Abhijit Mahalle

amahalle60@gmail.com | linkedin.com/in/abhijitmahalle | Portfolio: abhijitmahalle.github.io | GitHub: github.com/abhijitmahalle

Domain Skills: Robotics, Perception, Computer Vision, Autonomous Systems, Machine Learning, Deep Learning, Software Development, Data Structures, Algorithms

EDUCATION

Master of Engineering, Robotics | University of Maryland, College Park **GPA: 3.64** | 05/2023

Bachelor of Engineering, Mechanical Engineering | University of Mumbai, India **GPA: 3.86** | 05/2018

SKILLS

Languages: C++, Python, MATLAB

Softwares & Tools: Linux, ROS, Gazebo, RViz, Git, Docker, CMake, GCP, SolidWorks

Libraries: OpenCV, PyTorch, TensorFlow, NumPy, SciPy, sklearn, pandas, Matplotlib, GTest, pytest

Deep Learning Architectures: MLP, CNN, Autoencoder, VAE, RNN, LSTM, GAN

EXPERIENCE

Perception and Robotics Group, UMD — College Park, MD

Research Assistant **06/2022 - 05/2023**

- Achieved an IoU of 0.75 for **segmentation** and Average Endpoint Error of 0.15 m/s for **motion estimation** task by training encoder-decoder networks to estimate scene **depth**, **camera** and **object poses** from the **event camera** output.
- Combined the two estimates to generate **optical flow** to jointly optimize the two networks.
- Created a real-world indoor ground-truth dataset by fusing data-streams from 3 event cameras, classical camera, and 2 IMU to train VIO and SLAM models. Used Vicon motion capture system and Mujuco simulator.
- Constructed a pipeline to **calibrate** event and classical cameras simultaneously by reconstructing grayscale images from the event-stream using deep neural network.

Jacobs Engineering — Mumbai, India

Design Engineer **09/2018 - 07/2021**

- Designed piping systems using CAD tools for fluid transfer within process plants considering process requirements.

PROJECTS

Structure from Motion | Python, OpenCV — Link

- Reconstructed a **3D scene** and simultaneously obtained **camera poses** from a set of images using their feature point correspondence, **epipolar geometry**, triangulation, **PnP, bundle adjustment**, linear, and non-linear optimization.

April tag detection and tracking | Python, OpenCV — Link

- Detected and tracked an April Tag in a video sequence by background removal using FFT and Harris corner detection.
- Decoded the tag by identifying its position and orientation using homography.
- Superimposed a custom image and placed a **3D virtual** cube on tag using **homography, calibration**, **projection** matrices.

Home Organizing Robot | ROS, C++, Software Development, SLAM — Link

- Developed a **ROS** package in **C++** by **Agile** Iterative Process with GitHub CI and test-driven development using **GTest** for Tiago mobile manipulator for indoor search and object manipulation.
- Used **MoveBase** for autonomous navigation, **MoveIt** for manipulator control, OpenCV for filtering and object detection.
- Maintained software version control using Git, checked build using **Travis CI**, and achieved a code coverage of 80%.

VOXL m500 drone | ROS, C++, Aerial Robotics

- Developed a C++ pipeline for **real-time detection** and **tracking** of humans using monocular camera and YOLO-v5 classifier.
- Used 1D Lidar sensor for depth sensing. Programmed PD controllers for precise yaw and depth control of the drone.
- Constructed a ROS node for off-board position control of drone to autonomously follow a moving April tag, move through set way-points, and to move in a figure-of-eight.

Design, control, and simulation of Mars Rover | CAD, SolidWorks, ROS, Gazebo

- Designed a CAD Model of Curiosity Rover in SolidWorks with a 5-DOF KUKA KR5 robotic manipulator attached to it.
- Formulated **DH parameters** to compute **Forward** and **Inverse Kinematics** of KUKA KR5, used MoveIt for **motion planning**, and simulated the model on a Mars-like uneven terrain in Gazebo.

Robot Path Planning Link

- Implemented **BFS, Dijkstra, A***, and **Real Time-RRT*** algorithms on differential-drive robot ROS TurtleBot.

Multiple object detection and tracking | PyTorch, Python Link

- Resulted in a MOTA of 0.691 on multi-human tracking task by implementing a one-shot CNN tracker with two independent branches for **detection** and **re-identification** tasks.
- Jointly optimized the network for two tasks.

Image Classification | TensorFlow, Python Link

- Increased the accuracy by 40% on monkey species classification task by leveraging **transfer learning** to fine-tune the pre-trained VGG-16 model.
- Trained the model on small dataset having 10 classes and 140 images per class.

Image Segmentation | PyTorch, Python Link

- Achieved 99% accuracy and 0.97 dice score for semantic segmentation task by training **U-Net** network on Carvana dataset.

Hand-written digit recognition | Machine Learning Link

- Implemented **Linear SVM**, **Kernel SVM** with linear, polynomial, and RBF kernels, **Logistic Regression**, and **LeNet-5 CNN** architecture on MNIST dataset to recognize hand-written digits without using any in-built functions. Compared their results.