Problem Statement and Goals
SE 4G06

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

Table 1: Revision History

Date	Developer(s)	Change
9/21/2024	Calvyn Siong	Added changes to stakeholder, environment and goals section
...	...	...

1 Problem Statement

[You should check your problem statement with the problem statement checklist. —SS]

[You can change the section headings, as long as you include the required information. —SS]

1.1 Problem

1.2 Inputs and Outputs

[Characterize the problem in terms of "high level" inputs and outputs. Use abstraction so that you can avoid details. —SS]

1.3 Stakeholders

1. **Primary Stakeholders: Dr. Stephen Kelly's TPG Research Team**

   Dr. Kelly and his research team are the primary stakeholders of this project. As the original developers of the TPG framework, they are directly invested

in and benefit from improving the framework's software engineering practices and expanding its applicability to more complex environments. The success of this project will directly contribute to their ongoing research efforts in reinforcement learning (RL) and genetic programming. The additional integrations created for TPG will allow the group to more efficiently explore new research avenues, including the application of TPG in robotics and advanced simulations.

2. **Secondary Stakeholders: The GEGELATI Team**

   The GEGELATI team, a French-based TPG research organization, is a secondary stakeholder. They are adopting the TPG algorithm for research purposes that differ from Dr. Kelly's project. Dr. Kelly has stated that this group references his TPG project for their own research purposes. While they are not directly involved in the development of this project, they stand to benefit from its outcomes. Improvements in the TPG framework, particularly in terms of software reliability and scalability, will provide valuable insights and tools that can be incorporated into their research. The establishment of an interface between TPG and modern robotics simulators may also open up new possibilities for their applications of the TPG algorithm.

3. **Tertiary Stakeholders: The Wider Reinforcement Learning (RL) Community**

   The broader reinforcement learning community is a tertiary stakeholder in this project. Researchers and developers interested in the intersection of evolutionary algorithms and machine learning may benefit from the enhanced TPG framework. The improvements made through this project, including standard software engineering practices and expanded testing environments, will increase TPG's accessibility and usability. Additionally, the publication of project results in international conferences could raise awareness of TPG's capabilities, potentially encouraging wider adoption and collaboration within the RL community.

1.4 Environment

The capstone project will primarily be based on the existing TPG framework environment, which will consist of the existing C++ codebase which will be refactored and modified to comply with standard software engineering principles, such as continuous integration, automated testing, and improved documentation. In addition, a key objective of the capstone is to extend the TPG framework beyond its current application in Atari video games by integrating it into **MuJoCo (Multi-Joint dynamics with Contact)**, a widely used physics engine developed by **Google DeepMind**. MuJoCo provides highly accurate physical simulations, particularly suited for robotics. By interfacing TPG with MuJoCo, the project aims to evaluate the performance of the framework in controlling complex robotic systems, such as quadrupeds. This integration

will allow for more realistic testing environments, expanding the boundaries of TPG's current capabilities.

2 Goals

1. **TPG Framework must adhere to a modern software engineering Dev Ops pipeline**

   Set up fully automated CI pipeline to ensure all code changes in the TPG framework are continuously tested, verified and follow proper guidelines

   Measurable Outcomes: To accomplish this goal, there should be a complete integration of a CI tool such as Github Actions or Gitlab CI. The CI pipeline should be invoked when changes are made to the framework via new PRs and should not result in blocking errors (either accept or reject changes after CI pipeline analyzes changes).

   Reasoning: Establishing a CI pipeline will standardize software development workflows, reduce human error, and improve code reliability. It ensures that the framework is stable with each update, promoting scalability and maintainability

2. **The TPG Framework should be well tested**

   Develop a comprehensive testing suite for the TPG framework, including unit tests, to ensure major components are thoroughly validated and code coverage is maximized.

   Measurable Outcomes: To accomplish this goal, unit and integration tests will be implemented for core TPG functions, ideally hitting at least 80% code coverage (subject to change after further analysis). The tests must be able to be run as part of the CI pipeline, ensuring that all functionality remains valid after code changes.

   Reasoning: Introducing a well-structured testing suite is critical for ensuring the reliability and correctness of the TPG framework. Unit tests will verify that individual components behave as expected, and adding integration tests will confirm the system functions correctly as a whole. High test coverage ensures that potential issues are caught early, reducing future bugs and improving maintainability. This goal also aligns with the project's emphasis on adopting best practices in software engineering.

3. **The TPG framework must have an Agent-Environment Interface between itself and MuJoCo**

   Develop a robust interface that allows the TPG framework to communicate and control robotic agents within the MuJoCo physics engine.

   Measurable Outcomes: To accomplish this goal, the TPG framework should contain a fully functional interface for a basic agent (e.g., a program that controls a simple joint robotic arm) in MuJoCo. By the end of the

project, this can be further expanded to having the interface to support more complex agents.

Reasoning: This interface will enable the TPG framework to be tested in more realistic and complex environments. Successful integration with MuJoCo will validate TPG's performance in a cutting-edge robotics simulator, essential for showcasing its real-world capabilities. Furthermore, the current testing environment for TPG is stationary (all states are known/given), and integrating with Mujoco enables for testing with dynamic environments, where the agents will have to change over time.

4. **The TPG Integration in MuJoCo should have determinable behavior when ran**

   Perform a series of experiments to evaluate the behavior and performance of agents controlled by the TPG algorithm within the MuJoCo environment.

   Measurable outcomes: Analyze and record key metrics such as agent performance, model selection process, task success rate, and behavior patterns in various scenarios.

   Reasoning: Extending upon the previous goal, these experiments will provide insights into how well TPG can control complex agents in a physics-based simulator like MuJoCo. Understanding the agent's behavior in a dynamic environment will be essential for evaluating the practical applicability of TPG, similar to the previous goal.

3 Stretch Goals

1. **The TPG framework is integrated with additional Robotics Simulators**

   Extend the agent-environment interface to support additional simulators

   Reasoning: Supporting multiple simulators will increase the flexibility of TPG, allowing it to be tested in diverse environments. This would expand the framework's use cases, enhancing its appeal to the wider RL and robotics communities.

2. **The TPG framework is linked to a User-Friendly GUI for Experiment Management**

   Create a graphical user interface (GUI) for researchers to manage, run, and visualize experiments within the TPG framework.

   Reasoning: A user-friendly GUI would make the TPG framework more accessible and easier to use for the stakeholders involved. This could result in more efficient usage of the framework.

4 Challenge Level and Extras

The challenge level of this project is deemed to be **general**, as it builds upon an existing foundation of work developed by Dr. Kelly's research team.

While the project involves expanding the Tangled Program Graphs (TPG) framework and integrating it with new environments, the foundational research, including the algorithmic structure of TPG and its application in simpler environments, has already been established. This provides a stable starting point, reducing the complexity typically associated with new algorithm integrations.

Although the integration of TPG with advanced robotics simulators such as MuJoCo introduces new technical challenges with an unexplored and wider scope (in regards to extending the application to robotics simulations), these are manageable due to the existing knowledge base and the prior successful implementation of TPG in simpler environments like Atari video games. The task primarily involves adapting and extending the current work rather than developing entirely new algorithms.

Additionally, the software engineering improvements, such as implementing continuous integration, introducing a testing suite, and generating comprehensive documentation, follow standard best practices and are well-supported by available tools. These tasks, while requiring careful planning and execution, are considered general development work and are feasible within the scope of the project.

One of the extras to accompany this project will involve implementing **user documentation** to support the newly integrated features and enhancements. This documentation will be designed to guide users through the updated functionalities, including the integration of the TPG framework with the MuJoCo simulator, as well as any improvements made to the codebase and testing suite.

Appendix — Reflection

[Not required for CAS 741 —SS]

The purpose of reflection questions is to give you a chance to assess your own learning and that of your group as a whole, and to find ways to improve in the future. Reflection is an important part of the learning process. Reflection is also an essential component of a successful software development process.

Reflections are most interesting and useful when they're honest, even if the stories they tell are imperfect. You will be marked based on your depth of thought and analysis, and not based on the content of the reflections themselves. Thus, for full marks we encourage you to answer openly and honestly and to avoid simply writing "what you think the evaluator wants to hear."

Please answer the following questions. Some questions can be answered on the team level, but where appropriate, each team member should write their own response:

1. What went well while writing this deliverable?
2. What pain points did you experience during this deliverable, and how did you resolve them?
3. How did you and your team adjust the scope of your goals to ensure they are suitable for a Capstone project (not overly ambitious but also of appropriate complexity for a senior design project)?