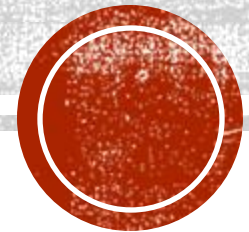
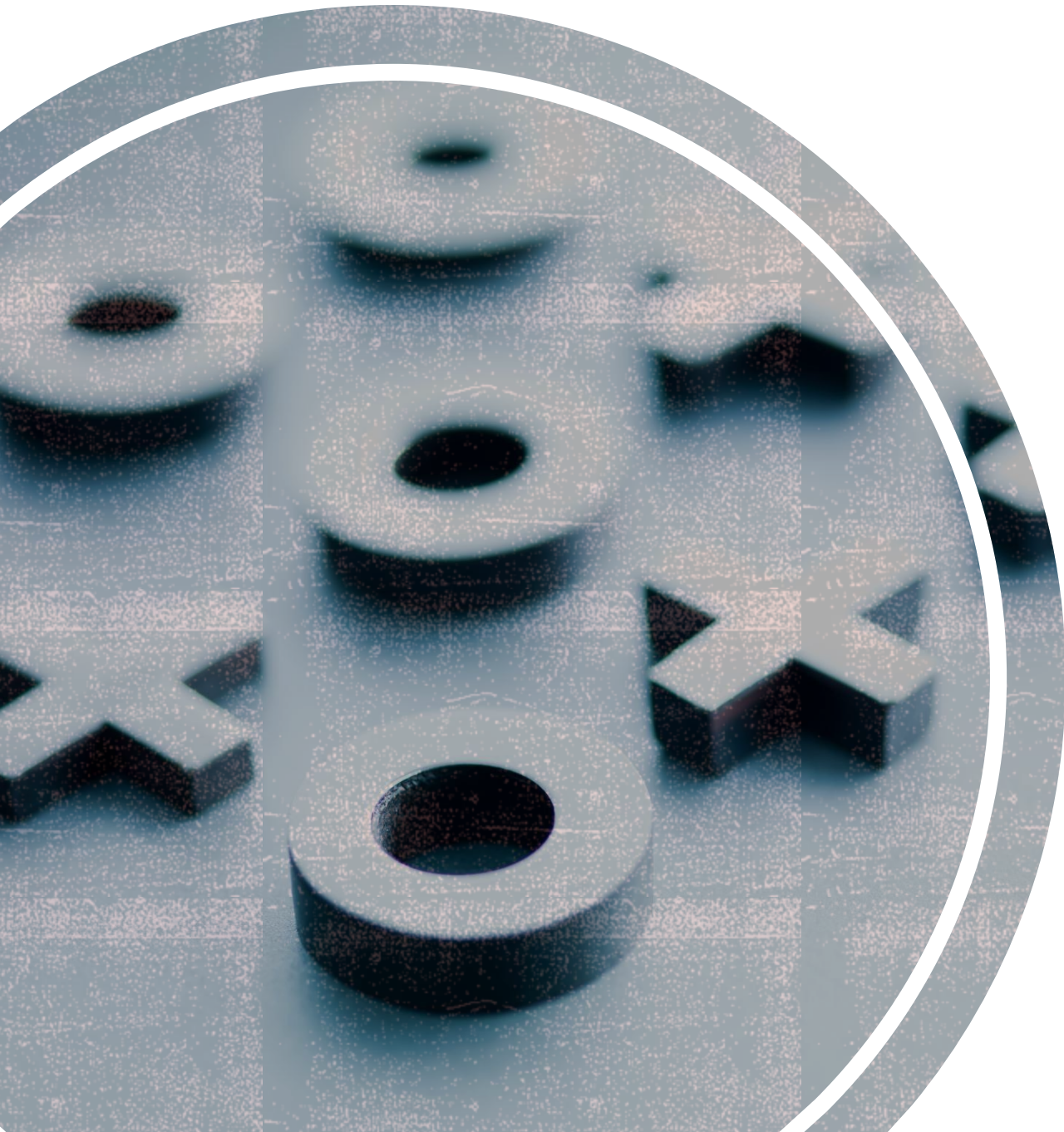

GROUP PRESENTATION — GRIDS GRANNIES AND GRATUITOUS VIOLENCE

Janusz Krakowiak, Kate
Leggott, Stuart Philpot,
Abigail Smith, Aidan
Wood





GAME CONCEPT

Our game is inspired by classic Battleship and equally as classic Angry Birds with our own stylised twist. The inspiration can be seen with the mapping (grid) of battleship and the movement mechanics of Angry Bird

Exchanging the war scene with a neighbourhood rivalry. Our game takes place in a suburban garden. With you throwing the objects over the fence in the hope you commit vandalism!



FUNCTIONS AND PARAMETERS

Ball

Targeting

Score


Enemy Player

Player feedback/text



BALL/PROJECTILE

We needed a way for the program to use projectile motion, we decided to do this by including a ball they could toss to determine what co-ordinated they hit



We did this by creating a ball that can be fired in the direction the user clicks in. This is done by calculating the distance of the click from the ball which determines the power of it and the angle of the click from the ball which determines how much power goes vertically vs horizontally



We then use this to determine information to update the balls position each frame, we also add download velocity at the same time to simulate gravity until the ball lands, this creates a satisfying projectile.



- We needed a way for the ball to result in co-ordinates for the player after it is fired, to do this we came up with a target hitting system.
- We did this by recording the horizontal position of the ball when it meets two requirements:
 - Firstly, checking if it's x co-ordinates are further than a wall that we made on the targeting screen allowing the users to recover from misclicks and encourages purposeful aiming
 - Then the ball with record its position when its Y position is just above the bottom of the screen.
- We decided which segment it has landed in by splitting the area the ball can land into 5 divided sections, we then compare the balls position to the segments and assign the segment number it has landed in to the Y co-ordinate the player will shoot



TARGETING



SCORE

- Our scoring system as the game progresses is measured by how many objects are broken.
- We do this by first checking if the x co-ordinate hit has any objects in it, if not then the score does not increase, and the player is informed of the miss. If there is an object, then the code checks for an object on the selected Y and X co-ordinates. If there is not object, then the same happens.
- If an object is located there then the player is told it's a hit and the number of hits is increased by one. If the number of hits is equal to 5 then the game recognizes that all objects are destroyed and displays a win message before resetting to menu



ENEMY PLAYER

- We needed another player to score against so we decided to make an enemy that would shoot back at the players objects.
- We did this by making the program generate two random numbers which are assigned at the computers X co-ordinate and Y co-ordinate guess for that turn. We then compare these against the players hit and the enemies score increments by one.
- When the enemy's score is equal to 5 then the player loses



PLAYER FEEDBACK

- We needed a way for potential players to know what they and each of the functions were doing without looking into the code itself.
- We did this by using blit and the draw shape commands in Pygame. Using these we created many buttons and texts that would be called to deliver feedback or info to the user.
- E.g. we created buttons and text at the beginning to tell the user to choose co-ordinates to place their objects. Along with a "hit" and "miss" text to deliver feedback after they fire their shot.



OVERVIEW OF SIMULATIONS



At the end of production our game runs to our expectations.



Initially the frame rate was thrown off so we corrected it by limiting the clock tick (`clock.tick(60)`) to 60. This allowed for a clear and smooth showing of our projectiles(Ball).



We tested code individually throughout the progress allowing for independent development of each function.





AREAS FOR IMPROVEMENT

- Validate user co-ordinates to prevent double placement of objects.
- While the computers objects are randomised. Validation is required to prevent this in future versions
- If the ball lands ontop of the fence it goes into it getting stuck. An if statement could be created to send the ball back to the initial projectile site.
- A way the running of the game could be improved is by including more environment building and action animations as the game feels still
- Can hit the same object over and over to reach the 5 point win score. Reattempt making the code to prevent reselection of co-ordinates



INDIVIDUAL CONTRIBUTION

- Abigail - Created the backbone of the code along with one other
- Stuart – Assisted with creating the backbone of the code. Frankensteined the project together ensuring everybody's contributions worked together.
- Kate – Created the code for the buttons. Created the scenes for the games and characters as well as putting together the final presentation for the group
- Janusz - Created the initial game logic along with coordinate registration
- Aiden – Progressed the game logic and combined it with the backbone

