Peizhi Chen

✉ peizhic@andrew.cmu.edu | 🏠 peizhic.com | github.com/chenpeizhi66 | linkedin.com/in/peizhi-chen/

Education

Carnegie Mellon University *Pittsburgh, US*
MSc in Computational Design and Manufacturing *Aug 2022 - Dec 2023*

- **Courses:** computer system, computer vision, Data Structure for Application, software construction, Visual Learning and Recognition

Dalian University of Technology *Dalian, China*
Bachelor in Mechanical Engineering *Sept 2018 - June 2022*

- GPA: 3.8/4.0
- **Courses:** Discrete Mathematics, Data Structure and Algorithms, C Programming, Microcomputer principle, control theory

Experience

Distributed Linux performance analysis and monitoring *Pittsburgh, PA*
CERLAB

- Utilized **Dockerfile** to specify sources and dependencies such as **cmake, grpc, proto, etc.**, to build the entire project environment.
- Implemented a factory method to construct an abstract class for the monitor, defining interfaces for CPU status, system load, soft interrupts, memory, and network monitoring.
- Built the server and client using the **gRPC framework**. The server is deployed on the server to be monitored, and the client generates a library for the monitor module and display module to call. To reduce coupling, each project module is independent and can be disassembled, only connecting remotely through the gRPC service.
- Used the **protobuf serialization protocol** to build the entire project's data structure.
- The display module is divided into two parts: **UI construction** and **datamodel construction**. The UI interface was built using **QWidget**, **QTableView**, **QStackedLayout**, **QPushButton**, etc. The datamodel was constructed by inheriting **QAbstractTableModel**, building the corresponding cpu_model, softirq_model, mem_model, etc., refreshing the data every 3 seconds.

Projects

Website Development for Personal Blog (github) *Pittsburgh, PA*
Carnegie Mellon University

- Implemented a highly customizable sidebar using Django's function-based views (FBV), allowing for sorting, display toggling, and embedding of custom HTML content.
- Fulfilled the front-end interfaces of registration, login, and personal information modification using customized **CSS** and **Bulma**.
- Deployed the **Django** application on **AWS Elastic Beanstalk** using an **EC2** instance with a **MySQL database**.

Developing a lightweight webserver on Ubuntu20.04 (github) *Pittsburgh, PA*
Carnegie Mellon University

- Using IO multiplexing technique **Epoll** and a **thread pool** to implement a multi-threaded **Reactor** high-concurrency model
- Using **regular expressions** and **finite state machine** to parse HTTP request messages
- Using STL containers to encapsulate characters to implement an **automatically growing buffer**
- **Implement a timer** based on a min-heap(vector based) to close inactive connections that have timed out
- Using **webbench** to perform stress test, and achieve QPS over ten thousand

Hero breakout game (github) *Pittsburgh, PA*
Carnegie Mellon University

- Implemented a third person shooting game based on Object Oriented Programming by **C++** and **OpenGL**
- Constructed super classes of virtual props with extensible methods as interfaces in convenience of inheritance

3D point cloud processing *Pittsburgh, PA*
Carnegie Mellon University

- Implemented precise positioning of window frames in a warehouse model using 3D point cloud processing technology based on laser radar data, the method achieved a good performance with an accuracy rate of about **85%** in object detection.
- Implement **RANSAC** algorithm to eliminate ground interference factors from the point cloud data returned by the radar.
- Cluster the point cloud with **DBSCAN** algorithm.
- Implement **PointNet++** method to accurately classify the objects.

Technical Skills

Programming Language	Python, C/C++, Java, JavaScript, TypeScript, HTML & CSS, Matlab
Tools and frameworks	PyTorch, Numpy, Scipy, Tensorboard, Git, Latex, Node JS, OpenGL, OpenCV, Open3D, PCL
Environment	AWS, Linux, ROS, GNU Bash, Conda, Vim, Docker