Creating a Mobile game with a custom built physics library

## By Michael Shalaby

## Supervised by Keith Mannock Student ID: mshala02 Msc Computer Science Project Proposal

## 

Abstract:  
This proposal will outline the design requirements of to create a Physics Library for mobile game development using open source graphics libraries. The proposed library will be built using Kotlin as the language of choice in order to specialise the physics library for android app development. In order to assess the viability of the physics library, the library will require a lot more than unit testing to see if it works. Therefore the ideal testing ground for this library would be to use it to simulate the daring dam-busters raid “Operation Chastise” carried out by the British during WW2.

### Disclaimer:

This proposal is substantially the result of my own work, expressed in my own words, except where explicitly indicated in the text. I give my permission for it to be submitted to the JISC Plagiarism Detection Service.

Contents

[By Michael Shalaby 1](#_Toc6262285)

[Supervised by Keith Mannock Student ID: mshala02 Msc Computer Science Project Proposal 1](#_Toc6262286)

[Abstract: 1](#_Toc6262287)

[Disclaimer: 1](#_Toc6262288)

[Chapter 1. Introduction: 4](#_Toc6262289)

[1.1 History of physical simulations in video games. 4](#_Toc6262290)

[1.2: Why simulate Operation Chastise: 4](#_Toc6262291)

[1.2.1: A brief history of operation chastise 4](#_Toc6262292)

[1.2.2: Why create a physics library for Operation Chastise 5](#_Toc6262293)

[1.3: Why build a new physics library for mobile apps 5](#_Toc6262296)

[1.4 The physics Library Incorporate: 6](#_Toc6262297)

[1.5 Road Map 6](#_Toc6262298)

[Chapter 2: Background 7](#_Toc6262299)

[2.2: Understanding motion. 7](#_Toc6262300)

[2.4 Representing motion as vectors 11](#_Toc6262301)

[2.3: Understanding Circular motion 12](#_Toc6262302)

[Forces, collisions and momentum 14](#_Toc6262303)

[Quantifying forces 15](#_Toc6262304)

[Quantifying Friction 15](#_Toc6262305)

[Momentum and impulse 20](#_Toc6262308)

[Chapter 3: Analysis Requirements and Design 23](#_Toc6262309)

[Language of choice 23](#_Toc6262310)

[Tools required for the project 23](#_Toc6262311)

[Iterative process: 24](#_Toc6262312)

[Predicting motion through an iterative method: 25](#_Toc6262313)

[Chapter 4: Experimentation and Evaluation 27](#_Toc6262314)

[Chapter 5: Timescale 27](#_Toc6262315)

[Glossary 28](#_Toc6262316)

[Refrences 29](#_Toc6262317)

# Glossary

|  |  |
| --- | --- |
| Symbol | Definition |
| s | the displacement of an object (how much has the position changed) |
| u | initial recorded velocity |
| v | Final Velocity |
| a | Acceleration |
| t | Time |
| g | Gravitational acceleration |
| F | Force |
| Fr | Fricional Force |
| D | Drag |
| Cd | Drag co-efficient |
| ρ | density |
| p | momentum |
| P | Pressure |
|  | angle turned |
|  | angular velocity |
|  | initial angular velocity |
|  | angular acceleration |
| L | Force of Lift |
| G | Vortex (for magnus effect |

# Chapter 1. Introduction:

In this section, the history and importance of physics libraries will be discussed with regards to how they relate to simulating the daring air raid, Operation chastise.

## 1.1: Why simulate Operation Chastise:

During my undergraduate studies in 2013, one of my projects was to build a web app to simulate some aspects of the bouncing bomb operation. However, due to having an extremely limited understanding in programming, the amount of physical properties that was simulated was very limited. However, as my understanding in programming has improved, so has my understanding on how to implement a more realistic simulation of a bouncing bomb by using less constants like having the velocity of the bomb along the i direction dynamically be affected by skimming the surface rather than a flat value of velocity reduced every time it skims. [3][4]

### 1.1.1: A brief history of operation chastise

On the night of the 16th and 17th of May, the RAF launched one of the most daring bombing raids in World War 2. The objective of this raid was to destroy three dams located by the industrial heartland of Germany. The reason why dams were chosen in specific was because it was believed that by destroying the dams, it would flood the surrounding areas around the dam and prevent any electricity being generated by the dams. The idea behind this was that by destroying the dams, the British could cripple Germany’s industrial capabilities a lot more effectively than launching conventional bombing raids directly over the city. However, as every dam was protected by a series of torpedo nets, it was crucial for a new type of bomb that would skim over the water to avoid being caught by torpedo nets

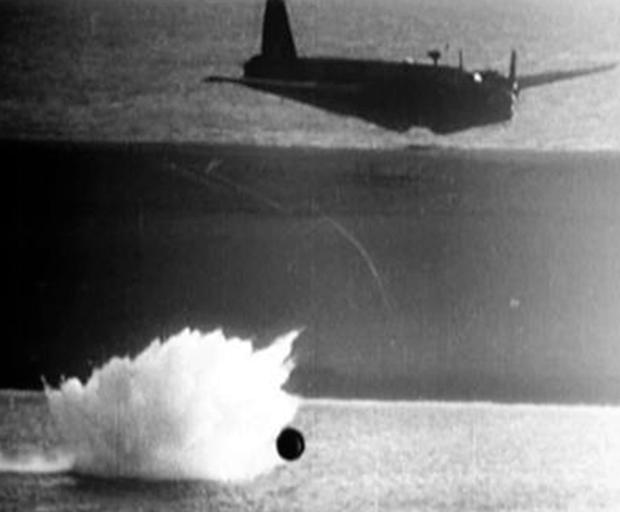


Image taken from <https://www.helensburghadvertiser.co.uk/news/15475746.eye-on-millig-helensburghs-crucial-role-in-testing-dam-busters-bouncing-bombs/>

### 1.1.2: Why create a physics library for Operation Chastise

As the dam where protected by anti-torpedo nets. The bombs dropped where designed to bounce across the lakes in a similar manner to ‘skipping stones’. Thus, negating the effect of torpedo nets underwater. Therefore, this particular event provides many interesting physical aspects to simulate. This would include:

* The height in which the bomb is dropped. During the original operation, the plane was required to fly 60ft above the surface of the water. Therefore, the user can be given the option to adjust the height in which the bomb is dropped from.
* The rate in which the bomb was spinning. During the original operation, the bouncing bomb was required to have a back spin at 700 rpm. Although this did very little to affect the way the bomb hit the water, it seems more likely due to the Magnus effect caused by the rotation generating lift.
* As the bombs velocity is entirely dependent on the aircrafts velocity before it is launched, allow the user to adjust the aircrafts air speed to see how that will affect the characteristics of the bouncing bomb   
    
  [3][4]

## http://www.chm.bris.ac.uk/webprojects2001/moorcraft/Image25.gif

## 

Demonstration of how the bouncing bomb worked. Image taken from: <http://www.chm.bris.ac.uk/webprojects2001/moorcraft/The%20Bouncing%20Bomb.htm>

## 1.2 The physics Library Incorporate:

* Physics models
  + This would require the computation of a variety of physical aspects such as:
    - The initial position of an object in vector form (for 2d <i,j> for 3d <i, j, k>
    - The velocity of an object in vector form
    - The acceleration of an object (acceleration can be derived from the force exerted by an object and the mass assigned to an object which will be discussed in the forces in play)
    - Basic material properties necessary for simulation i.e. the density which will affect the mass of an object depending on its size)
* Objects manager
* Collision detection and response
  + The collision detection model should be able to determine which objects will collide with each other based on the geometry and position of each object
  + The collision response model should incorporate how objects would react when colliding i.e. is there two objects in motion colliding or is an object colliding with a static object, how much energy is lost as a result of collisions and how the rotation of an object would affect the way they respond (note that these will be two independent entities but the collision response is heavily dependent on collision detection)
* Force’s in play
  + The forces exerted by an object i.e. thrust generated from an aircraft
  + The forces exerted onto an object i.e. air and surface friction.

# 1.3 Road Map

This roadmap will explain the main areas of interest for the completion of this project.

* In chapter 2, the Background information relating to the mathematics behind real life physical phenomena would be derived and analysed
* In chapter 3, The analysis of different languages for Android applications will be discussed as with how to model physical properties into usable code to model a variety of aspects relating to Operation Chastice
* In chapter 4, The methods of testing the physics library would be discussed as well as how to test the Simulation of operation chastise.
* Chapter 5 will elaborate on how the production schedule of the library as well as detailing when certain aspects of the physics library will be completed and tested.

# Chapter 2: Background

As this project relies almost entirely on physical equations and properties. This chapter would discuss the necessary equations needed to model motion, model collisions and collision response and the internal and external forces in play.   
While this proposal will only very briefly skim through these concepts. The Report will cover a lot more material on how each of these equations are derived.   
  
2.1: Simulating motion

One of the most important aspects of any physics library is to describe how objects in motion work. This can be derived almost entirely from first principle equations. This section will very briefly discuss how these will be derived from first principles and how these can be applied into very basic code with diagrams included.

## 2.2: Understanding motion.

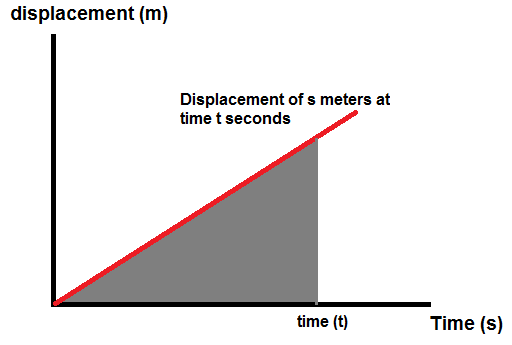
In the simplest sense, an expression for an object in motion can be expressed using the following equations.

*Figure 3. Equations of motion*

Where:

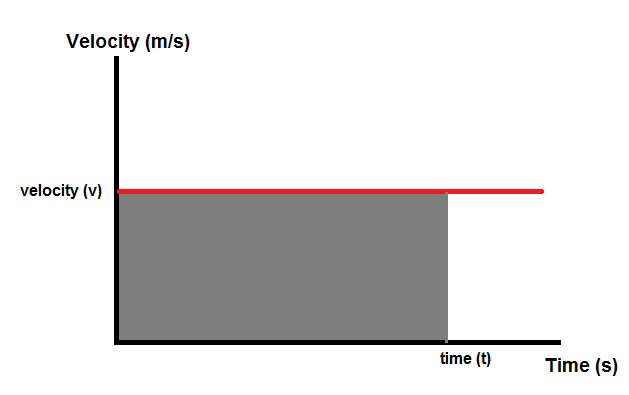
* V = the velocity of an object at time t
* u = the initial velocity of an object
* s = the displacement of an object
* a = the acceleration of an object in motion
* t = the time (how long the object has been recorded in its motion)

Therefore, when plotting a graph of distance against time for an object with a constant velocity, a very obvious trend can be observed as demonstrated in figure 4.



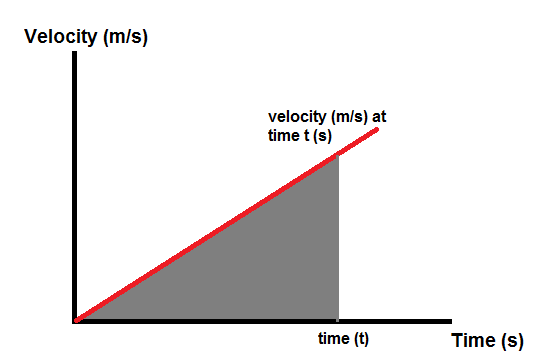
*Figure 4. displacement under a constant velocity plotted against time.*

Where the velocity can be observed as the gradient function of the change in displacement. Also, If a graph of the same constant velocity against time is plotted, another trend can be easily observed as follows.



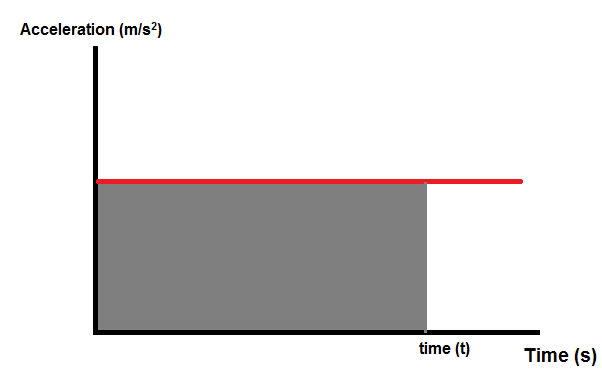
*Figure 5. constant velocity against time*

Where the total displacement of an object can be determined by calculating the area under the line. However, If the velocity is not constant but is increasing under a constant acceleration, The same trend observed in the displacement time graph can also be observed as follows.



*Figure 6. velocity under constant acceleration against time*

From the graph in figure 6, the acceleration of the velocity can be derived from the gradient function of the graph. Much like the velocity can be derived from the gradient function of the graph depicting displacement against time.

Similarly to the previous examples, if a graph plotting acceleration against time is plotted as shown below:

*Figure 7. velocity under constant acceleration against time*

From this graph, the velocity can be derived in exactly the same way displacement was derived from the examples showing velocity against time graphs by calculating the area under the line.

Therefore, it can be observed that velocity is a measure of the rate of change of displacement against the rate of change against time and acceleration is the rate of change of velocity against the rate of change in time. Therefore, These relationships can be expressed in differential equations (equations that measure the rate of change in an instance infinitely small) as shown below in figure 8:

*Figure 8: velocity and acceleration expressed in differential equations.*

Also the velocity can also be measured from the area under the line of the acceleration/time graph as the displacement can be measured as the area under the line in the velocity/time graph. Therefore, these relationships can be expressed as integral equations (a measure of an area under a curve in iterations infinitely small).

*Figure 9: displacement and velocity expressed with integral equations with time t as the limit.*

Therefore, if the either the displacement, velocity or acceleration or all of them are non-uniform i.e. varying all the time, it will still be more than possible to get values for each of them in a given instance of time.  
  
[1]

## 2.4 Representing motion as vectors

The previous section discusses how to determine the value of displacement, velocity and acceleration of an object. However, these equations only calculate the scalar values and not the vector values of motion. Therefore, these equations wont provide the position of an object or the direction of an object in motion. The position, velocity and acceleration of an object can be expressed relatively easily in two dimensions (2D) using vectors i and j as shown below:

Figure 10: expressing motion in 2D vector form

If the absolute values for velocity and acceleration are required, these can easily be calculated through two methods. Using Pythagoras theorem or using trig equations

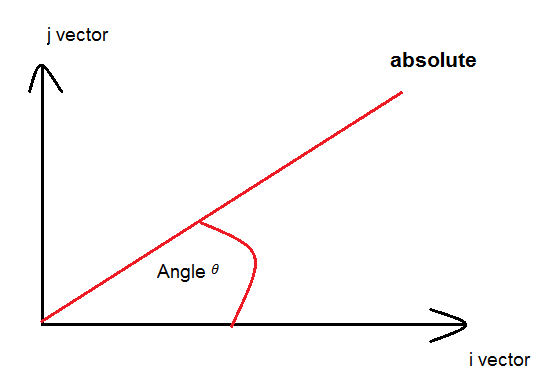


Figure 11: finding absolute values for velocity and acceleration using Pythagoras

Figure 12: finding absolute values for velocity and acceleration using trigonometry

[1]

## 2.3: Understanding Circular motion

While Circular motion is used to describe the rotation of an object rather than the movement of an object travelling from one point to another. The behaviour in which an object rotates can be expressed with almost identical equations to the equations of motions shown in figure 3 as demonstrated below:

*Figure 13: Equations for circular motion*

Note: all angles are assumed to be taken in radians unless expressed otherwise.

Where:

* = the angular velocity
* = the angle rotated
* = the angular acceleration
* t = time of rotation.

As seen in figure 3 in chapter 2.2, the equations for circular motion at the centre of a circle are identical to the equations of motion. Therefore, it can be assumed that the relationship between the angle rotated, the angular velocity and the angular acceleration would be also be identical to the relationship between the displacement of an object, the velocity of an object and the acceleration of an object in motion where:

Figure 14: using differential expressions to view the rate of change of circular motion

Figure 15: using integral expressions for circular motion

However the further away from the centre of rotation, the greater the velocity of travel. Therefore by examining some basic circle geometry as demonstrated in the diagram below

*Figure 16: Diagram showing a variety of aspects of a circle (Diagram created by the author)*

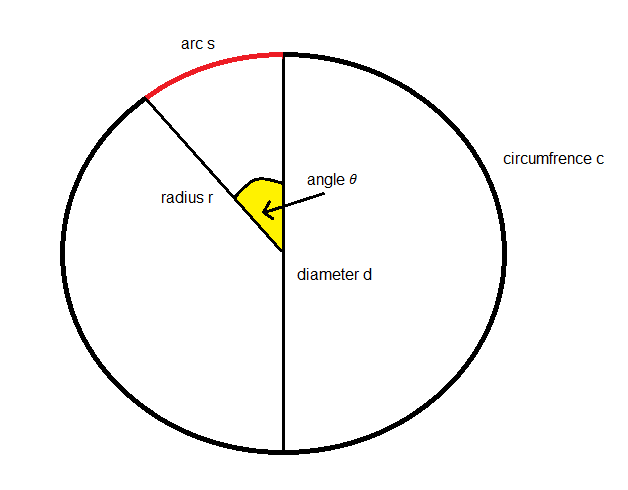


Diagram created by the author on paint

Where the circumference is calculated as follows on the circle is the total perimeter around the circle and can be calculated as follows:

Figure 17: Equation for the circumference of a circle

Where:

* d = the diameter of the circle (the total width of the circle)
* r = half the width of the circle
* c = the circumference of the circle (the total perimeter of the circle)

From the graph, it can be observed that the arc is a portion of the circumference. Therefore an expression for the arc can be derived from the ratio of the angle. Therefore the length of the arc can be expressed as follows

Figure 18: deriving an expression for the arc

Where:

* = angle traversed by the arc
* s = total length of the arc (this will later be known as the displacement traversing the circle as we move into circular motion. Hence why it shares the same variable name)

Therefore, if angular velocity is a measure of the rate of change of the angle rotated and the arc is a measure of a section of a circle rotated, an equation for the speed and acceleration around a circle can be shown as follows:

Figure 19: expressing velocity and acceleration around the perimeter of a circle by using its circumfrence

## Forces, collisions and momentum

Understanding motion is one crucial aspect of designing a physics library but the other important aspect of motion is understanding what creates motion. This can be expressed through directly quoting Isaac Newton’s three laws of motion:

1. Newton’s First Law  
   *“A body continues in a state of uniform rest or motion unless acted upon by an external force.”*
2. Newton’s Second Law  
   *“The acceleration produced when a force acts is directly proportional to the force and takes place in the direction in which the force acts.”*
3. Newton’s Third Law  
   *“To every Action, There is an equal and opposite reaction”*

From Newton’s laws, it can be observed that:

* If an object that is either in motion or motionless would stay in motion or motionless.
* The acceleration of an object is direcly proportional to the force exerted
* If every action has an equal and opposite reaction, this implies a conservation of momentum

### Quantifying forces

As force is directly proportional to the acceleration of an object, Force can be expressed as where K is an unchanging constant. In all cases K is the mass of an object which was derived from measuring the force of impact from a dropping object (apple in Newtons case) as the acceleration due to gravity is allways constant. Thus giving us the expression:

Figure 20: Equation for Force

Where:

* F = the force of an object
* g = the acceleration of an object due to gravity (typically 9.81 on Earth).

### Quantifying Friction

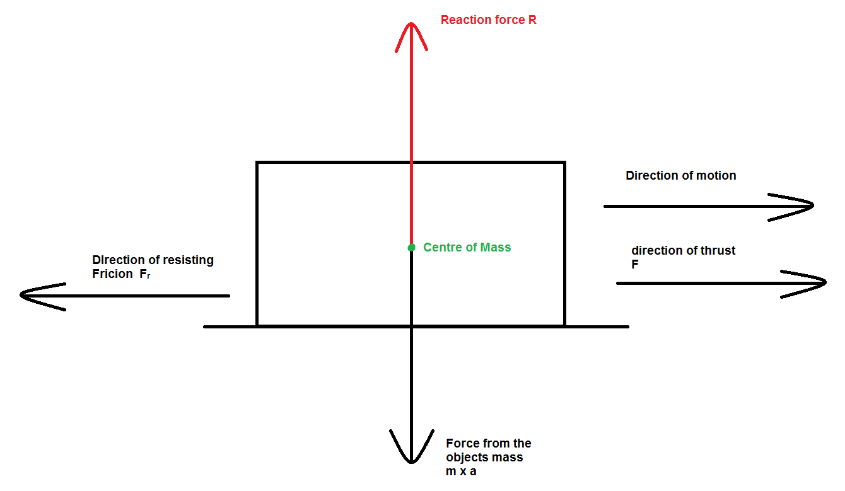
In a vacuum in empty space, an object will continue to accelerate for ever (or until it reaches close to the speed of light) however, there are many other forces in play that will prevent an object on earth typically accelerating towards infinity. All these forces can be quantified as Friction.

* static friction – The force between two surfaces preventing them from moving against each other
* Kinetic Friction – The force that acts against an object in motion

Friction can occur on the surfaces of solids or in a medium like the friction inside a liquid and gas.

#### Surface friction:

Surface friction is typically a measure of the resisting forces between two rough surfaces in contact. It typically can take the form of static friction or surface friction where it can prevent any motion i.or resist against the force of motion i.e. car tyres on a roads surface. Also the harder the force of contact, the greater the force of friction between these objects. Therefore, Surface friction can be expressed as follows

Figure 21: example of Surface friction in play on a level surface

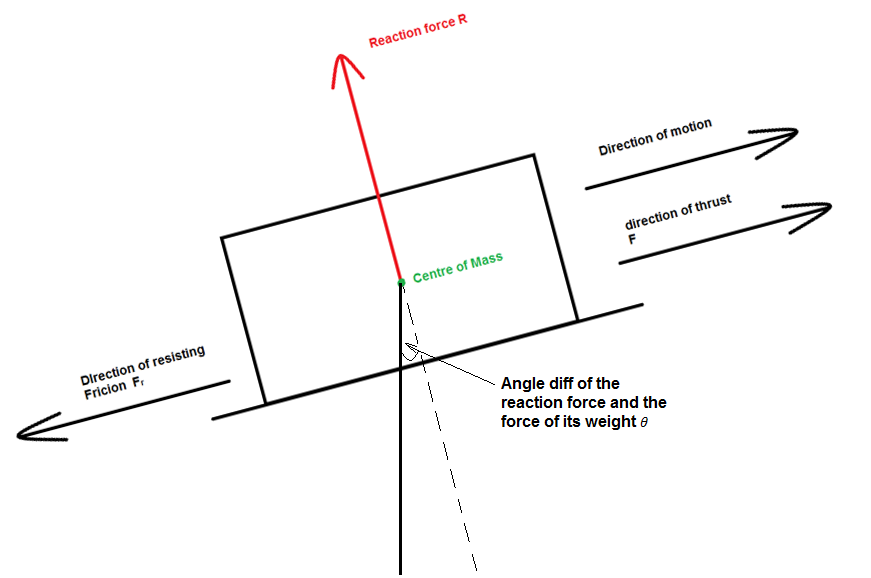
If Friction is directly proportional to the reaction force R an Expression for friction can be expressed as follows:

Figure 22: Equation for value of friction

Where

* Fr = Force of Friction
* R = the reaction force(the force exerted by the surface keeping the object from sinking in)
* = the co-efficient of friction. Its value is typically 0<<1 where the higher its value the more rough the surface.

However, the only demonstrates the force of friction on a level surface. If it were to be subjected to a slant, then the Reaction force would be different. As shown in the diagram below

Figure 23: Example of surface friction on a slope. 

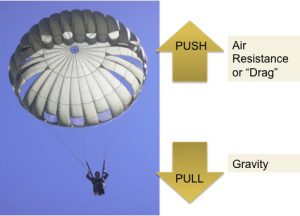
Where the reaction force can be expressed as:

As a result of surface friction, if a needs to move, the force exerted on it will need to be greater than the force of friction resisting it (F > Fr)

#### Drag (Friction in fluids)

Unlike Surface friction, Drag is the result of the fluid an object is in opposing the force of motion. As a result, it tends to be more kinetic friction rather than static friction as drag only increases with velocity and doesn’t have an upper limit.

Fig 24: Drag force slowing down the parachutes decent



In the image above, the drag force is slowing down the velocity of decent. The velocity the parachutist falls at will continue to fall until the drag force is equal to his weight where at that point the velocity will remain constant (and hopefully safe). The force of drag is heavily dependant on the size of the object, the density of the fluid the object is traveling through and the speed in which the object is travelling as expressed below:

Fig 25: Equation for calculating Drag

Where:

* **Cd** = the drag co-efficient
* = density of the fluid
* v = velocity of the object in motion
* A = the surface area of the object in motion.

The Drag co-efficient Cd is usually calculated by recording the drag force of an object when subjected to air travelling at certain speeds and therefore is usually determined experimentally. [11]

#### Lift Force

Lift force is a measure of the force at a right angle to the object in motion. This force is responsible for keeping aircraft in the sky or keeping racing cars on the ground. An example of lift can be can be demonstrated using bernoulli’s conservation of energy as demonstrated below as demonstrated below:

Figure 26: Lift explained by Bernoulli’s Principle

### Bernoulli's Explanation of Aerodynamic Lift

This example demonstrates lift being generated as a result of a pressure difference below and above the aerofoil as a result of differing Fluid Velocities. Therefore, the higher the velocity, the lower the pressure. As seen from the diagram, there is a lower velocity airflow at the bottom of the aerofoil then there is at the top of the aerofoil. As a result, the pressure below the aerofoil will be greater than the pressure above the aerofoil and therefore a resultant upwards lift force generated due to the pressure difference. Although there is a lot more to lift then just Bernoulli’s principle such as the transfer of momentum of air particles to the bottom of the aerofoil (Newtonians principle), For the purpose of the physics library, Bernoulli’s principle is sufficient. [12][13]

#### The Magnus effect:

During Operation Chastice, the bombs where given a backspin of 500 rpm before being dropped. While specific details as to why the backspin was applied is lost, It can be observed from Touvia Miloh’s journal *“Ricochet off water spherical projectiles”*, applying spin to a barrel would have almost no effect in the way the barrel bounced across the water. However, in Miloh’s journal it also mentioned an equation for the entry angle of the barrel as stated below:

Figure 27: equation for the critical angle

Where:

* = critical angle (maximum angle of entry before it stops bouncing and starts sinking)
* = relative density of the projectile

Therefore, it could be assumed from Miloh’s journal that the backspin applied to the bouncing bombs where to keep the angle in which the bomb hit the water below the critical angle by reducing the speed the bombs fell. The lift generated from a spinning object is known as the Magnus effect which can be demonstrated from the image below:

Figure 28: Diagram depicting the magnus effect.

# https://www.norwegiancreations.com/wp-content/uploads/2019/02/magnus_force.png

As seen from figure 23, the way lift is generated from the Magnus effect is identical to the explanation using Bernoulli’s principle where lift generated is due to differing fluid velocity above and below the barrel creating the same pressure difference observed from figure 23 to generate lift. Therefore the force of lift can be expressed from the kutta-Joukowski lift theorem below

Figure 29: Kutta-Joukowski lift theorem

Where:

* L = Lift of force generated by the magnus effect
* = density of fluid (in this case air)
* = the vortex of strength of the barrel

The vortex strength G is simply the relationship of the cross-sectional circles geometry and the velocity in which it rotates as demonstrated below:

Figure 30: equation for the vortex

Where:

* vr = the rotational velocity of the circles circumference  
  (note how the Vortex is calculated from the circles circumference by its velocity) [15]

### Momentum and impulse

Momentum is the relationship between the mass an object and its velocity. As interpreted from Newtons first and third law, Momentum will remain constant unless the object interacts with some external forces

Momentum p can be defined as follows:

Figure 31 Equation of momentum

An object with mass and constant motion will always be travelling with momentum as long as its in motion. However if another object was to change that momentum, it would undergo a force due to a change in velocity which therefore generates a value for acceleration (the rate of change of velocity) this force can be quantified as shown below:

Figure 32: the equation of impulse force

Where:

Is also the calculation acceleration (rate of change in velocity) that is derived from re-arranging Equation 1 from Figure 3.

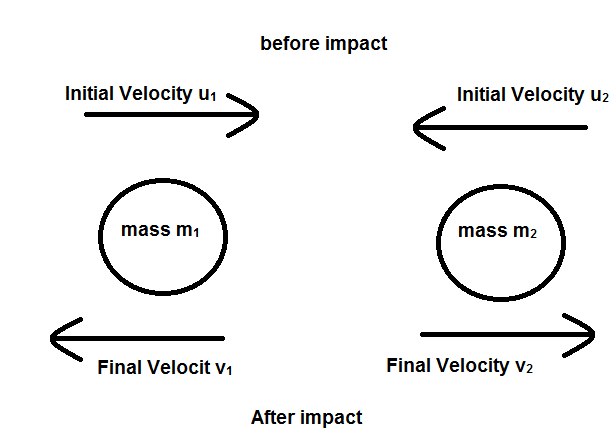
By observing the objects in motion below:

Figure 33: image demonstrating two objects colliding

If momentum is always conserved, then the total initial momentum in this example will be equal to the final momentum. This can be quantified as follows below:

Figure 34: Equation for the conservation of momentum

In a Frictionless System, Momentum conserved will only stay between these two particles in collision with each other. However, in realistic situations, momentum tends to get transferred to other factors in the form of sound by transferring momentum to air particles or by the object not being perfectly rigid. Therefore, the co-efficient of restitution is the ratio of momentum lost from external factors as demonstrated below:

Figure 35: Co-efficient of restitution:

Where:

* e = co-efficient of restitution
* u = velocity before collision
* v = velocity after collision

In almost all scenarios, objects don’t collide directly head on. Therefore there will allways be some deviation between objects in collisions. In a two dimensional plane, this can be demonstrated from the diagram below: [1]

#### Implementing direction to momentum

In most scenarios, objects tend to collide at an angle as demonstrated below:

#### Figure 36: collisions with directon

Therefore, the conservation of momentum can be expressed with direction as followed.

Figure 37: equations for momentum with direction

where:

* pi sum of momentum along the i plane
* pj sum of momentum along the j plane [1]

# Chapter 3: Analysis Requirements and Design

This chapter will revolve around how the game and library will be designed and implemented into a functioning mobile app.

## Language of choice

As this project revolves around programming this simulation for the android platform. As of 2017, Google has begun to provide full support for kotlin as one of the programming languages of choice for android app development. While java is the default language for android development, Kotlin also runs on the Java Virtual Machine too. Another advantage of using kotlin over other languages like Java is the added improvements to functional programming over java. As this project will require a series of functional loops to simulate motion, easier to use lambda functions in kotlin will make the programming of some equations a lot less memory intensive.

## Tools required for the project

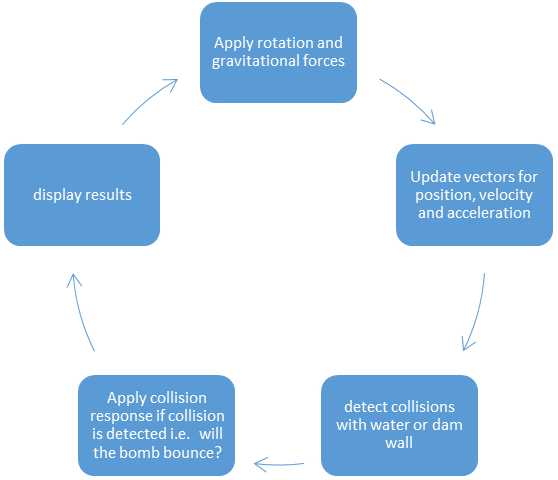
There will be a number of third party tools required for the completion of this project. The IDE of Choice will be Android Studio 3.0 for the development due to it being the Default IDE for android app development with full kotlin support. This project will also rely heavily on the Kotlin maths library in order to carry out most of the equations discussed in chapter 2.   
As creating a graphics rendering library will create far too many requirements to complete this project, The use of third party graphics libraries will be required. Therefore the graphics renderer of choice will be OpenGL as it can be used with kotlin to create a GUI to host the simulation on and is perfectly compatible for use with Kotlin.

## Iterative process:

As everything is simulated through an iterative process where each iteration is a value of time t. Each Iteration should have this process. Each iteration of time should incorporate the new values for variables like an objects position, velocity and acceleration. An example of this iterative loop is demonstrated below.

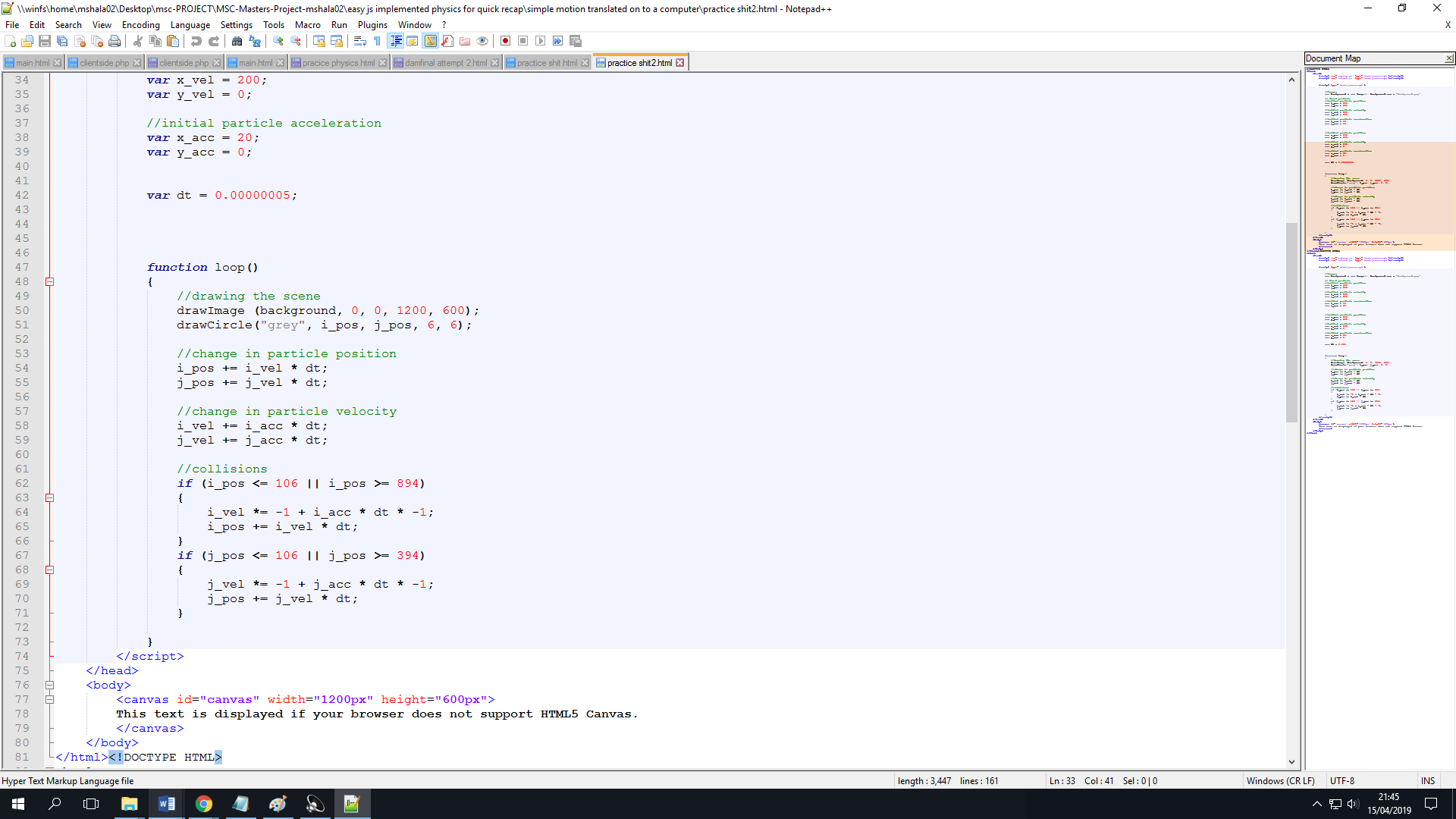
Figure 38: the iterative process for the bouncing bomb simulation

Bomb dropped with its motion influenced by the planes



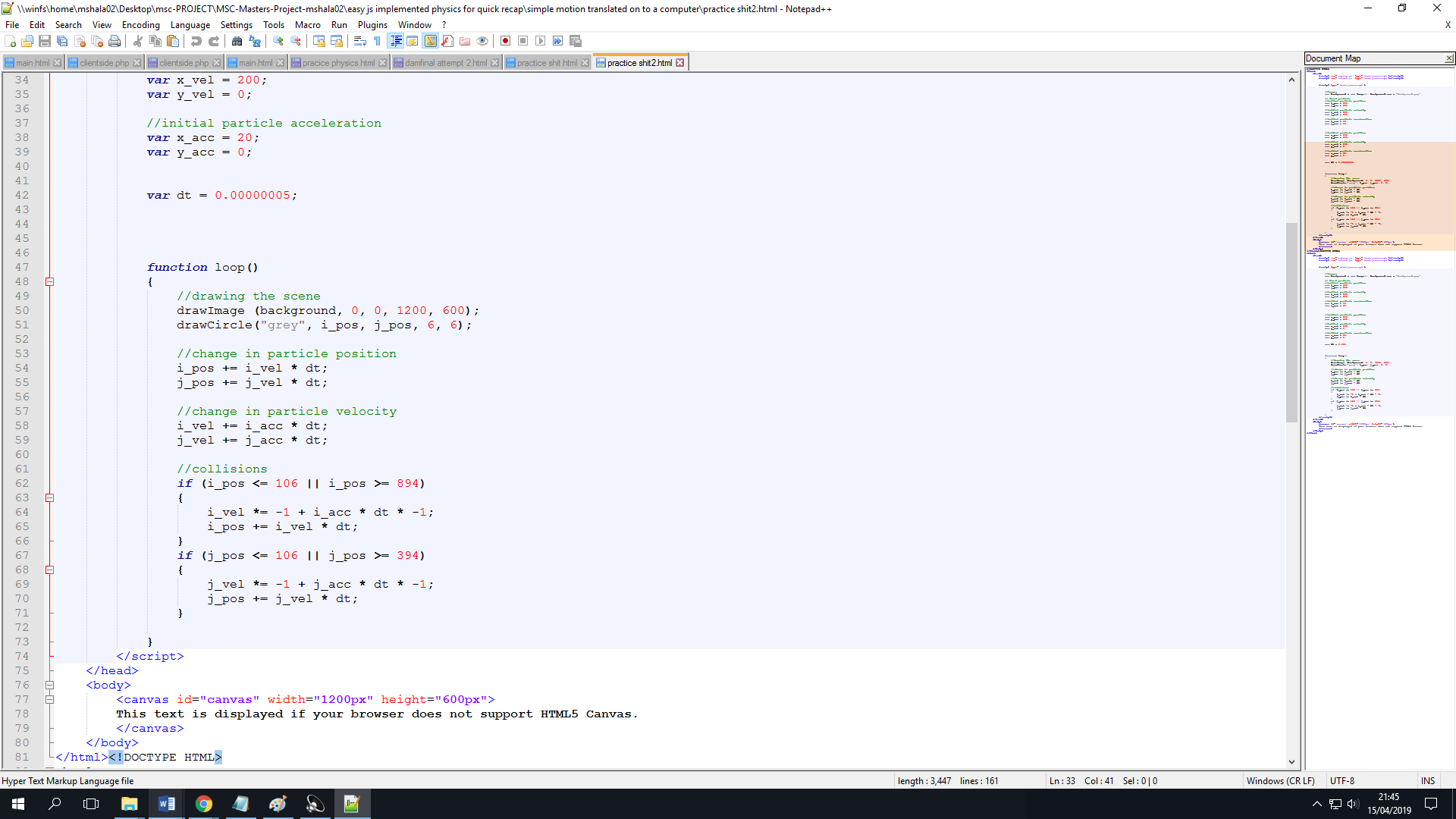
As a result of the Simulation being run per each iteration of time, a more step-by-step approach for calculating the displacement, velocity and acceleration of an object must be calculated through a more iterative method that incorporates units of time with units less than 0. An example of this can be seen in a simple javascript simulation of motion as demonstrated below:

Figure 39: position and velocity from each iteration



Where the position and velocity is updated on each interval of time. However, for collisions, boundaries for each object must be created. Therefore collision detection and response could take the form of conditional statements as shown in the figure below

Figure 40: basic collision model

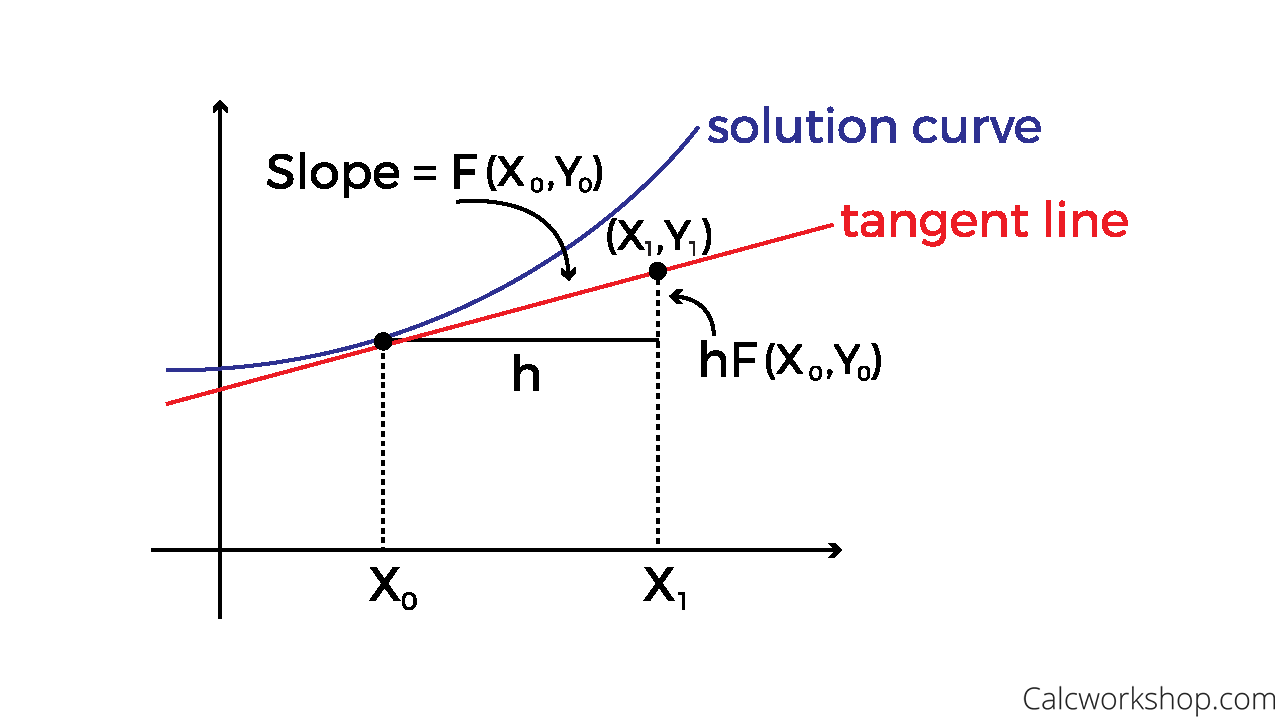


### Predicting motion through an iterative method:

As it will not be possible to predict motion through the use of differential or integral equations as t > 0. An iterative process will be required to predict the future motion of the bomb based on its current motion and position. There are two methods potential methods of measuring motion through an iterative process. This can be accopplished through using an iterative method such as Euler’s method

#### Euler’s method

Eulers method is a way to find a linear approximation for a non-linear trend as shown below:

Figure 41: demonstration of Euler’s method

This approximation can be expressed in the following equation:

Figure 42: equation for Eulers method

where:

* yn is the predicted value for the next iteration of
* F is the gradient function of the slope being measured at positions (xn-1 ,yn-1)
* h is the iteration size

In terms of linear motion, this equation can be re-written using displacement ‘s’ as an example as follows:

Figure 43: Euler’s method to calculate displacement

and this can be broken down even further into vector form of displacement to get the position vectors si and sj   
[19]

#### Circular motion

In order to rotate an object by an iterative process. the position of the shape must be represented in vectors. Therefore, through each iteraton of time, the a new shape will be translated onto the screen with new positions of its i and j vectors as shown below:

Figure 44: rotation matrix

Where   
i and j = the original position vectors  
inew and jnew are the new position vectors.

From the matrix above, the following equations can be obtained:

Figure 45: equation for rotated position

However, the equation above does not incorporate for translation around angle . Therefore, a 3x3 matrix is required to implement the translation of 2D rotation as shown below.

Figure 46: Matrix Trans

where the number 1 represents the translation of the new matrix position. the equations derived will still remain the same as the equations in figure 44. [9]

# Chapter 4: Experimentation and Evaluation

One key area for experimentation would be to assess the most optimal step size for time t. As seen from more basic examples built on JavaScript, the step size must be optimised for the time t as if the size is too large, then the value of error from using Euler’s method will become significant. too small however, and the simulation will start to become increadibly slow.

One method to evaluate against the accuracy of motion using Euler’s method would be to create unit tests to evaluate the accuracy of values for motion gained from Euler’s is to calculate exact values separately to assess the error.

Another area for evaluation would be to create unit tests for each of the functions used to simulate physics to aid with the programming of a viable physics library.

However, the overall accuracy of the simulation will not be apparent until the first working version is complete. Therefore it is important to run each version of the simulation and compare the simulation to real life results from the original operation and re-creations.

# Chapter 5: Timescale

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **sSimulation of the Bouncing Bomb** | **week 1** | **week 2** | **week 3** | **week 4** | **week 5** | **week 6** | **week 7** | **week 8** |
| **Set up requirements:** |  |  |  |  |  |  |  |  |
| familiarise self with kotlin |  |  |  |  |  |  |  |  |
| familiarise self with open GL |  |  |  |  |  |  |  |  |
| **Production** |  |  |  |  |  |  |  |  |
| develop unit tests |  |  |  |  |  |  |  |  |
| develop library of physical properties |  |  |  |  |  |  |  |  |
| **Post production** |  |  |  |  |  |  |  |  |
| User testing |  |  |  |  |  |  |  |  |
| Post production fixes |  |  |  |  |  |  |  |  |
| **Report write up** |  |  |  |  |  |  |  |  |

# 

# Refrences

1. [Physics for Game Developers, 2nd Edition](https://learning.oreilly.com/library/view/physics-for-game/9781449361037/) by Bryan Bywalec; David M Bourg  
   (viewed from https://learning.oreilly.com/library/view/physics-for-game/9781449361037/pr02.html
2. A summarised comparison of kotlin vs java  
   <https://www.konstantinfo.com/blog/kotlin-vs-java/>
3. History of the bouncing bomb operation

<http://www.dambusters.org.uk/the-dam-raids/the-bomb/the-bouncing-bomb/>

1. How the bouncing bomb worked  
   <http://www.chm.bris.ac.uk/webprojects2001/moorcraft/The%20Bouncing%20Bomb.htm>
2. Circular motion  
   <http://web.mit.edu/8.01t/www/materials/modules/chapter06.pdf>
3. Circular motion equaitons  
   <https://www.khanacademy.org/science/ap-physics-1/ap-centripetal-force-and-gravitation/introduction-to-uniform-circular-motion-ap/a/circular-motion-basics-ap1>
4. <https://www.intmath.com/differential-equations/11-eulers-method-des.php>
5. <https://www.helensburghadvertiser.co.uk/news/15475746.eye-on-millig-helensburghs-crucial-role-in-testing-dam-busters-bouncing-bombs/> (bouncing bomb and bomber image)
6. Easy to understand matrix transformation   
   <https://www.alanzucconi.com/2016/02/10/tranfsormation-matrix/>
7. A direct refrence to newtons laws of motion  
   <http://www.oxfordreference.com/view/10.1093/oi/authority.20110803100232420>
8. Drag and parachute  
   <https://prosserlibrary.info/2017/10/push-pull-forces-parachutes/>
9. Force of lift  
   <https://www.grc.nasa.gov/www/k-12/airplane/drageq.html>
10. Lift from Bernoulli’s  
    <https://www.mpoweruk.com/flight_theory.htm>
11. Bouncing steel balls on water. by Michael de Podesta, 2007 Phys. Educ. 42 466
12. Magnus Effect in Duct Flow by Jesse Batko, Cameron Clarke, Kenneth Smith.   
    <https://www.slideshare.net/GLK21fm/magnus-effect-in-duct-flow-jbatko-cclarke-ksmith>
13. Advantages of kotlin  
    <https://business.udemy.com/blog/kotlin-vs-java-9-benefits-of-kotlin-for-your-business/>
14. comparison of kotlin to java  
    <https://kotlinlang.org/docs/reference/comparison-to-java.html>
15. Google officially supporting kotlin   
    <https://www.theverge.com/2017/5/17/15654988/google-jet-brains-kotlin-programming-language-android-development-io-2017>
16. explanation of the Eulers Method  
    <http://tutorial.math.lamar.edu/Classes/DE/EulersMethod.aspx>
17. explaining forces  
    <https://www.physicsclassroom.com/class/newtlaws/Lesson-2/The-Meaning-of-Force>