VANIER
CÉGEP / COLLEGE

Project Management Plan

[Billiards Simulation]

Presented to Samad Rostam Pour

Author: [Karim Botros]
Last Updated: February 19, 2019

Table of contents

Project Definition

Project Sponsor: [Samad Rostam Pour]
Project Members: [Karim Botros] [Vincent Bruzzese] [Dylan Bobb] [Daniel Cicciarelli]
Estimated Project Start Date: [2019/02/14]
Estimated Project End Date: [2019/04/07]

Executive Summary of Project Charter

The initial plan would be to implement a physics simulation of kinetic and elastic energy. The main aspect that will be looked at are the ball to ball contact. The speed of the impact, along with the angle with which it will be hit at, will come into play. The table itself will implement friction, with different amounts of friction available for the player to change to show different results, and an elastic border which will change the reaction of the balls upon contact. A physics simulation will also affect the inclination of the board, which will therefore influence the balls. The classic game rules of billiards will be implemented along the way, along with a retro themed style UI, and menus.

Project Purpose

The purpose of this simulation is to show the physics behind the classic game of billiards. The game has multiple layers beneath the classic ball to ball contact pastime. The game itself should include the simulation of a real game, along with a sandbox mode allowing the player to play around with physics related aspects such as the friction on the table and the mass of the balls.

Purpose of Project Management Plan

The project management plan (which will now be referred to as PMP) will detail what each team member will be specializing in, as well as giving an overview of the functionalities and what the final product will look like.

Project Scope

Our project will implement two game modes: Sandbox mode and Billiards mode. In the Sandbox mode, there will be different buttons and sliders allowing the user to play around with various physics elements in between every shot. Before the game starts, the user can also change the size of the cue ball to see the effects of a greater radius on that ball. The user will be allowed to change the friction on the table, the energy absorbance of the walls, the mass of the balls, and the tilt of the table before they take each shot. There will also be graphs that appear in a pop-up window that detail the changes in position and velocity of each ball, distinguishing each ball in the graph with its respective color. The Billiards mode will play a classic game of billiards, with no changes in the physics elements allowed.

Project Dependencies

The project will be developed using NetBeans and written in Java. It will be packaged as an executable jar file.

Project Constraints

Certain constraints that could cause issues are heavily implicated in the physics department of the game. The outcome of the project is dependent on the computer being able to properly simulate the contact and reaction of the balls throughout the game. The computer can cause issue, especially with the constraints the hardware can have, as having 9 active moving entities can cause certain issues. Other possibilities revolve with us having an overly simplified game or lacking the knowledge and experience to implement more complex features in the sandbox mode, such as ball spin.

Work Breakdown Structure

Front End

Vincent Bruzzese is responsible of everything UI related such as the menus, buttons, sliders, the color-theme of the project, and the game assets. He will program all these JavaFX controls, coding what happens when each control is used and where each control is available to press.

Game Rules

Daniel Cicciarelli oversees deciding what game settings will be available in each game mode (sandbox and the pool game). He will define the game rules such as play turns, fouls, scoring in a pocket, as well as showing position and velocity graphs for each of the balls.

Back End

Karim Botros and Dylan Bobb will be implementing the physics related to how billiard balls behave in a table. Physics elements such as momentum, friction, elevation, and collisions will be applied onto the billiard balls.

Project Management Plans

Milestones

The table below lists the milestones for this project, along with their estimated completion timeframe.

Milestones	Estimated Completion Timeframe
The project plan, along with all major ideas should be completed.	2019 - 02 - 14 Week 4
The main framework of the project, along with all physics simulation, the UI and the GUI should all be completed. The code should be completed as well, along with the entirety of the project. Both game modes should be completed	2019 - 04 - 07 Week 11
Oral presentations based off the full project will be presented to our target audience.	2019 - 04 - 14 to 2019 - 04 - 28 Week 12 -14
Individual and team reports will be presented to the involved committee.	2019 - 05 – 05 Week 15

Testing and Integration

Significant amount of testing will be done to ensure a bug free environment. Overloading certain commands and reviewing the outcome along the way. Having outside sources, unrelated to the class or project, test the environment and game itself would help with the collection of data and be able to see the game being run through its courses. Any bug or issue will be ironed out to ensure a smooth, professional program.

Testing Procedures

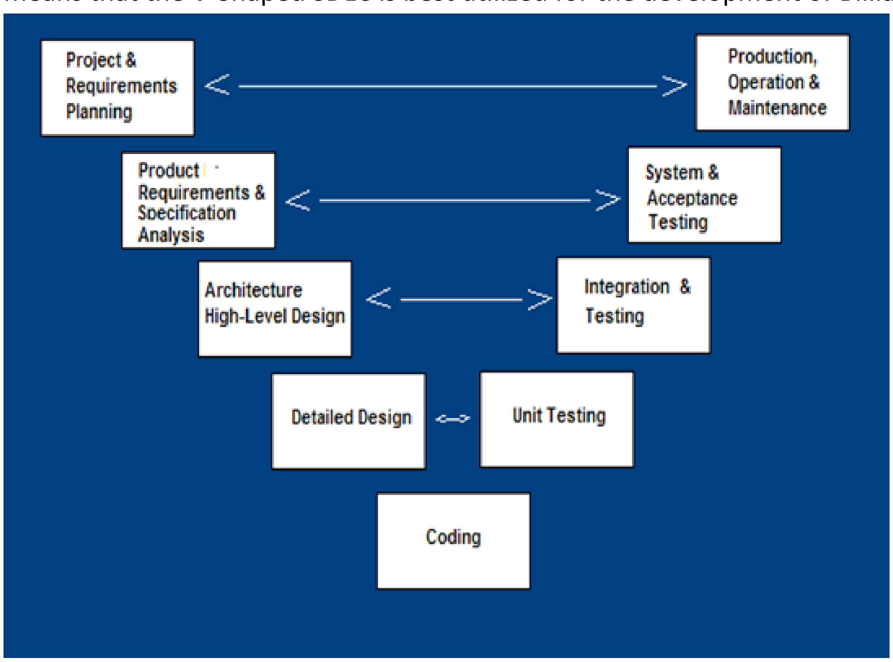
1. Implementer of a feature checks whether actual output matches the expected output.
2. The method in question is then sent for unit testing.
3. The method is then attempted to be broken with unexpected inputs (This step can be included in unit testing).

Version Control

To ensure a smooth process of integrating code between members, git will be used (through the use of bitbucket) to facilitate version control. In the event of a merge conflict, the code shall be rolled back to the latest stable version, while the origin of the conflict is being found.

V-Shape SDLC

Our project needs to be constantly tested, in order to make sure that all the physics aspects of the program are working correctly. That means that in unison, the code will be developed and tested. This means that the V-Shaped SDLC is best utilized for the development of Billiards Simulation.



•Project and Requirements Planning – All Team members will have access to Vanier computers, along with the programs necessary to complete the project. All our inter-Team documents shall be available for use, such as google documents, containing our documentation of the program and our coding database to allow for everyone to work on the same document. If need be, every team member has access to their personnel computer to work from remote locations.

•Product Requirements and Specification Analysis – Our program must properly demonstrate a physics simulation within a game of 8 ball pool, and the sandbox mode. This means that the system must be able to complete the equations of elastic kinetic energy and of linear momentum without any issues. It must also be able to detect collisions, and react according to the environment it is found in. The software program must also include a seamless UI which allows the user to use it with ease.

•Architecture or High-Level Design – Defines how software functions fulfill the design.

•Detailed Design – The detailed design will be properly shown in the flow chart, showing the main algorithms behind our billiard simulation.

• Production, Operation and Maintenance – Determining the necessary resources to produce the project and to maintain it.

•System and acceptance testing – Check the entire software system in its environment.

•Integration and Testing – Review *Testing and Integration*, as a detailed analysis into Integration and testing has been discussed. In short, the program will be tested regularly, ensuring that the code used is in working order. As well, team members will be continuously working on integrating their modules into the program, insuring that the program is working harmoniously.

•Unit testing – Each member will be assigned to make sure that their individual code is working flawlessly, in order to make sure that the final product be in working order. That means, overloading certain buttons or actions, to make sure that the code behind theme is up to par. This will mean, ensuring that the menu bar will bring up the code necessary, or that the cue ball reacts the way it should while colliding with any objects.

• Coding – Transform algorithms into software.

One of important aspects of the v-shaped SDLC is the fact that the code and program must constantly be tested, especially during the early stages of development. This will prove crucial for our project as we must make sure that the physics portion of our simulation is working to perfection, before anything else can move forward.

We make sure that we can easily verify and validate the program, but we must make sure that we can easily test it, to make sure it is in working order. Another reason why this SDLC was chosen is because we know all the requirements that our program needs to work properly, and the outcome is well know as well. The elastic reactions produced by the balls encountering each other are predictable and based on real-world physics.

Human Resource Management

Issues with the PMP

In the event of a member wishing to change the contents of this project management plan, a vote shall be initiated within the team. A majority of ¾ will be required to pass the vote. Once the vote is passed, the proposal shall be sent to Samad Rostam Pour (the professor for this project), who shall approve or deny the proposal. Samad will make the final decision, regardless of how many group members wish to approve the proposal.

Issues within team members

If two (or more) members disagree with the implementation or details of the PMP, the final decision on the interpretation of the PMP shall be made by the individual responsible to the section of the project in question.

Exception to this rule: If the individual responsible to the section of the project in question is failing to meet the project requirements, as described in the PMP, a vote shall be held. If a majority of ¾ is reached, the individual responsible for the section in question shall have two choices:

1. Implement the changes as requested by the group
   OR
2. Allow the other members of the team to implement the feature they are requesting.

Note: This vote does not need to be passed forward to Samad Rostam Pour for authorization.

Further Note: This vote can only be initiated if the feature being requested by the other members is already described in the PMP.

Logs of these votes, and descriptions of the features being voted on, shall be kept in a separate document, distributed on the team's Slack workspace.

Risk Management

Risk Assessment

Following the approval of a new feature (as described in section 7 of the PMP), a risk assessment shall be initiated as follows, in order of priority and importance:

1. Value to the project
2. Time requirement of proposition
3. Complexity/Difficulty of completing the task
4. Chance of improper implementation, failure
5. Risk of harming existing codebase

A vote shall be initiated to ensure that the feature is not an unnecessary risk as described by the risk assessment above. If following the assessment, ½ of the team members deem the feature an unnecessary risk, the addition shall be paused until the risk can be reduced, in which case, the process described in section 9 shall be re-initiated for the feature.

Risk Log

A document containing a written risk assessment of each newly proposed feature during the development process will be distributed to all team members on the team's Slack workspace.

Issue Management

In the presence of issues in our code, the following plan will be followed:

Procedure to Fix Issues

If the problem is between the compatibility of codes between two team members:

1. Asses the complexity of each member's code.
2. Determine the importance of each person's code (i.e. button functionality is not as important as the physics of the collisions.)
3. Whoever's code is less complex shall adjust their code to be compatible with the other member's.
4. If the complexity levels are the same, whoever's code is not as critical as the other's will adjust their code.

In Case of Need for Help

If the problem is within one person's code and that they are not capable of dealing with it, proceed in the following order until issue is solved:

1. Another team member steps in to help.
2. Another team member takes over the task.
3. Seek for external help such as a library that facilitates the achievement of the goal.

Procurement Management

Throughout the entirety of the project, Vanier College will be providing work space and computers to allow the development of the project. If need be, the project can be continued during the personnel time of every team member, as the completion is vital to a perfect grade and presentable project. Every team member has access to their own personal computers to ensure maximum production output when needed.

General Algorithm

In both the sandbox and regular pool game modes, the simulation will be separated into the two following phases: **Cue Phase** and **Action Phase**. During the **cue phase**, all objects on the table are stationary except for the cue stick which can be controlled by the user. The user will input a direction and a force using the cue on the cue ball, which then initiates the **action phase**. During the action phase, the user is stripped of all control and objects are put into motion, colliding into each other and being affected by various physics elements such as momentum, energy transfer, friction, collision, and gravity. Balls can fall into pockets during the **action phase** and game rules will be applied during the game loop. The **action phase** persists until all objects become stationary, where it shifts back to being the **cue phase**.

This simulation shall be made using an animation timer in JavaFX. The loop shall be on pause during the **cue phase**, and active during the **action phase**. During the **cue phase**, the player will use the mouse to pointer to angle the cue, giving the cue ball a direction to travel to, and a force slider that will be transmit the cue's force into the cue ball that will then receive the kinetic energy. As the **action phase** initiates, the game loop resumes, moving the cue ball to its target location and checking whether any collision has occurred with the walls or other balls. If there is a collision, new trajectories shall be calculated as described above for the objects concerned. Furthermore, friction of the table is applied on all moving objects and slows them down to a halt. In the sandbox game mode, more elements such as slopes and gravity will have effects on the objects' trajectories and behaviors. At the end of the **action phase**, the simulation will check whether any balls have been potted and apply specific game rules such as scratches onto the next player's turn.

Upon pressing the button to access the sandbox mode, the program will be receiving the information that said button was enabled and will open up a new stage allowing for the user to successfully play the second game mode. From here, the user will have access to switches which will modify the friction of the board by changing the friction factor in the formula. Also, if the user desires to manipulate the elevation of the board, there will be switches as well to modify it to his desires. These modifications will take place once the player has taken his shot in the **action phase**. This has been developed in detail in the paragraph above, pertaining to all the steps the program is doing to ensure that the shot is doing what it is supposed to do. The manipulation of the board will be done before every shot, and this will be done in a continuous loop, every time the player gets a ball in the pocket or scratches the cue. The loop would only be broken if the player desires to manually exit the game, or to pocket all the balls.

As for the Classic Billiards mode, pressing that button will open a new stage and a new Billiards game will begin. The program will automatically assign a Player 1 and Player 2, and then randomly assign one player to the first shot. That player will input the shot angle and shot power and press a button when they're happy with their inputs. Once that button is pressed, the program will simulate the shot's trajectory with the user's inputs and will calculate and simulate the collisions if there are any. The goal of the game is to get all the balls of the same suit in the pockets, and following the pocketing of your balls, pocketing the 8 ball. The two users will shoot until a player gets a single ball in a pocket in one turn. In that case, the suit of the ball the player pocketed will be assigned to that player by the program, and the user will get to shoot again. After each shot, the program will check to see if there was a scratch on the shot. A scratch occurs when:

- The player does not hit any ball.
- The player hits a ball of the opponents suit before hitting his/her own.
- The player pockets the cue ball.
- No balls hit any of the walls or are pocketed.
- The player hits the 8 ball before hitting his/her own.

Following the event of a scratch, the program will follow the scratch procedure (refer below for scratch procedure). If there was no scratch, the user hit a ball of his own suit and possibly pocketed a ball. When a ball is pocketed, the program will check to see if the ball pocketed belonged to the same suit as the user. If the ball did not belong the user, nothing happens. If the ball did belong to the user, the program sets a boolean variable to true that allows the player to shoot again. If the player pockets the 8 ball, the program will check to see if the player pocketed all of his/her balls. If not, the player automatically loses the game, and if so, the player wins the game.

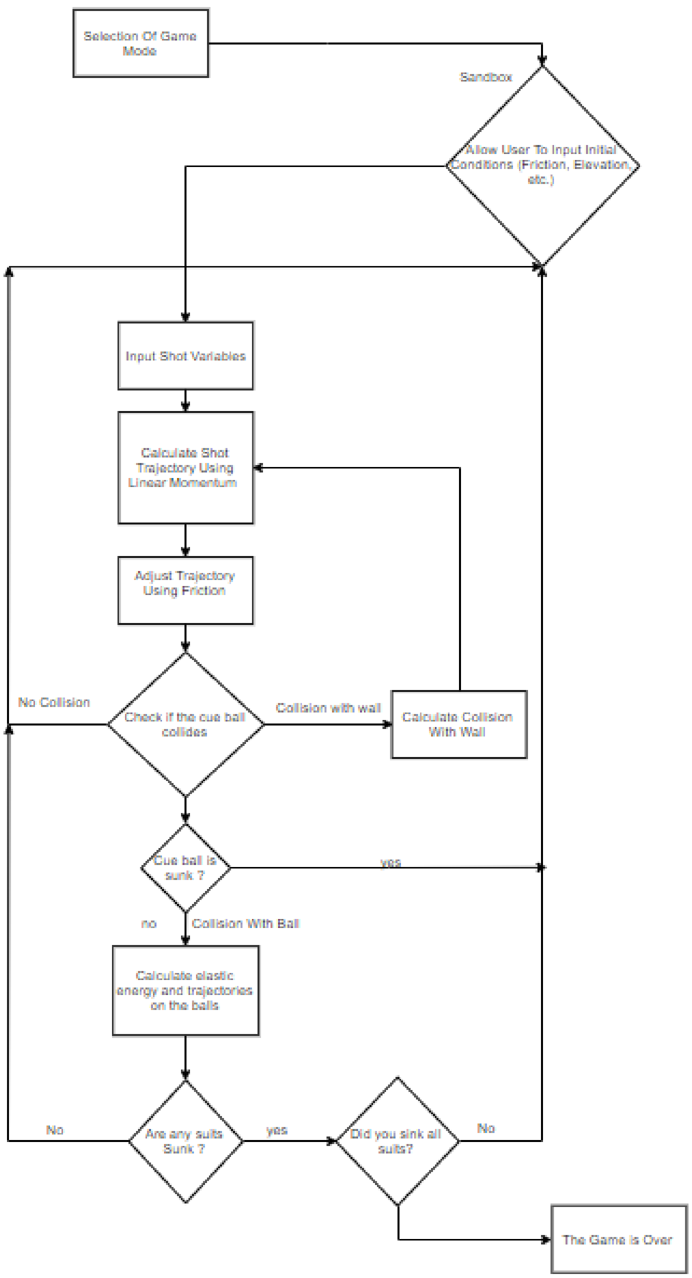
Scratch Procedure

When a ball needs to be scratched, the scratch flag (variable) shall be set to true which allows the player who DID NOT scratch the ball to move the cue ball to any position on the table that he/she desires. This variable shall enable a mouse movement listener, which will move the cue ball to the position of the mouse. Once the player clicks, the ball shall be locked into place, and the scratch flag shall be set to false, disabling the mouse movement listener. Afterwards, the shot procedure shall re-commence normally (it shall be the turn of the player who just repositioned the cue ball) by initiating the **cue phase**.

Functionalities

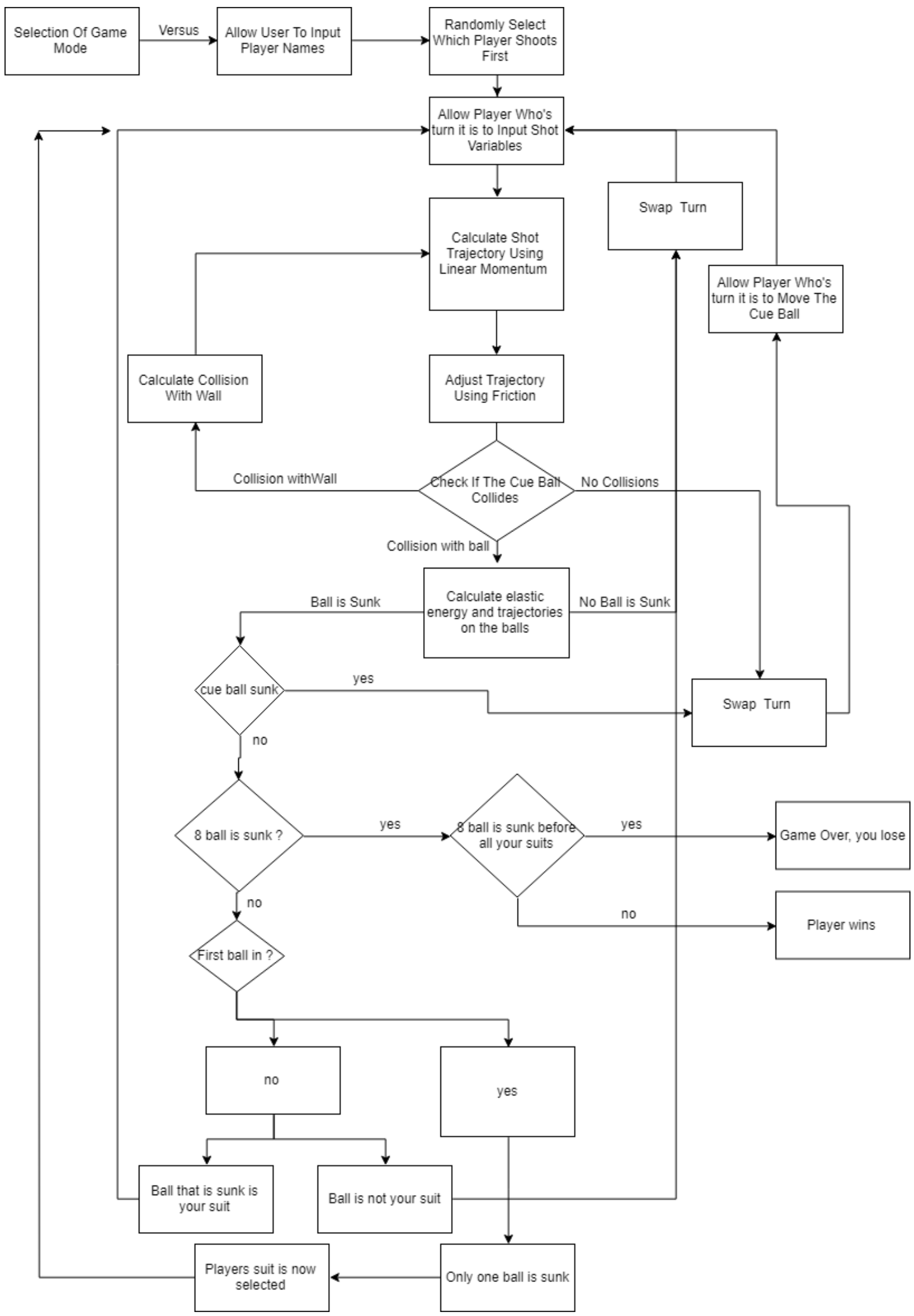
Sandbox Algorithm Flowchart

The following flowchart describes the algorithm for the sandbox portion of the simulation.



Billiards Algorithm Flowchart

The following flowchart describes the algorithm for the sandbox portion of the simulation.



User Interface

Implementation of the user interface will be done by team member Vincent Bruzzese. A full Asset manager filled with appropriate background images, sounds and images will be present. The goal will be to have an easy to use and visually appealing project. Keeping a general theme and art style will ease the user and simplify the look and feel.

Game Rules

Game rules will be implemented by team member Daniel Cicciarelli. Our simulation will have two game modes: Classic Billiards and Sandbox Mode. The Classic Billiards game mode will follow a regular game of pool, with standard friction, ball masses and collision elasticity. The Sandbox game mode will be the more important of the two game modes, as it will include the ability to change many of the physics related aspects of the game.

Physics

The physics portion will be implemented by Karim Botros and Dylan Bobb.

The behavior of the balls in the game of billiards is dictated by a set of physics formulas and rules that are described in the following section:

1. Conservation of Linear Momentum:

mA V1A = mA V2A + mB V2B ---> V1A = V2A + V2B
The initial speed of the cue ball is equal to the final speed of the cue ball and the final speed of the ball it comes into contact with. As the masses of the two balls are equal, mass is irrelevant in this equation. In the sandbox mode, the player has the option to change the mass of the cue ball, in which case the first equation is necessary.

2. Collision Detection: 2nd

As the balls move around the table, their movement is represented by vectors. As the ball positions change, the algorithm will detect when two vectors collide and calculate the angle of impact to be able to give the balls a new trajectory.

3. Conservation of Elastic Kinetic Energy:

½ mA (V1A)^2 = ½ mA (V2A)^2 + ½ mB (V2B)^2 --->
(V1A)^2 = (V2A)^2 + (V2B)^2
The energy of the balls is conserved, and if the masses of the balls are the same, the initial speed of the cue ball squared is equal to the final speed of the cue ball squared and the final speed of the ball it comes into contact with squared.

4. Friction Modifier:

$f = \mu\eta$

Balls that have a stored kinetic energy and momentum will be affected by the friction of the table's surface, slowing down the ball as it rolls around.

5. Wall Energy Absorption (Elastic/Inelastic Collisions):

mv' = mv-p

Where p represents the energy lost to the collision

As a ball hits the sides of the table, the impact makes it lose some of its stored kinetic energy, affecting the momentum. The impact absorbs some of the energy in the elastic collision. **Note** The energy lost to a collision will be able to be altered in sandbox mode. In the game mode, this will come preset with a value that feels "natural", and will be scaled based on the speed of the ball colliding with the wall (no exact value is given here because this value must be found experimentally).

6. Elevation (Gravity):

m*g*sin(Θ)

In sandbox mode, it is possible to have slopes placed around the table that can affect the ball's' trajectory. As a ball goes over a slope, it will either slow down if going upwards (against gravity), or speed up if the slope is downwards (pulled by gravity). A ball's velocity will be affected by the acceleration from gravity's pull multiplied by the ball's mass. To find the horizontal acceleration, trigonometry is used with relation to the slope angle.

Physics Schedule

The table below lists the functionalities for this project that need to be done, along with their estimated completion timeframe.

Physics Element	**Estimated Completion Timeframe**
Conservation of Linear Momentum Collision Detection	Week 5
Conservation of Elastic Kinetic Energy	Week 6
Friction Modifier Wall Energy Absorption (Elastic/Inelastic Collisions)	Week 7
Elevation (Gravity)	Week 8

Documentation

Throughout the entirety of the project keeping tabs on the project and the progress will be essential. When coding the program, commenting on every important section, will ensure that the next person working on the section will be able to know what is happening and be able to pick up the work where it was last left off. However, that will only be one part of the process, keeping up a full document on work being done on the code will be done on a separate Google document that every team member will have access to, and be able to modify at any given time. When any major modification is done, updates to the existing documentation will be altered to ensure that it is corrected, and with need be, the original documentation will be kept, to ensure if any other problems occur, the past problem and solution will be there to be used. Keeping track of big obstacles will be done as well, to show the progress of the project. Information will be kept easily acceptable so that anyone can be able to work on the project and get up to speed.

Team Members

Print Name: Vincent Bruzzese

Title: Student

Role: Graphical User Interface

Print Name: Daniel Cicciarelli

Title: Student

Role: Game Rules/ Logic, Interactions, and Graphs

Print Name: Karim Botros

Title: Student

Role: Physics Back-End

Print Name: Dylan Bobb

Title: Student

Role: Physics Back-End

References

Document Name	Description	Location
The Physics of Billiards	Reference Material for physics formulas	https://www.real-world-physics-problems.com/physics-of-billiards.html
Pool Hall Lessons: Fast, Accurate Collision Detection Between Circles or Spheres	Reference Material for collision detection and basic physics (linear momentum and energy loss)	http://www.gamasutra.com/view/feature/131424/pool_hall_lessons_fast_accurate_.php?
User Documentation	In depth description of how to use the software	A copy of this document can be found in the help menu of the application, or on the project's git repository hosted through BitBucket
Risk Log	Document containing risk assessments of all proposed changes to the PMP	A copy of this document can be found in the team's slack workspace or in the project's git repository hosted through BitBucket
Human Resources Log	Document containing any proposed changes and the results of their votes	A copy of this document can be found in the team's slack workspace or in the project's git repository hosted through BitBucket
Code Documentation	In depth description of all the methods of the code as well as a brief description of changes from the previous version (This document shall have several versions throughout the SDLC)	A copy of this document can be found in the team's slack workspace or in the project's git repository hosted through BitBucket