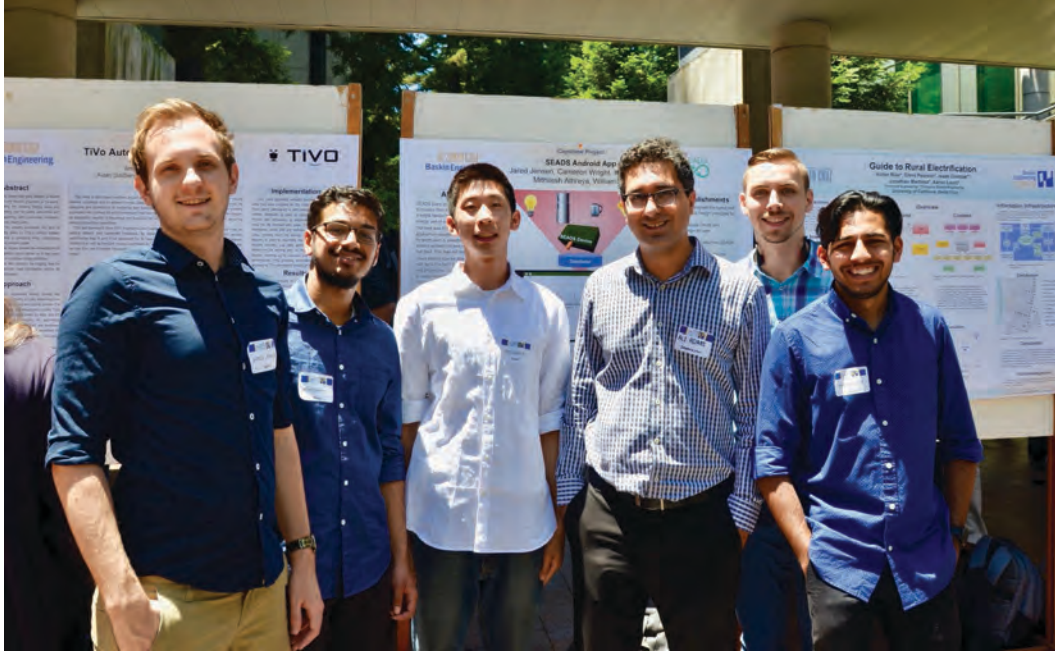
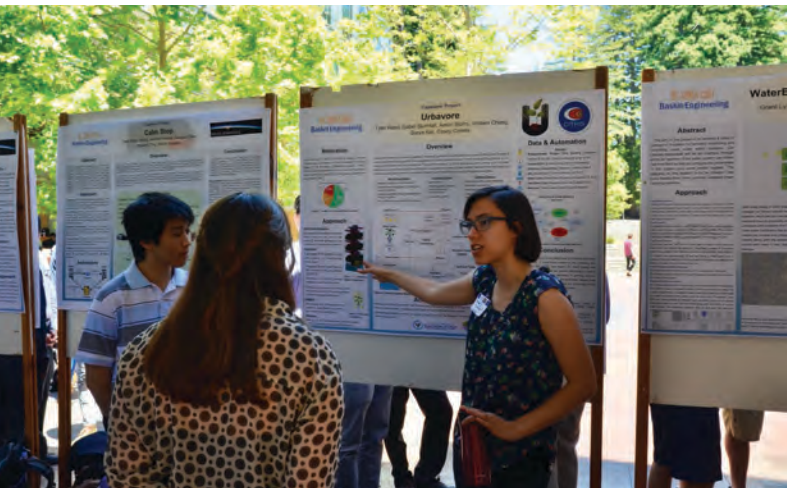


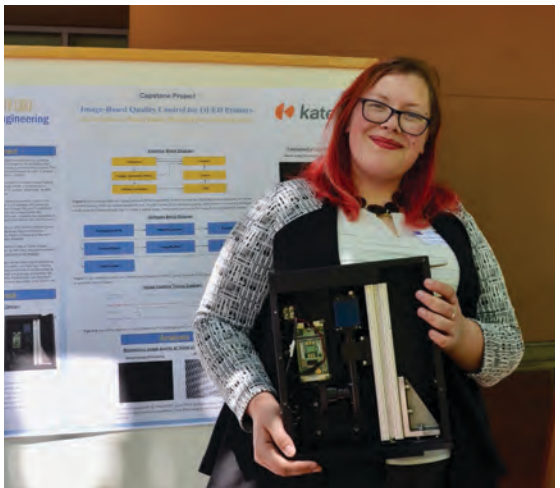
PARTNERS' DAY

2018 PROGRAM

CORPORATE SPONSORED
SENIOR PROJECTS

Baskin Engineering UC SANTA CRUZ





INTRODUCTION

This publication highlights the seventh year of the **Corporate Sponsored Senior Project Program (CSSPP)** at the Baskin School of Engineering. The publication also includes a selected group of this year's capstone projects from student teams in Computer Science, Computer Engineering and Electrical Engineering working on faculty/student initiated projects.

CSSPP provides students with a unique opportunity to experience working on real-world engineering projects as part of their undergraduate education. Throughout the academic year, students interact with teammates; some make visits to their corporate sponsor's worksite, and all are required to solve problems along the way. By working with mentors at corporate partner companies, students learn important skills, take on interesting challenges, and begin to understand what it means to be a professional engineer.

We appreciate our corporate sponsors for their willingness to support this year-long program, mentor our students and provide them with challenging projects to work on. And we appreciate our students, who have worked hard and have enriched our lives through their energy, intellect and determination.

Alexander L. Wolf
Dean
Baskin School of Engineering

Baskin Engineering UC SANTA CRUZ

ACKNOWLEDGEMENTS

We would like to acknowledge and thank the faculty, teaching assistants and staff who have been so instrumental in the Corporate Sponsored Senior Project Program:

SENIOR DESIGN FACULTY
(CORPORATE SPONSORED SENIOR PROJECT PROGRAM) 2017-18

Patrick Mantey
Director of Senior Capstone and Professor, Computer Engineering

Gabriel Elkaim
Associate Director of Senior Capstone and Professor, Computer Engineering

Stephen Petersen
Teaching Professor, Electrical Engineering

Richard Jullig
Lecturer, Computer Science

Tela Favaloro
Lecturer, Electrical Engineering

David Munday
Lecturer, Computer Engineering

TEACHING ASSISTANTS
(CORPORATE SPONSORED SENIOR PROJECT PROGRAM) 2017-18

Michael Grimes
Ahsan Habib
Kavya Jha
Dylan Rothfled
Reihaneh Torkzadehmahan

CORPORATE SPONSORED SENIOR PROJECT PROGRAM STAFF

Lisa Coscarelli
Special Agreements Officer

Russ Evans
Development Engineer, Baskin Engineering Lab Support

Tim Gustafson
BSOE Technical Lead/BSOE Webmaster

Angelina Gutierrez and Angel Dominguez
Event Coordination

Liv Hassett
Associate Campus Counsel

Frank Howley
Senior Director of Corporate Development

Alexander Wolf
Dean, Baskin School of Engineering

Maureen McLean
Director of Resource Planning and Management

Christian Monnet
Development Engineer, Baskin Engineering Lab Support

Lynne Sheehan
Network Administrator, Facilities/Machine Room

Bob Vitale
Director of Laboratories and Facilities

SPONSORS

Special thanks to our sponsors for your generous support of our Corporate Sponsored Senior Projects Program. Your time, experience and financial support were beneficial to our students and the success of their Senior Design Projects.

UC SANTA CRUZ
Baskin Engineering

Capstone Project

Slug IoT Web Console

Bryan Ko, Darwin Li, John Wilde,
Joseph Robinson, Nadal Alyafaie, Sherif Elsaid

amazon Lab126

Abstract

Amazon Lab 126, creators of Internet of Things (IoT) devices like the Dash button, see the need for an easier way to analyze the data that these devices produce. The vision is a web console that allows any kind of user or business, from farmer to hardware engineer, to see visualizations, recognize trends, and derive value from such data.

We designed this console around three types of accounts: users, administrators, and developers. Developers can see all data produced and make improvements accordingly. Administrators can oversee many accounts, and regular users can enjoy an interface focused on their device data. All three can receive live, event-based email or SMS notifications alerting them of problems, reminders, or other changes in the data.

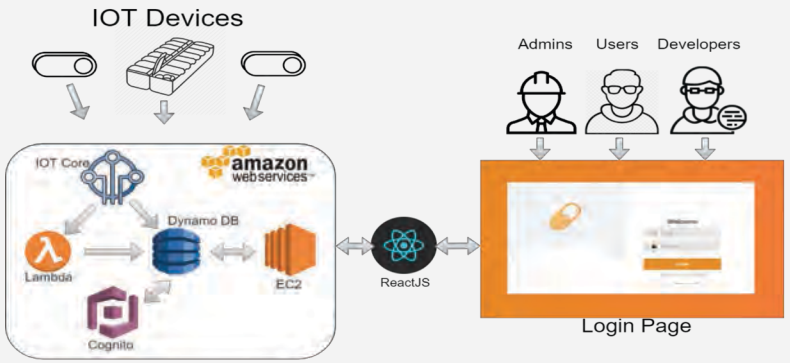
Approach

To narrow the scope of the project, we chose to solve a specific problem: tracking medication progress via an IoT Pill Box. This allowed us to develop specifically for users as patients, admins as doctors, and developers as engineers improving these IoT Pill Boxes.

Since this IoT Pill Box is only a device concept, we generate pseudo-data based on our vision of the Pill Box, as well as use actual Dash buttons to simulate opening a particular compartment of the box; all of this IoT-like data is logged into DynamoDB.

We then create visualizations by graphing this log of a patient's medication history. This allows patients to see their own progress and allows doctors to see aggregate graphs across multiple patients. Lastly, the developers get a different view: they see all IoT data logged in DynamoDB as a searchable data table as well as selectable graphs.

Architecture



Administrator/User

- Admin can view different users' data
- Calendar view of prescriptions
- Scatterplot of IoT data against prescription data
- SortedActivity feed of IoT data
- Simple statistics visualized with donut graphs

Developer

- Paginated table showing IoT data
- Column sorting, filtering. Can be combined with any combination
- Table can be exported to CSV file
- Line graph to visualize IoT data based on data entry points and different IoT data types

Acknowledgments

Thanks to Amazon Lab126 sponsors Sukwon Noh, Dan Rhodes, and Michael Lee for giving us this opportunity. Thanks to Prof. Richard Jullig and our TAs Scott Davis and Reihaneh Torkzadehmahani for their continuous support and guidance throughout the development process.

Technologies

Frontend

ReactJS: Javascript web framework built with reusable components, which are easily managed with rendering and state management.

AWS Backend

IoT Core: Service that handles the setup, connection, and messaging of new IoT devices.
Lambda: Serverless functions that can trigger when an event, such as a dash button click, occurs.
DynamoDB: NoSQL database we use to store all of our IoT data, which is then accessed for visualizations.
EC2: Virtual machine running Ubuntu used to host the React web application online. AWS Route53 service allows the instance to be linked to a domain name.
Cognito: Secure user authentication service. Also allows restricted access to AWS services depending on the user level role.
Dash Button: IoT device that connects with AWS and allows 3 different button press types.
API Gateway: AWS custom API service to can connect to AWS services securely.

Results

Admin Page: Can select different users and see data visualization and statistics based on that user. We can also see the history of data for that user.
Developer Page: This page gives developers debug tools to access and interpret IoT data. This page features a datatable in which you view, sort, and filter raw IoT data. Data from the table can also be exported to a CSV file. Data visualization allows developers to see trends in different aspects of their data.
Hardware Integration: Our web app can receive data from an Amazon IoT Dash Button and through AWS Lambda, we can generate a data payload that we can use with our webapp.
Secured & Serverless System: Webapp is hosted with HTTPS giving secured connection. Access to IoT and user data is handled through secured AWS giving us a serless system with an additional level of security.

UC SANTA CRUZ
Baskin Engineering

Capstone Project

Greens Only

An Tran, Kevin Ajili, Cesar Neri,
Arindam Sarma, Eric Su, David Munoz

ATOLLOGY

Abstract

The goal of the project was to detect contaminants in the produce packaging process. Contaminants found in major produce factories are costly and potentially harmful. Our team was tasked with using open source software, computer vision, and inexpensive hardware to detect and flag contaminants in spinach processing line. The team focused on finding contaminants on the top layer of the spinach, because it is difficult to see through thick layers of produce.

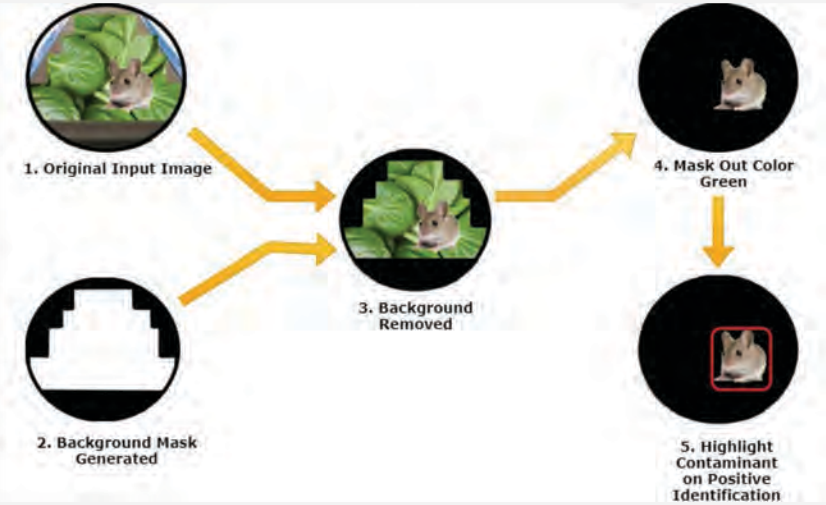
Approach

The team explored multiple methods for contaminant detection, including color analysis and edge detection. A feature-based combination of several methods was the initial approach. Instead, a pipeline was created to process images (1) from a video feed. The first step was to mask a static background (2) from an input video (3), then proceed to mask all green colors (4). Afterwards the remaining color palette is simplified and any regions of a certain area of contiguous color is isolated / flagged (5).

Overview

Atollogy specializes in building IIoT-based (Industrial Internet of Things) big data solutions for enterprises. These provide composite analytics by placing non-invasive sensors in the manufacturing environment. One of Atollogy's customers is one of the nation's leading produce packaging firm, averaging a daily 1 million pound throughput of spinach alone. Unfortunately, a contaminant found in any part of the process can cost the company thousands or even hundreds of thousands of dollars. This is where Atollogy and our project come in, to be able to detect these contaminants more reliably than finding contaminants manually.

Architecture



Acknowledgments

Rob Schoenthaler	(CEO)	Duck Ha Hwang	(Lead Engineer)
Jan Jannink	(CTO)	Frank Poon	(Relations)
Tony Tarantino	(Relations)	Richard Jullig	(Professor)
Reihaneh T.	(TA)	Kavya Jha	(TA)

Analysis

The pipeline allowed us to modularize the process by breaking it down into steps. Masking the background means the focus is on the areas of interest in each frame. Removing all green colors from the resulting image isolates the contaminants in the area of interest. Compressing the color palette of the masked images allows for more efficient detection of potential results.
Detecting the polygon in the black background was the most challenging part of the process.The math behind detection techniques becomes complex and explores concepts such as Gaussian matrices and Green's Theorem.

Future Work

In the end, the best solution we could come up with, considering the software and hardware limitations we were operating under, still has a number of known flaws and areas for improvement. One significant improvement for the system is to add functionality for black and green contaminants. Testing and research in infrared cameras is a promising area the team looked into briefly. Multi-angle camera placements can help with finding contaminants not visible on the surface.

Pneumatically Actuated Feedback Control

Joseph Legnitto, Roger Berman, Shruti More, Baoxian Yang, and Jordan Tapia
University of California at Santa Cruz, Kateeva

Abstract

Kateeva uses pneumatic actuation to precisely position its Organic LED printing machinery. Pneumatic devices are difficult to automate due to nonlinearities such as air compression and static friction. We created a PID controller for the vertical position of a pneumatic actuator and load system that achieved the designated position within 5 seconds and high precision.

Introduction

A pneumatically actuated feedback system for position control was built using a Festo pneumatic regulator integrated with Arduino Uno R3 microcontroller. We implemented two different algorithms for our system to control pressure which translated to z-axis motion of an air cylinder piston [1]. These were an Adaptive PID algorithm and State Machine algorithm. Through experimentation we determined the Adaptive PID algorithm to be our best solution.

Experimental Test-Bench Set-up

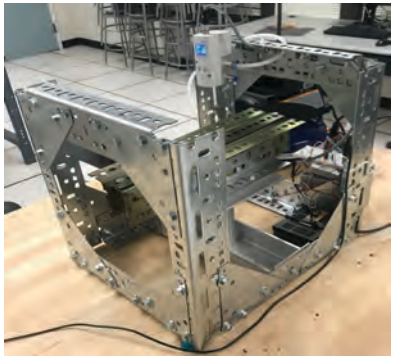


Figure 1: Test-bench

Methods and Materials

The Test-bench shown in ***Figure 1*** is modeled after Kateeva's pneumatic system. It is composed of a sliding piston air cylinder, Festo pneumatic regulator, and an Arduino Uno that implements the feedback control loop shown in ***Figure 2***. The GREEN box consists of the Arduino microcontroller. The RED box consists of an actuator/plant. The filter and gain circuit are in between, along with the linear position in the return loop.

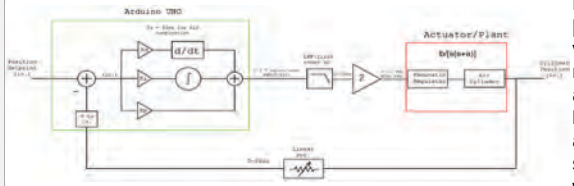


Figure 2: Control Loop

Mathematical modeling for our system [2] assisted in determining a range of values for the PID parameters for manual PID tuning. PID values were adapted depending on the difference between a desired reference position and the actual position. Additionally a state machine was also implemented, which altered pressure values depending on proximity to the reference point.

Red: **Weight**, Green**: Sensor**, Orange: **Cylinder Rods**, Yellow: **Sensor Notch**.

① Air cylinder remains stationary.
② We supply constant pressure to maintain air cylinder in constant velocity.
③ PID controller activates when it is less than 1.4 inch from the setpoint of 3-inch.

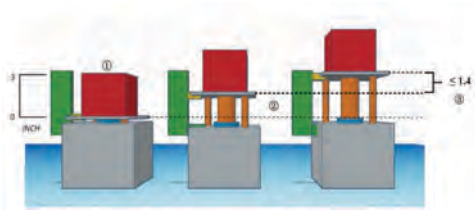


Figure 3: Adaptive Control

Results and Discussion

Reliable position control is not 100% possible due nonlinearities such as air and static friction lead to inconsistent velocity control and unrepeatable settling times. However, we achieved 100% accuracy within our test data.

In ***Figure 4***, we observe short settling times and consistent movement. Note that 1 of the 10 tests have longer settling times due to the nonlinear attributes of the system. Data from the State Machine experiment in ***Figure 5*** shows a large overshoot and numbers of oscillations around the 2-inch setpoint.
Such unwanted oscillation leads to a greater mean settling time than that of the PID.

Reference Position(in.)	PID	SM
1	100%	80%
2	100%	10%
3	100%	20%

Figure 6: Result of 10 Tests

Through the data in ***Figure 6***, we see that the PID control algorithm is more consistent than the S.M. for tests which had settling times less than 5 seconds.

Resulting Plots (PID and S.M.)

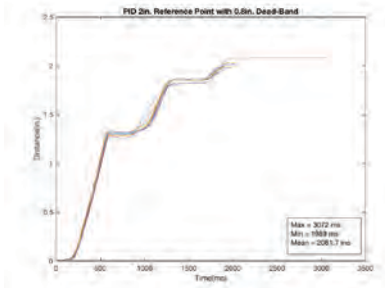


Figure 4: PID Time vs. Position

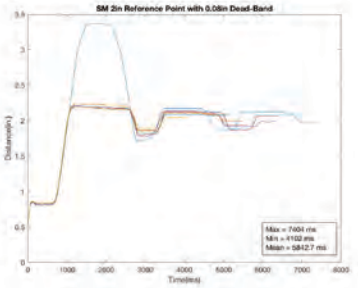


Figure 5: S.M. Time vs. Position

Conclusions and Future Direction

We successfully accomplished a functional position controller through Adaptive PID tuning with settling times as rapid as 2.5s. However, inherent nonlinearities lead to inconsistent settling times between run-tests; this essentially eliminated velocity control. There is much room for enhancements to this non-linear system in terms of statistical analysis and research on precise control using pneumatic actuators.

Contact Information

Roger Berman	raberman@ucsc.edu	Robotics Engineering
Joseph Legnitto	jdlegnit@ucsc.edu	Robotics Engineering
Shruti More	spmore@ucsc.edu	Electrical Engineering
Jordan Tapia	joetapia@ucsc.edu	Electrical Engineering
Baoxian Yang	bayang@ucsc.edu	Electrical Engineering

References

1. Franklin, G, et. al. "Feedback Control of Dynamic Systems." Prentice Hall. 2001. 7th ed. pp. 160-165.
2. Milutinovic, D. "Re: Senior Design Project on Controls." Message to Kateeva Senior Design Project Team. 5 Mar 2018. E-mail
3. Beauregard, B. "Arduino PID Library." playground.arduino.cc. Web. 11 Jan. 2018.

Acknowledgements

Prof. Stephen Petersen
Prof. Dejan Milutinovic
Prof. Tela Favaloro
Azzam Qureshi
Karl Mathia
Daniel Ruatta

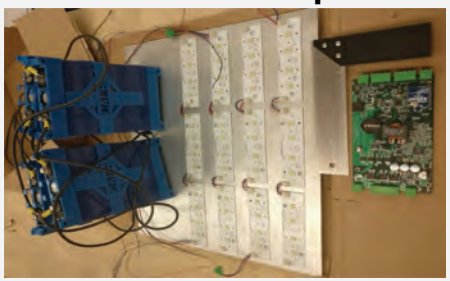
UC SANTA CRUZ
Baskin Engineering

Capstone Project

Mirabella Hybrid Solar Lamp

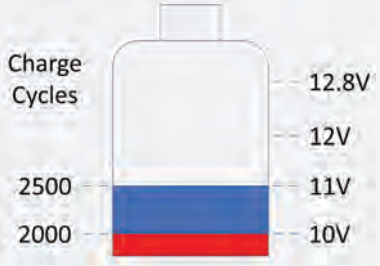
Keyan Chang, Eduardo Rodriguez, and Daniel Castro
Department of Electrical Engineering, University of California, Santa Cruz

1. Increase Input



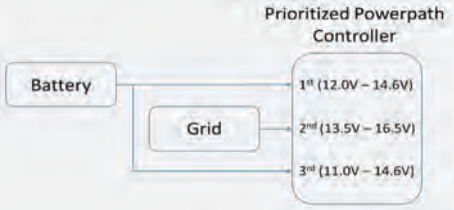
Mirabella upgraded their solar panels resulting in higher input voltage. To accommodate this change, we replaced the Battery Charger's resistors to increase its maximum input voltage from 20V to 40V.

2. Longevity



Battery lifetime is shortened the deeper the discharge. We utilized an under voltage lockout circuit to stop the battery from draining below 11V, thus extending the battery's life to 2500 cycles.

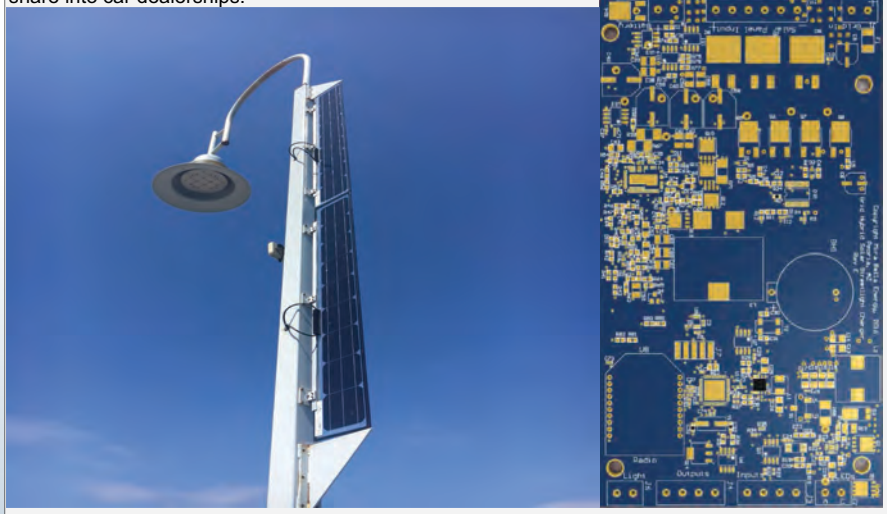
3. Uniform Circuit



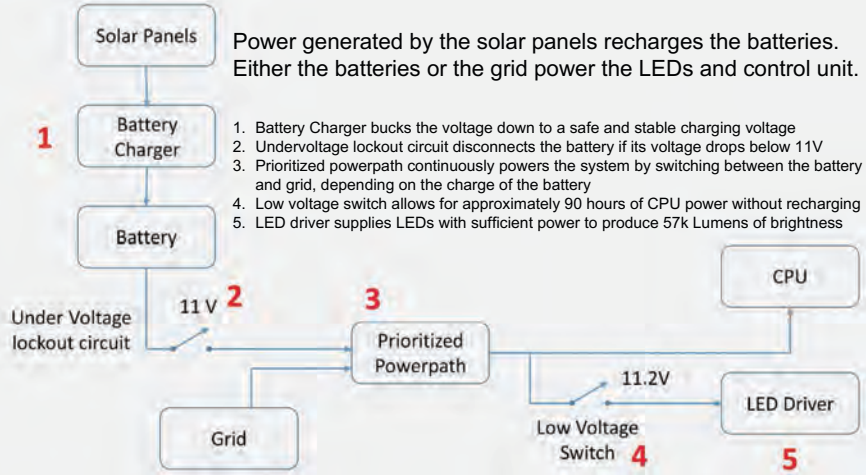
If the grid is connected, the battery will drain to 12V then the system will swap over to the Grid as a power source. We needed a circuit that could function with or without a grid. To accommodate this, we added a third path that reconnects the battery to the system allowing the battery to be used down to 11V.

Introduction

Mirabella is a company utilizing bi-facial solar panels attached to streetlights, which allow for hybrid solar streetlights. Their goal is to make their lights brighter in order to expand their market share into car dealerships.



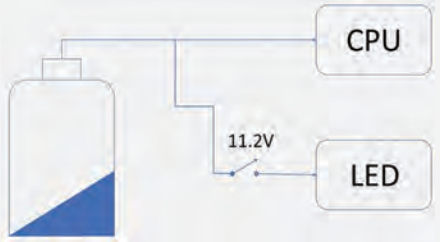
High Level System Redesign



Power generated by the solar panels recharges the batteries. Either the batteries or the grid power the LEDs and control unit.

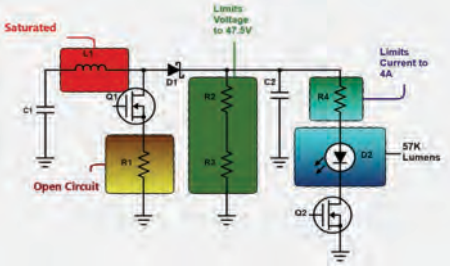
1. Battery Charger bucks the voltage down to a safe and stable charging voltage
2. Undervoltage lockout circuit disconnects the battery if its voltage drops below 11V
3. Prioritized powerpath continuously powers the system by switching between the battery and grid, depending on the charge of the battery
4. Low voltage switch allows for approximately 90 hours of CPU power without recharging
5. LED driver supplies LEDs with sufficient power to produce 57k Lumens of brightness

4. Power Saving Mode



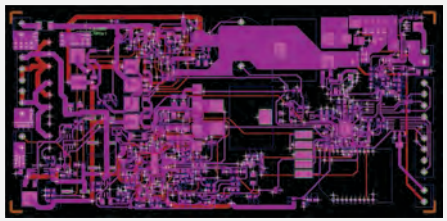
For off-grid integration during cloudy days, we want to ensure the CPU doesn't lose power. We integrate an LED shutoff at 11.2V providing approximately 90 hours of battery life to the CPU after the LEDs are disconnected.

5. LED Brightness



Through analysis of the LED driver we noticed that the boost converter's inductor was being saturated and that the sense resistor was experiencing too much power. We upgraded these components allowing us to produce 47.5V at 4A resulting in approximately 57,000 lumens.

Conclusion



After we designed a system capable of producing 57K lumens of brightness, we redesigned the board layout. This required us to create footprints, route components, and add heatsinks to hotspots that had potential of overheating. We created Gerber files and sent them to our client in order to be milled, assembled, and tested.

Capstone Project

Throughput Diagnostic System to Improve Efficiency in Automated Testing

Nicholas Ivy, Tyler Veness, Milo Webster

UC SANTA CRUZ Baskin Engineering

CITRIS AND THE BANATAO INSTITUTE

nanometrics

Abstract

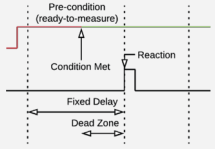


Figure 1: Throughput Optimization

- To increase the throughput of their semiconductor wafer metrology system, Nanometrics needs to characterize the timing of the non-deterministic, throughput-relevant signals in their network of controllers and measurement instruments.
- They cannot currently identify which components of their metrology system are limiting the throughput and their software cannot perform this monitoring without affecting its timing characteristics.
- An external device is, therefore, required for the non-invasive monitoring of throughput-relevant signals.
- This **Throughput Diagnostics System** (TDS) must indicate where existing fixed delays, which are used to preserve ordering of non-deterministic events, can be reduced in order to increase the throughput of the Nanometrics' wafer metrology system.

Constraints

To provide insight into the timing characteristics of Nanometrics' system, the **Throughput Diagnostic System** must sample and display the state of all metrology signals at a 1ms resolution. It must then analyze the signal data in post-processing, generating a gantt chart-type timing diagram. The post-processing software must identify which signals, in a provided list of sequential states, are introducing unnecessary delays in the metrology system.

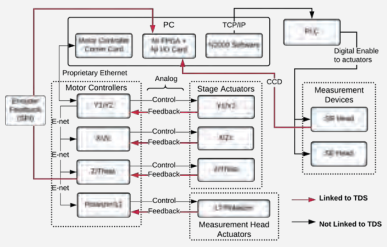


Figure 2: Nanometrics Metrology System Diagram

Client Application Constraints

- Record 19 digital and 10 analog data streams at 1ms period for up to 1 hour
- Generate timing diagram in post-processing, exposing which dependent signals are introducing dead zones

Data Acquisition Unit (DAQ) Constraints

- Timestamp all signal states to eliminate future timing constraints
- Process a 250k baud SPI data stream as a non-invasive slave device
- Process 8 500kHz (max) sin/cos analog incremental motor encoders
- Process 19 digital inputs
- Transmit all data to Client at sufficient throughput to prevent congestion
- Put little to no electrical load on existing system
- Maintain galvanic isolation between all Nanometrics system entities and our DAQ unit
- Interface with existing wiring harness on the machine

Design

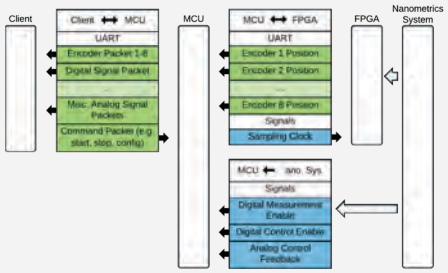


Figure 3: TDS Data Flow Diagram

The Throughput Diagnostic Systems is composed of:

- Client application for visualization and analysis, written in Python with the Qt GUI framework
- STM32F4 microcontroller for real time processing and recording
- Xilinx Artix-7 FPGA for high speed encoder processing
- A collection of peripheral hardware on a PCB for galvanic isolation, analog signal conditioning, and connectors

DAQ Software

- Samples raw digital and analog signals at 1ms
- Receives encoder counters over UART from FPGA
- Receives feedback data over SPI as slave device
- Sends collated, timestamped data over UART to client software.

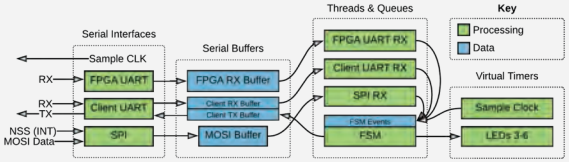


Figure 4: TDS DAQ Software Flow Diagram

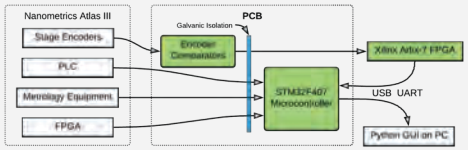


Figure 5: TDS DAQ Hardware Diagram

DAQ Hardware

- The hardware consists of a PCB centered around an STM32F407 microcontroller, and externally interfaces with a Xilinx Artix-7 FPGA board.
- It galvanically isolates all signals tapped from the existing wire harness, connecting encoder signals to the FPGA and the rest to pins on the microcontroller for measurement.

Client Software

- The data from the DAQ is received over UART in a separate process from the GUI.
- Complete packets are pushed onto a thread-safe queue.
- GUI process displays data from queue and conditions it for use in the timing diagram.

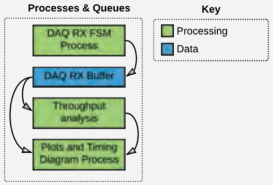


Figure 6: TDS Client Software Flow Diagram

Acknowledgments

Many thanks to Paul Doyle, Vladimir Risko and Tom Baycura at Nanometrics for their guidance in design of the PCB, as well as general support for our project. Also, thanks to Patrick Mantey for providing guidance as our faculty advisor.

Testing / Validation

- We initially tried latching onto the motor controller communication bus to gather data on motor controller performance. We abandoned this data stream after determining that it would require reverse engineering the proprietary protocol.
- Real-time software issues in this project were non-trivial, but thanks to the RTOS we used, very manageable. The accuracy and precision of signal recording were paramount in this project, and the RTOS provided facilities such as virtual timers that allowed for highly precise, periodic execution in parallel with many asynchronous tasks.
- Much of the PCB was the same circuit replicated many times across the board, due to its role as a galvanic isolator from the various signals on the Nanometrics device. For the non-trivial circuits, we purposefully designed the board to allow for minor modifications after fabrication. This allowed us to test the end-to-end system with functional hardware, and then iterate on the PCB in parallel.

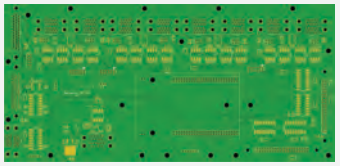


Figure 7: PCB Layout

Results

With the combination of a flexible data acquisition unit and client application, the Throughput Diagnostic System provides the means for test engineers at Nanometrics to identify dead zones in wafer metrology throughput. We were able to automate some portions of dead zone identification, but due to the vast set of events in the metrology system there is still work to be done in future iterations of the TDS. Below is a screen capture of signals depicted in the client GUI application.

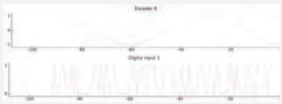


Figure 8: GUI Sample

Conclusion

Identifying input/output constraints of a complex system is challenging; this project was no exception. We spent a substantial amount of our time determining where and how to non-invasively listen in on the Nanometrics' metrology machine for signals and data streams that were relevant to its throughput. We found that some of the, rather rich, data streams could not actually be non-invasively monitored without an advanced spoofing mechanism. In future iterations of this project, we would be excited to see more advanced work be done on the automated statistical analysis of recorded data streams.

UC SANTA CRUZ
Baskin Engineering

Capstone Project

Spine Image Recognition

Madeline Hawkins, Yona Edell, Aryan Samuel,
Stephanie Shi, Hanifah Solachuddin, Davie Truong

nevro

Abstract

In order to prescribe therapy to patients with Nevro's implanted Senza System, x-ray images of the patient's spine containing the implant are analyzed by trained physicians and clinicians. To speed up the process of prescribing therapy, Machine Learning technologies have been implemented to create an image recognition algorithm and an accompanying Graphical User Interface which detects the vertebrae and implanted electrodes, then displays boxes around the detected elements.

Figure 1. Nevro's Senza System that is implanted in a patient's spine.

Approach

The project goal was to create a fully automated prediction based on a Machine Learning model which detects spines, vertebrae, and implanted leads in an x-ray image. This was achieved through the use of technologies such as Keras and Convolutional Neural Networks to create a successful spine image recognition algorithm.

The information is then passed from the Machine Learning backend to the Graphical User Interface created in Python with the Kivy Python framework, which then parses through the data from the Machine Learning backend and translates it into bounding boxes that are layered on top of the displayed image (x-ray). Independently, the Graphical User Interface can also function without a Machine Learning backend and can be used to manually select elements of a loaded x-ray image.

Clinicians can then use the information presented by the user friendly interface to recommend therapy for patients.

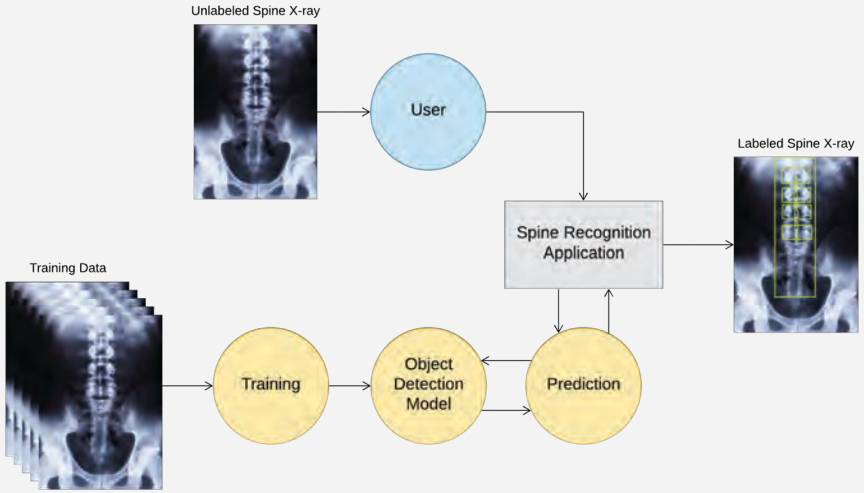
Overview

Nevro Corporation is a medical device company that invented the Senza System, a spinal cord stimulator which uses HF10 therapy to treat chronic back pain.

The project goal was to make identifying vertebrae and implanted leads easier for Nevro clinicians by creating a software to automate the process of detecting and labeling these elements. This expedites the process for clinicians and allows doctors to prescribe the best method of treatment for chronic pain using Nevro's technology.

A fully automated image detection model was developed using Machine Learning that recognizes spines, vertebrae, and leads in a given x-ray image, as well as a Graphical User Interface (GUI) that allows clinicians to manipulate the detection data in a user friendly manner.

Architecture



Acknowledgments

Special thanks to Nevro Corporation, our sponsor for this project, and Sean Knudsen for his guidance and continuous support through the duration of the project. The team would also like to thank Professor Richard Jullig for his help with project management and capstone project support.

Results

Goals achieved by the Graphical User Interface (GUI):

- User will be able to import and export an x-ray image along with its metadata (locations of the vertebrae, leads, discs, as well as the confidence rating of the machine learning algorithm).
- User will be able to draw, edit, and label boxes on an x-ray image to mark locations of vertebrae and leads.
- User will be able to label drawn boxes.

Goals achieved by the Machine Learning model:

- Model will be able to detect with confidence the locations (coordinates) of vertebrae in a spine.
- Model will be able to detect the locations of implanted leads in a spine.
- Model will be able to send all data to GUI for displaying the prediction results.

Figure 2. Screenshot showing GUI and labeled spine X-ray.

Conclusion

The project goal of developing a software to aid in the treatment of patients with Nevro spinal implants was ultimately achieved. The team created a graphical user interface that allows users to import a spinal x-ray image and label the vertebrae or leads in the spine with boxes. The detection of the vertebrae and leads is done by the machine learning model in the background. The end goal is to revolutionize patient treatment and contribute to Nevro's innovative and life changing products.

UC SANTA CRUZ
Baskin Engineering

Capstone Project

TruffleGo Compiler

Trevor Ching, Kyle Remmert, Petar Zaninovich, Vince Kim

ORACLE

Abstract

Oracle's Multilingual Engine (MLE) allows multiple languages to freely interoperate. This enables users to work with a language of their choosing when working with Oracle's database products. The MLE relies upon the Truffle framework and Truffle languages to execute source code within the database process space. The goal of this project was to incorporate a subset of Go into Truffle's suite of languages.

Approach

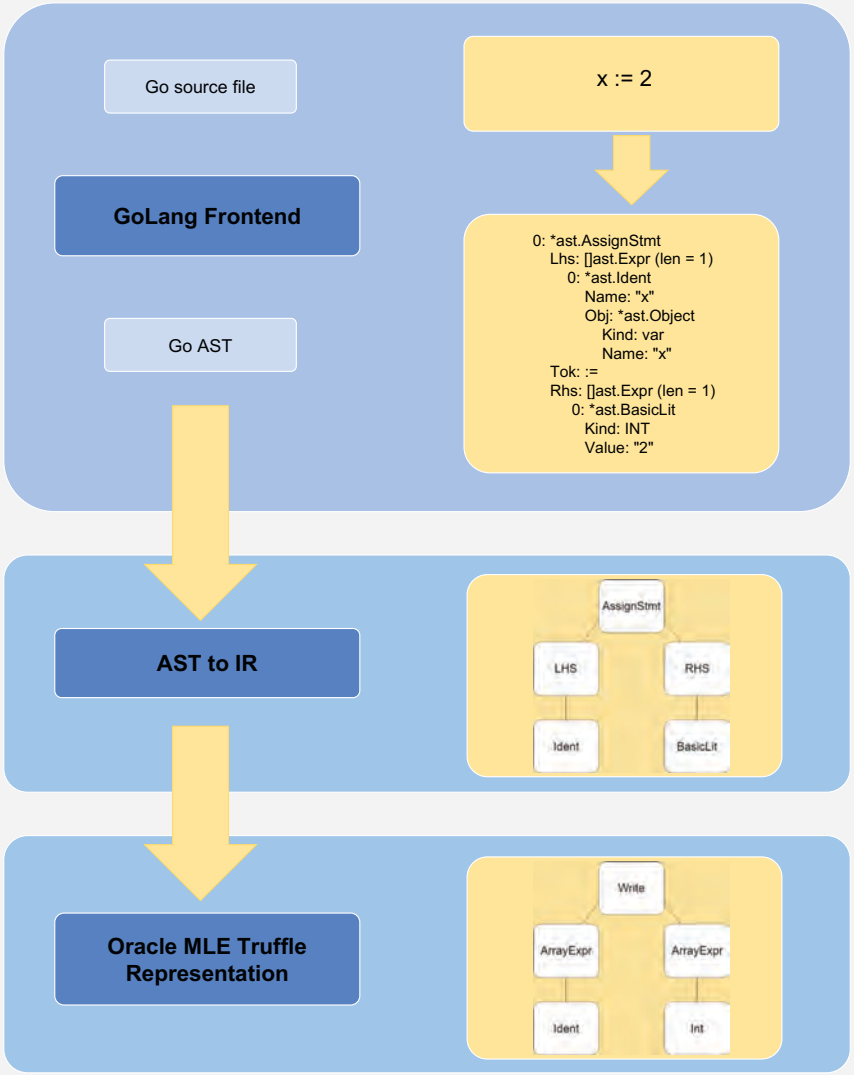
The project followed an iterative approach using the following steps. The team began by defining an increasingly larger subset of the Go language to implement. Then, there were three major stages:

GoLang Frontend
A compiler frontend handles the initial reading of code using a tool called a Parser and a Lexer. Rather then creating a parser and lexer frontend for the Go language, the project team used Go's own Abstract Syntax Tree (AST) package to reduce the workload. Go's AST package parses Go source code and constructs a corresponding AST.

Abstract Syntax Tree (AST) to Intermediate Representation (IR)
An AST is a tree representation of the syntactic structure of source code written in a programming language. Using the information from the frontend the project team formed an intermediate representation layer. This stage transforms the tree created by the frontend into another tree like representation. This representation is used to collect extra information like scoping and perform type checking or transformations. This was simpler than going straight to Truffle, as it allowed the team to get data necessary to form the Truffle representation.

Intermediate Representation to Truffle Representation
Using the visitor design pattern, the compiler performs a post order traversal of the intermediate representation and then generates a Truffle node counterpart with the data it contained. The Truffle node generated corresponds to one node or several nodes of our intermediate representation. Each node has an execute function that performs a specific operation with its attributes. Truffle optimizes performance by specializing each node with a call to Graal, a Just-in-Time compiler.

Architecture



What is Truffle?

Truffle is a language framework that simplifies the process of implementing guest languages in Java using a self optimizing Abstract Syntax Tree interpreter. Guest languages are executed by Graal, a dynamic compiler, via the Truffle framework. Truffle allows for type and object specialization during runtime when combined with the use of annotations. The annotations inform the compiler of what children and attributes each node contains. This enabled the project team to create GoLang semantics by informing Truffle of expected types. This allows the framework to generate code specialized for each type during execution. When the execution count of a Truffle node reaches a predefined threshold, it triggers a call to Graal to apply partial evaluation to specialize the code. If the specialization fails, the node can be reverted to a more generic version. The specializations allow for an increase in performance.

Accomplishments

Through the use of Truffle, the project team was able to implement a subset of Go comprised of the following features:

- Basic Operators
- Data Types (primitive, user-defined)
- Conditionals
- Variables (Read, Write)
- Loops
- Functions (some built-ins, user-defined)
- Imports
- Pointers
- Structs

Future Work

This project's goal was to implement the basic language constructs of Go. To complete the language, the team would need to implement features such as Channels and Interfaces. The team would also work to replace the current frontend, in order to not rely on Go's AST package. These are a few of the core features the team would work towards implementing, along with many of Go's built-in packages.

Acknowledgments

The project team would like to thank Pit Fender, our sponsor, for working with the team and guiding our progress throughout the project process. Special thanks to Richard Jullig, Scott Davis and Dylan Rothfeld for keeping track of our work.

UC SANTA CRUZ
Baskin Engineering

Capstone Project

Seagate Active Drive™ Devices

Hesham Assabahi, Thomas Shum,
Richard Le, Tarik Zeid

SEAGATE

Goal

Explore the computational ability of Seagate Active Drive™ devices for computations on massive datasets, with an emphasis on genetics research.

Approach

We considered several genomics-related computational algorithms for a proof-of-concept implementation using a network of Seagate Active Drives™.

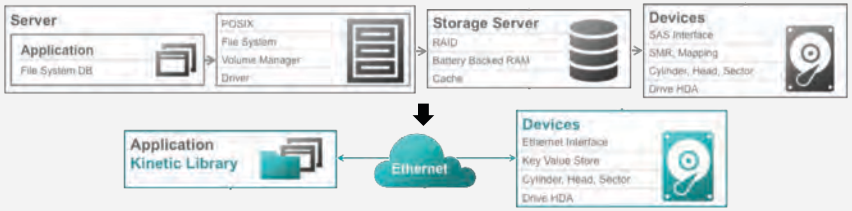
Our first choice, the compression of genetics data, turned out to be too difficult given the team's limited knowledge in bioinformatics.

Instead, we decided to tackle string matching. A crucial aspect of genome sequencing involves finding DNA sequence chunks that overlap each other so that they could be pieced together; so searching for matching patterns would be the first step. As this did not require as much of a background in bioinformatics, we found this easier to tackle.

Challenges

Since we had to use the Active Drive™ API for the host machine to load and execute applets and for the devices to access their local key:value store, any existing applications would have to be adapted to work with the API. This was the biggest reason for why we had to drop the idea of testing distributed genomic data compression, since we lacked the bioinformatics knowledge to tackle trying to port the libraries to build such a system.

Overview

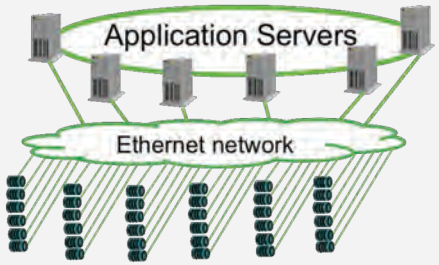


Integrating the functionality of several servers into one appliance.

Seagate Active Drive™ devices are hard drives that build on the Kinetic Device standard, an Ethernet-connected hard drive that provides an interface to a key:value store. Instead of having to manage a database server, storage server, and storage device separately for each key:value store, you only need to manage a single Kinetic Device. This cuts down operational costs and space requirements since you only have one-third as many devices to manage, and reduces latency for any data operations since everything is done on only one device.

Active Drive™ devices take this a step further by integrating computational power into the drives, giving the drives the ability to execute applets. If this can be shown to be practical, then these devices have the potential to even further reduce the data center footprint. The cost of entry barrier for research and development could be significantly lowered with less infrastructure to maintain.

Architecture



Active Drive™ devices connect to your data center's network, just like any other appliance. To control applet execution, there would be at least one other server that manages sending out applets to the drives and telling them when to execute the applet.

Acknowledgments

Richard Jullig (Prof.)
Reihaneh Torkzadehmahani(T.A)
Dylan Rothfeld (T.A)
Chris A Markey (Seagate)
Jon D Trantham (Seagate)
Bryan Wyatt (Seagate)

Results

We were able to get a distributed string search application up and running entirely with the Active Drive™ devices; only one extra computer was needed to do the initial distribution of the applet and the data to the drives, and to aggregate the results from the drives on every run.

Analysis

Not all tasks are suited to run on the drives. The biggest thing to note is that these drives are not powerful; you are not going to get any good results by trying to run computationally intensive tasks on them. However, since the CPU has relatively direct access to its drive's key:value store, and since each drive has its own CPU + RAM, any parallelizable I/O-heavy task is going to benefit the most from this setup. Our string matching program demonstrates that perfectly: even when parallelized on the same machine, the bottleneck is going to be retrieving each string from storage. With our drive setup, each CPU has its own storage to work with, so there wouldn't be any situation with multiple CPUs fighting for access to the same storage device.

Conclusion

Active Drive™ devices do have a practical use cases. Though these devices aren't suited for every application, they can be used in any part of a pipeline that's I/O-heavy, bringing down infrastructure maintenance costs.

UC SANTA CRUZ
Baskin Engineering

Capstone Project

Smart Solar Siting

Christopher Smith, Nicki Thompson, Sam Singh,
Mattheo Ioannou, Andrew Guterres, Henry Hargreaves

Abstract

Solar siting is a process that allows someone to determine the amount of solar energy available at a certain location, e.g. for planning home gardens and photovoltaic systems. The existing methods and tools are either expensive or reliant on approximate measurements.

The high-level goals of the project were to create an application that would reduce the cost of surveys, as well as the human error in overall calculations.

Approach

The main camera view of the app uses augmented reality libraries to superimpose the real solar path on the sky. The green points represent 10-minute intervals of an average day for the given month, and the average day is computed using the Grena-3 solar position algorithm, storing the results in arrays.

The user captures a panoramic image in order to capture the full 180 degree solar path. The app then processes the image using Google Vision (detecting the labeled months and points) and OpenCV (creating a binary image for obstruction identification).

The final stage of the process displays solar availability in graph format with the option to save the solar site or export the data. Solar availability is obtained from NREL data based on user-defined settings and multiplied by the fraction of solar path points visible (without obstruction) for each hour.

Technologies Used

Android Studio: The IDE used to develop the app in Java which provides access to Android tools and libraries.

Google Vision API: An image recognition API which was used for text recognition in order to calculate the amount of available energy per month and hour.

Firebase: The database used for user authentication and storing of data, calculations and images.

OpenCV: An image processing library used to mask sky and obstructions in the captured image, and verify points recognized by Google Vision.

NREL: The National Renewable Energy Lab provides the PVWatts API, which is accessed through an API call based on user-defined system and data settings.

Overview

Smart Solar Siting is an Android application that allows users to perform a solar site survey and obtain monthly and annual data on how much solar energy is available at a certain location. Through the application the user is able to identify obstructions of the solar path as well as see how much energy is potentially available for a specific site. The current methods for doing this include using a device called the SunEye, which costs ~$3000, or by using manual devices that are cheap but require doing everything by hand, including drawing obstructions in and adding up each part of the solar path to find the total. With this application, everything is automated with a nice, clean and simple user interface.

With the average monthly solar path as reference (shown in green with name of month), users can take a picture of the solar site they wish to be analyzed. As the camera turns, the solar path readjusts itself.

The app performs automatic obstruction detection, and users can draw in obstructions that were missed, if any, before confirming the picture to be analyzed.

Once calculations are done by the app, the results can be viewed on charts, which contain average monthly and annual solar energy availability. Users can save the results to their account, or export them (as jpeg or csv).

Acknowledgments

We would like to thank our sponsor Kevin Bell for his guidance in the process of solar site surveying as well as giving us a chance to improve our development skills. We would also like to acknowledge Professor Richard Jullig and our TA Dylan Rothfeld for their support and guidance throughout the entire development process.

Challenges

Image processing to detect obstructions on the solar path, such as trees and buildings, was a major challenge. Two tools are used in the app to bridge the different types of processing needed. OpenCV is used to create a binary image of sky and obstructions (watershed algorithm) and check the values of the solar path points detected by Google Vision.

To provide complete and accurate data, the app has to compensate for the difference between the fish-eye lens typically used in solar siting and the standard phone camera lens. By taking several photos and stitching them together into one panoramic image, the full solar path can be captured. Users still have the option of adding fish-eye lens hardware and extending the field of view in app settings.

Key Features

View average monthly solar path - in the form of green dots over the camera view

Create panorama - Take multiple photos and stitch them into a panorama for a greater field of view

Identify obstructions - between the sky and the phone, automatically and manually if any are missed

View solar availability - in the form of graphs by month or time of day, based on the part of the solar path available and obstructions

Save and export results - store them to the app, download as csv or jpeg or email them externally

Customize - within the settings, change panel configuration for precise calculations

Conclusion

Overall, developing this application has been a valuable learning experience. Using technologies and methods that we never used before gave us good insight into how they could be used in the real world. Through this project we were able to learn more about solar siting and the processes that go along with it, which no one in our group had any prior experience with. In developing this application, we were all able to gain new skills and knowledge that we can apply to future problems.

We are very pleased to include posters for the Senior Design Projects that were done without industry sponsors. Some of these projects were instigated and/or sponsored by research at the Baskin School while others were created by students with the assistance of faculty mentors and TAs.

We have selected three of these projects for presentation in the program, and all were invited to display their posters that summarize their projects.

SproutLabs Mesh | SproutLabs Smart Plant Care & Agriculture | Smart Solar Siting

UC SANTA CRUZ
Baskin Engineering

Capstone Project

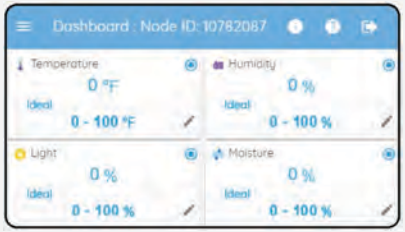
Sproutlabs Mesh

Miguel Calleja, Lucas Dekker, Dianna Kwan, Kevin Lee, Ryan Ortiz

SproutLabs
Data Driven Plant Care & Agriculture

Goal

The goal of this project was to add mesh networking functionality to Sproutlabs devices. Adding mesh network capability decreases the cost and difficulty of setting up and maintaining a set of Sproutlabs devices. It also adds flexibility since a setup will only require one point of internet access. Updating the devices will also be an easy process with this functionality.



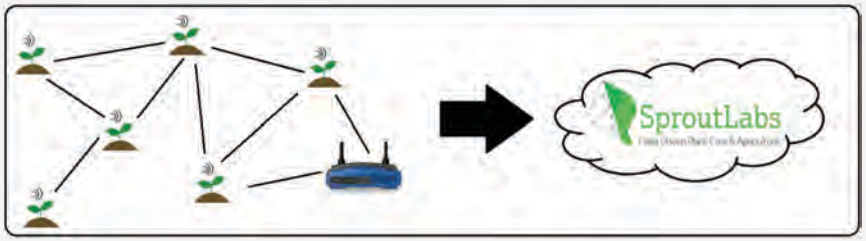
View of data collected by SproutLabs node

Approach

The design of the mesh network was separated into three main phases: connectivity between two devices, passing information through a chain of multiple devices, and automatically configuring a mesh of multiple devices. After adding the functionality to connect two devices, an algorithm was designed for automatically switching between server and client which allowed for transmitting data along a chain of nodes. WIth the chain mesh in place, WiFi credentials could be distributed among a mesh of nodes. A new algorithm was designed for distributing WiFi credentials and automatically registering devices with the Sproutlabs server.

Overview

Sproutlabs devices are WiFi connected smart sensors for agricultural applications. Previously each device required a direct connection to an access point. While this was effective with short range and open space applications, the cost and complexity for the user would quickly increase in situations where the WiFi signal was not optimal. Sproutlabs Mesh adds **mesh networking** functionality to the devices that greatly **reduces** the **cost and complexity** of setting up and maintaining a set of Sproutlabs devices.



Architecture

The initial device is registered to the Sproutlabs website and configured manually. Additional devices that are in range of the router are autoconfigured by the initial device to connect to the router. Devices outside of the range of the router are then configured to connect to the WiFi connected nodes.

To post the sensor data to the website, the child node farthest from the access point enters client mode and sends its information to the nearest server node. The child mode then goes back sleep. The server node then formats its data and appends it to the received data, and enters client mode. This process repeats until the nodes connected directly to the router receive the data. They parse the data fand post the sensor data collected by each node. After sending its data, the device goes into sleep mode for a default of three hours.

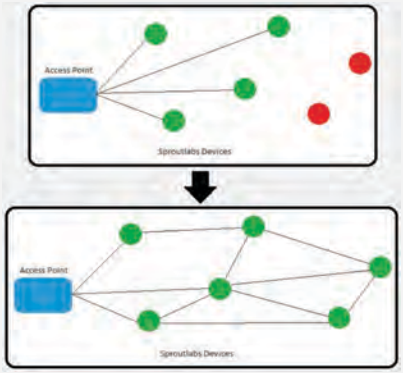
Acknowledgments

We would like to thank our sponsors Babandeep Singh and Ewing Lin for providing us with the opportunity to work with them, and assisting us with this project.

Functions

Sharing credentials If a node is configured manually by the user, it can automatically send the access point's WiFi credentials to all of the child nodes, allowing them to obtain internet access.

Auto-registration Each device can automatically register with the Sproutlabs server so that it can begin posting sensor data to be viewed by the user. This removes the hassle of having to configure each device manually.



Conclusion

The mesh network implementation allows an end user to setup and maintain a large network of Sproutlabs devices with no added infrastructure (other than an existing wifi connection) and minimal configuration time.

Future directions for this technology may include the option of adding automatic OTA firmware updates as well as enabling/disabling functionality for individual devices from the user dashboard.

UC SANTA CRUZ
Baskin Engineering

Capstone Project

Smart Plant Care & Agriculture

Alejandra Buznego, Aaron Le, Anthony Lee,
Alan Duncan, and Anubhav Murali

SproutLabs
Data Driven Plant Care & Agriculture

Abstract

SproutLabs (SL) enables proper and water-efficient plant care by providing users with historical and real-time data collected by sensors in the soil. This project extends user access to plant data beyond mobile and web applications to smart home devices. Users now have convenient, conversational access to plant and sensor health data via Amazon Alexa and Google Home.

Approach

The Amazon and Google Home services were configured to link the user's SproutLabs (SL) account with the corresponding services. The SL API was then integrated with the Amazon Alexa and Google Home platforms using a Lambda function written in JavaScript and running on Node.js. Asynchronous function calls retrieve the data from the database and send it to the smart home platform upon a user's request. An interaction model defines the questions users can ask about their plants (and sensors) and the responses provided by the smart home device. Users can inquire about both plant and sensor data.

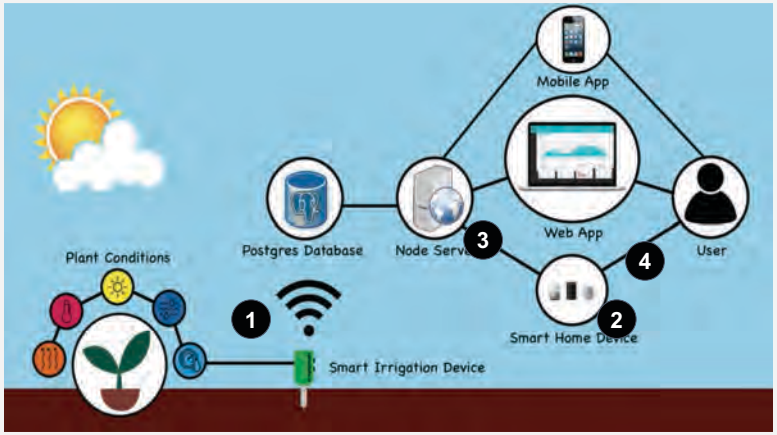
Overview

SproutLabs provides a solution for data-driven plant care and agriculture by introducing a sensor device that reads the temperature, humidity, light, and moisture directly from a plant's surrounding environment. Thus, SproutLabs' approach is two-fold:

- Gather the data from sensors and store in database
- Access the data through mobile/web apps or smart home devices

To facilitate this access, our team has integrated the SproutLabs API with the Amazon Alexa and Google Home services. Now the user can ask Alexa or Google, "How is my plant doing?" or whether their SproutLabs device needs charge. Caring for plants just became that much easier!

Architecture



1)Sensors collect temperature, humidity, sunlight, and moisture levels of the plant and log the readings into the database
2)The interaction model maps the user's question (request) to a corresponding database query (intent)
3)The database is queried via a call to the SL server through a Lambda function running on the Node.js server
4)Using the interaction model, the smart home service formulates a response to convey the data retrieved through the smart home device

Results

Using the OAuth 2.0 protocol, the SproutLabs (SL) application for Amazon Alexa and Google Home establishes a link to the user's SL account to allow the access of the data through these services. Once the end user's credentials have been validated, the user can interact with the smart home device via voice commands and easily access the information about:

1) SL device battery levels
2) Amount of water used
3) Out-of-bounds plant readings
4) Status of plant groups
5) Inactive SL devices

Conclusion

Enabling Smart Home services will facilitate data access and further promote adoption of SproutLabs devices for users who prioritize the convenience of using a smart home device to properly care for plants.

Our modular code base allows for implementation of new capabilities from the SproutLabs devices and provides a foundation for other smart technology implementations in the future.

Acknowledgments

Special thanks to Babandeep Singh, Ewing Lin, Bobby Lyons for their support as sponsors for this project; and to Professor Richard Jullig and Scott Davis for guidance through this project.

Capstone Project

Autonomous Bulldozer Development

James M Trombadore, David Kooi,
Donald Avansino, Kiefer Selmon

UC SANTA CRUZ Baskin Engineering

TOPCON

Objective

Research and develop technology to enable a Bulldozer's autonomous approach and dig operation.

Primarily rely on camera input for the operational decisions of:

- Pile identification and alignment
- Optimal approach calculation
- Dig verification

Target Identification & Alignment

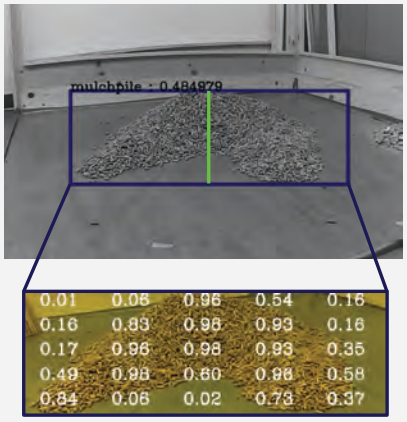
Use Neural Networks for object detection & texture analysis.

Object Detection
- Uses YOLO (You Only Look Once) algorithm
- Trained with public domain construction stockpile images
- Outputs a bounding box and confidence of identified object

Texture Analysis
- Built on the Tensorflow framework
- Trained with laboratory images
- Bounding box is segmented and run through texture network
- Outputs target texture accuracy for each segment

Alignment
- Compute center of confidence based on texture segments
- Center of confidence indicated by green line

Verification
- Discard "false-positive" object detections with invalid texture
- Retry dig operation if bucket has low texture score

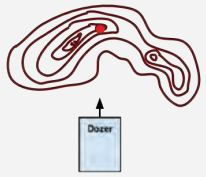


*Networks trained on:
Hummingbird Computational Cluster
UC Santa Cruz Research Computing

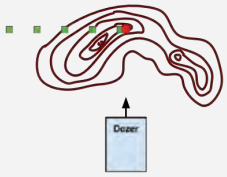
Optimal Approach Calculation

Given a misshapen pile, how to determine the best entry point?

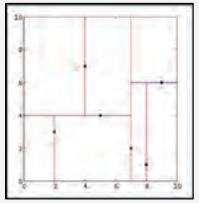
1.) Find the peak straight ahead



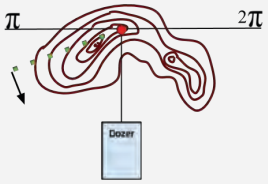
2.) Create a reference plane with origin at the peak.



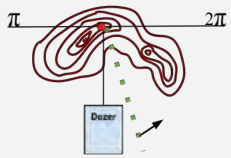
3.) Create a KD-Tree of the point-cloud



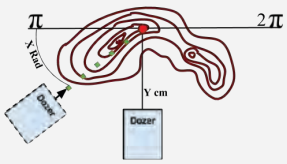
4.) Sweep the reference plane from π to 2π and use the KD-Tree to get the intersection between the pointcloud and the reference plane.



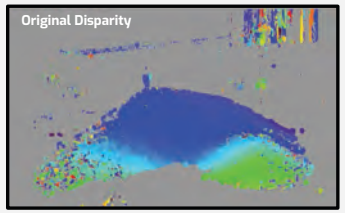
5.) For each intersection compute the area beneath the curve.

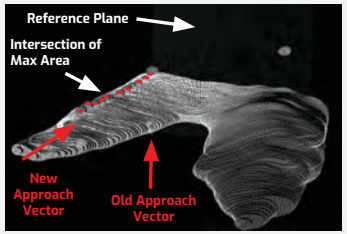


6.) Transform Dozer to align with intersection of maximum area

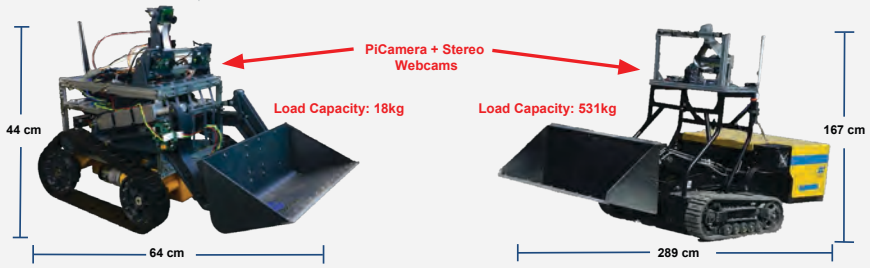


Stereo Camera Feed



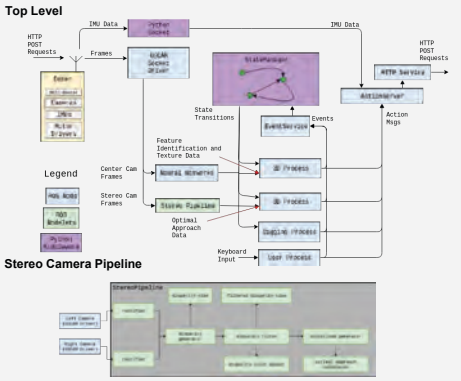


Bulldozers



Software Architecture

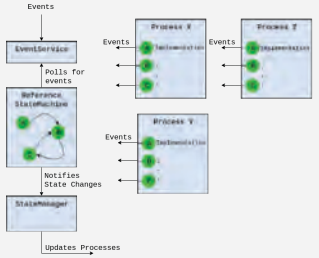
Top Level



State Synchronized Processes

How to separate concerns for parallel development and modularity?

- Create N number of 'processes' to 'observe' a reference state machine
- This is based on the canonical Object-Oriented Observer design pattern



Conclusion and Results

Several key components for an autonomous bulldozer have been developed:

- Autonomous pile approach and alignment
- Optimal approach calculation
- Movement and bucket control command sequencing
- Successful dig validation

Acknowledgments

We would like to thank the following for their help and support:

Prof. David Munday, Michael Grimes, Donna Kelley, Joseph Parsons, Miguel Duenas and the rest of the TopCon Engineering team!

CONTACT US

Patrick Mantey
CITRIS Campus Director
Director of ITI
Jack Baskin Endowed Professor, Computer Engineering
mantey@soe.ucsc.edu

Frank Howley
Senior Director of Corporate Development
fhowley@ucsc.edu

Visit our website:
csspp.soe.ucsc.edu

Join us on Facebook:
facebook.com/BaskinSchoolofEngineering