# RISHIKESH BHYRI

+1 (716)-923-5597 | bhyririshikesh@gmail.com | GitHub | LinkedIn | Buffalo, NY

## EDUCATION

**Master of Science in Computer Science and Engineering** — **Aug 2024 - June 2026**

SUNY, University at Buffalo (UB), United States — **GPA - 3.9/4.0**

- Research Track with specialization in Computer Vision and Deep Learning
- Courses: Machine Learning, Parallel and Distributed Processing, Reinforcement Learning, Deep Learning

**Bachelor of Technology in Electronics and Communication Engineering** — **July 2017 - June 2021**

Vellore Institute of Technology, India — **CGPA - 9.48/10**

## SKILLS

- **Programming** - Python, C++, OpenMP, MPI, CUDA
- **Tools and Frameworks** - PyTorch, ArmNN, ONNX, TFLite, OpenCV, Scikit-learn, Git, CMake, Splunk, GCP, Slurm
- **Concepts** - Deep Neural Networks, Computer Vision, Vision Transformers, Diffusion Models, Object Detection, Post-training model compression, Data Structures, Linear Algebra, Object Oriented Programming

## WORK EXPERIENCE

**Research Assistant, Visual Computing Lab, University at Buffalo** — **Feb 2025 - Present**

- Developed a high-density object counting solution using **GroundingDINO** integrated with Swin-B and BERT encoders.
- Implemented a **feature fusion** module with bidirectional cross-attention across text and image modalities, enhancing model interaction.
- Improved performance through **semantic-guided visual and textual prompt tuning** combined with domain-specific loss functions, achieving a Mean Absolute Error **(MAE) of less than 1**.

**Tech Program App Dev 2 - C10, Citi, India** — **July 2021 - July 2024**

- Co-owned and led the development of a web-based end-to-end tool initiative for data creation and conditioning, reducing the time and effort required for data handling. Engineered the tool using ReactJS and NodeJS, optimizing test data creation and conditioning workflows
- Developed API layers in .NET using minimal-API, exposing Selenium-C# automation functionalities and enabling remote headless execution which **decreased data creation time by 5x**, reduced cross-team dependency and **turnaround time by 80%**
- Designed C# automation scripts for Fund Transfer module, **increasing the test coverage** for 7 market countries **by 50%** and **saving cost** in licensing tools **by 30%**

**Computer Vision Intern, PlaEye LLC (Portland, OR), Remote** — **Jan 2021 - Jun 2021**

- Developed a **mobile-based brand analytics** system for **logo detection and classification** using the LogoDet-3k dataset, optimized for arm64-v8a devices with ArmNN. Addressed challenges in dataset imbalance and diverse image features by implementing scaled loss functions and class augmentations
- Created a novel algorithm for automated tennis **court calibration using Homography and Cross ratios** to overcome **single-view calibration limitations** arising from elusive feature points in occluded views. I also worked on optimizing the SubSENSE background subtraction algorithm for GPU using CUDA

## PROJECTS

**MindVault: AI-Powered Bookmark Organizer (UBHacking'24 Best AI/ML Project)** — **Nov 2024**

- Developed a context-aware bookmark manager extension for Chrome
- Implemented web scraping using BeautifulSoup, content indexing with NLTK, and feature extraction via Hugging Face Sentence Transformers
- Designed dynamic clustering system using cosine similarity and DBSCAN for content-based bookmark organization

**Depth Estimation from single-view images** — **Jan 2020 - Mar 2020**

- Created a depth estimation algorithm that focuses on 2D-to-3D translation. The algorithm utilizes perspective projection and shadow point computation using ray geometry to determine the 3D world position of a tennis ball from single-view images

**Parking Assistance System** — **Jan 2020 - Mar 2020**

- Developed a solution for drivers to view and book parking slots in advance via a website, including slot availability and charges. Led the creation of a MobileNetSSD-based car detection algorithm, deployed on a Raspberry Pi, which communicated real-time data to a PHP backend via MQTT