Neet Mehulkumar Mehta

India | neetmehta19@gmail.com | +91 9484508208 | GitHub: https://github.com/neetmehta
Website: neetmehta.github.io

MACHINE LEARNING ENGINEER

Perception AI Engineer with 2+ years of experience building **production-grade autonomous driving perception systems**, specializing in **multi-sensor fusion, BEV-based 3D object detection, and embedded deployment**. Proven track record of improving model accuracy and latency through **geometry-aware learning, CUDA/TensorRT optimization, and end-to-end MLOps pipelines**, and shipping perception stacks to vehicles under **SIL/HIL and V-model processes**. Recently applied **image-conditioned generative vision models** as an extension of perception expertise, focusing on **spatial alignment, consistency, and controllable generation**, without shifting away from core perception and robotics fundamentals.

EDUCATION

Worcester Polytechnic Institute (WPI) **Worcester, MA**
Master of Science- Robotics Engineering, GPA- 3.85/4.00 Dec 2022
Nirma University **Ahmedabad, India**
Bachelor's in mechanical engineering, GPA- 7.8/10.00 May 2020

KEY SKILLS

- **Programming Skills**: C++, CUDA, Python, MATLAB
- **Tools and Libraries**: ROS2, Pytorch, TensorRT, ONNX, CARLA simulator, AWS, PCL (Point Cloud Library), OpenCV, Docker, Git, ComfyUI, Blender 3D, Solidworks.

WORK EXPERIENCE

Aavaran **Bhuj, India**
Applied Generative Vision Engineer, Self Employed Mar 2025 – Present
Pytorch, ComfyUI, Docker, Vastai

Applied generative vision techniques as an extension of my perception and geometry background, focusing on spatial alignment, scale consistency, and deployment efficiency.

- Designed and deployed **image-conditioned generative vision pipelines** for interior visualization and virtual try-on, emphasizing **geometry consistency, spatial alignment, and controllable generation**.
- Built **text- and image-conditioned scene editing** workflows using **Qwen 2.5.11 Image Edit** for realistic in-context placement of home textiles with preserved scale, occlusion, and lighting.
- Created an **AI brand character** using **FLUX.2-dev**, training a **LoRA** for identity consistency across poses and viewpoints, and applied **ControlNet** for pose and composition control in advertising imagery.
- Developed a **virtual try-on system for traditional apparel**, transferring garments onto real subjects and AI characters while preserving **fabric texture, drape, and pose alignment**.
- Extended image pipelines toward **short-form generative video** using **WAN 2.2**, focusing on **temporal coherence**.
- Optimized inference using **quantized GGUF / FP8 models** on **RTX 4090 (Vast.ai)** and consumer GPUs via **custom ComfyUI workflows**.

TORC Robotics **Austin, TX**
ML Engineer II Feb 2023 – Feb 2025
C++, CUDA, ROS2, Python, Pytorch, AWS, TensorRT

- Served as **SCRUM Master for the perception team**, driving sprint planning, backlog grooming, cross-team coordination, and delivery tracking, while ensuring alignment between research, systems, and deployment milestones for a complex multi-sensor autonomy stack.

Learned Sensor Fusion

- Defined **clear upstream and subsystem interface requirements**, specifying latency, synchronization, and data quality constraints for **image streams (ISP), LiDAR point cloud aggregation, and depth map inputs**, translating system-level constraints into well-scoped component requirements.
- Conducted a **POC evaluation of learned multi-sensor object detectors** (including **BEVFusion** and related architectures), ultimately shortlisting BEVFusion as the baseline for further architectural innovation and production feasibility.
- Implemented **depth-aware BEV unpooling**, generating **dense 3D camera points and features** from monocular depth outputs, significantly improving camera-to-BEV feature richness and fusion quality.
- Designed and implemented an **attention-based learned spatio-temporal feature aligner** to jointly align **LiDAR BEV features and dense 3D camera features** across **multiple sensors and time steps**, achieving a **+2.5% mAP improvement** over vanilla BEVFusion on proprietary datasets.
- Built a robust **model conversion and deployment pipeline**, converting models from **PyTorch → ONNX → TensorRT**, enabling production-grade inference on embedded platforms.

- Applied **post-training quantization (PTQ) to FP16**, reducing end-to-end inference latency by **~50%** while preserving detection accuracy.
- Designed and implemented a **custom TensorRT BEV pooling plugin** capable of efficiently pooling **~1M camera points into a 3D BEV grid**, addressing a critical performance bottleneck.
- Developed **custom CUDA kernels** for performance-critical operations including **voxelization, Non-Maximum Suppression (NMS), and bounding box decoding**, demonstrating deep expertise in low-level GPU optimization.
- Integrated the optimized TensorRT model into a **proprietary Inference Engine**, creating a clean abstraction layer for deployment across embedded hardware targets.
- Architected a **highly configurable ROS2 perception node** with **sensor-level abstraction**, allowing camera and LiDAR vendors to be swapped without impacting internal data formats; the node supports **arbitrary numbers of cameras and LiDARs** and is extensible to **radar integration** via modular data converters.
- Delivered outputs in **team-aligned interface formats** defined through **ICP documents**, enabling smooth downstream consumption by planning and control teams.
- Executed **full system integration and deployment on a production truck**, following a rigorous **V-model development process** including **unit testing, SIL, and HIL validation**.
- Implemented **rich perception visualization tools** using **RViz and Foxglove**, enabling efficient debugging, performance analysis, and cross-team collaboration.
- Built a **fully automated, end-to-end model lifecycle and MLOps pipeline** spanning **training → versioning → conversion → validation → deployment**, enabling seamless onboarding of newly trained models or minor architectural changes; models were **versioned and tracked in a centralized registry using Comet**, automatically pulled with all required artifacts, converted **PyTorch → ONNX → TensorRT** for embedded compute, validated through **unit, component, and performance tests**, and deployed via **CI/CD**, requiring only a **model version update in the ROS2 node** to trigger conversion, validation, and metric generation for the new model.

Domain Adaptation

- ***Implemented a Proof-of-Concept (POC) for Unsupervised Domain Adaptation (UDA) based on the GUDA (Geometric Unsupervised Domain Adaptation) framework to address sim-to-real domain gaps in autonomous driving perception models.***
- ***Utilized large-scale synthetic datasets (thousands of images) provided by Parallel Domain, enabling geometry-aware adaptation without requiring labeled real-world data.***
- ***Designed and trained a geometry-guided adaptation pipeline for semantic segmentation and monocular depth estimation, enforcing cross-domain consistency using depth, camera pose, and geometric alignment constraints instead of pixel-level supervision.***

Monocular Depth Estimation

- Developed a state-of-the-art unsupervised monocular depth estimation model for autonomous driving scenarios, leveraging view synthesis and photometric consistency losses to learn depth without ground-truth supervision.
- Designed and implemented a self-supervised training pipeline using monocular video sequences, jointly optimizing depth and ego-motion through differentiable image warping and multi-scale reconstruction losses.
- Incorporated geometric constraints including smoothness regularization, edge-aware depth priors, and occlusion masking to improve depth stability and reduce artifacts in dynamic driving environments.
- Achieved robust depth prediction performance across diverse lighting and weather conditions by applying temporal consistency losses, data augmentation strategies, and scale-invariant normalization techniques.
- Evaluated adapted models on real-world driving data, demonstrating **improved generalization for both segmentation and depth estimation** compared to non-adapted baselines, validating the feasibility of GUDA-based sim-to-real adaptation.

TORC Robotics — Blacksburg, VA

Perception Engineer – Co-Op — Jan 2022 – Aug 2022

C++, Python, Pytorch, AWS, TensorRT

- Worked on Data extraction and data postprocessing for deep learning architectures. Established extendable pipeline to generate detailed metrics report for each Deep learning model.
- Developed automated hyperparameter tuning stage in AWS Sagemaker. Used Bayesian search to find optimal hyperparameters.

PROJECTS

Synthetic Data Generation for Self Driving Cars using CARLA simulator — Aug 2022 – Present

Python, Pytorch, CARLA

- Designed and developed a **scalable synthetic data generation toolkit using CARLA**, enabling **unlimited dataset creation** for autonomous driving perception tasks including **depth estimation, optical flow, instance segmentation, semantic segmentation, 2D object detection, and 3D object detection**.
- Implemented a **highly configurable sensor simulation framework**, allowing dynamic modification of **sensor placement, intrinsics/extrinsics, field-of-view, and update rates**, supporting realistic multi-sensor perception experiments.

- Built controls to **programmatically vary scene complexity**, including **traffic density, pedestrian count, and agent behavior**, enabling systematic stress-testing of perception models under diverse operating conditions.
- Enabled **configurable data capture frequency and synchronization**, supporting fine-grained control over temporal resolution for video-based and multi-frame learning pipelines.
- Generated **multi-modal, time-synchronized annotations** across tasks, facilitating research in **sensor fusion, self-supervised learning, domain adaptation, and sim-to-real transfer**.
- Designed the repository with **modular, extensible architecture**, allowing easy integration of new sensors, annotation types, and export formats for downstream training pipelines.

June 2022 – July 2022

Self-Supervised Monocular Depth Estimation (Monodepth2) from scratch

Python, Pytorch

- Implemented **Monodepth2 from scratch in PyTorch** to gain a deep understanding of **self-supervised monocular depth estimation**, including view synthesis, photometric reconstruction, and scale-invariant depth learning.
- Reproduced the full **training pipeline** using monocular video sequences, implementing **pose estimation, differentiable image warping, multi-scale supervision, and automasking** as described in the original paper.
- Developed and integrated key loss components including **photometric reprojection loss, edge-aware smoothness regularization, and minimum reprojection selection** to handle dynamic scenes and occlusions.
- Validated implementation correctness through **qualitative depth visualization and quantitative evaluation** on standard driving datasets, ensuring consistency with reported Monodepth2 behavior.

3D Object Detection in Point Cloud using Voxelnet

Sept 2021 – Dec 2021

Python, Pytorch, OpenCV

- Implement a 3D detection network (VoxelNet) on the KITTI vision (Point Cloud) benchmark dataset to unify feature extraction and bounding box prediction into a single-stage, end-to-end trainable deep network.

Feb 2022 – May 2022

Multinet-2: A Multitask learning architecture for Semantic, Depth, and Normal prediction

Python, Pytorch

- Implemented Deep CNN architecture that can predict Semantic mask, estimate Depth, and normal simultaneously.
- Increased combined inference speed to 1.75x with a slight accuracy drop.

"Attention Is All You Need" – Transformer Reimplementation (Learning Project)

Apr 2025– May 2025

Python, Pytorch

- Reimplemented the **Transformer ("Attention Is All You Need") from scratch in PyTorch** for English–German translation to gain a deep understanding of self-attention and encoder–decoder architectures; validated results qualitatively.

BERT from Scratch

Jun 2025 – Present

Python, Pytorch, OpenCV

- Reimplementing **BERT from scratch in PyTorch** to deeply understand **Transformer-based language models**, including token embeddings, multi-head self-attention, encoder blocks, and masked language modeling.
- Developing a **financial sentiment analysis pipeline** on top of the implementation, fine-tuning the model for domain-specific text classification.
- Validating implementation through **qualitative analysis and basic quantitative checks**, focusing on architectural correctness and training dynamics.

Real-time hand gesture recognition using SSD-MobileNet and Transfer Learning

Oct 2021– Dec 2021

Python, Tensorflow, OpenCV

- Trained object detection model consisting of 5gestures by Transfer Learning to a pre-trained SSD-MobileNet model and TensorFlow object detection API on RTX 2060 MAX-Q GPU.
- Achieved 80% accuracy for a class.
- Trained lightweight model suitable for real-time hand gesture recognition.

Popular CNN architectures

Jan 2022 – Jan 2023

Python, Pytorch

- Implementing popular Deep Learning architecture for 2D/3D object Detection, Semantic and Instance Segmentation, and Depth Estimation.
- The purpose of these projects is to develop a strong foundation of the theoretical and practical aspects of Deep Learning.
- You can find all the projects on my GitHub. Some of them might still be in development.

Implementation and Visualization of Autonomous Robot Path Planning Algorithms

Feb 2021 – May 2021

Python

- Implemented discrete and sampling-based algorithms such as A*, Weighted A*, Dijkstra, Probabilistic Road Map (PRM), Rapidly exploring Random Tree (RRT), RRT*, and Informed RRT* to navigate through obstacles in a 2D environment.