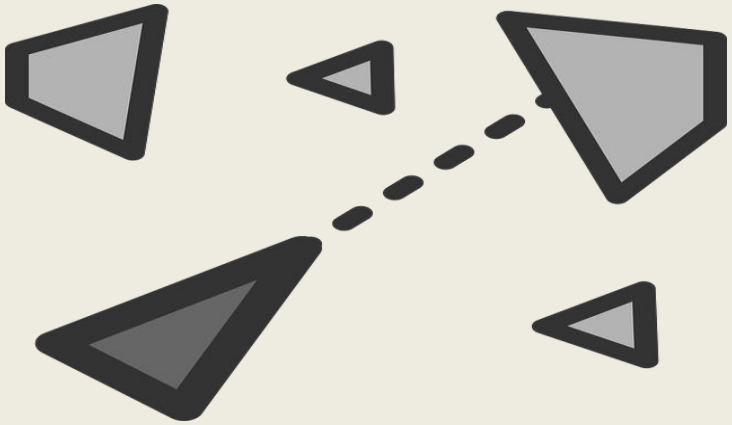
Asteroids

Software Requirements Specification



Team 4 - Astronoids

Damien Girard
Siddhart Ghandi
Thuy-Anh Le
Genevieve Nantel
Vincent Petrella
Chi-Wing Sit

2/10/2013

1. Behaviours that we did not have time to implement

All the basic features required for the project were completed. However, there are certain behaviours that would need to be enhanced. For example, when the alien shoots the player, the player automatically dies. Instead of automatically dying, the player should lose a life. Moreover, all the asteroids are worth the same amount of points. If we had more time we would assign points differently, i.e., smaller asteroids would be worth more than big asteroids. Below you will find the list of requirements that were stated in the system requirement document. Beside each requirement a red "Yes" or "No" is assigned to show which requirements have been met and which have not.

Requirements	Percentage Completion (%)
User interface that allows you to manage Game State.	100 %
Graphical Implementation of the game (multi-sized asteroids, aliens, bonus drops, weapon drops, spaceship animations and spaceships)	100%
Game Control to allow control of the ship	100%
Implementation of shooting, firing rate control and impact detection	100%
Implementation of game physics for both asteroids and spaceship	100%
Implementation of bonus life drops	100%
Implementation of sound system	100%
Simple AI implementation for Alien ships	100%
The high score list with statistics	100%
2 player mode, alternating turns	100%
Different levels of difficulty	100%

Detailed System Requirements

3.1.1 INITIALIZATION

3.1.1.1 GUI

3.1.1.1.1 [Essential/Easy] System available with a GUI, developed using java swing components. **(Yes, remarks: GUI Developed with JOGL)**

3.1.1.2 DATABASE

3.1.1.2.1 [Essential/Medium] Creating and initializing a database to store information upon first execution **(Yes)**

3.1.1.2.2 [Desirable/Medium] Loading and saving your current game **(No)**

3.1.1.2.3 [Essential/Easy] Only one game per username can be save **(No)**

3.1.1.2.4 [Essential/Medium] Opening and loading information from database when user wants to view statistics and high scores **(Yes)**

3.1.1.2.5 [Essential/Medium] Saving information in database when user ends or saves game **(Yes)**

3.1.2 USER INTERFACE

3.1.2.1 IDENTIFICATION WINDOW

3.1.2.1.1 [Essential/Easy] User must be identified by entering user name before he can see game panel **(Yes)**

3.1.2.2 HELP WINDOW

3.1.2.2.1 [Essential/Easy] The help window shall provide a way of viewing documentation about the game. **(Yes)**

3.1.2.3 MAIN WINDOW

3.1.2.3.1 [Essential/Easy] After user logged in successfully, the game panel will be replaced by the user information: You are logged in as: username. **(Yes)**

3.1.2.3.2 [Essential/Easy] User game Interface allows you to manage the game state: seeing your statistics and high scores, new game, game rules, player mode and change settings. **(Yes)**

3.1.2.3.3 [Essential/Easy] When user clicks on new game, the game option panel will disappear and be replaced by the game itself. **(Yes)**

3.1.2.4 CHANGE SETTINGS WINDOW

3.1.2.4.1 [Essential/Easy] Two player mode, alternating turns **(Yes)**

3.1.2.4.2 [Desirable/Medium] Two player team up with simultaneous play **(No)**

3.1.2.4.3 [Essential/Easy] Different levels of difficulty **(Yes)**

3.1.2.5 STATISTICS WINDOW

3.1.2.5.1 [Essential/Easy] User will see he high score list with statistics **(Yes)**

3.1.2.6 GAME WINDOW

2.2.6.1 [Essential/Medium] The graphical implementation of the game will include multi-sized asteroids, aliens, bonus drops, weapon drops, spaceship animations and environment wraparounds. **(Yes)**

3.1.3 FUNCTIONALITIES

3.1.3.1 ARTIFICIAL INTELLIGENCE

3.1.3.1.1 [Essential/Medium] A HOSTILE AI implementation for alien ships **(Yes)**

3.1.3.2 SCORING

3.1.3.2.1 [Essential/Easy] Implementation of point system **(Yes)**

3.1.3.2.1.1 - Large Asteroid: 50 points **(No, remark all asteroids: 10 points)**

- Small Asteroid: 100 points

- Large Saucer: 200 points

- Small Saucer: 1000 points

3.1.3.2.2 [Essential/Easy] Implementation of life system **(Yes)**

3.1.3.2.2.1 [Essential/Easy] Set starting lives at 3 **(Yes)**

3.1.3.2.2.2 [Essential/Easy] Implementation of bonus life drops; every time the player shoots an asteroid or a flying saucer, there is a certain percentage change that it will be a bonus life or a weapon **(Yes)**

3.1.3.3.2.3 [Essential/Easy] Collision with an asteroid results in loss of life **(Yes)**

3.1.3.2.2.4 [Essential/Easy] Game is over when player has lost all his lives **(Yes)**

3.1.3.3 SOUND SYSTEM

3.1.3.3.1 [Essential/Medium] Implementation of the sound system for the feature of various sound effects. **(Yes)**

3.1.3.4 WEAPONS

3.1.3.4.1 [Essential/Medium] Implementation of shooting, firing rate control and impact detection. **(Yes)**

3.1.3.4.2 [Desirable/Easy] Implementation of different types of weapons: regular, spread, screen clear **(No)**

3.1.3.5 SPACESHIP

3.1.3.5.1 [Essential/Medium] Player controls a triangular-shaped ship that can rotate left and right, fire shots straight forward, and thrust forward. **(Yes)**

3.1.3.4 GAME CONTROLS

3.1.3.4.1 Game controls to allow the control of the ship

3.1.3.4.1.1 [Essential]/ [Medium] Implementation of game physics for spaceship **(Yes)**

3.1.3.4.1.2 [Essential]/ [Medium] Player will be able to change the direction of the navigated space shuttle. **(Yes)**

3.1.3.4.1.3 [Essential]/ [Medium] Player will be able to change the speed of the navigated space shuttle. **(Yes)**

3.1.3.4.1.4 [Essential]/ [Medium] Momentum is not conserved, the ship eventually comes to a stop again when not thrusting **(Yes)**

3.1.3.4.2 Basic Game Control

3.1.3.4.2.1 [Essential]/ [Easy] Stop Game **(Yes)**

3.1.3.4.2.2 [Essential]/ [Medium] Pause Game **(Yes)**

3.1.3.5 ASTEROIDS AND SAUCERS

3.1.3.5.1 [Essential/Medium] Implementation of game physics for asteroids and saucers **(Yes)**

3.1.3.5.1.1 [Essential/Medium] Each kind of asteroids/saucers should be able to change their speed. They have different minimum and maximum speed. **(Yes)**

3.1.3.5.1.2 [Essential/Medium] Saucers will appear periodically on one side of the screen to the other before disappearing. **(Yes)**

3.1.3.5.2 Saucers

3.1.3.5.2.1[Essential/Medium] Implementation of saucers: Large saucers fire in random directions, while small saucers aim at the player's ship **(Yes)**

3.1.3.5.2.2[Desirable/Medium] Implementation of different types of flying saucers: regular, big boss. **(No)**

3.1.3.5.3 Asteroids

3.1.3.5.3.1 [Desirable/Medium] Implementation of two different kinds of asteroids: One is small and fast and does not change its direction until it reaches the boundary of the game board. The other is big and slow, and it may change its direction any time. **(Yes)**

3.1.4 DIFFICULTY

3.1.4.1[Essential/Medium] Every time the screen has been cleared of all asteroids and flying saucers, a new set of large asteroids appears. The number of asteroids increases each round up to a maximum of 12. **(Yes, Remark: 5 stages)**

3.2 QUALITY REQUIREMENTS

3.2.1 PERFORMANCE

3.2.1.1 [Essential/Easy] Minimum delay between user input and game reaction, and more generally a minimum delay between two game frames. The desired frame rate is a constant 60 fps on current mid-range hardware, which is equivalent to a 16ms delay between two game refresh. **(Yes)**

3.2.1.2 [Desirable/Easy] Query and reporting times (initial loads and subsequent loads) should load in less than 20 seconds on current mid-range hardware. **(Yes)**

3.2.1.3 [Essential/Medium] Calculations to keep the score of the player up to date should be executed instantaneously (i.e, during the frame of the change) **(Yes)**

3.2.1.4 [Essential/Easy] A high score table will appear in less than 5 seconds after “game over” (processing time) on current mid-range hardware. **(No)**

3.2.2 USABILITY

3.2.2.1 [Essential/Easy] The language used for the game is English **(Yes)**

3.2.2.2 [Desirable/Easy] Shortcuts on the keyboard should be implemented to facilitate the user’s game. **(Yes)**

3.2.2.3 [Essential/Easy] Game interface designed in a way to appeal to users of all ages and you can learn how to play while playing the game, i.e. Asteroids must be simple to play: The user interface and the rules of playing Asteroids must be intuitive. **(Yes)**

3.2.3 RELIABILITY AND AVAILABILITY

3.2.3.1 [Essential/Medium] In case the system crashes, all high scores will be maintained in the database. **(Yes)**

3.2.3.2 [Essential/Difficult] All predictable exceptions should be handle by the program during execution, and should not produce a crash. If an exception occurs forcing the program to terminate, the current state of the session should be saved before termination. **(Yes)**

3.2.4 SAFETY

3.2.4.1 [Essential/Easy] If the user clicks outside of the system's window, the game is paused. **(No)**

3.2.5 Maintainability

3.2.6.1 [Essential] The code should be fully documented and modular in order to permit future modifications. **(Yes)**

3.2.6 EXTENSIBILITY

3.2.8.1 [Desirable/Medium] The game should be extensible and allow easy changes or improvements in the future. **(Yes)**

3.3 DESIGN CONSTRAINTS

3.3.1 CONSTRAINTS ON THE APPLICATION OR ITS DEVELOPMENT

3.3.1.1 Small size of the deployed product **(Yes)**

3.3.1.2 Less than 100mb of ram during runtime **(Yes)**

3.3.1.3 Software must be independent of the Java platform. All the code must be implemented in Java so that it can run on any system with JVM installed. **(Yes)**

3.3.1.4 Hardware environment must have support for Java version 7 or higher. **(Yes)**

3.3.1.5 Game lag must be avoided at all cost as it takes away from user experience and breaks immersion. **(Yes)**

3.3.2 GAME AESTHETIC

3.3.2.1 Game graphics, including sprites, animations, backgrounds, etc. must be visually pleasing, professional and conform to the original game's concept. **(Yes)**

3.3.2.2 The final system should not include any graphical glitches. **(Yes)**

3.3.3 USER INTERFACES

3.3.3.1 The interface of the system should be user-friendly for all ages. Users of all level of computer experience should be able to easily engage in the game. **(Yes)**

3.3.3.2 The elements involving game states (save/pause, etc.) not directly involving game play should be intuitive as well as visually appealing to the user while retaining a professional look. **(Yes)**

3.3.3.3 The interface must accurately inform the user in case of input errors. **(No)**

3.3.4 ERROR HANDLING

3.3.4.1 All exception raised during runtime must be handled so that the system meets a minimal standard of stability. The game should never crash but a realistic best case scenario should make a game crash a rare occurrence. **(Yes)**

3.3.4.2 In extreme or high stress conditions, the system must retain stability. The conditions should be determined in testing. **(Yes)**

3.3.4.3 Errors raised by the program must be minimized. **(Yes)**

2. Future Improvements

If more time was allocated to this project, the following improvements would have been made:

- When player is shot by alien, results in only one life lost for the player
- Implementation of a more elaborated point system
- Loading and Saving a game
- Two player mode: simultaneous play
- Different types of weapons
- More types of aliens with their corresponding AI
- Special sound effects
- Different types of weapons (regular, spread, screen clear)

Also, we would have spent more time on testing. A lot of time was spent for the implementation and few time remained for the testing. We should have spent more time testing in order to be sure that no unexpected bugs are present.