PyGame Tractor Pong Tutorial - Part 3

Contents

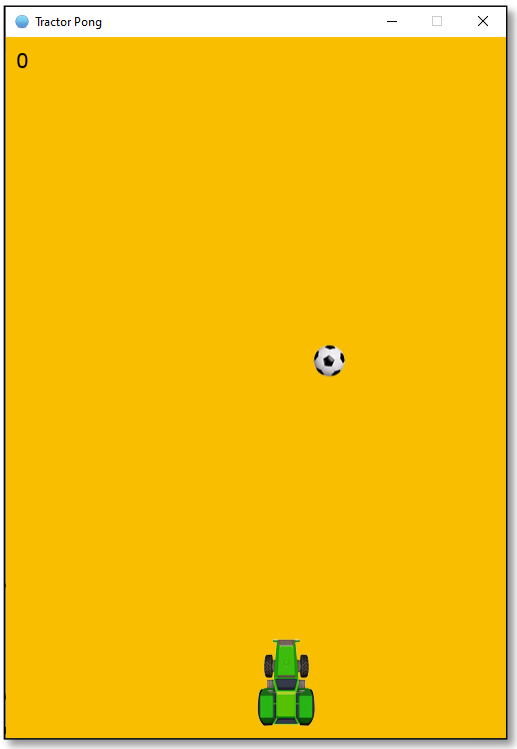
Time required: 30 minutes

Preview of the Game

Atari. - the year: 1973 - the date: - November 29th -

That game is called Pong . . . . Then there was Tractor Pong.

Tractor Pong Demo Video



Time to Bounce

1. Save **tractor_pong_2.py** as **tractor_pong_3.py**
2. Modify the following code.

```
15  class TractorPong:
16      def __init__(self):
17          # Initialize the pygame library
18          pygame.init()
19
20          # Create the game surface (window)
21          self.surface = pygame.display.set_mode(
22              (config.WIDTH, config.HEIGHT)
23          )
24
25          # Set window caption
26          pygame.display.set_caption("Tractor Pong")
27
28          # Setup computer clock object to control the speed of the game
29          self.clock = pygame.time.Clock()
30
31          # Load the ball image from the file system into a variable
32          self.ball = pygame.image.load(
33              "assets/soccer_ball.png").convert_alpha()
34
35          # Create a rectangle the same size as the ball
36          # rect is used to set the location of the ball
37          self.ball_rect = self.ball.get_rect()
38
39          # Initial postion of the ball rectangle x random, y/top = 10
40          self.set_ball_location()
41          self.ball_rect.y = 10
42
43          # Ball speed in pixels for x, y
44          self.set_ball_direction()
45          self.speed_y = 3
```

Randomization is a way to make a game more interesting. The ball will randomly appear at a different horizontal location and direction.

Add these methods.

```
47  # ------------------------ SET BALL LOCATION ------------------------------#
48      def set_ball_location(self):
49          """Set random initial ball direction along the x axis"""
50          # Randomly determine the initial x coordinate of the ball
51          # along the x-axis (left or right)
52          self.ball_rect.x = randint(20, config.WIDTH - 20)
```

self.ball_rect.x = randint(20, config.WIDTH - 20) - This line sets the x-coordinate of the ball's position (`self.ball_rect.x`) to a random value between 20 pixels (to ensure it's not too close to the edges) and `config.WIDTH - 20` pixels (to ensure it's not too close to the right edge of the screen).

This effectively initializes the ball's position along the x-axis randomly within the game boundaries.

```
54  # ------------------------ SET BALL DIRECTION ------------------------------#
55      def set_ball_direction(self):
56          """Set random initial ball direction along the x axis"""
57          # Randomly determine the initial direction of the ball
58          # along the x-axis (left or right)
59          ball_direction_x = randint(0, 1)
60
61          # If the randomly chosen direction is 0 (left),
62          # set the horizontal speed of the ball to move to the right
63          if ball_direction_x == 0:
64              self.speed_x = 3
65
66          # If the randomly chosen direction is 1 (right),
67          # set the horizontal speed of the ball to move to the left
68          else:
69              self.speed_x = -3
```

ball_direction_x = randint(0, 1) - This line generates a random integer either 0 or 1, representing the initial direction of the ball along the x-axis (left or right).

if ball_direction_x == 0: - This line checks if the randomly chosen direction is 0, indicating that the ball should move to the right.

self.speed_x = 3 - If the ball is meant to move to the right, this line sets the horizontal speed **self.speed_x** to 3, indicating movement towards the right.

else: - If the randomly chosen direction is not 0 (i.e., it's 1), indicating that the ball should move to the left.

self.speed_x = -3 - This line sets the horizontal speed **self.speed_x** to -3, indicating movement towards the left.

```
59  # --------------------------- GAME LOOP ---------------------------------#
60      def game_loop(self):
61          """Infinite game loop"""
62          while True:
63              self.check_events()
64
65              # ------------------- UPDATE BALL ----------------------------#
66              # Move the ball position every frame
67              self.ball_rect.x = self.ball_rect.x + self.speed_x
68              self.ball_rect.y = self.ball_rect.y + self.speed_y
69
70              # --------------- DRAW ON BACKBUFFER -------------------------#
71              # Draw everything on the backbuffer first
72              # Fill the surface with Cougar Gold
73              self.surface.fill(config.COUGAR_GOLD)
74
75              # Draw the ball on the surface
76              self.surface.blit(
77                  self.ball,       # Image to draw
78                  self.ball_rect   # Location to draw the image
79              )
80
81              # -------------- UPDATE SURFACE -------------------------------#
82              # From backbuffer, update Pygame display to reflect any changes
83              pygame.display.update()
84
85              # Cap game speed at 60 frames per second
86              self.clock.tick(60)
87
88
89  # Create game instance
90  tractor_pong = TractorPong()
91  # Start the game
92  tractor_pong.game_loop()
```

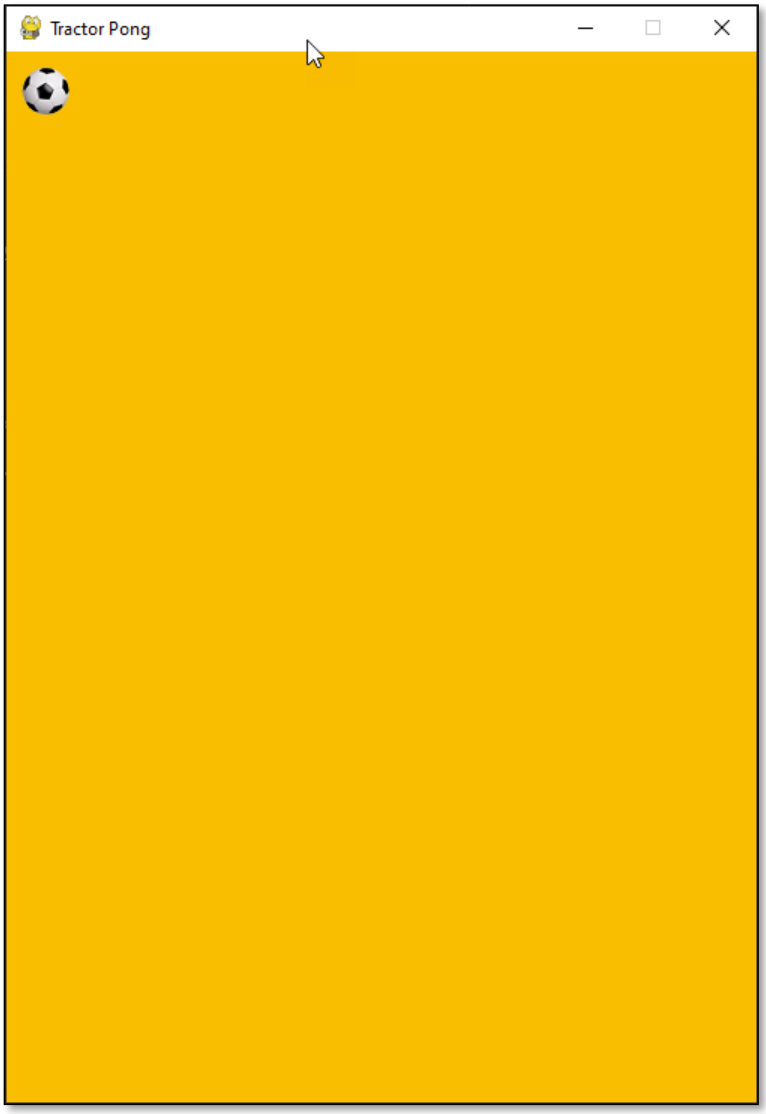
This code updates the position of the ball in the game by adding its current speed along the x and y axes to its current position.

self.ball_rect.x = self.ball_rect.x + self.speed_x - This line updates the x-coordinate of the ball's position (self.ball_rect.x) by adding its current x-axis speed

(self.speed_x). If **self.speed_x** is positive, it moves the ball towards the right; if negative, towards the left.

self.ball_rect.y = self.ball_rect.y + self.speed_y - This line updates the y-coordinate of the ball's position (self.ball_rect.y) by adding its current y-axis speed (self.speed_y). If **self.speed_y** is positive, it moves the ball downwards; if negative, upwards.

These lines update the game's state in each frame, causing the ball to move continuously based on its current speed along the x and y axes.



The ball bounces around the screen off the walls.

Assignment Submission

1. Attach all tutorials and assignments.
2. Attach screenshots showing the successful operation of each tutorial program.
3. Submit in Blackboard.