Reflection and Traceability Report on TPG

Team 3, Tangle
Calvyn Siong
Cyruss Allen Amante
Edward Gao
Richard Li
Mark Angelo Cruz

1 Changes in Response to Feedback

This section summarizes the changes made over the course of the capstone project in response to feedback from sources such as TAs, the supervisor and other teams. The associated commits can be found by clicking on the associated issue created.

1.1 SRS and Hazard Analysis

Here is the feedback we received on the SRS and Hazard Analysis documents, and the changes we made in response to that feedback.

Table 1: Feedback and Changes for SRS Documentation

Feedback Source	**Feedback Item**	**Response**	**Issue**
TA Feedback	Formalization	Attempted to improve formalization of documentation where possible.	#311
TA Feedback	Extension of Knowledge	Mentioned and cited sources where terms are taken from.	#310
TA Feedback	Verifiable Requirements	Updated requirements to ensure they were testable and measurable.	#309
TA Feedback	Traceable Requirements	Added traceability matrix to enhance traceability.	#308
TA Feedback	What not How (Abstract)	Revised some requirements to focus on "what" the system should do rather than "how" it should do it.	#307
TA Feedback	Content of SRS (Functionality and Specificity)	Revised functional requirements and clarified ambiguous sections.	#305

Feedback Source	Feedback Item	Response	Issue
TA Feedback	Formatting and Style	Modified formatting according to feedback.	#306
Peer Review	Project Goals	Modified project goals associated with peer review.	#106
Peer Review	Verifiability	Adjusted specified requirements for verifiability.	#105
Peer Review	User Business	Clarified problem context.	#104
Peer Review	Dev Planning	Updated development planning section with metrics.	#102
Peer Review	Data Dictionary and Scope	Revised data dictionary.	#101
Peer Review	Maintainability, Supportability, Adaptability Requirements	Adjusted requirements for maintainability, supportability, and adaptability.	#107
Peer Review	Fix Functional Requirements	Revised concerned FR-6 for specificity.	#103

Table 2: Feedback and Changes for Hazard Analysis

Feedback Source	Feedback Item	Response	Issue
TA Feedback	Recommended Actions	Modified actions to be more actionable.	#314
TA Feedback	Hazard Identification	Adjusted concerned sections with feedback.	#313
TA Feedback	Spelling and Grammar	Corrected spelling and grammar errors and implemented other feedback specified.	#312
Peer Review	Inconsistent Hazard Reference	Fixed inconsistency between hazard references.	#136
Peer Review	Potential Missing Hazard for FMEA	Added missing hazards to the FMEA analysis.	#135
Peer Review	Priority Assignment	Revised priority assignments based on updated risk assessments.	#133
Peer Review	No Mitigation Strategy	Modify mitigation strategies for hazards.	#132
Peer Review	Prioritization Justification	Provided detailed justification for hazard prioritization.	#130
Peer Review	SRS Linking Roadmap	Linked SRS in roadmap to hazard analysis.	#128
Peer Review	Ambiguous Terms	Clarified ambiguous terms in the hazard analysis.	#134

1.2 Design and Design Documentation

Here is the feedback we received on the design documents (MG and MIS), and the changes we made in response to that feedback.

Table 3: Feedback and Changes for Module Guide

Feedback Source	Feedback Item	Response	Issue
TA Feedback	Quality Information	Fixed all addressed concerns with issue.	#346
Peer Review	Lack of Links to Other Documents	Added links and references to related documents/sections for better traceability.	#242
Peer Review	Module Decomposition	**Did Not Fix**: Decomposition was deemed unnecessary for the current scope.	#240

Table 4: Feedback and Changes for Module Specification Interface

Feedback Source	Feedback Item	Response	Issue
TA Feedback	Sketches for Enough to Build	**Did Not Fix**: Did not include additional sketchecs or examples, as current level of detail seemed sufficient for our project scope.	#347
Peer Review	Confusion on TPG Experiment Module	Clarified confusing sections in the module specification interface.	#245
Peer Review	Lack of Info for Independent Developer	Added additional details to support independent developers.	#243
Peer Review	Incorrect "Uses" in MIS	Corrected "Uses" subsections in the module specification interface for modules.	#244

1.3 VnV Plan and Report

2 Challenge Level and Extras

2.1 Challenge Level

[State the challenge level (advanced, general, basic) for your project. Your challenge level should exactly match what is included in your problem statement. This should be the challenge level agreed on between you and the course instructor. —TPLT]

2.2 Extras

[Summarize the extras (if any) that were tackled by this project. Extras can include usability testing, code walkthroughs, user documentation, formal proof, GenderMag personas, Design Thinking, etc. Extras should have already been approved by the course instructor as included in your problem statement. —TPLT]

3 Design Iteration (LO11 (PrototypeIterate))

The design and implementation of our capstone project were driven by the needs of our client, Dr. Kelly, and his research group. As contributors to their existing codebase, our efforts focused on addressing pain points in current workflows and enhancing the overall developer experience, aligning with the goal of facilitating research on the TPG framework within more complex environments.

3.1 MuJoCo Environment Integration and Expansion

Our initial work involved integrating MuJoCo environments to expand the capabilities of the TPG framework. Dr. Kelly's group had previously implemented the "Ant" environment. Recognizing the need to evaluate the TPG framework in more advanced physics engines, we implemented support for additional environments such as Inverted Pendulum, Humanoid Standup, and Half Cheetah. This directly addressed Dr. Kelly's desire to transition to MuJoCo, enabling experimentation within more sophisticated and realistic simulations.

This effort culminated in addressing Dr. Kelly's primary research question: evaluating TPG's performance in multi-task scenarios. Building upon the individual environment implementations, we iterated on combining MuJoCo environments, enabling the evolution of policies capable of learning across multiple tasks. This progression demonstrates a clear evolution driven by the client's research objectives, moving from basic integration to tackling more complex, multi-task learning problems.

3.2 Migration to .csv Logging

A significant change involved modifying the logging system. Dr. Kelly expressed a desire to transition data analysis processing from R to Python-based tools, citing maintainability and the flexibility of Python libraries as key motivations. To accommodate this transition, we migrated the logging format from `.std` and `.err` files to `.csv` files.

The choice of `.csv` over the previous `.std` format offers several advantages from the perspective of user needs:

- **Organization:** `.csv` files provide a structured, tabular format. This makes it easier to import and analyze data using Python libraries like Pandas, streamlining the data processing pipeline. The previous `.std` format required more complex parsing and was less amenable to automated analysis.
- **Accessibility:** `.csv` is a widely supported and easily accessible format. Researchers can readily open and manipulate `.csv` files in various software packages, fostering collaboration and simplifying data sharing.

- **Specific Metric Capture:** We designed the `.csv` logs to capture specific metrics from each stage of the evolutionary process. This allows for targeted analysis of performance and behavior, providing Dr. Kelly and his team with the data they need to evaluate the TPG framework effectively.

By migrating to `.csv` logs, we not only accommodated Dr. Kelly's preference for Python-based tools but also improved the organization, accessibility, and analytical potential of the logged data, directly benefiting the research workflow.

3.3 CLI Tool Development

Initially, the execution of experiments relied on script-based execution, requiring developers to navigate to specific directories and execute commands with specific parameters. This workflow was identified as a pain point, particularly when running multiple experiments concurrently.

In response to this usability issue, we developed a Command Line Interface (CLI) tool to simplify experiment execution. This change was presented to Dr. Kelly, who agreed that a streamlined workflow would significantly improve the developer experience.

The initial "MVP" version of the CLI tool supported essential functions such as:

- `tpg evolve [environment]`: Evolving a policy for a given environment.
- `tpg replay [environment]`: Replaying the best-performing agent in an environment.
- `tpg plot [environment]`: Plotting relevant statistics.

These commands mirrored the functionality of the original scripts but offered a more intuitive and descriptive interface. After merging the initial version and providing documentation, Dr. Kelly's research group provided valuable feedback. They requested additional functionality, such as:

- `tpg clean`: Removing old log files to maintain a clean working environment.
- `tpg kill`: Terminating MPI processes.

Based on this feedback, we iterated on the CLI tool, incorporating these new features to further streamline the user experience and address the specific needs of the research team. The final CLI tool represents a significant improvement in usability, allowing researchers to execute, manage, and analyze experiments with greater ease and efficiency.

In summary, our design iterations were continuously guided by the needs of Dr. Kelly and his research group. By focusing on improving existing workflows, accommodating new technologies, and responding to user feedback, we were able to develop a system that directly supports their research goals and enhances the overall developer experience.

4 Design Decisions (LO12)

[Reflect and justify your design decisions. How did limitations, assumptions, and constraints influence your decisions? Discuss each of these separately. —TPLT]

5 Economic Considerations (LO23)

[Is there a market for your product? What would be involved in marketing your product? What is your estimate of the cost to produce a version that you could sell? What would you charge for your product? How many units would you have to sell to make money? If your product isn't something that would be sold, like an open source project, how would you go about attracting users? How many potential users currently exist? —TPLT]

6 Reflection on Project Management (LO24)

[This question focuses on processes and tools used for project management. —TPLT]

6.1 How Does Your Project Management Compare to Your Development Plan

[Did you follow your Development plan, with respect to the team meeting plan, team communication plan, team member roles and workflow plan. Did you use the technology you planned on using? —TPLT]

6.2 What Went Well?

[What went well for your project management in terms of processes and technology? —TPLT]

6.3 What Went Wrong?

[What went wrong in terms of processes and technology? —TPLT]

6.4 What Would you Do Differently Next Time?

[What will you do differently for your next project? —TPLT]

7 Reflection on Capstone

This section focuses on the key learnings and reflections gained during the course of the capstone project.

7.1 Which Courses Were Relevant

Several courses proved highly relevant to the successful completion of our capstone project. Among the most impactful were:

1. **2AA4 - Introduction to Software Design:** This course provided a foundational understanding of collaborative software development practices. The experience of working on a group project using GitHub, including managing issues, utilizing Kanban boards, and creating pull requests, was invaluable. It instilled the fundamentals of effective team-based development. Furthermore, the emphasis on software design principles, such as SOLID and GRASP,

was directly applicable to the capstone project. Although the course was primarily focused on Java, the object-oriented design patterns learned were transferable to our project, which utilized Python and C++. Specifically, we leveraged the Observer pattern to implement a refined logging system within the TPG codebase.

2. **3RA3 - Requirements Engineering:** This course provided a solid foundation in defining project requirements prior to implementation. Learning to articulate and prioritize user needs was instrumental in guiding our development efforts. The user-based design principles learned in 3RA3 helped us focus our efforts and prioritize features that aligned with Dr. Kelly's research objectives. The reports we submitted throughout the capstone project mirrored the structure and content of assignments completed in 3RA3, demonstrating the practical application of the course's principles.

7.2 Knowledge/Skills Outside of Courses

Beyond the formal curriculum, our capstone project required the acquisition of specific knowledge and skills in areas not explicitly covered in our coursework. These included:

1. **Genetic Programming:** Gaining an understanding of genetic programming, a specialized research area, was essential. Since none of us had prior exposure to this domain, our learning primarily involved direct instruction from our supervisor, Dr. Kelly, as well as in-depth analysis of the existing TPG codebase. This process required us to quickly grasp complex concepts and apply them to our project.

2. **C++ Proficiency:** C++ was the language of choice for performance-critical aspects of the project, a decision driven by Dr. Kelly's preference. While our curriculum included object-oriented languages like Java, C++ was not a primary focus. We therefore had to develop proficiency in C++ independently, leveraging our understanding of object-oriented principles to navigate the language's syntax and intricacies.

3. **Docker Containerization:** The project made extensive use of Docker for containerization, a topic not typically covered in our academic coursework. While some of us had gained exposure to Docker during co-op placements, a deeper understanding of how containers work and their benefits was necessary. We independently learned how to install, run, and utilize Docker, developing a valuable skill essential in modern software development workflows. Given the prevalence of containerization in industry, incorporating some introductory material on this topic into the curriculum could benefit future students.