

2 Dimensional, Multiplatform Space Invaders Using XNA Game Studio

By Stephen McQueen, Supervised by Dr F.C. Langbein and Moderated by Prof R.R.Martin

Project Description

My intended project is to re-create the classic game of Space Invaders that runs on modern platforms. The platforms in question are the Xbox 360 Console, a Windows OS PC and a Windows 7 Mobile Phone.

The game itself will be in 2 Dimensions and will include all of the original gameplay elements found in the original game. One of the challenges I will be faced with is adopting the traditional joystick and two button control system as found with typical arcade units with more modern hardware such as an Xbox 360 Rumble Controller, Keyboard and touch sensitive screens. Finally the main goal of this project is to add more variety and content to the original game and to break away from what would now be considered linear game play. I intend to create a more unpredictable game which should make it more challenging and fun for the player.

Project Aims and Objectives

- Space Invaders – Clone
 - The basic (backbone) aim which shall provide the overall structure in which to develop the project is to create the basic game of space invaders. The objective here is to produce a “clone” of the original game so that it contains all of the basic gameplay elements which can run in a modern hardware environment (i.e. modern games consoles such as the Xbox 360). The version of the game I intend to clone is the original arcade version released in 1978. Once this aim is achieved the game can be expanded upon.
- User Interface
 - Getting the user interface right will be crucial to the overall success of this project. This aspect is most important for the mobile phone version of the project as age range and ethnicity of potential players will be diverse. The game must rely heavily on aesthetics and the user interface to draw players in to download the game and play it. Therefore looking at the project from a Computer Science point of view the main focus of this objective will be the design and implementation of an engaging and self-intuitive user interface.
- Collision Detection
 - This element of the project is key as it will control interactions between objects in 2 Dimensional space (e.g. allow lasers to blow up the bad guys). The challenging aspect of this element will be achieving maximum efficiency in collision detection loops as the lasers will travel up and down the screen pixel by pixel (whether it is fired by the player or the enemy). This is a top priority as if a laser does hit an enemy, the enemy explodes (what looks like) immediately rather than continues unawares until the relevant code is finally executed after several long loops (i.e. an

infinite loop will be required to check for collisions and another loop to update the progress of the lasers on screen).

- Game Artificial Intelligence – Optional
 - One of the major gameplay deficits in the original space invaders game was the lack of game progression and Game Artificial Intelligence (AI), this mainly being due to hardware constraints and AI being an early concept recently introduced to the gaming industry. The reason the term “Game AI” is used is because often the “AI” code used actually contains very little AI and the industry is notorious for using tricks to mimic intelligent behaviour. Due to basic hardware available to society now being vastly more powerful than the original arcade platform and due to modern games investing millions of dollars into incorporating some of the most advanced Artificial Intelligence into their game engines I shall also attempt to add a modern flair to the very basic enemies found in the original game. One of the ideas I have in this regard is for the enemies to work together (collaborative behaviour) in killing the player. One of the ways to do this will be for the enemies to shoot one side of the player forcing the player to manoeuvre out of the way, however another enemy can judge the players reactions and fire a laser to intercept the player before he/she can react. Another interesting idea will be for the enemies to break away from the main pack and attempt to physically collide with the player. This will involve the use of path finding and again collaborative behaviour can be used here to allow for multiple enemies flying towards the player.
- Networking (Multiplayer) - Optional
 - This aim is an advanced feature that will be interesting concept to consider even if it does not get implemented as this will be reliant on the amount of time left towards the end of the implementation phase. The possibility of turning a simple game involving a player and AI into a two player strategy game sounds very inviting. The theory would involve allowing two players both using Windows 7 Phones playing the same game over a WiFi network (other networking possibilities such as Bluetooth will require further study as well as ethical discussions to ensure privacy). The structure of the game itself along with its objectives from the player controlling the ship would remain unchanged. However instead of Game AI producing instructions for the enemies, the second player would have the ability to select an instruction from a set list for the enemies to respond accordingly. This would provide a much more challenging and engaging game experience for both players as they attempt to out manoeuvre their intelligent opponent rather than predictable Game AI code.

Initial Project Requirements (Subject to change throughout project lifecycle)

The main objective of the project is to create a fully functional and multiplatform space invaders game. The aims of the game itself must:

- Have a core game engine
 - This will be based on the original space invaders game as mentioned in the aims and objectives section.
- Be a single player game
 - The player must:

- be able to move horizontally but not vertically
 - be able to shoot at the enemies on screen
 - have a fixed number of lives at the start of the game
- Contain enemies
 - The enemies must:
 - Have different scores associated with them. i.e. enemies at the back of the pack are worth more as they are harder to kill
 - Move together from one side to the screen to the other. Once they reach the side of the screen they will all move down the screen one increment.
 - Have convincing artificial intelligence and fire at any given time
 - Get progressively quicker as the round level increases
- Have a mystery enemy
 - This enemy must be quicker than the others and will appear at the top of the screen (thus being protected from the others for a short period of time)
 - Must be worth a random number of points (which are to be assigned before the enemy is created)
 - Can either travel from left to right or vice versa. This also is to be decided before the enemy is created.
- Have barriers
 - Will be made up of several individual barrier blocks
 - Must be destructible via laser fire (from the player or enemy)
 - Will provide the player with shelter from laser fire. I.e. each barrier block can absorb 3 lasers before being totally destroyed.
- Handle collisions
 - Whether it involves a laser and the enemy or the enemy lasers and the player, the game must acknowledge when a collision occurs and act accordingly
- Be efficient
 - This mainly applies to the mobile version of the game due to hardware limitations.
- Multiplatform compatible
 - The game must be fully compatible with the Xbox 360, Windows OS PC and the Windows 7 Phone.
 - The must also be able to handle the different control devices for the game and act accordingly.
- Be in 2 Dimensions
 - This means that the graphics used and the objects involved will only have two dimensions; namely width and height.
- Tombstone effectively for Windows 7 Phone
 - This means that the game will pause but keep running in the background if the user has to take a phone call and will allow the user to return to the game as normal afterwards.
- Include animations
 - Ranging from explosions to fire coming out of players jet engines.

- Include Game Artificial Intelligence
 - Allow for collaborative behaviour between enemies to eliminate the player.
 - Allow for enemies to break away from the main group and fly at the player.
- High Scores
 - Highest scores to be stored on memory whether it be a memory card or hard disk so the user can continually aim to improve their performance.

Optional features:

- Power-ups
 - Once destroyed, certain enemies can drop a random power up that increases the players advantage of survival by some degree. Alternatively the player could purchase upgrades using a point/money system to increase survivability.
- Enemy bosses
 - One significantly difficult enemy that will use a set routine of actions and movements to defeat the player.
- Multiplayer options
 - Players strategically attempt to out manoeuvre one another in a cunning game of strategy.

Project Milestones

For this project I intend to adhere to the waterfall model but add an iterative twist between phases to ensure constant reviewing of requirements and design before applying them to the software. Therefore with this in mind I have set down project milestones which will be used to determine the project progress:

- Requirements/analysis
 - Project analysis
 - Background research
 - Existing versions of game
 - Similar games
 - What IDE/programming language should be used?
 - Requirements
 - What elements of the game will be included
 - What must the game achieve?
- Design
 - Class/objects design using CRC cards
 - Gameplay design
 - i.e. What must the player do in the game etc.
 - UI and home screen designs
 - Player/enemy designs and animations
- Implementation (For specific implementation milestones see project timetable)
 - Xbox 360 Console
 - Windows OS PC
 - Windows 7 Phone

- Testing
 - Alpha phase
 - Testing done by myself
 - Closed Beta phase
 - Testing done primarily by myself and close friends to gain valuable feedback
 - Open Beta phase
 - Game open for public testing and feedback
 - Fixing bugs and patching game
 - Based on tester feedback
- Evaluation
 - In depth discussion on final product emphasising on the strengths and weaknesses. Self-Criticism and feedback provided by external parties e.g. Project Supervisor is to be analysed in depth. The second part of the evaluation will involve writing a concise reflection on how the project went, discuss what went well and what did not. Finally to make note of any potential further developments to the project and if done again what could have been done differently to achieve a better outcome.
- Conclusion
 - This will be a summary of the main points carried out in the project and discuss briefly what was achieved from it.

Interim Report Deliverables

As per the project guidelines the interim report is to account for 25% of the overall mark and therefore the work done should reflect this. Therefore I have decided that the project deliverables for the interim report will consist of:

- Outcomes of research
 - The interim report will contain large amounts of background research into various parts of the game, namely;
 - Space Invaders
 - Visual Studio and the XNA Game Studio framework
 - Game AI
 - Hardware Platforms (Xbox 360 and Windows 7 Phone)
- Project analysis and approach
 - Breakdown the project and discuss feasible milestones for project progression.
- Game Design
 - Undertake the design stage of the project and provide:
 - Detailed classes and objects
 - User Interface
 - Gameplay design

Final Report Deliverables

As the final report accounts for the remaining 75% of the mark I will focus on providing

- Core game engine
 - At this stage I intend to have completed the main core engine of the game ready for expansion for the more challenging aspects of the project (i.e. Game AI)
 - Produce a project diary discussing the challenges and advancements made during the core engine development phase.
 - This will also allow time for modifying the game engine to be more efficient.
- Detailed analysis of implementation
 - Including a project diary to reflect how I went about tackling the Game AI and what challenges I faced along the way.
- Testing
 - I will also provide a thorough testing section for my final report emphasising on the importance of continuously testing a product to ensure a more complete solution.
- Complete Game
 - Describing functionality and comparing completed solution to initial requirements.
- Project findings and achievements
 - Discuss the outcomes of the projects and acknowledge any/all achievements made.
- Evaluation and Conclusion
 - Compare and contrast the finished solution to the initial specification.
 - Reflect on how the project went; what went well and what didn't, if repeated what would have been done differently and what could be expanded upon if there was more time?
 - Provide a summary of the project outlining what can be learned from undertaking it.