# ANISH BISWAS

Google Scholar ◇ GitHub ◇ LinkedIn
anishbiswas@utexas.edu

## EDUCATION

**The University of Texas at Austin** *Aug 2026 – Present*
Ph.D. in Computer Science
*Dean's Strategic Fellowship*

**Manipal Institute of Technology** *Jun 2018 – May 2022*
B.Tech. in Computer Science and Engineering

## CONFERENCE PUBLICATIONS

**Sutradhara: An Intelligent Orchestrator-Engine Co-design for Tool-Based Agentic Inference**
*ASPLOS 2027 (to appear).*
**Anish Biswas**, Kanishk Goel, Srivarshinee S., Jayashree Mohan, Alind Khare, Anjaly Parayil, Chetan Bansal, Ramachandran Ramjee

**ModServe: Modality- and Stage-Aware Resource Disaggregation for Scalable Multimodal Model Serving**
*ACM Symposium on Cloud Computing (SoCC), 2025.*
Haoran Qiu, **Anish Biswas**, Zihan Zhao, Jayashree Mohan, Alind Khare, Esha Choukse, Íñigo Goiri, Zeyu Zhang, Haiying Shen, Chetan Bansal, Ramachandran Ramjee, Rodrigo Fonseca

**Awkward Array: JSON-like Data, NumPy-like Idioms**
*Proceedings of the 19th Python in Science Conference (SciPy), 2020.*
Jim Pivarski, Ianna Osborne, Pratyush Das, **Anish Biswas**, Peter Elmer

## RESEARCH EXPERIENCE

**The University of Texas at Austin** – Graduate Researcher *Aug 2026 – Present*
*Advised by Aditya Akella.*

- Investigating how concurrency across agents affects correctness and scheduling in long-running agentic workflows.

**Microsoft Research India** – Pre-Doctoral Research Fellow *Jul 2024 – Jul 2026*
*Advised by Jayashree Mohan, Alind Khare, and Ramachandran Ramjee.*

**Sutradhara: Orchestrator-Engine Co-design for Agentic Inference**

- Studied 6,000 production-derived agentic requests and identified three central bottlenecks – tool-dominated tail latency, serialized tool and LLM execution, and KV-cache thrashing.
- Proposed Sutradhara, a cross-layer design that exposes agent structure to the inference engine, enabling parallel execution and workload-aware cache management. Sutradhara supports up to 77% higher load at the same median first-token-rendered latency, or up to 15% lower median latency at the same load.

**ModServe: Resource-Efficient Multimodal Model Serving**

- Characterized the serving behavior of six multimodal models and production Azure workloads, showing that monolithic deployments overprovision resources and inflate the tail latency during image-driven traffic bursts.
- Co-designed ModServe, which separates modality-specific inference stages and manages their resources independently. ModServe achieves 3.3–5.5× higher throughput than monolithic vLLM serving and requires upto 41% fewer GPUs while meeting tail-latency objectives.

**Manipal Institute of Technology** – Planning Lead, Project Manas *Feb 2019 – Apr 2021*

**Autonomous Ground Vehicles**

- Led a 5-member planning subteam responsible for mapping, path planning, and sensor integration for the SOLO and EVE autonomous ground vehicles.
- Developed core path-planning algorithms for SOLO, which won the 2019 Intelligent Ground Vehicle Competition Grand Prize; EVE subsequently placed among the top five teams in the Mahindra Rise Prize Driverless Car Challenge.

## INDUSTRY EXPERIENCE

**Microsoft** – Software Engineer, Azure Storage *Jul 2022 – Jun 2024*

- Developed adaptive rate-limiting mechanisms to protect the load balancer against bursty traffic.
- Created a framework for testing load-balancing policies without provisioning full storage clusters.

**Microsoft** – Software Engineer Intern, Windows Service Delivery *May 2021 – Aug 2021*

- Developed a microservice that automates software-package versioning during release rollouts.

## OPEN SOURCE CONTRIBUTIONS

**CERN** – Software Engineer Intern *Jan 2022 – Jun 2022*

- Redesigned Awkward Array's GPU backend around a direct Python-CUDA architecture, accelerating representative particle-physics workloads by up to 44×.
- Extended forward- and reverse-mode JAX automatic differentiation to complex nested-array operations.

**IRIS-HEP** – Fellow *Jan 2021 – Apr 2021*

- Enabled zero-copy data exchange between Awkward Array and JAX through DLPack.
- Extended JAX PyTree support to trace Python container structures backed by C++ memory, enabling automatic differentiation through Awkward Array slicing operations.

**Google Summer of Code** – CERN-HSF, Student Developer *May 2020 – Sep 2020*

- Developed a CUDA backend for Awkward Arrays, porting approximately 30 irregular-array operations from CPU to GPU using parallel scan and reduction algorithms.
- Developed Python/C++/CUDA interoperability and native CuPy support for seamless movement between CPU and GPU execution.