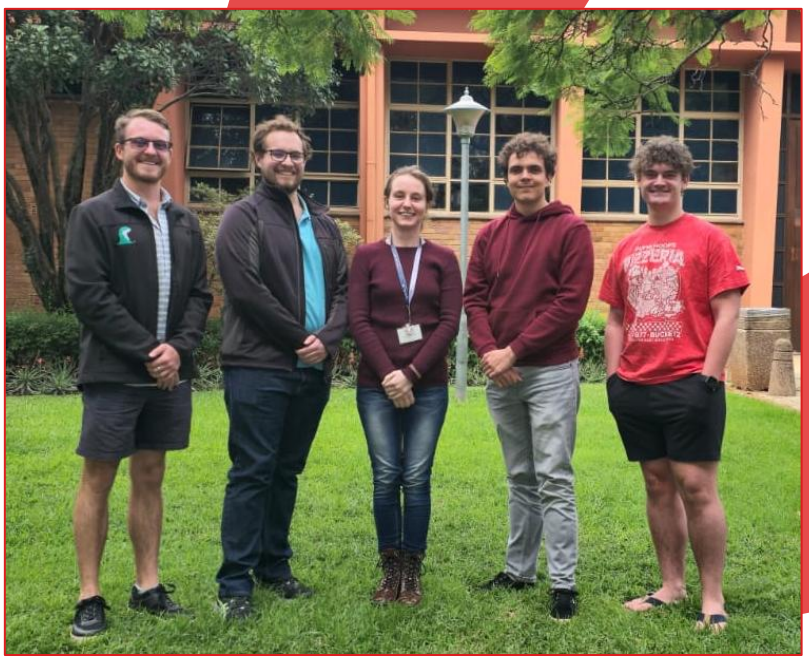
SuperLap Racing Line Optimization System

EPI-USE



Quintessential

Amber Ann Werner [u21457752]

Milan Kruger [u04948123]

Qwinton Knocklein [u21669849]

Sean van der Merwe [u22583387]

Simon van der Merwe [u04576617]

Requirements

Functional Requirements

R1: Track Image Processing

R1.1: Image Conversion

- The system will convert top-down racetrack images into binary maps for AI analysis.
- The system will load data from saved csv files for comparison.

R1.2: Boundary Detection

- The system will accurately detect and distinguish track boundaries from off-track areas.
- The system will store this information for future use.

R2: Racing Line Optimization

R2.1: Reinforcement Learning

- The system will apply Reinforcement Learning (RL) to simulate and refine racing lines.
- The system will use image data to train the CNN.
- The system will use the track lines to help train the ACO.

R2.2: Path Evaluation

- The system will iterate through multiple paths to determine the fastest racing line.

R3: AI Training and Simulation

R3.1: Training Data Input

- The system will train AI agents using simulated/ game-based datasets.

R3.2: Physics Modelling

- The system will incorporate physics-based models to ensure realistic performance.

R4: Result Visualization

R4.1: Line Overlay

- The system will overlay the optimized racing line on the track image.
- The system will allow for adjustments to the overlay.

R4.2: Performance Metrics

- The system will display key performance indicators such as estimated lap time and braking zones.

R5: Infrastructure Integration

R5.1: Computation Support

- The system will support GPU-accelerated or equivalent computational resources for efficient RL training.

Wow Factors

R6: Game Intergradation

R6.1: Record a lap in MotoGP 18

- The system will allow users to race laps within MotoGP and save this lap to the system to compare it to the AI generated optimum lap.

R7: Enhanced Visualization

R7.1: Heatmap of Speed/Acceleration Zones

- The system will showcase where users must increase speed or break on a given track.

Technology requirements

Client-Side Technology Requirements

Desktop Application Platform

- **Unity Engine 6.0+**: Primary development platform for desktop client
- **Unity Hub 6.1**: Project management and Unity version control
- **Target Platform**: Windows 11 (primary), with potential Linux support
- **Programming Language**: C# for Unity components and business logic

Client System Requirements

Minimum Requirements:

- **Operating System**: Windows 11
- **CPU**: Intel Core i5 or AMD equivalent
- **RAM**: 8GB (end users), 18GB (development), 32GB (optimal Unity performance)
- **Storage**: 2GB available space for application
- **Network**: Stable internet connection for API communication
- **Graphics**: DirectX 11 compatible GPU

Recommended Requirements:

- **CPU**: Intel Core i7 or AMD Ryzen 7
- **RAM**: 16GB+ for optimal performance
- **Graphics**: Dedicated GPU with 4GB+ VRAM for smooth visualization

Backend Technology Stack

API Gateway and Orchestration

- **Node.js**: Runtime environment for API gateway

- **Express.js**: Web framework for RESTful API development
- **JavaScript/TypeScript**: Primary programming languages
- **Swagger/OpenAPI**: API documentation and testing

AI and Image Processing Services

- **Python 3.9+**: Primary language for AI and image processing
- **TensorFlow/PyTorch**: Deep learning frameworks for CNN models
- **OpenCV**: Computer vision library for image processing
- **NumPy/SciPy**: Scientific computing libraries
- **Pillow (PIL)**: Image manipulation library
- **Scikit-learn**: Machine learning utilities

Racing Line Optimization

- **Python**: For Particle Swarm Optimization (PSO) algorithms
- **C#**: Integration components with Unity client
- **Physics Simulation Libraries**: Custom physics models for motorcycle dynamics

Development and Deployment Technologies

Version Control and CI/CD

- **GitHub**: Repository hosting and collaboration platform
- **Docker**: Containerization for microservices

Development Tools

- **Visual Studio Code**: Primary IDE for development
- **Postman**: API testing and documentation
- **Unity Editor**: Game engine development environment

Containerization and Orchestration

- **Docker**: Service containerization
- **Dockerfile**: Individual service container definitions
- **docker-compose.yml**: Multi-service orchestration
- **Linux**: Container runtime environment

External Integrations

Game Integration (Wow Factor)

- **MotoGP 18**: Target racing simulation game
- **Steam Platform**: Game distribution platform
- **Sim Racing Telemetry (SRT)**: Telemetry data extraction tool
- **CSV Processing**: Data import/export capabilities

Component	*Primary Technology*	*Alternative Options*	*Compatibility Notes*
Client UI	Unity 6.0 + C#	Unity 2022.3+	Requires Windows 11
API Gateway	Node.js + Express	.NET Core, Spring Boot	Cross-platform
AI Processing	Python + TensorFlow	PyTorch, scikit-learn	GPU acceleration preferred
Database	MongoDB	PostgreSQL, MySQL	NoSQL preferred for flexibility
Containerization	Docker	Kubernetes, OpenShift	Docker Compose for development
Image Processing	OpenCV + Python	ImageJ, MATLAB	Python integration essential

Domain Model