Raghav Nallaperumal

Worcester, MA • raghavnallaperumal753@gmail.com • (508) 615-3070
linkedin.com/in/raghav-nallaperumal • github.com/N-Raghav • Portfolio

EDUCATION

Worcester Polytechnic Institute — Worcester, MA
Master of Science in Robotics Engineering | *GPA: 4.0/4.0* — Expected Aug 2027
- Relevant Coursework: Robot Dynamics, Robot Controls, Machine Learning for Robotics, Computer Vision

Shiv Nadar University — Chennai, India
Bachelor of Technology in Computer Science and Engineering (IoT) | *GPA: 8.3/10.0* — Graduated June 2025

TECHNICAL SKILLS

- **Programming:** C++, Python (PyTorch, OpenCV, Kornia), MATLAB
- **Robotics:** ROS 2 (Humble), MoveIt 2, Nav2, Gazebo, PyBullet, Simulink
- **Machine Learning:** Deep Reinforcement Learning (REINFORCE, Actor-Critic, A3C), Supervised Learning, CNNs, Policy Gradient Methods, Value Function Approximation
- **Controls & Planning:** PID, LQR, MPC, System Identification, Jacobian-based Control, Forward/Inverse Kinematics, Trajectory Optimization
- **Perception:** Camera Calibration, VIO (MSCKF), SLAM, Structure from Motion, Bundle Adjustment, RANSAC, Optical Flow
- **Hardware:** Franka Emika Panda, Intel RealSense D435, Crazyflie 2.0, Beckhoff TwinCAT PLC
- **Tools:** Docker, Git, SLURM/HPC Clusters, CUDA, Blender

EXPERIENCE

Graduate Research Assistant — Manipulation and Environmental Robotics Lab — Aug–Dec 2025
Advisor: Dr. Berk Calli | *GitHub* — Worcester Polytechnic Institute
- Built a ROS 2 perception and control pipeline for the Franka Panda that picks and sorts waste objects from a moving conveyor belt, using an Intel RealSense and an antipodal grasping network.
- Replaced the existing Arduino conveyor controller with a Beckhoff TwinCAT PLC for deterministic real-time actuation.

IoT Software Intern — May–Jul 2024
HCLTech | *GitHub* — Chennai, India
- Built a stress-test simulator for industrial IoT protocols (ModBus, CAN, BACnet, PROFINET) by generating high-throughput mock sensor streams for validation.
- Developed an Android app for scanning and tracking BLE asset tags in a manufacturing environment.

PROJECTS

EinsteinVision: Monocular Perception Pipeline | **GitHub** | *Python, PyTorch, Blender* — Jan–Mar 2026
- Built end-to-end monocular perception pipeline for autonomous driving: trained custom YOLOv6 on BDD100K (mAP50 0.564), integrated DepthAnythingV2 for metric depth, FCOS3D for 3D pose, and UFLDv2 for lane detection.
- Implemented ego-motion estimation via RANSAC-based visual odometry; estimated vehicle velocities, predicted collisions, and detected brake lights and speed bumps using classical CV and deep learning.
- Rendered 13 urban driving scenes (27K frames) in Blender using procedural geometry and WPI's GPU cluster via SLURM.

Deep RL for Robotic Picking | **GitHub** | *Python, PyTorch, PyBullet* — Feb–Apr 2026
- Implemented REINFORCE, Actor-Critic, and A3C algorithms for LunarLander-v2 (mean reward 100+) and PyBullet Kuka grasping (25%+ success rate over 100 episodes).
- Designed CNN-based actor-critic architecture for image-based RL; achieved stable training using gradient clipping, advantage normalization, and bootstrapped returns.
- Deployed asynchronous multi-worker A3C training with shared global network for parallelized experience collection.

Structure from Motion and NeRF | **GitHub** | *Python, OpenCV, NumPy, PyTorch* — Feb – Mar 2026
- Implemented classical SfM pipeline from scratch: RANSAC-based feature matching, 8-point fundamental matrix estimation, essential matrix decomposition, cheirality-based camera pose disambiguation, and linear/nonlinear triangulation.
- Built PnP solver with RANSAC for camera registration from 2D-3D correspondences; refined poses and 3D points jointly via sparse bundle adjustment, reducing reprojection error by 36% (33.2 to 21.2 pixels).
- Trained NeRF (Neural Radiance Fields) on WPI's GPU cluster for novel view synthesis using volumetric rendering with positional encoding; achieved 27.3 dB PSNR / 0.90 SSIM (Lego) and 27.5 dB PSNR / 0.84 SSIM (Ship).

Quadrotor Trajectory Control & System Identification | **GitHub** | *Python, PyBullet, Crazyflie 2.0* — Nov – Dec 2025
- Implemented PD and LQR controllers in PyBullet and validated on real Crazyflie 2.0 hardware; performed system identification reducing sim-to-real RMSE from 4.8mm (simulation) to 0.8mm (hardware), with LQR maintaining position error below 2cm.
- Implemented polynomial trajectory generator for smooth, dynamically feasible 3D flight paths.