StarCraft II UED

Daniel, Richard, Hyunsung, Youngwoo

Introduction

Real-time strategy (RTS) games like StarCraft present significant challenges for artificial intelligence (AI) due to their intricate blend of strategic planning, resource management, and dynamic combat scenarios. Developing an AI bot for StarCraft exploration decision-making and autonomous control, pushing the boundaries of AI applicable to complex, real-time environments. This project focuses on creating a competitive StarCraft bot that leverages the strengths of the Terran race, known for its early-game solid potential, versatile ranged units, and formidable defensive capabilities. Building upon the well-established Terran 1-1-1 build, our bot utilizes multiple types of units to perform a coordinated attack around the 6-minute mark. By utilizing precise micro-control and effective unit deployment, the bot seeks to overwhelm opponents swiftly. Additionally, the strategy adapts dynamically against adversarial races, ensuring flexibility and resilience in various gameplay scenarios.

Related Work

During the initial planning phase of our project, we explored existing strategies within the StarCraft II community for our approach. A significant influence on our strategy is the Terran 1-1-1 build, which is well-known and widely used in the competitive SC2 ladder. As detailed on Liquipedia, the 1-1-1 strategy is a foundational build order for many Terran players due to its high flexibility. This build integrates various unit types, including infantry, mechanical, and air units, enabling players to adapt effectively to different opponents and game situations. While the 1-1-1 strategy is known for its versatility, one of its weaknesses is the relatively minor number of combat units.

We address this problem by enhancing individual unit control, allowing our bot to efficiently manage Marines, Siege Tanks, and Battlecruisers to counter larger enemy forces. Furthermore, community-driven studies on optimal build orders and strategic flexibility in real-time strategy games influenced our exploration of unit control mechanics and resource management techniques. By integrating these things, our strategy aligns with successful prior methods and contributes new tactics to improve overall gameplay performance.

System Overview

Our StarCraft II bot relies on a modular system architecture that efficiently manages the game's various demands. Key components such as Economy Management, Building Construction, Unit Control, Offense, and Defense work harmoniously to optimize performance and adaptability within the dynamic game environment. Central to our system is the BasicBot file, which orchestrates the interaction between these modules by initializing the bot, managing the primary workflow, and facilitating seamless communication. This hierarchical organization ensures that each module operates independently while contributing cohesively to our overarching coordinated attack strategy. By maintaining a clear separation of concerns, our design allows for flexible and efficient strategic decision-making, from resource management to unit control, ultimately enhancing our bot's competitive edge and ability to execute swift and practical strategies on the battlefield.

Effective resource management is essential for sustaining unit production and maintaining strategic flexibility. Our Economy module is responsible for optimizing the allocation of resources, managing SCV tasks, and ensuring a steady flow of minerals and gas. Initially, the bot assigns an appropriate number of SCVs to gather minerals and gas, dynamically adjusting based on the game's progression and resource availability. To address early inefficiencies, we implemented checks to prevent excessive SCV allocation to mineral gathering and ensure balanced gas collection. Additionally, the module oversees population capacity and technology tree management, enabling timely upgrades and expansions that improve our offensive and defensive capabilities.

The Building module orchestrates the construction of essential structures, following a predefined build order that aligns with our coordinated attack strategy. This sequence begins with the construction of Barracks, followed by a Factory with a Tech Lab, and culminates in Starport and Fusion Core. By implementing a complete and optimized build order, we ensure that critical infrastructures are established promptly, facilitating the rapid production of Battlecruisers. Advanced functionalities, such as the ability to swap the location of Starport with Factory for Tech Lab acquisition, enhance our strategic flexibility. Additionally, the module incorporates algorithms to calculate optimal building placements, preventing SCVs from obstructing mineral paths and ensuring efficient resource gathering.

Effective defence is crucial for our strategy, which utilizes advanced calculations and strategic planning to protect key areas while efficiently managing diverse units and structures. By accurately identifying and securing choke points and narrow passages that control enemy movement, we restrict opponents' offensive maneuvers and create opportunities for counter-attacks through optimal defensive positioning. The bot manages different units and buildings' varying sizes and spatial requirements, ensuring that structures such as Supply Depots and Missile Turrets are strategically placed to maximize protection without hindering resource gathering or unit production. We also employ a layered defence by clustering defensive buildings, enhancing their effectiveness against various enemy strategies. Dynamic adjustments based on the evolving game state allow UED to raise or lower defences as threats emerge or recede, maintaining both economic efficiency and defensive robustness. This combination of precise map analysis, strategic structure placement, and adaptive defence mechanisms enables us to sustain a strong defensive stance while supporting coordinated attacks.

A significant aspect of our strategy involves advanced unit control, integrating sophisticated targeting and kiting mechanisms to optimize combat effectiveness. Inspired by diverse targeting techniques, our bot employs three distinct targeting methods: threat-based, score-based, and proximity-based targeting. Threat-based targeting prioritizes enemy units with the greatest immediate danger, enabling swift neutralization of critical threats. Score-based targeting assigns numerical values to potential targets based on factors such as armour type, enemy density, one-shot potential, and distance, allowing units like Siege Tanks to prioritize high-value targets. Proximity-based targeting focuses on engaging the nearest enemies, which is

particularly effective for units that benefit from maintaining optimal engagement ranges. This approach ensures that each unit type effectively engages enemies to their unique attack patterns. Complementing this targeting system, our bot employs kiting strategies inspired by skilled human players, which involve strategically maneuvering units to retreat from or advance toward enemy forces. Our kiting algorithm follows a three-step process: attack whenever possible, move away when enemies get too close, and advance when enemies retreat or reposition. This adjustment maintains advantageous positions during combat, enabling our units to avoid unfavourable engagements or exploit openings in the enemy's defenses.

Our offensive strategy allows it to locate and attack the enemy base quickly. It starts by sending one SCV to scout all possible enemy locations until it finds the Townhall. Once the enemy base is identified, the bot builds Barracks, Factory, and Starport to produce Marines, Siege Tanks, and Battlecruisers. These units gather at the base and launch a coordinated attack around the six-minute mark. By attacking with air and ground forces simultaneously, the bot puts much pressure on the enemy. During extended battles, damaged units retreat to get repaired while new reinforcements are continuously deployed, keeping the bot's forces strong. This strategy takes advantage of early ground and aerial attacks, making the bot flexible and effective through careful unit deployment and adaptable tactics.

Evaluation

To determine the performance of our bot, we created a series of trials against AI opponents of the Terran, Zerg, and Protoss races on various maps. Each race was tested on three unique maps: Proxima Station LE, Bel'Shir Vestige LE, and Cactus

Valley LE, with AI opponents set to the VeryHard difficulty level. One hundred matches were played, allowing for a thorough evaluation of the bot's performance across various settings and hostile methods.

Win Rate	Terran	Zerg	Protoss	Total
Proxima Station LE	100%	100%	81.82%	93.94 (31/33)
Bel'Shir Vestige LE	95.45%	100%	100%	100%
Cactus Valley LE	100%	100%	91.67%)	97.06% (33/34)
Total	96.97% (32/33)	100% (33/33)	91.18% (31/34)	96.00% (96/100)

As illustrated in our table, our bot achieved a win rate of 96.00%, securing victories in 96 out of 100 matches. Breaking down the results by race, the bot maintained a perfect win rate of 100.00% against Zerg opponents, demonstrating exceptional proficiency in countering Zerg strategies. Against Terran and Protoss opponents, the win rates were 96.97% and 91.18%, indicating strong performance across all races, with slightly more challenges encountered against Protoss. When analyzing performance by map, the bot consistently performed well across all three environments. On Proxima Station LE, the bot won 93.94% of the games, while on Bel'Shir Vestige LE and Cactus Valley LE, the win rates were 96.97% and 97.06%. These high win rates across different maps demonstrate the bot's adaptability and effectiveness in executing the coordinated attack in varied strategic landscapes.

Citations

- https://liquipedia.net/starcraft2/111_Expand