CMPE 249 Project Proposal

Camera-only 3D Semantic Occupancy Prediction via BEV Backbone and OccFormer

Option 2 (Fine-Tune & Train)

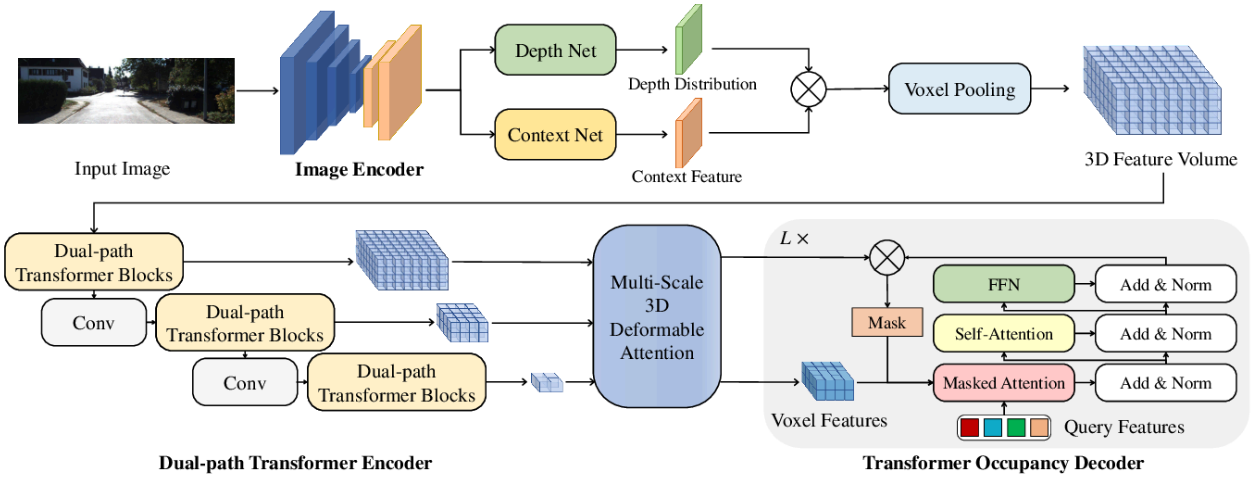
Su Hyun Kim (018219422)

Motivation & Use Case

This project studies camera-only 3D semantic occupancy as a dense scene representation for autonomous driving. Instead of predicting boxes alone, the model estimates which 3D space around the ego vehicle is occupied and by which semantic class (e.g., road, vehicle, pedestrian). This representation is less brittle under occlusion and is directly consumable by planning. Public benchmarks on nuScenes and Waymo now provide compatible labels and protocols, which lets us pretrain on nuScenes and fine-tune on Waymo to analyze domain shift. We will report both accuracy (mIoU) and systems metrics (memory, model size) on the server and discuss the trade-offs between them.

Selected Model

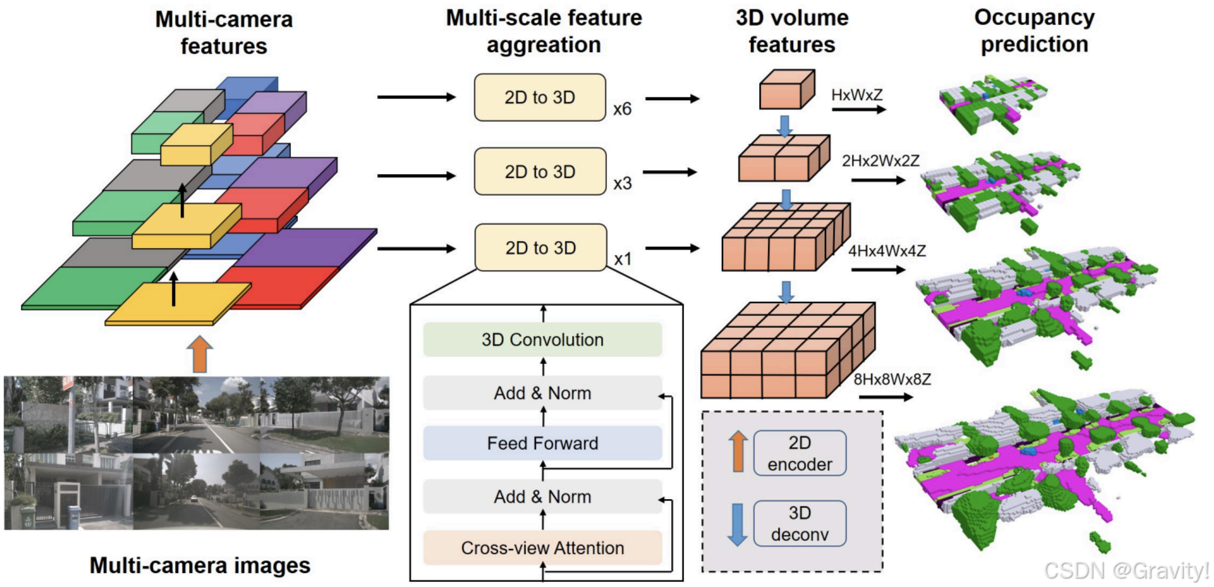
We are going to select **OccFormer** (ICCV' 23) as the base model. OccFormer is a camera-only 3D semantic occupancy network that lifts multi-view images into a 3D voxel feature volume and performs transformer-based 3D reasoning to predict per-voxel semantic labels. It was proposed specifically to overcome the limited receptive field and heavy compute of classical 3D CNNs by introducing dual-path transformers for long-range yet efficient 3D encoding.



The model is contains four steps

1. **Image encoder** extracts multi-scale features from synchronized, calibrated cameras.
2. **Image-to-3D lifting** estimates discrete depth per pixel (LSS-style) and aggregates features into a 3D voxel volume in the ego frame.
3. **Dual-path transformer encoder** mixes local windowed attention in 3D with a global BEV context path, then fuses them adaptively to balance range and efficiency.
4. **Transformer occupancy decoder** predicts voxel-wise semantic masks (Mask2Former-style formulation).

We select OccFormer because it is a well-established camera-only occupancy model with public code/checkpoints, strong reported results on nuScenes-based occupancy benchmarks, and an architecture that scales reasonably while retaining long-range 3D reasoning.



Datasets & I/O

We adopt a two-corpus setup for domain adaptation. The starting point is OccFormer pretrained on nuScenes semantic occupancy; the target is **Waymo** (Occ3D-Waymo) as the new dataset for fine-tuning. Using both allows us to compare in-domain performance and cross-dataset generalization while keeping inputs and outputs aligned.

Compared to nuScenes, Waymo uses **five surround cameras** (front, front-left, front-right, side-left, side-right) and differs in scene statistics and semantic ontology, which creates a meaningful domain shift. We follow the Occ3D-Waymo occupancy volumes and evaluation masks (e.g., FOV/visibility) to ensure fair comparisons. Training code ingests Waymo's five streams through the same multi-view interface, using per-frame intrinsics/extrinsics and timestamps. Unless noted, we report **mIoU** under each dataset's native class set; for qualitative cross-dataset figures we may also show a reduced, shared label subset (e.g., drivable, vehicle, pedestrian, infrastructure) for readability.

All experiments are camera-only. Each sample consists of synchronized RGB from the available cameras (six for nuScenes, five for Waymo), their calibrations, and ego poses. The network outputs a voxelized 3D semantic occupancy volume in the ego-centric frame over a fixed spatial range around the vehicle. Although Waymo has five cameras (not full 360° like nuScenes), we still predict over the full ego-centric grid; evaluation masks exclude unobserved regions so the model is not penalized for areas outside the camera frustums. This keeps the representation consistent across datasets while respecting each dataset's field of view.

Milestone

Dates	Goal	Key Deliverables
09/22–28 (W1)	Setup & sanity	● Pull OccFormer repo + pretrained weights ● Run one nuScenes-val inference
09/29–10/05 (W2)	Understand the model	● Walk through code (encoder → lifting → dual-path → decoder) ● Document I/O shapes and loss
10/06–12 (W3)	Integrate Waymo (Occ3D-Waymo)	● Add Occ3D-Waymo loader ● Run metric script and do **zero-shot** eval (nuScenes-pretrained on Waymo-val)
10/13–19 (W4)	Main fine-tune	● Fine-tune OccFormer on Waymo with LR sweep {2e-4, 1e-4, 5e-5}
10/20–26 (W5)	Main fine-tune	● Fine-tune OccFormer on Waymo with LR sweep {2e-4, 1e-4, 5e-5}
10/27–11/02 (W6)	Efficiency pass	● Export best model and measure latency/FPS/memory/model size on server GPU
11/03–09 (W7)	Cross-checks	● Re-evaluate best Waymo-FT on nuScenes dataset (generalization check)
11/10–16 (W8)	Draft results	● Write short Results section (what improved, how much it cost)
11/17–23 (W9)	Polish	● Tighten figures/tables ● Finalize discussion & limitations
11/24–27 (W10)	Finalize	● Finalize PDF + slides