jamaisyan18@gmail.com
248-835-5911

Charlie (Jim) Yan

335 Holyoke Street
San Francisco, CA, 94134

EXPERIENCE

Pneubotics / Canvas **San Francisco, CA, Apr 2016—May 2019**

- Software and architecture:
  - Built a user interface and architecture and infrastructure that used PyQT, OpenGL drawing functions, STL meshes of various robotics components for rendering simulations and widgets for robot operator interaction, taking into account joint angles for a couple very kinematically different arms
  - Worked with threading and concurrency and rendering smoothness, ray-picking for advanced user selection
  - Utilized software design patterns like decorators, factories, strategies, and duct-typing / python interfaces throughout
  - Infrastructure supported various robotics use cases like eye-hand calibration, robot tele-operation, surface-prep, mobile base path planning and execution simulation
  - High level conductor that talked with ROS over services / topics to see robot state, and coordinate various events related to sensing, planning, execution, safety systems and robot state. Implemented interfaces in C++ / Python related to how individual components worked with each other, and be able to simulate / debug one part of the system independent of others
  - Inverted the conductor logic fundamentally to allow for higher degree of concurrency and safety
  - Data structure to organize arm waypoints given user selection, functional programming to account for various situations of continuity that require robot behaviors in between discontinuous selections
  - C++ templating for sensor processing and daisy-chaining 'sources' and 'processors', and pure abstract classes for encoders depending on robot configuration to allow us to scale to different hardware and communication protocols (USB / serial vs. Ethernet / modbus) on different mobile bases quickly
  - Telemetry for automatically starting ROSbagging, and logging throughout the system of various robot activities to support business and post-Morten troubleshooting
- Path planning and algorithms:
  - Hybridized strategy for mobile base planning of RRT and Reeds-Shepp which allowed for robustness but also solving for an analytically feasible and exact theoretical path within a production-level scale of time. Taking into account various types of obstacles that define collisions differently, optimizing collision checking with binary vs linear search, and making it robust to noisy obstacle data (false negatives and positives)
  - First pass at a analytic solution to spraying and sanding to cover an rectangular wall area
  - No go zone algorithm to split and route tool-paths given polygons that the tool cannot touch using decision tree search and minimization, branch-and-bound for Airbus windows and features
- Computational geometry:
  - Touch-off behavior and underlying geometry to support of path shifting based on imperfect mobile base or manual base positioning, where the spot on the wall does not change but the waypoints in base frame do, or where the spot on the wall changes because of imperfect sensing or perception
  - Side-finding and room-finding graph based branch-and-bound algorithm to find non-overlapping 'sides' and from those 'sides' 'rooms' given a collection of 2D line segments that represent walls
    - Robust to order of processing and noise between walls at intersection points etc.
  - Geometry to support saving pose estimates and re-using that pose estimate assuming no mobile base repositioning, to obviate the need to re-map a space after shutdown or crash
  - Given a trajectory's progress, and planned path, get a 'expected' pose to support error quantification and correction, arc normal error correction calculation
- Kinematic, trajectories, estimation, optimization:
  - Forward kinematics, iterative and derivative inverse kinematics, analytic inverse kinematics for UR10 robot arm for GUI rendering of STLs and for closing the loop with mobile base positioning to go from a parameter based to a arm reach-ability criteria
  - Piecewise sinusoidal trajectory for mobile base execution interruption if range sensor detects unexpected jerk in range (operator safety)
  - Symmetrical velocity trajectory given acceleration and velocity limits using NLopt library and setting up the correct equality / inequality constraints
  - Steering and drive calibrated encoders to pose estimation and simulation
- Machine vision:

jamaisyan18@gmail.com
248-835-5911

Charlie (Jim) Yan

335 Holyoke Street
San Francisco, CA, 94134

- Chilitags debugging and integration into robot base eye-hand calibration, 2D-3D registration with Kinect sensor and geometry to calibration video feed detection of 4 corner steel coupon against Kinect depth sensing and generating waypoints for robot surface prep
- First pass at a SLAM solution using Scanse range finder using line segmented, correspondence finder, and analytic geometric Kabsch algorithm to compute 2D pose estimate deltas
- SLAMbox with 2 Velodyne sensors, initial bringup and mount on robot arm, and integration into Cartographer to characterize noise and RMS pose estimation error
- Orbbec RGBD sensor + bring in a third party Agglomerative Hierarchical Clustering algorithm for point cloud segmentation to detect walls as planes and generate tool-paths for surface prep, used in several early jobs

- Hardware:
  - Vive as ground truth to calibrate steering encoders with arc fitting
  - Steering encoder serial protocol encoding with library
  - Modbus protocol digital input read and generate a mobile base 'auto vs. manual' mode for operator
  - Talking to industrial arm over various TCP sockets, sending programs over, monitoring state, sending 6-DOF trajectories, pause and resume state synchronization / transitions

Software Development Engineer, Microsoft Corporation WAC **Jan 2015—Dec 2015**

- Delivered bug fixes, performance improvements, typing features to Word, Powerpoint, OneNote Online
- Designed, implemented, and shipped a new activity feed feature to internal dog-food within a month

Software Development Engineer, Amazon.com **May 2013—Dec 2014**

- Designed, implemented, and shipped a new Java Spring service for ingesting data about billions in cash transactions per day to accounting platform using DynamoDB, SQS/SNS, and S3 AWS services. Migrated all global traffic to use new service on schedule, using automation and scripting to detect and fix regressions before they impacted production. Shipped a Ruby on Rails interface around the service that simplified cash related developer operations and reduced team monthly operational tickets by 15%.
- Worked with technical and business teams to onboard new international businesses and new payment use cases onto our platform. Increased global automated accounting coverage from 85% to 92%.
- Refactored all services to be fully configuration-driven, vastly reducing developer effort needed to onboard new businesses and accelerating efforts to reach 100% automated accounting.

Research Assistant, University of Michigan CS Dept. **Jan 2012—Jan 2014**

- Built an object detection pipeline using Matlab and Python for detecting automobiles in pictures and point clouds from a variety of environments.
- Designed and executed experiments to improve the detection performance using HOG and SIFT features from 2D images and clustering and scene understanding techniques on 3D velo-dyne data. Used classification techniques such as linear and RBF SVMs, Restricted Boltzmann Machines, and Deep Autoencoders. Implemented spatial pyramids and sliding window algorithms for detection. Used CUDA convnets to speed up computation.
- Presented research findings in an undergraduate research symposium.

Instructional Aide, University of Michigan ECE Dept. **Jan 2012—May 2013**

- Taught six laboratory sessions of the Digital Signal Processing Lab course to engineering upperclassmen.
- Mentored student teams with their capstone design projects, providing technical guidance on FPGAs, TI and Arduino embedded systems programming, and sensors.

EDUCATION

University of Michigan College of Engineering — Dual Degree in CS and EE

Graduated May-2013, GPA 3.71, *Magna Cum Laude*

- Algorithms, Operating Systems, Databases
- Machine Learning, Artificial Intelligence, Advanced Topics - Image Processing
- Digital Signal Processing, Control Systems
- Probability, Real Analysis, Linear Programming

PROGRAMMING LANGUAGES

- C++, Python, Ruby, Java, Matlab, Javascript