Camera Pose Estimation from Depth and Point Tracking

Deboparna Banerjee
CSE 527 Intro to Computer Vision

1 Goal

This project aims to estimate camera motion and reconstruct a 3D scene using only video frames, without LiDAR or stereo data. The pipeline combines recent depth and feature tracking models to compute camera trajectories and produce a global 3D reconstruction. The focus is on achieving temporal depth consistency, accurate relative poses, and low-drift trajectory estimation.

2 Approach

2.1 Data Collection

Short handheld videos ($\sim$10 s) will be captured in static scenes using an iPhone 15 (non-Pro) at 30 FPS and 1080p–4 K resolution. Frames will be extracted with OpenCV, and approximate intrinsics (focal length $\approx$ 6.8 mm, principal point at image centre) estimated from EXIF metadata or calibration.

2.2 Depth Estimation

Depth maps will be obtained using **Video Depth Anything (VDA)**, which predicts temporally consistent depth $D_t(u, v)$ for each frame I_t. This reduces scale drift compared to frame-wise models such as MiDaS. A small amount of guided or bilateral filtering will remove local noise and improve edge alignment. Each depth pixel will then be converted to 3D camera coordinates.

2.3 Feature Tracking

Dense 2D correspondences will be computed with **CoTracker3**, which produces long, stable tracks (x_i^t, y_i^t) across frames and confidence scores for occlusion handling. These tracks replace traditional SIFT or optical-flow features and serve as the basis for pose estimation.

2.4 Pose Estimation

Given tracked points and depths, 3D coordinates are reconstructed for consecutive frames t and t+1. The rigid transform between them is estimated using the **Umeyama SVD** alignment:

$$p_{t+1} = R_t p_t + t_t,$$

where $R_t \in SO(3)$ and $t_t \in R^3$ minimise

$$\min_{R_t,t_t} \sum_i \|p_{t+1,i} - (R_t p_{t,i} + t_t)\|^2.$$

These pairwise poses are chained to form an initial global trajectory.

2.5 Joint Optimisation

To improve accuracy, a short-window **photometric bundle adjustment (PBA)** will jointly refine depth and pose over 5–10 frames by minimising

$$L = L_{\text{photo}} + \lambda_1 L_{\text{smooth}} + \lambda_2 L_{\text{depth-cons}} + \lambda_3 L_{\text{pose-reg}},$$

where the main term enforces photometric consistency:

$$L_{\text{photo}} = \|I_t - \text{warp}(I_{t+1}, D_t, R_t, t_t)\|_1.$$

Gradients are computed with PyTorch autograd. A global **pose-graph optimisation** (Levenberg–Marquardt in Open3D/g2o) will then reduce accumulated drift across the sequence.

2.6 3D Reconstruction and Visualisation

Each frame's depth map is back-projected into a local point cloud and transformed to the world frame using the refined $(R_{global,t}, t_{global,t})$. Successive clouds are merged incrementally and coloured using RGB values to form a global reconstruction. The combined 3D scene and camera path will be displayed in **Open3D** and exported as a `.ply` model.

2.7 Evaluation

- **Pose accuracy:** Compare trajectories with pseudo ground truth from **MapAnything**, **VGGT**, and π^3 using Absolute Trajectory Error (ATE) and Relative Pose Error (RPE).
- **Depth quality:** Compute per-pixel RMSE and check temporal consistency of depth maps.
- **Photometric consistency:** Evaluate mean L_1 error between real and reprojected frames.
- **Visual quality:** Inspect reconstruction sharpness and loop-closure stability.

3 Expected Challenges

- Keeping depth scales consistent across frames when using monocular video models.
- Minimising drift when chaining relative camera poses into a global trajectory.
- Ensuring convergence and stability of the joint optimisation while keeping computation reasonable.