بسمه تعالی

سیستم‌های نهفته

گزارش پروژه

بازی تانک دو نفره

استاد

دکتر انصاری

نویسنده

علیرضا هنرور (۹۸۱۰۲۵۵۱)

سیدعلی نجیبی (۹۸۱۰۶۱۲۳)

دانشگاه صنعتی شریف

نیم‌سال دوم ۱۴۰۲ – ۱۴۰۱

مقدمه

در این پروژه قصد داریم بازی تانک دو نفره را با استفاده از Raspberry Pi و سنسورهای حساس به صدا و فاصله، پیاده‌سازی کنیم. سنسور حساس به صدا، مربوط به تعیین زاویه‌ی پرتاب، و سنسور حساس به فاصله، مربوط به تعیین شدت پرتاب می‌باشد. بازی با استفاده از زبان برنامه‌نویسی Python نوشته شده، و لوازم استفاده‌شده در پروژه به صورت زیر می‌باشد:

- Raspberry Pi
- Breadboard
- MF Cables
- MM Cables
- Ultrasonic Sensor
- Sound Sensor
- Right and Left Keys (Bonus)

هدف پروژه

هدف در این پروژه، پیاده‌سازی بازی تانک دو نفره می‌باشد. این بازی نوبتی می‌باشد، و نوبت هر شخص که باشد، با دکمه چپ و راست اجازه حرکت دارد (اختیاری و بدون نیاز)، و ابتدا برای تعیین زاویه‌ی پرتاب بازیکن، یک خط‌چین در حال تغییر زاویه می‌باشد و با تشخیص صدا توسط سنسور، زاویه خود را ثابت می‌کند، و سپس به سراغ تعیین شدت پرتاب می‌شود. برای این کار، سنسور تشخیص فاصله، فاصله ابتدایی و فاصله انتهایی را می‌سنجد و تقسیم بر زمان کل می‌کند تا سرعت پرتاب بدست آید. به عنوان مثال، بازیکن ۳ ثانیه فرصت برای تعیین قدرت پرتاب دارد. در نتیجه تفاضل فاصله انتهایی و ابتدایی دست از سنسور تشخیص فاصله، تقسیم بر ۳ (ثانیه) می‌شود. اگر تیر به بازیکن مقابل برخورد کند، جان آن کاهش پیدا می‌کند تا اینکه یک بازیکن تماماً جانش تمام شود و ببازد.

توضیح کد و روند انجام‌شده

```python
def run(self):
    running = True
    clock = pygame.time.Clock()

    turn = 0
    turn_array: List[Tank] = [
        Tank("tank2.png", 100, GAME_HEIGHT - TANK_HEIGHT // 2, 45, self.ground_points),
        Tank("tank1.png", GAME_WIDTH - 100, GAME_HEIGHT - TANK_HEIGHT // 2, -45, self.ground_points)
    ]

    state = GameState.CHOOSE_ANGLE
    aim_indicator = AimIndicator(turn_array[turn])

    bullet = None

    self.sound_detector.turn_on()

    while running:
        clock.tick(FPS)

        self.window.fill(BLACK)
        pygame.draw.polygon(self.window, GREEN, self.ground_points)
```

در کد بازی، ابتدا تصاویر و مکان بازیکن‌ها لود می‌شود. سپس، در یک حلقه‌ی while می‌رویم، و بازی شروع می‌شود. درون بازی، یک ساعت داریم که زمان و FPS بازی را در نظر می‌گیرد. سپس عامل سیاه بودن آسمان و سبز بودن زمین بازی اعمال می‌شود. توجه کنید که دلیل وجود این قطعه در حلقه while این است که هنگام نمایش نماد تعیین زاویه و گلوله و ...، باید نمایش‌های قبلی از بین بروند و تنها یک گلوله در هر ثانیه قابل مشاهده باشد. در نتیجه هر لحظه آسمان را سیاه می‌کنیم، تا تنها گلوله رسم‌شده در آن لحظه مشاهده شود. همچنین، در ابتدا، تشخیص صدا برای بازیکن اول فعال می‌شود.

حال، به بخشی از کد می‌رسیم که چندین if و else if وجود دارد و وضعیت یا state را چک می‌کنیم و بر اساس آن یک کار انجام می‌دهیم. وضعیت‌ها به شکل زیر می‌باشند:

```
if state == GameState.CHOOSE_ANGLE or state == GameState.CHOOSE_POWER:
    if keys[pygame.K_LEFT]:
        turn_array[turn].left()
    if keys[pygame.K_RIGHT]:
        turn_array[turn].right()
```

این وضعیت، صرفا جهت حرکت روی زمین می‌باشد و لزومی ندارد اجرا شود.

```
if state == GameState.CHOOSE_ANGLE:
    if self.sound_detector.is_sound_detected():
        self.sound_detector.turn_off()
        self.speed_detector.turn_on()
        state = GameState.CHOOSE_POWER

    aim_indicator.indicate(self.window)
```

این وضعیت، تشخیص زاویه را انجام می‌دهد و از Sound Sensor کمک می‌گیرد.

```
elif state == GameState.CHOOSE_POWER:
    if self.speed_detector.is_speed_detected():
        bullet = Bullet(turn_array[turn].x, turn_array[turn].y,
                        aim_indicator.get_shoot_angle(),
                        self.speed_detector.detected_speed,
                        self.ground_points)
        self.speed_detector.turn_off()
        state = GameState.SHOOT
    aim_indicator.draw(self.window)
```

این وضعیت، تشخیص قدرت را انجام می‌دهد و از Ultrasonic Sensor کمک می‌گیرد.

```
        elif state == GameState.SHOOT:
            if bullet.hit:
                turn = (turn + 1) % 2
                self.sound_detector.turn_on()
                aim_indicator = AimIndicator(turn_array[turn])
                turn_array[turn].check_hit(bullet)
                state = GameState.CHOOSE_ANGLE

            else:
                bullet.fire(self.window)
                self.indicate_power(bullet.power)
```

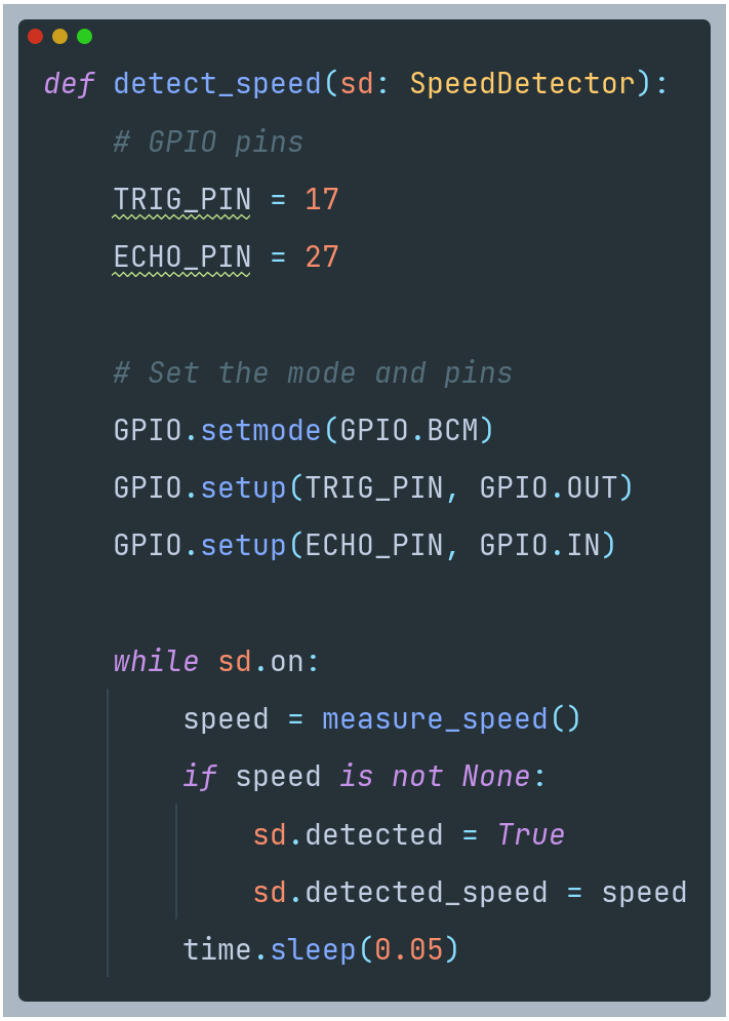
این وضعیت، جهت انجام انیمیشن پرتاب تیر می‌باشد و عملیات منطق حرکت تیر به سمت حریف را نیز دربر می‌گیرد.

```
for tank in turn_array:
    tank.draw(self.window)

pygame.display.flip()
```

در نهایت، باید وضعیت فعلی تانک‌ها در این لحظه را رسم کنیم چرا که ممکن است حرکت کرده باشند و یا جان‌شان تغییر کرده باشد.

حال سراغ بررسی Sound Detection و Distance Detection می‌رویم. این دو قابلیت، به وسیله قابلیت Thread پیاده‌سازی می‌شوند:

```
def detect_sound(detector: SoundDetector):
    # GPIO SETUP
    channel = 17
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(channel, GPIO.IN)

    Ali Najibi
    def callback(channel):
        if GPIO.input(channel):
            detector.detected = True
        else:
            detector.detected = True

    GPIO.add_event_detect(channel, GPIO.BOTH, bouncetime=300)  # let us know when the pin goes HIGH or LOW
    GPIO.add_event_callback(channel, callback)  # assign function to GPIO PIN, Run function on change

    # infinite loop
    while detector.on:
        time.sleep(1)
```

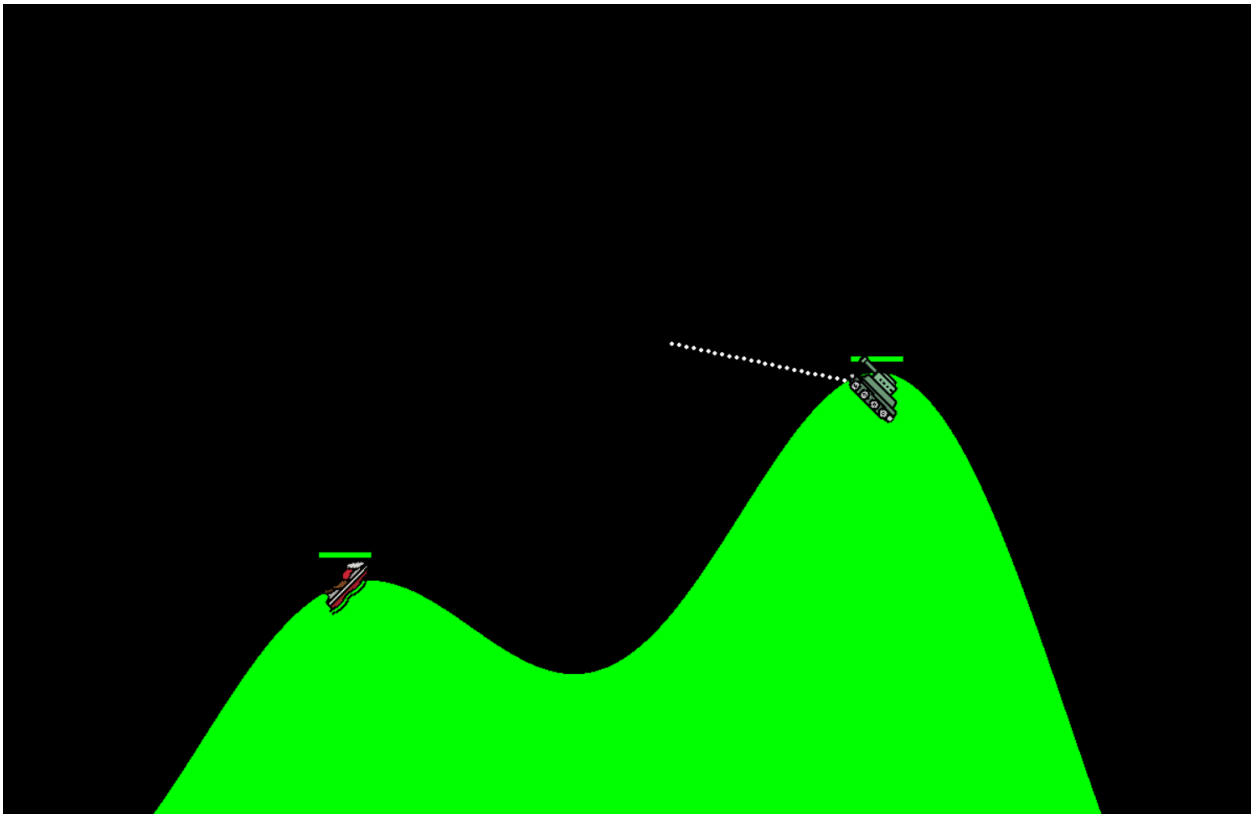


```python
def detect_speed(sd: SpeedDetector):
    # GPIO pins
    TRIG_PIN = 17
    ECHO_PIN = 27

    # Set the mode and pins
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(TRIG_PIN, GPIO.OUT)
    GPIO.setup(ECHO_PIN, GPIO.IN)

    while sd.on:
        speed = measure_speed()
        if speed is not None:
            sd.detected = True
            sd.detected_speed = speed
        time.sleep(0.05)
```

```python
def measure_distance():
    # Send a short pulse to trigger the ultrasonic signal
    GPIO.output(TRIG_PIN, True)
    time.sleep(0.00001)
    GPIO.output(TRIG_PIN, False)

    # Wait for the echo signal
    pulse_start = time.time()
    while GPIO.input(ECHO_PIN) == 0:
        pulse_start = time.time()

    pulse_end = time.time()
    while GPIO.input(ECHO_PIN) == 1:
        pulse_end = time.time()

    # Calculate the duration of the pulse
    pulse_duration = pulse_end - pulse_start

    # Speed of sound at 20 degrees Celsius (343 meters/second)
    speed_of_sound = 343.0

    # Calculate the distance (round-trip)
    distance = (pulse_duration * speed_of_sound) / 2.0

    return distance
```

```python
def measure_speed():
    # Get the initial distance
    initial_distance = measure_distance()

    # Wait for some time (e.g., 1 second)
    time.sleep(0.01)

    # Get the final distance
    final_distance = measure_distance()

    # Calculate the speed
    speed = abs(final_distance - initial_distance) / 1.0  # Change in distance per second

    return speed
```

نتایج بدست آمده



در این بازی، همانطور که مشاهده می‌شود، یک زمین تولید می‌شود. بازیکن‌ها قابلیت حرکت دارند (امتیازی). جهت و شدت پرتاب مشخص می‌شود. و همچنین در نهایت بازیکنی که جانش تمام می‌شود، می‌بازد.