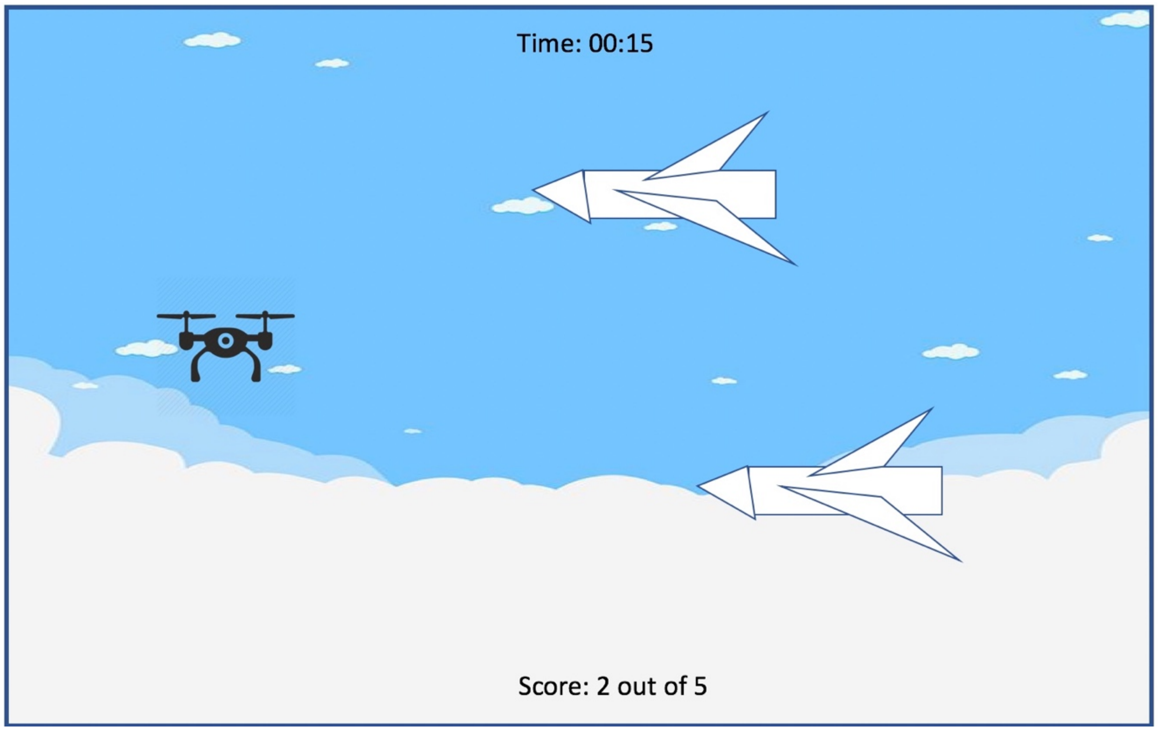
A User Controlled Drone

Storyline:

Our project is based on a simple drone game, which implements a user-controlled drone, airplanes, and a timer.

Like typical games, collisions between the drone and airplanes have a penality.



The user-controlled drone wins by passing through all the airplanes without running into any (i.e. by avoiding collisions). Each airplane is moved with a steady speed (1 pixel in the x-coordinate every some millisecond). The user can control the position of the y-coordinate using of the drone by using (UP arrow key) or (DOWN arrow key).

The airplanes array consists of anywhere from 1 to 6 airplanes . They have any speed from 1pixel per 100ms to 1000ms. The drone needs to avoid hitting airplanes in order to keep flying.

Drone motion:

The drone can avoid collision with airplanes by moving up, down, forward and backward. The drone has a fixed speed when the user is not using the keys to move it. The direction of the drone’s movement is from left to right

Airplane Motion:

The airplanes move from right to left using fixed speed. It appears at random y position on the far right of the screen on the Y-coordinate and start moving toward the left. They always start from a fixed position on the X-coordinate that is near the far right of the frame.

Timer

All the airplanes need to move in the scene. Use a timer to increment the x positions for the airplanes every certain ms.

Stopwatch

The time for the game is 1:30 minutes. If the drone does not hit more than 2 airplanes during the time, then the game finishes, the user’s score is incremented by one, and the time is reset.

Score

When the drone finishes its flying for 1:30 minutes with 2 collisions or less, the player score should increase one point indicating that the user wins the current game. If the drone hits more than 2 airplane within the 1:30 minutes time frame, then increment the total number of the games played only.

Collision

When a collision is detecting between the airplanes and the drone, there is a penalty of freezing the drone for 5 seconds while everything else moves. All the airplanes keep moving. This will make the game more challenging for the user.

Additional Implementation

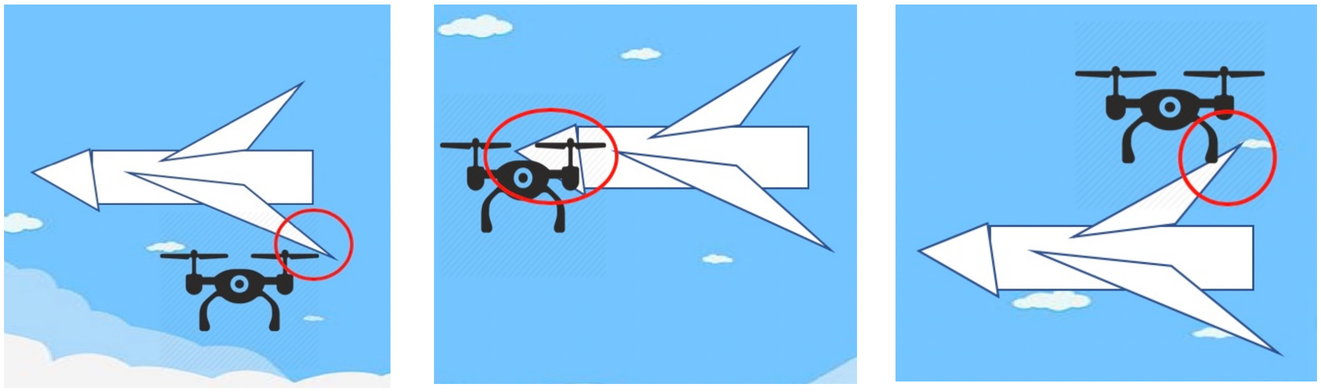
- Firing range for the drone is a limit to 1/4 the length of the screen.
- Airplane disappears completely when the drone shoots at it as they no longer pose a threat to the drone. Each airplane has its own velocity. Currently, we allow airplanes to pass through each other.
- A background image for a sky that covers the width and the height of the screen

Approach

The Airplane Field

The airplane field class uses an array to keep track of all the airplanes. You can define procedures to allow you to accurately predict collisions with airplane. These fields also are updated if an airplane is destroyed by the drone.

Collision Detection



Collision Detection Optimization

We can start the collision detection method using JComponent for drone and airplane. This approach is a much faster method for collision detection than using hyperplanes of convex polygons. Determine if the drone enters into the boundaries of another airplane. If so, it is a collision.

Firing into a Field of Airplane

Shooting at airplanes requires yet another form of collision detection: checking for the intersection of a line (laser beam or bullets) with the component (airplane). This involves finding the intersection of a beam line with the airplanes.

You can use the spacekey on the keyboard or implement a JButton key for firing the laser beam/bullet.

User-Controlled Drone:

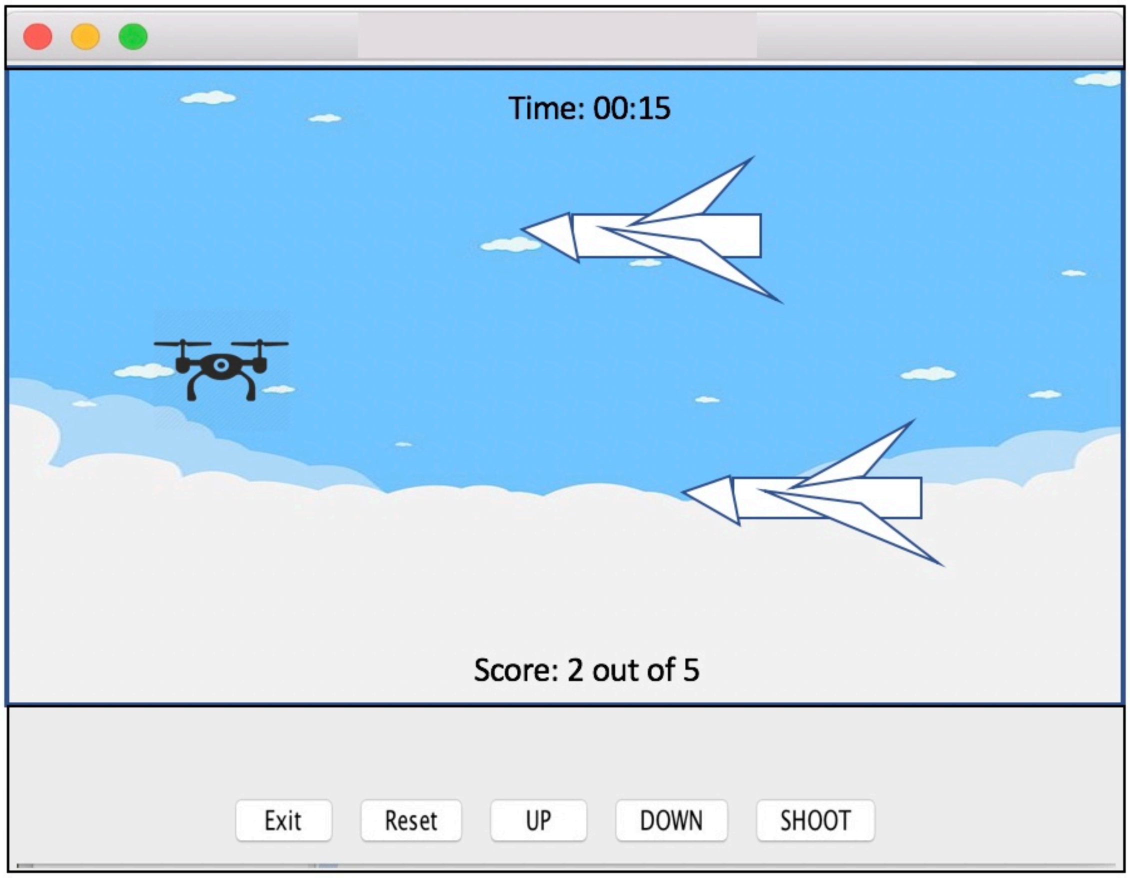
You have two ways to move the drone. Choose one and implement it in your code:

- **You can control the drone using keyboard** up arrow key for moving the drone up in the window and down arrow key to move the drone down
- **Another way is to have two JButton** keys as shown in the figure below to move the drone

The Code

Our project is written in Java, using object oriented development. The code is logically separated into the following possible classes (feel free to have more/less classes).

- DroneGame.java (the main class)
- DPoint.java, MovingPlain.java, Drone.java, Airplane.java, ConfigurationSpace.java, Timer.java, Scores.java, etc.



Have fun!