Motivation

Use Case: Safety Requirements with Deadlines
The Performance Architecture Team (PAT) needs to ensure that safety requirement deadlines will be satisfied.

- Understand the performance of the software
- Detect and analyze execution time outliers

Motivation

Deadline Estimations for Autonomous Vehicles

Tegra system-on-a-chip (SoC)
Project Goal

To provide an pipeline for a user to benchmark the performance of a workload on a Tegra board and visualize the results.
Pipeline Overview

User -> React -> Input API -> django -> Benchmark Output

Linux VM

MySQL

E3550

Maglev

Redis
Live Demo
Control Application

Front-end

- React provided flexibility and modularity
- Served the React app using an Express.js (Node.js) web server managed by PM2

Back-end

- Utilized Django web service
- Contains
  - Benchmarking program
  - Cross compiler
  - MagLev client
  - Automation scripts for the whole pipeline
Benchmarking Framework

- Design Decisions
  - Python-generated C code
  - MySQL for code storage
- Multi-threading
- Data collection
Benchmarking Framework

● Design Decisions
  ○ Python-generated C code
  ○ MySQL for code storage

● Multi-threading

● Data collection
Benchmarking Framework

- Design Decisions
  - Python-generated C code
  - MySQL for code storage
- Multi-threading
- Data collection
Data Visualization

- MagLev and Redash
- A query is dynamically tied to a visualization
Data Visualization

<table>
<thead>
<tr>
<th>Table</th>
<th>Frequency Vs. Execution Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>bins</td>
<td>frequency</td>
</tr>
<tr>
<td>65,000</td>
<td>1</td>
</tr>
<tr>
<td>67,000</td>
<td>1</td>
</tr>
<tr>
<td>68,500</td>
<td>1</td>
</tr>
<tr>
<td>69,000</td>
<td>1</td>
</tr>
<tr>
<td>69,500</td>
<td>4</td>
</tr>
<tr>
<td>70,000</td>
<td>1</td>
</tr>
<tr>
<td>70,500</td>
<td>2</td>
</tr>
<tr>
<td>71,000</td>
<td>3</td>
</tr>
<tr>
<td>71,500</td>
<td>2</td>
</tr>
<tr>
<td>72,000</td>
<td>4</td>
</tr>
<tr>
<td>72,500</td>
<td>9</td>
</tr>
<tr>
<td>73,000</td>
<td>6</td>
</tr>
</tbody>
</table>

1. SELECT floor(exec_time / {{ Bin Size }}) * {{ Bin Size }} AS bins, count(*) AS frequency
2. FROM pat_interns_benchmark.{{ Test Name }}
3. GROUP BY floor(exec_time / {{ Bin Size }}) * {{ Bin Size }}
4. ORDER BY floor(exec_time / {{ Bin Size }}) * {{ Bin Size }}
Data Visualization

![Data Visualization Diagram](image-url)
Conclusion

Goal: To build a platform that measures and visualizes system software performance on Tegra

User Interface

Benchmarking

Visualization
Future Work

● Run benchmark tests on additional platforms (HV+L, QNX)
● Implement a graph to compare between two runs
● Benchmark higher-level workload(s)
● Run benchmark tests in the background and send completion notification emails to users
Special Thanks

- Allen Martin
- Bill Armstrong
- Waqar Ali
- Mitch Luban
- Professor Mark Claypool
- Performance Architecture Team
nvPlayfair
Amanda Chan, Chau Do, Xinzhe Jiang, Kevin Nguyen