Team Laser Combat

A player-friendly approach to turn-based tactical strategy by

Michael Henson

Project Proposal

Introduction

From the point of view of some video game enthusiasts, commercial strategy games are often plagued by two key inhibitors:

1. Playing through a single level or scenario often requires a significant time commitment.

2. There may be a fair amount of reading involved due to the massive reference manuals detailing the game's multitude of elements.

For hardcore gamers who regularly commit several hours a day to the habit, this may not be a problem at all. However, the popularity of social games on the Internet and casual party games for the Nintendo Wii suggests that there may be an emerging market segment consisting of players who crave a more in-depth gaming experience but lack the time or the inclination to research available products and subject themselves to the steep learning curves that often accompany commercial strategy games.

My proposition is to develop a new type of strategy game that plays more like a board game with a lighthearted laser tag theme. Aside from rolling dice, moving characters, and firing at opponents, players' decisions will be limited to a few character enhancements, such as attack power, defense strength, movement ability, hit point upgrades, and random chance improvements. The interface will provide brief descriptions of each of these options so novice players can make meaningful strategic decisions. For a full explanation of all of these options and much more, refer to the game design document (File TLCDesignDoc.docx. URL pending.).

Development of the project is already underway, but the gameplay elements have barely been touched, and thus provide a good candidate for the focus of my capstone.

Current Baseline Features

The development of the game so far has followed the design document very closely, with the GDD being occasionally updated to reflect changes in the accepted requirements. Currently, the following critical features have already been implemented, though at various degrees of completion.

- Game mode selection

- User profile creation
- Team creation
- Character selection and customization
- Client/server connection and communication
- Game state initialization thread
- Character movement and pathfinding
- Team inventory UI and backend support

Some of the most critical elements yet to be developed include the following:

- The turn-based combat system, including the UI, combat factor evaluation, artificial intelligence, audiovisual indicators, and the combat results summary screen.
- Environmental interactions, such as the random bonus spaces on the board, random hazards, and destructible containers (which depends on the combat system).
- AI goals other than the combat system mentioned above, such as the desire to keep bonus items out of opponents' reach, to select the best opponent to "pick on", and to strategically avoid danger if conditions are unfavorable for a confrontation.
- Advanced options such as manual map selection, Alliance mode, map cycling, customization of victory conditions, and persistence of game state and user profile data.
- Update/rewrite of map editor utility, which only works reliably on Windows XP. It freezes up randomly on Windows Vista and corrupts its own data on Windows 7.

Milestones

For the purposes of a capstone project, I feel that the highest priority is the heart of the game play. Therefore, my three milestones are all directed at producing a playable version of the game as specified by the rules set forth in the game design document.

Milestone 1: Combat Initiation

The largest, most complex, and most important system will come first: The combat system. It's a turn-based, multi-phase system that involves one player choosing to attack another, and the selecting which resource (combat token) to spend on the attack. Then the "victim" of the attack gets a chance to defend himself based on the defensive tokens in his inventory. Finally, the results of the interaction are resolved by the game server and the updated values are broadcast. Milestone 1 will focus on the initiation of a combat interaction.

At the completion of Milestone 1, the following functional requirements will be verifiable:

- The system shall perform a pre-validation of actions available to the player when it is that player's turn.
- The system shall construct a UI (the Actions Menu) which allows the selection of an available action based on validation results.
- If the Attack action is available and is selected from the Actions Menu, the system shall display the directions in which an attack is possible and allow selection of one of these directions.
- The system shall provide an Auto Attack option that players may select if they would rather have the game select their tokens for them automatically after issuing the Attack command.
- After the player has selected a direction for attack, the system shall display the offensive combat tokens from the player's inventory and allow the selection of a token to spend IFF the Auto Attack option is turned off.
- AI players shall be able to issue Attack commands that are subject to the same validation rules as the human players.

Milestone 2: Combat Resolution

The second milestone picks up where the first leaves off and completes the player-to-player combat system.

At the completion of Milestone 2, the following functional requirements will be verifiable:

- When a unit is attacked, the system shall notify the player controlling the unit by presenting a Defend UI which allows selection of a defensive combat token from the player's inventory.

- The game server shall resolve combat interactions by evaluating the stats of the characters and their selected combat tokens and updating the data model accordingly.
- The game server's combat evaluator shall handle all combat token types: the basic combat tokens, the four types of offensive Power Tokens, and the four types of defensive Power Tokens as detailed in the GDD.
- The game server shall broadcast a summary of combat results to all connected players.
- The system shall provide a Combat Results toggle that allows players to opt out of seeing the detailed combat summaries at the conclusion of each combat interaction.
- The user interface shall display a combat summary screen when combat results are received from the game server IFF the Combat Results option is turned on.
- The system shall provide an Auto Defense option that players may activate if they would rather have the game select their tokens for them automatically when their units are attacked.
- The system shall display an event log on the HUD which records the actions taken by all units in the game.
- AI players shall be able to defend against attacks subject to the same rules as the human players.

Milestone 3: The Game World Environment

The completion of the combat system paves the way for the implementation of the various interactive elements of the game world. The third milestone will include the functionality for all of the random bonus spaces and hazards as specified in the GDD, the addition of destructible containers which are operated via the combat system's Attack action.

At the completion of Milestone 3, the following functional requirements will be verifiable:

- The system shall place six bonus items in random, unobstructed locations on the game board at the start of each match: a coin, a token, a snack, a weapon upgrade, an armor upgrade, and a boot upgrade.
- When a unit travels onto a space occupied by a bonus item, the system shall apply the effect of the item to the unit or the unit's team as detailed in the GDD.

- The system shall remove a bonus item from the board after its effect has been applied.
- The system shall place one invisible hazard in a random, unobstructed location on the game board at the start of each match.
- When a unit travels into the space occupied by the random hazard, the system shall deduct one hit point from the unit.
- When a unit is damaged by a random hazard, the user interface shall display a notification to all players indicating which unit was damaged but not revealing which space contains the hazard.
- While loading the data from a map file, the system shall recognize identifiers for destructible containers and instantiate them onto the board accordingly.
- The attack validator (to be updated from Milestone 1) shall recognize destructible containers as valid targets to be attacked.
- When a container is attacked, it is removed from the board and replaced by a random bonus item: a coin, a token, a snack, a weapon upgrade, an armor upgrade, or a boot upgrade.