School of Computer Science
University of St Andrews
2023-24

CS4303 - Video Games
Physics Practical: Ballista Command

This Practical comprises 20% of the coursework component of CS4303. It is due on Monday 12th February at 21:00 (NB MMS is the definitive source for deadlines and weights). The deliverables comprise 3 elements:

- A report, the contents of which are specified below.
- A Player's Guide for your game.
- The Processing source code for the video game you will write.

Carefully read the practical specification, as it contains many details about how the game should be implemented. The practical will be marked following the standard mark descriptors as given in the Student Handbook (see link below).

Background

This practical is intended to give you the opportunity to learn the Processing language and implement some of the concepts from the Physics component of the module. The task is to implement **in Processing** a *variant* of the classic video game **Missile Command**. If you are not familiar with this type of game, your first task is to read the following summary:

https://technologyuk.net/computing/computer-gaming/gaming-landmarks-1960-1985/missile-command.shtml

There are many sites online where you can play the game. One of them is here:

https://archive.org/details/a8b_Missile_Command_1981_Atari_US_k_file

The basic controls are F1 to start, the keypad cursors to move and the keypad 0 to fire.

Ballista Command - The game

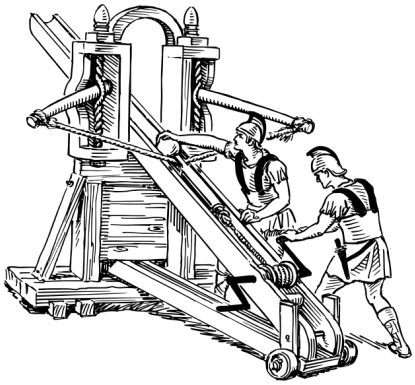


Illustration of a ballista, from https://en.wikipedia.org/wiki/Ballista

Summary

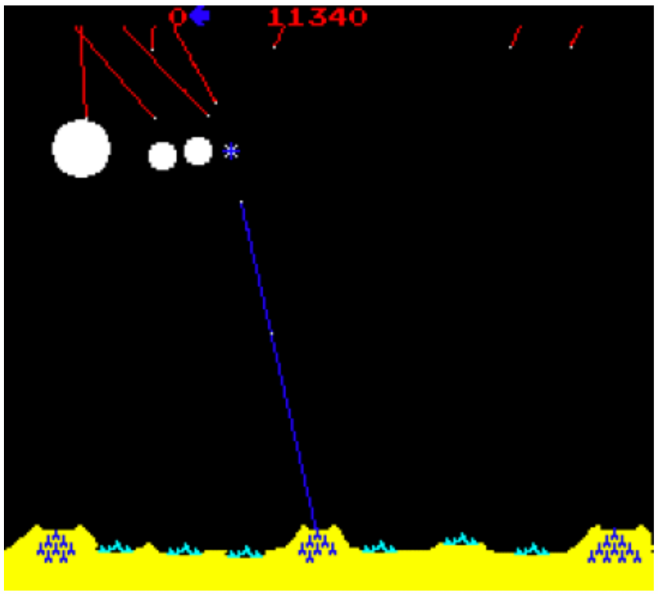
Ballista Command is a single-player variant of Missile Command that you will implement in this practical. It takes place after the events of Missile Command, where you were able to successfully prevail over the barrages of enemy missiles that were endlessly fired at you.

After spending so much in missiles, everyone is out of fuel and therefore no missiles are available. Still, the enemy has been able to invent a device to reroute asteroids at will, and the next attack looms over your head … As the reserve of missiles is exhausted, three giant ballistae are installed to protect the cities once again. Unlike traditional ballistae, they are not equipped with rocks or bolts, but bombs instead! Using these bombs, your goal is to protect the cities from meteorites falling from above.

In this game the player will control three ballistae, which can throw bombs at the incoming meteorites to protect the cities. To make it a bit more interesting, we are going to subject all projectiles to physics. Think of the behaviour of bombs and meteorites as **rock projectiles**: affected by gravity and drag, unlike the missiles of the original game which had a constant thrust.

The Play Area

The play area should be arranged as per the original Missile Command, as shown in the following screenshot:



Missile Command. Atari 1980.

The missile batteries occupy the left, right and centre positions on the ground, with two groups of three cities in between. This general arrangement should remain the same in your implementation of Particle Command. Conceptually, the ballistae will substitute the missile batteries, but the graphical appearance of these elements is left for you to decide *and document*.

Waves

The game proceeds in waves. In each, a set number of meteorites fall towards the cities with the intent of destroying them. The number of meteorites should increase with the wave number so that the level of challenge in the game increases as the player progresses. The number of waves is unlimited.

The game is over when all cities have been destroyed.

Meteorites

The meteorites will appear falling from the sky (the top of the play area). They should have a **random initial velocity**, but you should use the wave number to influence the initial velocity of each meteorite so as to increase difficulty in later waves.

In Missile Command, enemy missiles had a static trajectory and a constant speed. In our case the meteorites **must be affected by both gravity and drag**, as described in lectures. If a meteorite hits a city, the city is destroyed. If a meteorite hits a ballista, the ballista is disabled for the current wave. Finally, if a meteorite is hit by an explosion, it should also explode, as in the original Missile Command.

A careful implementation of the meteorites will be crucial for achieving a balanced and enjoyable game.

Ballistae

Each ballista has a stock of ten bombs per wave. The number remaining in each ballista should be clearly visible to the player. The ballistae are controlled independently by the player, who may choose which of the ballistae fires a bomb.

Aiming is performed via a crosshair as per Missile Command. The control scheme for aiming and firing should be the following: The mouse is used to aim the crosshair, and the vector defined by the position of the ballista and the position of the crosshair should be used to determine the initial instantaneous force when firing a bomb.

In Missile Command, a missile is a particle with a constant thrust force propelling it in the direction of its orientation. In our case it is different, as **bombs start with an instantaneous force at the point of firing, but then gravity and drag affect them until they slow down and fall.** To visualise how it should behave, think about the parabola formed when you throw a rock, or the sling in Angry Birds.

Explosions

In Missile command, when a missile reaches its target position it explodes. In our case it will be different, as bombs will be detonated on request. That is, a key or button should be pressed to make all the bombs in the screen detonate. Bomb explosions should happen in the same order as they were fired.

Any meteorite caught in the blast radius of the explosion should itself explode. The blast radius of these additional explosions may trigger further meteorites to explode, and so on. This should allow you to fire various bombs and chain explosions if timed properly.

Scoring

Basic scoring is as follows:

- The player should be awarded 25 points for each destroyed meteorite, and 100 for hitting the bomber or satellite enemies if they are implemented (see the following section).
- At the conclusion of each wave, 100 bonus points are awarded for each surviving city, and 5 for each unused bomb.

However, scoring is multiplied according to the wave following the same scheme as Missile Command:

- Waves 1–2: 1x

- Waves 3–4: 2x
- Waves 5–6: 3x
- Waves 7–8: 4x
- Waves 9–10: 5x
- Waves 11+: 6x

A city should be restored (if any have been destroyed) for every 10,000 points earned. Cities are restored at the end of the wave in which the required number of points is reached.

The current score should be depicted at the top of the screen, as in Missile Command.

Additional Features

From wave 2 *onwards*, the following additional gameplay elements should appear:

- Some of the falling meteorites should split into two or more separate meteorites as they fall.
- Bomber and satellite enemies from the original Missile Command should appear intermittently and travel the play area horizontally, periodically dropping additional meteorites.

Report and Player's Guide

Your report is an integral part of the practical, so be sure not to neglect it. It should have at least an **Introduction** and a **Conclusion** section. The main body of the report should document the design and implementation of your game *in detail*. In particular, it should contain a detailed account of your implementation of the physics involved. In addition to that, include screenshots that show your game in operation and illustrate its features.

As previously stated, the player's guide must also be included with the submission. The guide should behave as a manual, explaining how the game works and its controls.

Marking

This practical will be marked following the standard mark descriptors as given in the Student Handbook. There follows further guidance as to what is expected:

- To achieve a mark of 7 or higher: A bare bones implementation of the game, consisting of a single wave of falling particles. This implementation should be adequately described in an accompanying report and game guide.

- To achieve a mark of 11 or higher: In addition to the above, the game should proceed in waves as specified, with an increasing level of difficulty. This

implementation should be well described in an accompanying report and game guide.

- To achieve a mark of 14 or higher: In addition to the above, some of the additional features should be implemented, together with a fully correct physics implementation. This implementation should be well described in an accompanying report and game guide.

- To achieve a mark of 17: the full basic specification above must be implemented, including both of the additional features. The report and game guide should be written to a high standard. The game should be well-balanced in terms of difficulty and having a good pace.

- To achieve a mark greater than 17: In addition to the requirements for a mark of 17, evidence of an exceptional achievement in terms of technical challenge, the original "smart bomb" enemies fully implemented. You will have to research how the "smart bombs" worked.

Pointers

Your attention is drawn to the following:

- Mark Descriptors:
  https://info.cs.st-andrews.ac.uk/student-handbook/learning-teaching/feedback.html
- Lateness:
  https://info.cs.st-andrews.ac.uk/student-handbook/learning-teaching/assessment.html
- Good Academic Practice:
  https://info.cs.st-andrews.ac.uk/student-handbook/academic/gap.html