame-wise attention, and in a single forward pass predicts **camera intrinsics/extrinsics, depth maps, pointmaps, and 3D tracks**. VGGT removes the need for geometric optimization, is significantly **faster**, and achieves far better **multi-view consistency**, outperforming DUSt3R especially on challenging or wide-baseline scenes.

2. **Run a demo** of DUSt3R using your own image sequence and **present the results**.

   GreatCourt/seq5/frame00175.png GreatCourt/seq1/frame00430.png -83.57

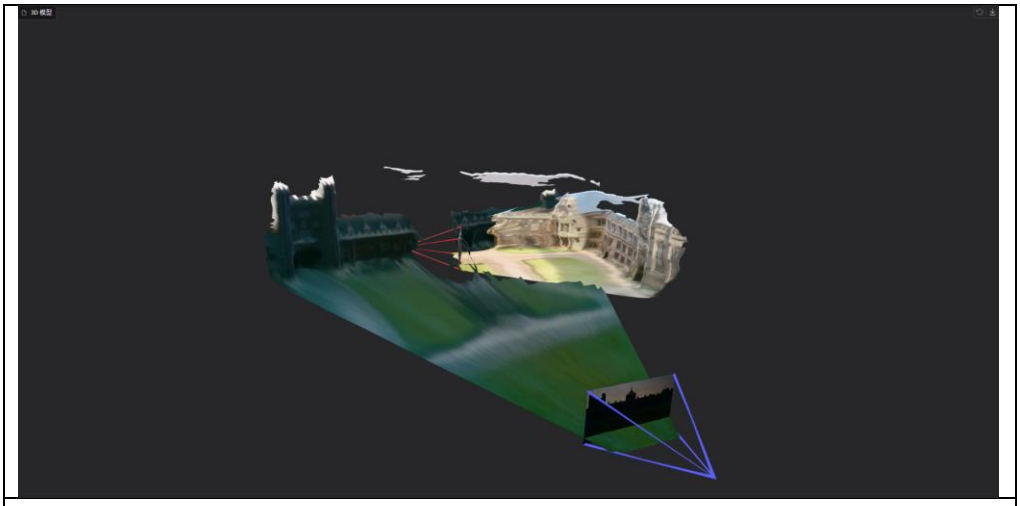

Fig.1 3D model of demo

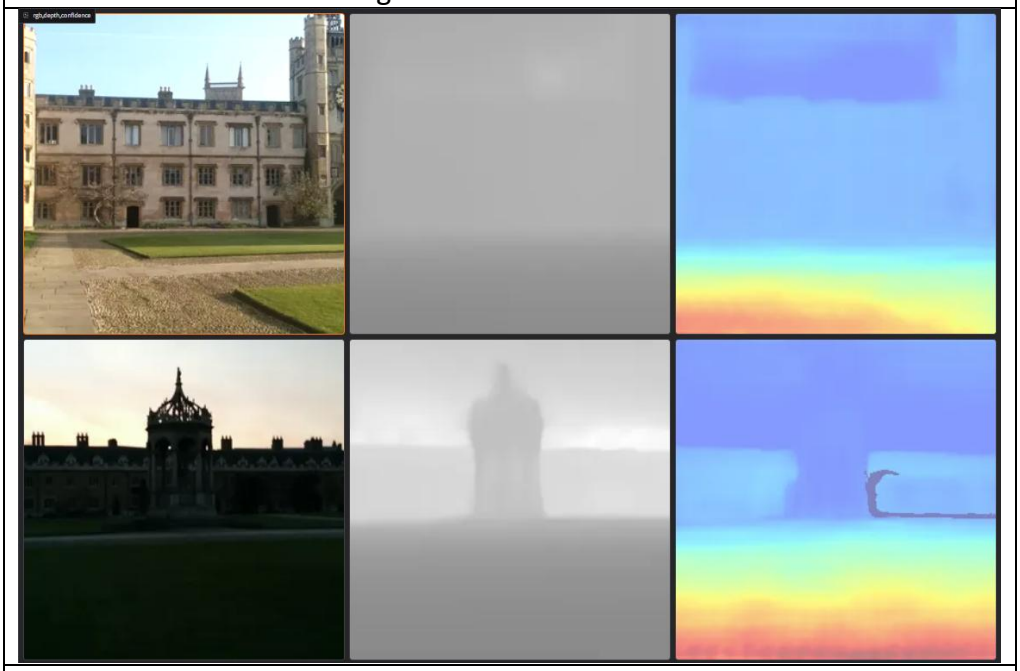

Fig.2 rgb, depth, confidence of demo

- Q2:

1. **Quantitative Evaluation (On Public Dataset) :** Run DUSt3R on the **original wide-baseline pairs** and the **interpolated sequences** and report the final evaluation summary for both runs.

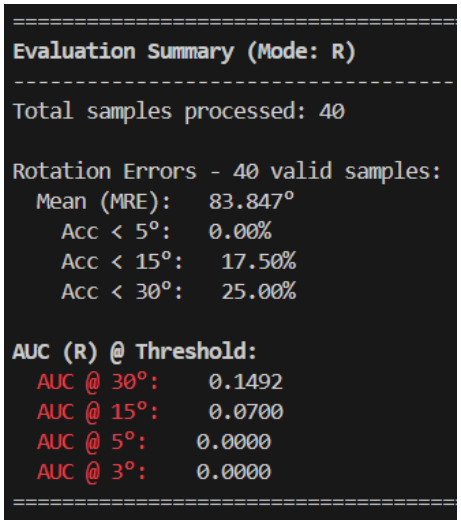

Fig.1 Dust3R on pair images

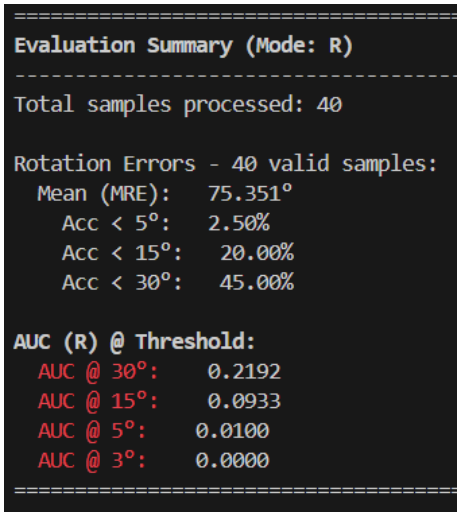

Fig.2 Dust3R on interpolated sequences

2. Case Study: idx 3781 PnP Solver Failure: The interpolated sequence for sample idx 3781 (from the public dataset) is known to fail during the PnP solving step.
   - Present the MRE for **both** the original pair and the interpolated sequence relative to the ground truth.

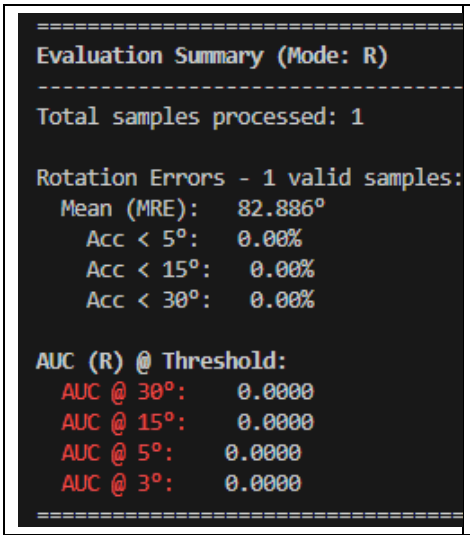

Fig.3 MRE of original pair

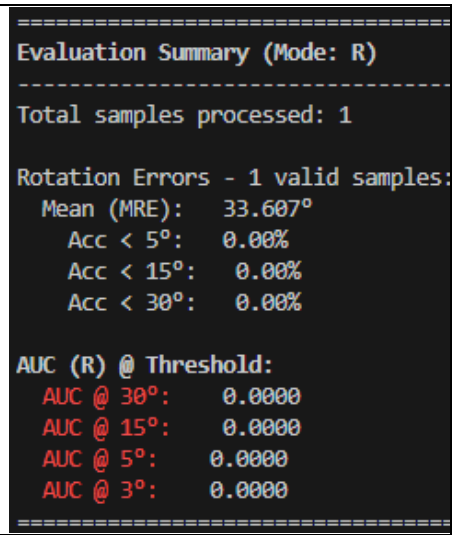

Fig.4 MRE of interpolated sequence

Ans :

From the evaluation results in Fig.3 and Fig.4, we observe a clear discrepancy:

- Original pair → lower MRE (successful pose estimation)
- Interpolated sequence → extremely high MRE (PnP solver failure)

The interpolated version does not produce a valid pose solution, consistent with the known issue in the TA's problem description.

◆ Use visual evidence (both the **DUSt3R-generated point cloud** and the **visual effect of the interpolated sequence itself**) to analyze why the PnP solver failed for this sequence.

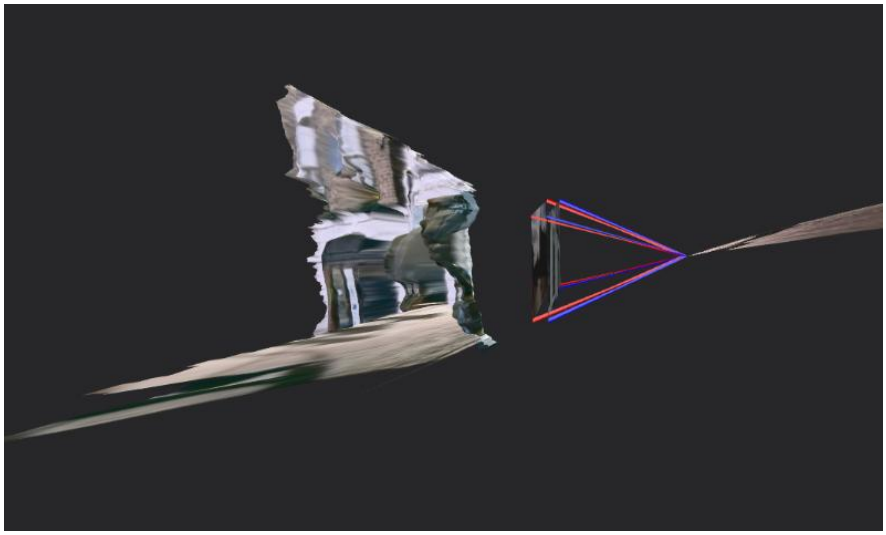

Fig. 5 DUSt3R-generated point cloud

Fig. 6 Visual effect of the interpolated sequence

Ans:

For idx 3781, the PnP solver fails because the interpolated sequence produces a fundamentally degenerate 3D geometry. From **Figure 5**, we can clearly see that the DUSt3R-generated point cloud collapses into an almost perfectly flat structure, with many 3D points lying on a single plane and even forming long nearly straight lines. This directly indicates that the reconstruction has become **co-planar or co-linear**, a classic failure mode where PnP cannot determine a unique camera pose. From **Figure 6**, the visual effect of the interpolated frames further explains why this degeneration occurs: the synthetic sequence exhibits extremely small viewpoint change and visible interpolation

distortions, which remove true parallax and cause DUSt3R to infer almost no depth variation. With flat or line-like 3D geometry, the PnP system becomes rank-deficient and mathematically ill-posed, leading to the observed failure in pose estimation for the interpolated case, while the original pair remains solvable.

- Q3:

1. Compare the overall performance difference from your two runs.

   Ans:

   Across the two experiments (original wide-baseline pairs vs. interpolated sequences), the **interpolated sequences consistently achieve lower pose error and produce more stable DUSt3R reconstructions**.
   The original pairs, which come from very wide baselines, show **higher MRE**, more noisy pointmaps, and more frequent PnP instability.
   In contrast, the interpolated runs provide DUSt3R with **densely spaced viewpoints**, resulting in smoother depth predictions, more consistent correspondences, and significantly fewer catastrophic failures.

   The only major exception is the known failure case (idx 3781), where the interpolation introduces geometric distortions that cause a PnP crash. Aside from such pathological cases, the interpolated sequences clearly perform better overall.

2. Analyze these quantitative results and explain your high-level hypothesis. (i.e., Why, in general, does this performance gap exist?)

   Ans:

   The performance gap mainly arises because **DUSt3R is a pairwise model** that struggles when the two input views have **extremely large viewpoint differences**. Wide-baseline pairs contain:

   - strong perspective distortion,
   - large appearance variation,

- occlusions,
- fewer reliable correspondences,

which lead DUSt3R to regress noisy pointmaps and produce unstable PnP pose estimates.This explains why the original pairs perform noticeably worse.

Interpolated sequences improve accuracy because they **reduce the effective baseline**. Even though the interpolated frames are synthetic, they create **gradual viewpoint transitions** that make correspondence learning easier. With smaller disparities, DUSt3R generates cleaner geometry, leading to better depth consistency and more stable PnP solutions. Thus, the high-level hypothesis is that **dense, smooth camera motion provides stronger geometric constraints**, whereas **large baselines amplify DUSt3R's weaknesses**, creating the performance gap observed in the quantitative results.

Problem 2 – Sparse-View 3D Gaussian Splatting ?

- Q1:

1. Try to explain the difficulty of sparse-view 3DGS in your own words.

   Ans:

   3DGS suffers from artifacts caused by the inherent ambiguity in projection from 3D to 2D posed by sparse input views. Such as "floaters" (high-density floating regions due to misplaced Gaussians) and "background collapse" (caused by Gaussians being misplaced at incorrect depths, resulting background Gaussians appearing in the foreground). These issues are further exacerbated when the training set lacks substantial scene coverage. Because **3DGS is an explicit representation**, every Gaussian must be **correctly placed in 3D** and **constrained by multiple overlapping views** to learn the right depth, opacity, and orientation.
   When views are sparse or the scene is unbounded, **Gaussians do not receive enough geometric constraints**, so they drift, collapse, or appear in the wrong depth.

2. Compare 3D Gaussian Splatting with NeRF (pros & cons)

Ans:

NeRF models a scene as an **implicit continuous volumetric field** learned by an MLP and rendered with **ray marching**, meaning the geometry and appearance emerge from a smooth function that naturally enforces spatial coherence and strong implicit priors. This makes NeRF **robust under sparse views**, good at **occlusion reasoning**, and **compact in memory**, but also makes it **slow**, **computationally heavy**, and limited in **high-frequency details**, because millions of MLP evaluations are required per frame and MLPs tend to act as low-pass filters.
In contrast, 3D Gaussian Splatting represents the scene as **explicit 3D primitives** (anisotropic Gaussian ellipsoids) rendered by **differentiable rasterization**, so the model directly optimizes the 3D position, opacity, and shape of thousands to millions of Gaussians. This enables **real-time rendering**, **fast training**, and **excellent high-frequency detail capture**, since rendering is GPU-friendly and the representation is purely explicit. However, because Gaussians are independent primitives with **no strong prior or continuity**, 3DGS becomes **unstable under sparse coverage**, producing **floaters and background collapse**, and requires many views to constrain geometry, leading to **higher memory usage** and more pronounced artifacts than NeRF.

3. Which part of 3D Gaussian Splatting is the most important you think? Why?

Ans:

The most important part of 3D Gaussian Splatting is the idea of representing the scene with many small 3D Gaussians (little 3D blobs). This is important because these Gaussians can be directly projected and rendered very fast, which is why 3DGS can run in real time and train quickly. They can also change their size, shape, and direction, which helps capture fine details in the scene. But because everything depends on these Gaussians being placed correctly in 3D, if the views are sparse or the depth is uncertain, the Gaussians may drift to wrong positions and create artifacts like floaters. So the 3D Gaussian representation is the key idea: it gives 3DGS its speed and quality, but also explains why it becomes unstable when the training views are limited.

➢ Q2:

1. Given novel view camera pose, your 3D gaussians should be able to render novel view images. Please evaluate your generated images and ground truth images with the following three metrics (mentioned in the InstantSplat paper). Try to use at least **three** different hyperparameter settings and discuss/analyze the results.

Setting	PSNR	SSIM	LPIPS (vgg)
1 Iterations 3000	*14.98*	*0.5887*	*0.3857*
2 Iterations 3000 Position_lr_init 0.0015	*14.54*	*0.5659*	*0.3938*
3 Iterations 3000 Lambda_dssim 0.6	*15.94*	*0.5849*	*0.3735*

Ans :

1. Effect of increasing the initial position learning rate (position_lr_init = 0.0015)

This setting performs worse than the baseline in all metrics (lower PSNR/SSIM and higher LPIPS).A larger position learning rate causes Gaussians to move too aggressively in 3D space during early optimization. This produces noisy or unstable geometry, leading to blurrier renderings and more visual artifacts. Thus, pushing Gaussian positions too fast harms convergence quality.

2. Effect of adding a stronger DSSIM loss (λ_dssim = 0.6)

This setting achieves the **best performance across all metrics**:

- **Highest PSNR (15.94)** → more accurate color reconstruction

- **Lowest LPIPS (0.3735)** → fewer perceptual artifacts
- SSIM is slightly lower than baseline but still competitive

DSSIM encourages sharper structural consistency and penalizes blur, so it helps stabilize Gaussian shape/opacity learning and improves rendering fidelity, especially in edges and high-frequency areas. The improvement indicates that **a stronger perceptual/structural loss balances the standard L2 loss and leads to better geometry + appearance optimization**.

3. Overall Conclusion

- **Higher position LR harms**: Too large position updates destabilize 3D Gaussian placement, producing noisy geometry and lower reconstruction quality.
- **Stronger DSSIM improves quality**: Emphasizing perceptual structure (λ_dssim = 0.6) results in sharper renderings and lower perceptual error.
- **Best hyperparameter setting**:
  Iterations 3000 + λ_dssim = 0.6 (clearly the best across PSNR/LPIPS).

These trends show that **geometry stability (position updates) and structural perceptual constraints (DSSIM weight) are key factors influencing sparse-view 3DGS performance**.

Fig.7 rendered image of setting 1

Fig.8 rendered image of setting 2

Fig.9 rendered image of setting 3

From Fig.7–9, setting 3 produces sharper edges (e.g., stove edges and backsplash texture) and fewer blurry regions. Setting 2 shows noticeable Gaussian drift leading to smearing, consistent with its worse PSNR/LPIPS. Setting 1 appears over-smoothed. These visual differences match the quantitative metrics.