# 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 for-	#311
		malization of documentation	
		where possible.	
TA Feedback	Extension of	Mentioned and cited sources	#310
	Knowledge	where terms are taken from.	
TA Feedback	Verifiable Require-	Updated requirements to en-	#309
	ments	sure they were testable and	
		measurable.	
TA Feedback	Traceable Require-	Added traceability matrix to	#308
	ments	enhance traceability.	
TA Feedback	What not How (Ab-	Revised some requirements to	#307
	stract)	focus on "what" the system	
		should do rather than "how"	
		it should do it.	
TA Feedback	Content of SRS	Revised functional require-	#305
	(Functionality and	ments and clarified ambigu-	
	Specificity)	ous sections.	

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

Table 2: Feedback and Changes for Hazard Analysis

Feedback Source	Feedback Item	Response	Issue
TA Feedback	Recommended Ac-	Modified actions to be more	#314
	tions	actionable.	
TA Feedback	Hazard Identifica-	Adjusted concerned sections	#313
	tion	with feedback.	
TA Feedback	Spelling and Gram-	Corrected spelling and gram-	#312
	mar	mar errors and implemented	
		other feedback specified.	
Peer Review	Inconsistent Hazard	Fixed inconsistency between	#136
	Reference	hazard references.	
Peer Review	Potential Missing	Added missing hazards to the	#135
	Hazard for FMEA	FMEA analysis.	
Peer Review	Priority Assign-	Revised priority assignments	#133
	ment	based on updated risk assess-	
		ments.	
Peer Review	No Mitigation	Modify mitigation strategies	#132
	Strategy	for hazards.	
Peer Review	Prioritization Justi-	Provided detailed justifica-	#130
	fication	tion for hazard prioritization.	
Peer Review	SRS Linking	Linked SRS in roadmap to	#128
	Roadmap	hazard analysis.	
Peer Review	Ambiguous Terms	Clarified ambiguous terms in	#134
		the hazard analysis.	

## 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 Informa-	Fixed all addressed concerns	#346
	tion	with issue.	
Peer Review	Lack of Links to	Added links and references to	#242
	Other Documents	related documents/sections	
		for better traceability.	
Peer Review	Module Decomposi-	Did Not Fix: Decomposi-	#240
	tion	tion was deemed unnecessary	
		for the current scope.	

Table 4: Feedback and Changes for Module Specification Interface

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

### 1.3 VnV Plan and Report

Here is the feedback we received on the VnV Plan and VnV Report, and the changes we made in response to that feedback.

Table 5: Feedback and Changes for VnV Plan

Feedback Source	Feedback Item	Response	Issue
TA Feedback	Inaccurate descrip-	Modified the two sections to	#315
	tions in 2.1 Sum-	better reflect rubric	
	mary and 2.2 Ob-		
	jective		
TA Feedback	Improve Spelling	Addressed feedback given in	#316
	and Grammar and	issue	
	Style of VnV Plan		
Peer Feedback	Add traceability in	Modified traceability matrix	#150
	VnV Plan	to reflect documentation	
Peer Feedback	Increase Detail in	Addressed feedback in issue	#151
	Nonfunctional Test	by adding details	
	Descriptions		
Peer Feedback	Survey Questions	Added survey questions	#155
	for Documentation		

Table 6: Feedback and Changes for VnV Report

Feedback Source	Feedback Item	Response	Issue
Peer Feedback	Limited Explana-	Did Not Fix: Deemed not	#398
	tion of Changes	relevant for VnV Report.	
	Due to Testing		
Peer Feedback	Missing Test De-	Updated the VnVReport with	#398
	scription for FR-	the test's full description in	
	SNL6 in both VnV	the same format as other tests	
	documents		
Peer Feedback	Traceability Matrix	Removed test to get rid of in-	#398
	Inconsistency	consistency	
Peer Feedback	Lack of Criterion	Added criteria for usability	#400
	for Usability Tests	tests	
Peer Feedback	Missing NFR test	Included proper NFR solution	#401
	case that is present		
	in the VnVPlan		
Peer Feedback	Lack of clarity with	Clarified what changes are	#402
	feedback changes	made in response to feedback	

# 2 Challenge Level and Extras

## 2.1 Challenge Level

The challenge level of the project is **advanced** as agreed upon by the course instructor since this project is an extension of the current Tangled Program Graphs repository created by Dr. Stephen Kelly.

#### 2.2 Extras

The extra tackled by this project is a Research Report. This report will cover the research and results obtained from incorporating dynamic memory to enhance reinforcement learning within MuJoCo using Tangled Program Graphs. It explores this through single-task (STL) and multitask (MTL) experiments on MuJoCo environments such as Inverted Pendulum, Half Cheetah, and Humanoid Standup. The Research Report can be found here.

## 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)

Since the project will eventually be open-sourced once the research work is finished, its primary market does not include direct commercial sales but rather the research community, AI practitioners, and robotics engineers and enthusiasts.

Attracting users will require a combination of strategies to enhance community engagement, increase project visibility, and improve usability. A few key approaches are:

- Open-Source Repository: Hosting the project on GitHub with a well-organized README, issue tracker and contribution guidelines would encourage potential contributors.
- Conference & Workshop Presentations: Presenting the project at ML and robotic conferences such as GECCO and Conference on Artificial Life would increase its visibility among researchers.
- Publishing Research & Documentation: Continue releasing research papers and technical reports explaining TPG's capabalities and how well it compares with alternatives.
- Showcasing Notable Results: Publishing blog posts, YouTube videos, as well as engaging with the reinforcement learning community through Hugging Face forums, GitHub discussions, Reddit and other online platforms.

TPG's potential lies in academic research, robotics applications, and open-source adoption. The long-term strategy would focus on gaining traction in research and industry collaborations, which could lead to funding opportunities, grants, or potential commercial applications in reinforcement learning for real-world robotic systems.

While it's difficult to estimate exact numbers of potential users, open-source RL-based GitHub projects often have thousands of users. For example, OpenAI's Gym has 35.6k stars GitHub, while Stable Baselines3 has around 10.2k stars. Given TPG's unique focus, it has the potential to attract a niche but highly engaged user base of researchers, engineers, and developers in the AI and robotics space.

# 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]

Our team properly followed our development plan with respect to the workflow plan, team member roles, and communication plan. As stated in our Development Plan, we used the "TPG Capstone" Discord server for team communication, which was consistently utilized throughout the project to discuss ideas, host meetings and notify the team for PR requests. Additionally, the

development stack and technologies mentioned in the Development Plan, such as Git, GitHub, and C++, were integral to the development of our project.

We made sure that each team member adhered to their designated roles and responsibilities by assigning each member tasks that matched their roles. Our team meetings were held regularly, and we successfully coordinated schedules to accommodate everyone, leading to regular team attendance. Overall, our project management was carried out efficiently, staying true to our original plan.

#### 6.2 What Went Well?

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

Our project management was effective, particularly in the implementation of MuJoCo environments, which proceeded smoothly. This success allowed us to build upon our initial work by exploring multi-tasking for the TPG-Mujoco interface. Additionally, we enhanced our project with the development of new CLI tools, which streamlined our workflow and improved the overall efficiency of our processes. Our team communication and collaboration were strong throughout the project. This ensured that all team members were aligned with project goals and timelines. The use of GitHub for version control and project management also contributed to our success, allowing us to track progress effectively and manage tasks efficiently through issues. Overall, these factors combined to create a productive working environment.

### 6.3 What Went Wrong?

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

One of the challenges we faced was the inability to implement all the MuJoCo environments we initially planned due to unforeseen complications. Additionally, the process of migrating from SCons to CMake proved to be quite challenging due to a lack of updated documentation. Debugging C++ was particularly difficult as changes were occurring regularly in the TPG framework, and our limited experience with C++ added to the complexity. These issues highlighted areas where we needed to improve our skills and adapt our strategies to better handle such challenges in the future.

### 6.4 What Would you Do Differently Next Time?

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

In future projects, we would aim to diversify our roles within the team, allowing each member to explore multiple areas of software development. This would encourage a more varied approach and broaden our collective skill set, as it challenges us to step outside our comfort zones and tackle different aspects of the development process. Additionally, experimenting with a different development technologies and frameworks could be both exciting and beneficial, as it presents new challenges and learning opportunities. While our current stack was effective, exploring newer frameworks or languages might enhance performance and developer efficiency.

We would also likely place a greater emphasis on more comprehensive unit testing to ensure robust code quality, as our testing scope was relatively limited. This would involve more inclusive testing strategies that cover a wider range of scenarios and edge cases. We would likely also have implemented more complex MuJoCo environments for evaluation, which would have provided a deeper understanding of the TPG framework's capabilities and limitations in handling sophisticated simulations.

### 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.