Jasher Grunau

Fork Union, VA | P: +1 (434) 825-6769 | email: jasherg7@vt.edu | website: https://jasherg7.github.io/

OBJECTIVE

Software Engineer who is versatile, but prefers data science, robotics, firmware design, machine learning, communication systems, and/or image processing. I'm open to anything; I just wanna be programming stuff.

EDUCATION

B.S. IN COMPUTER ENGINEERING — Blacksburg, VA 24061
Virginia Polytechnic Institute and State University — August 2019

SKILLS

Programming Languages: Python, Embedded C, C++, Javascript, Java, Matlab, Golang, Bash, Verilog, MIPS assembly
Markup Languages: HTML, JSON, XML, LaTeX
Libraries: Tensorflow, Anaconda Suite, Matplotlib, OpenCV, Qt, ProcessingJS
Databases: MySQL, MongoDB
Operating Systems: Cloud computers (usually Ubuntu), FreeRTOS, Unix based systems

WORK EXPERIENCE

AALTA — Seattle, WA (remote)
Software Engineer — Oct 2021 – Present
Work in Artificial Intelligence, Machine Learning, Web Development, Digital Image Processing, Predictive Analytics, Text Mining, etc

ROBOGRINDER (International Robot team for *RoboMaster*) — Blacksburg, VA
Software Engineer Team Member — 2017 – 2019
Work in Embedded Systems, Computer Vision, Artificial Intelligence, Robotics, etc

BRADLEY DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING — Blacksburg, VA
Electrical Engineer, part time — 2015 – 2016
Work in Signal Processing, Embedded Systems, Communications, etc

PROJECTS

MARITIME AERIAL OBJECT DETECTION AI — Project for Aalta
Proof of concept developed for an open project for the Navy. It was a computer vision project to detect ships in raw video aerial footage.

- Developed and programmed all ML models using Python with the framework of Tensorflow. Plotted and analyzed data using the Anaconda Library suite such as Matplotlib. Compiled code on Ubuntu VM and stored on git repository.
- Tried a simple neural network (NN) as a start, and then a deep NN, but the model was not effectively learning object detection
- Constructed a Convolutional NN, and performance still didn't improve
- Mapped the convolutions and poolings visually, and saw the objects were too small to be recognized in the noise of the sea
- Re-cropped dataset and immediately fixed performance and achieved 89% accuracy with the Cross-Validation Set
- Implemented sliding-window object detection
- Parsed, formatted, and labeled raw video footage into a dataset
- Built a ResNet50 Convolutional Neural Network architecture because it measurably performs well with satellite imagery
- Froze and transferred training weights from a model trained off of the COCO 2017 dataset
- Reached a Loss less than 0.002 with very original maritime aerial footage then concluded the Proof of Concept

IONIC LIQUID ANALYSIS AI Project for Aalta

We were contracted by a university's chemistry department to assist with a machine learning research project. Had a couple extra interns hired to work with.

- Developed the models using Python, Tensorflow, and Anaconda Libraries
- Used Python for automated tokenizing and data parsing
- Extracted image data manually and algorithmically from Chemistry PDF Textbook with the help of interns
- Cleaned and organized the collected data
- Transformed the image data with various image processing techniques to make the model run more accurately and quickly
- Implemented a Regression model in a Multi-layer Neural network to get a numerical output
- Expanded the image dataset by using image augmentation
- Implemented a small CNN and its analysis for the conclusion

SAM.GOV NATURAL LANGUAGE PROCESSING AI Project for Aalta

First Project I worked on for Aalta. The project ended after we discovered we terminally lacked legal access to specific data to make our intelligence work. I worked under Sr. Data Scientist and SME until its conclusion. I collected, cleaned, and prepared data to flow well into a NLP model of choice.

- Programmed a cloud computer to continuously extract data from Sam.gov with a python script
- Multithreaded the program to continuously gather 100M+ data points from an API as stably and quickly as possible
- Designed my own tokenizer (honestly that was because I didn't know better) using Python, Tensorflow, and Anaconda library
- Scraped web pages with Python script and web crawler using Ubuntu cloud computer from AWS
- Parsed and cleaned data into a MySQL database
- Migrated MySQL database into a MongoDB database

UNORTHODOX SEQUENTIAL MODEL AI Personal Project

Original algorithmic development of an AI model that takes its functional roots from engineering systems controllers I learned back in college, rather than the field of mathematical regression models. It works very differently than other models, and I don't see anyone else taking the same approach as I am with this experiment. For better or worse, I want to see how my concept will develop and perform compared to mainstream approaches.

- Automated data collection from cloud computers with Google Cloud computing
- Developed personal server with Raspbian Linux and Python Scripts, then coordinated it with the cloud computer
- Implemented SQL database management system
- Drafted and tested different system mathematical models to work in discrete systems using Python, Anaconda, and Matlab
- Experimented, tested, and refined algorithmic performance on low level (Python mostly)
- Compare performance visually with mainstream sequential models

BLOCKCHAIN SMART CONTRACTS FOR ITEM INVENTORY Project for Aalta

Proof of Concept to draft an inventory control system on a niche permissioned blockchain network with unique smart contracts and token system.

- Develop ERC-20 token system logic deployed in a simulated Ethereum blockchain system with permissioned accessibility for extra security using Solidity and Ethereum
- Make token design as the drafted permission settings to verify users into the network for added security
- Deploy nodes using docker images and manage the clusters using Kubernetes
- Develop Hyperledger smart contracts with Golang and a little C++
- Manage system on a Ubuntu VM
- Design network and software architecture of a private blockchain implementing the Hyperledger Fabric Framework
- Test network functionality with simple smart contracts to monitor system functionality and security

COMPUTER VISION FOR ROBOT NAVIGATION AND COMBAT RoboGrinder

Developed an image processing software in an International robot team called RoboGrinder primarily focused on a Chinese competition called Robomasters hosted by DJI in Shenzhen, China. I worked on the software subteam, where we focused on Computer Vision. Specifically we focused on image processing for our robots' guns and turrets to automatically target enemies without the use of a pilot.

- Design the image processing feed of our robots to auto-detect and target our enemy robots with our turret gun using C++.
- Use the OpenCV library to apply various image filtering to our feed in our Embedded Linux Environment
- Implement a Flood-Fill algorithm for object detection of that filtered image feed in C++
- Train our turrets to target special locations in a separate competition mini-game (hosted in Quebec). Programmed with Python then trained with Tensorflow and PyTorch libraries with Machine Learning techniques. Trained model is exported to our embedded system
- Work on team with a massive language barrier
- Document trips and handle technical writing for sponsors

MICROCONTROLLER INTERFACE FOR ROVER CONTROLS Capstone Project & RoboGrinder

Capstone project for programming, design, and microcontroller interfacing for 3 different rovers that incorporates engineering standards. As part of an engineering design team, I needed to design, implement, and debug multi-threaded software that operates under real-time constraints on PIC32 embedded computer systems.

- Program FPGA and microcontroller system using embedded C/C++ as the programming language, FreeRTOS as the embedded OS, and Harmony as the software development interface with the FPGA
- Implement a multithreaded environment using minimal FreeRTOS software components. The restricted environment needed us to use as much hardware as possible, and less software dependent solutions (because there was only around 250MB of memory and much less cache).
- Made a software system using multiple tasks, FICO queues from FreeRTOS, and ISRs as the skeleton of the system.
- Create a unique communication system for Debugging. Multithreading the microcontroller breaks all inbuilt debugging systems, so we had to create our own within the hardware using the GPIO outputs to send timed codes to a Logic analyzer.
- Created a second debug output from the embedded system by using a system that outputs debug codes as UART messages
- Implement Unit Testing within an embedded environment
- Use UART, I2C, SPI, and CAN to communicate with all the various modules integrated with the PIC32 board i.e. robotic arms, sensors, LEDs, motors, etc.
- Use interrupt handlers integrated with hardware timers to read ADC units

P.I.D. STABILITY CONTROLLER Capstone Project

In the feedback loop between my written software and the motor system in my rover robot, there are inconsistencies with response and a significant delay in the feedback. Correcting error in the environment in an efficient manner can be really difficult without leading to system instability. This problem was solved with a formal systems engineering method by employing the Proportional Integral Derivative controller.

- Develop an embedded system on a PIC32 board with FreeRTOS and Harmony as the Interface
- Program an autonomous parameter tuning system in Embedded C/C++ that tunes stability parameters based on the inputs sent to it via Server communications
- Implement Systems Engineering practices
- Tune my rover's motor systems to find stability and to consistently reach its coordinates with no more than +/- 1.0 cm error, every 30.0 cm of travel.
- Design low-level UDP communication to communicate with other Team members' PICs and my server system.
- Change P, I, and D constants in the embedded system through server communication for efficient debugging and testing.
- Send motor error real time to the server, and plot the error over time to see how to monitor the stability of the system.
- Use the stability plots to tune the PID control system via the PID constants on the server.

2D-GAME ENGINE Senior Design Project

We must design a fully functioning game using only P5.JS's simple graphic library within the frameworks of HTML, CSS, and Javascript. To have a functioning game, I needed to build and design a game engine. Its design was meticulous and difficult. It's performance is atrocious because our class's development platform doesn't allow us to store custom sprites in cache or in memory. Meaning each sprite is manually being regenerated many times a second. Just see it on my website: https://jasherg7.github.io/

- Create a video game from scratch for a senior design project in college.
- Limit in class with an extreme restriction of not being able to load images onto memory
- Develop a drawing program and game engine in Javascript that generates the javascript code that regenerates that rendered visual for each frame needed in the environment
- Implement unique rendering system that doesn't store predefined images onto memory (makes the game really slow, but it was required for the class)
- Make a particle system and tile map in Javascript that is rendered in p5js

DRONE COMMUNICATIONS IN EMBEDDED SYSTEMS Bradley Dep of ECE

The efficient deployment of multiple unmanned aerial vehicles (UAVs) with directional antennas acting as wireless base stations that provide coverage for ground users.

- Implement Software Defined Radio (SDR) integration with microcontrollers on UAV for research department
- Redesign 802.11ac communication protocol onto SDR with very low level signal processing design, tested with Matlab
- Implement Dr. Saad's EE theoretical communication protocols onto the SDR on embedded Linux environment
- Integrated the Linux OS with the SDR via python scripts which automates the SDR's behavior in testing
- Mobile device testing integration with SDR protocols to extend wifi coverage as a massive network of hotspots

MIPS ASSEMBLY SIMULATOR Class Design Project

Large individual project to simulate the MIPS Assembly compiler and its code execution.

- Code in modern C++, while using Git and CMAKE for the project management
- Create robust unit testing for 100% code coverage
- Write language Tokenizer and parser for MIPS assembly language
- Use object-oriented design for command executions previously parsed
- Manually allocate memory in C++ and fix any measured memory leakage
- Design a UI using the Qt graphics library for an easier access for clients
- Multi-thread environment and measure increased program execution speed

CONSOLE FIRMWARE AND VIDEO GAME DESIGN Class Design Project

Class Design project where we programmed a microcontroller with a self designed video game on a system state machine to interact with various hardware modules with a wide variety of communication protocols. **(C, C++, Harmony, FPGA)**

- Create a unique console system in a restricted firmware environment on an FPGA and program a unique game using Embedded C/C++ with Harmony as the interface
- Design LED behavior for debugging notifications
- Integrate a MAXSONAR sensor via UART into system state machine
- SPI and Analog to digital Converter for Joystick integrated with hardware timers
- Interrupts are used to access hardware timers and A2D converters
- I2C communication to access accelerometer
- Programmed and designed a videogame to be played on the OLED with as a UART output for each frame and with customized pixel graphics

ACHIEVEMENTS

3RD PRIZE for team RoboGrinder **ICRA 2019 International A.I. Challenge**
2ND PRIZE for team RoboGrinder **Robomaster 2018 Final Tournament**
1ST PRIZE for team RoboGrinder **Robomaster 2018 International Regional Competition**
3RD PRIZE for team RoboGrinder **Robomaster 2017**
EAGLE SCOUT