Project4: Interactive Drawing & Animation

Air Hockey... but it's Pinball

20191170 한의현

Content

1. Switching Lanes
2. Making Objects
3. Respawn, Game Over, and Reset
4. Recording Time
5. Designing Interface
6. Gameplay Screenshots

Switching Lanes

Camera + Balls Collision = Too Slow

Non-TPS top-down dodging game? BORING → changed concept!

Pool doesn't work like that anyway

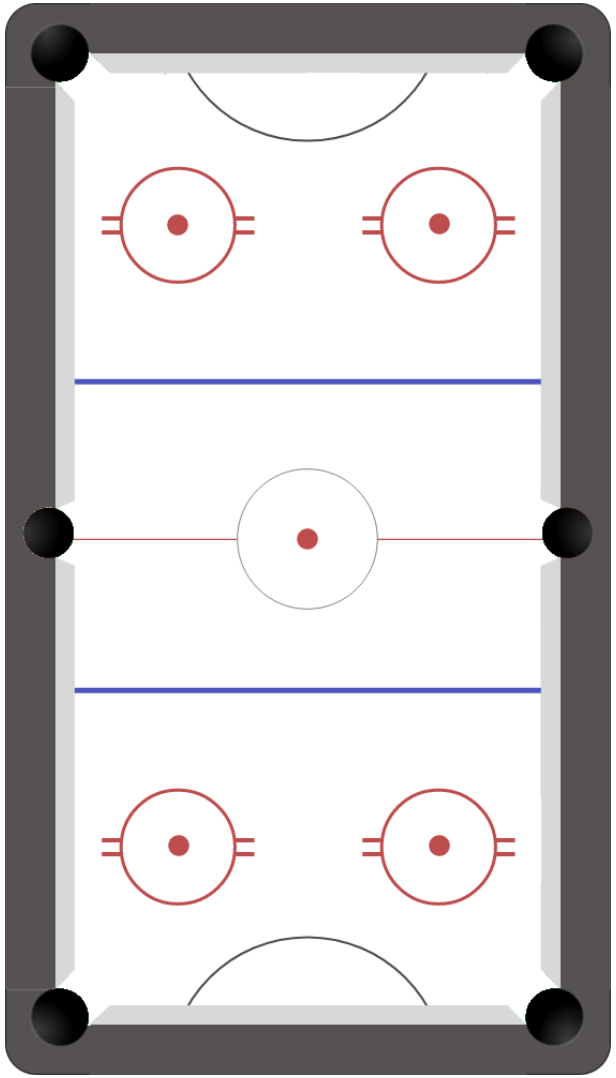
Switching Lanes

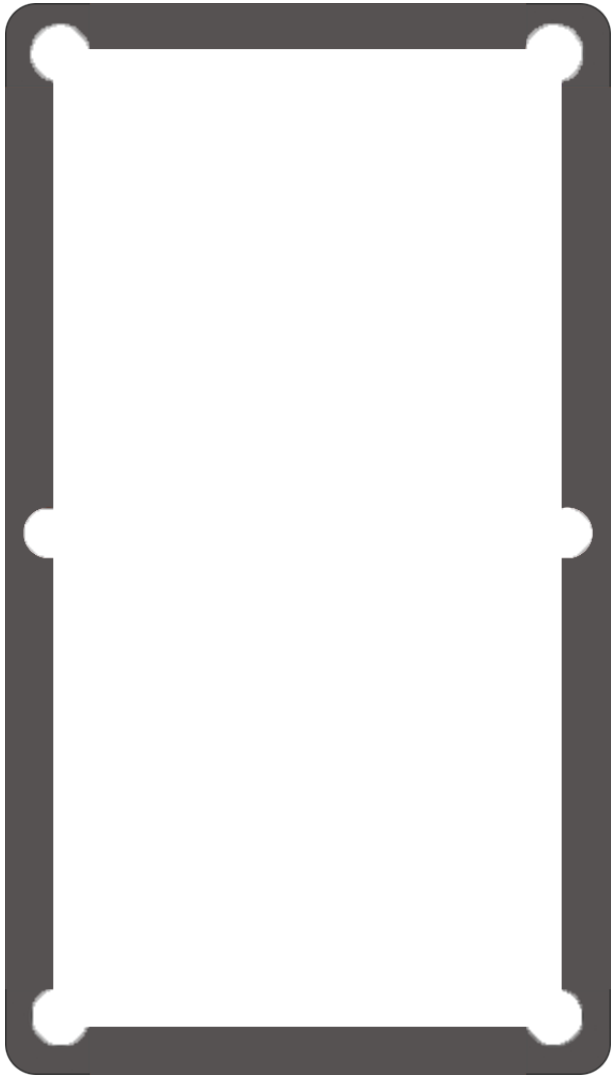
+

opposite balls act as pinball obstacles now

the more you collide and the longer you survive, final score gets higher

Switching Lanes





this one's called 'striker'

pucks are just... circles

For Puck's Sake!

하키에서 쓰는 공을 퍽이라 부른다
for one's sake는 '~를 위해' 같은 의미로
이를 풀이하면 '하키 퍽을 위하여!'가 된다.

절대 욕 같은게 아니다.

Making Objects

```
26    space = pymunk.Space()
27    static_body = space.static_body
```

```
43    def puck_striker(radius, pos):
44        body = pymunk.Body()
45        body.position = pos
46        shape = pymunk.Circle(body, radius)
47        shape.mass = 1500
48        shape.elasticity = 1
49        space.add(body, shape)
50        return shape
51
52    def puck(radius, pos):
53        body = pymunk.Body()
54        body.position = pos
55        shape = pymunk.Circle(body, radius)
56        shape.mass = 5
57        shape.elasticity = 1.01
58        space.add(body, shape)
```

pymunk의 물리법칙이 적용되는 차원을 만든다.

space 내에 물체(body)는 충돌 때마다 부딪힌

상대 물체와 질량, 속도를 비교하여 값을 변화시킨다.

따로 충돌 조건을 만들어줄 필요가 없어 코드가 가벼워짐

puck_striker 는 사용자가 조작하기 때문에

고유 속도 값이 없어서 질량을 크게 잡음

puck의 탄성계수는 1.01로, 충돌 시마다 1%만큼 속도가

증가함

Making Objects

```
60    pucks = []
61    puck_r = 18
62    rows = 5
63    for col in range(5):
64        for row in range(rows):
65            pos = (215 + (row * (puck_r * 2 + 1)) + (col * puck_r), 200 + (col * (puck_r
              * 2 + 1)))
66            new_puck = puck(puck_r, pos)
67            pucks.append(new_puck)
68        rows -= 1
69    white_puck = puck(puck_r, (WINDOW_WIDTH / 2, 740))
70    pucks.append(white_puck)
```

puck의 초기 위치를 설정하여 리스트에 저장함 (ball 101개 만들 때 한 것 처럼)

white_puck이 pinball에 해당하는 퍽, 게임 상에서 검은색으로 바꿨는데 깜빡하고 변수명은 걍 냅둠

Making Objects

```
72  def cushion(poly_dims):
73      body = pymunk.Body(body_type = pymunk.Body.STATIC)
74      body.position = ((0, 0))
75      shape = pymunk.Poly(body, poly_dims)
76      shape.elasticity = 0.99
77      space.add(body, shape)
78
79  cushions = [
80      [ (47, 75), (47, 471), (64, 464), (64, 92) ],
81      [ (500, 92), (500, 464), (518, 471), (518, 75) ],
82      [ (47, 519), (47, 918), (64, 900), (64, 526) ],
83      [ (500, 527), (500, 900), (518, 918), (518, 518) ],
84      [ (80, 47), (99, 65), (467, 65), (484, 47) ],
85      [ (99, 935), (81, 953), (484, 953), (467, 935)]
86  ]
```

puck의 elasticity를 1.01로 올리면 10초만 지나도 너무 빨라지므로 쿠션은 0.99로 하여 밸런스 맞춤 (그러나 퍽끼리 부딪히는 빈도가 훨씬 많기에 결과적으론 빨라짐) -- 점수 올리는 전략으로도 사용할 수 있음

Making Objects

```
220         for i, puck in enumerate(pucks[:-1]):
221             pygame.draw.circle(screen, RED, (puck.body.position[0], puck.body.
                position[1]), puck_r, 0)
222         # portal visualization
223         # pygame.draw.circle(screen, SKYBLUE, (pucks[9].body.position[0], pucks[9].
            body.position[1]), puck_r, 0)
224         # pygame.draw.circle(screen, GREEN, (pucks[2].body.position[0], pucks[2].
            body.position[1]), puck_r, 0)
225         pygame.draw.circle(screen, BLACK, (pucks[-1].body.position[0], pucks[-1].
            body.position[1]), puck_r, 0)
```

핀퍽을 제외한 퍽은 RED, 핀퍽은 BLACK으로 색상을 설정하고 pygame loop 밖에서 생성한 body들의 좌표를 부여한다. (주석 처리한 부분은 리스폰이 되는 것을 시각적으로 확인하는 코드)

Making Objects

```
227         striker_r = 25
228         if initial == True:
229             show_init_screen()
230             striker = puck_striker(striker_r, pygame.mouse.get_pos())
231             initial_snd.play()
232             pucks[-1].body.apply_impulse_at_local_point((0, 500 * -15), (0, 0))
233             initial = False
234         else:
235             temp_time = pygame.time.get_ticks() / 1000
236 
237         striker.body.position = pygame.mouse.get_pos()
238         screen.blit(striker_img, (striker.body.position[0] - striker_r, striker.body.
            position[1] - striker_r))
```

핀퍽은 게임 시작 직후에 1회 –y 방향의 힘을 받고 속도를 가져, 나머지 퍽들을 쳐서 당구처럼 게임이 시작

striker는 별도의 초기 위치가 정해져 있지 않아 loop 밖에서 정의가 불가한데 loop 돌 때마다 새로 생성되는 것을 방지하기 위해 initial == True 조건일 때 마찬가지로 1회 생성

Respawn, Game Over, and Reset

```
92    pockets = [ (50, 43), (40, 495), (50, 946), (512, 946), (525, 495), (512, 43) ]
93    respawn = [ (477, 911), (477, 494), (477, 88), (87, 88), (87, 494), (87, 911) ]
```

```
184             for p in pucks[:-1]:
185                 for i in range(len(pockets)):
186                     pocket = pockets[i]
187                     if dist(p, pocket) <= pocket_r:
188                         p.body.position = respawn[i]
189
190                 if dist(p, (WINDOW_WIDTH / 2, WINDOW_HEIGHT / 2)) > 1149:
191                     p.body.position = (
192                         np.random.randint(low=87, high=477),
193                         np.random.randint(low=88, high=911)
194                         )
195                 elif dist(p, pucks[-1].body.position) <= pocket_r * 2 + 1:
196                     score += 3 * multiplier
197                     hit_snd.play()
```

퍽과 포켓의 거리가 포켓 반지름 미만이면 공빠짐 판정,

각 포켓마다 미리 정해둔 리스폰 좌표로 순간이동

(하지만 속도는 유지되므로 나온 것처럼 보임)

간혹 퍽 속도가 너무 빠를 때 쿠션 뚫고

날아가버리는 경우, 랜덤 리스폰

핀퍽(?)과 충돌 시 점수 3점씩 추가, 효과음 발생

Respawn, Game Over, and Reset

```
199             for i in range(len(pockets)):
200                 pocket = pockets[i]
201                 if dist(pucks[-1], (WINDOW_WIDTH / 2, WINDOW_HEIGHT / 2)) > 1149 or dist
                    (pucks[-1], pocket) <= pocket_r:
202                     jackpot_snd.play()
203                     time = temp_time - time
204                     score_list[tries] = score
205                     show_go_screen()
206                     waiting = True
207                     while waiting:
208                         clock.tick(60)
209                         for event in pygame.event.get():
210                             if event.type == pygame.QUIT:
211                                 done = gameover = True
212                             elif event.type == pygame.MOUSEBUTTONUP:
213                                 waiting = False
214                                 gameover = True
```

핀퍽이 포켓에 빠지거나 쿠션을 뚫으면 게임오버 판정

잭팟 효과음을 발생하고 측정한 시간과 점수를 넘김

게임 오버를 다루는 별도 함수에서 이를 사용

Respawn, Game Over, and Reset

```
243     pymunk.Space.remove(space, striker)
244     tries += 1
245
246     reset = []
247     rows = 5
248     for col in range(5):
249         for row in range(rows):
250             pos = (215 + (row * (puck_r * 2 + 1)) + (col * puck_r), 200 + (col *
                (puck_r * 2 + 1)))
251             reset.append(pos)
252         rows -= 1
253     reset.append((WINDOW_WIDTH / 2, 740))
254
255     for p in range(len(pucks)):
256         pucks[p].body.velocity = (0, 0)
257         pucks[p].body.position = reset[p]
```

이전 판에 존재했던 striker를 제거한다.

모든 퍽의 속도를 0으로 하여 정지시킨 뒤 리셋 좌표로 이동시킨다.

Recording Time

```
228         if initial == True:
229             show_init_screen()
230             striker = puck_striker(striker_r, pygame.mouse.get_pos())
231             initial_snd.play()
232             pucks[-1].body.apply_impulse_at_local_point((0, 500 * -15), (0, 0))
233             initial = False
234         else:
235             temp_time = pygame.time.get_ticks() / 1000
```

```
199         for i in range(len(pockets)):
200             pocket = pockets[i]
201             if dist(pucks[-1], (WINDOW_WIDTH / 2, WINDOW_HEIGHT / 2)) > 1149 or dist
                (pucks[-1], pocket) <= pocket_r:
202                 jackpot_snd.play()
203                 time = temp_time - time
```

시작 화면이 끝나면 (initial == False), 그 때부터 temp_time이 기록되며, 게임 오버 때마다 초기값이 0인 time에 이전에 측정된 것만큼 빼서 업데이트

Designing Interface

```
115  def show_init_screen():
116      screen.blit(table_img, (0, 0))
117      draw_text(screen, "FOR PUCK'S SAKE", 64, WINDOW_WIDTH / 2, WINDOW_HEIGHT / 4,
         BLACK)
118      if tries == 0:
119          draw_text(screen, "Move your mouse to control", 22, WINDOW_WIDTH / 2,
             WINDOW_HEIGHT / 2, BLACK)
120      else:
```

```
133      draw_text(screen, "Click to begin", 18,
         BLACK)
134      pygame.display.update()
```

```
draw_text(screen, "HIGH SCORE", 26, WINDOW_WIDTH / 2, WINDOW_HEIGHT / 2 -
65, BLACK)
draw_text2(f"1st", 22, WINDOW_WIDTH / 2 - 71, WINDOW_HEIGHT / 2 - 30,   GOLD)
draw_text2(f"2nd", 22, WINDOW_WIDTH / 2 - 71, WINDOW_HEIGHT / 2,      SILVER)
draw_text2(f"3rd", 22, WINDOW_WIDTH / 2 - 71, WINDOW_HEIGHT / 2 + 30, BRONZE)
draw_text2(f"4th", 22, WINDOW_WIDTH / 2 - 71, WINDOW_HEIGHT / 2 + 60,  BLACK)
draw_text2(f"5th", 22, WINDOW_WIDTH / 2 - 71, WINDOW_HEIGHT / 2 + 90,  BLACK)
draw_text3(screen, f"{sorted(score_list, reverse=True)[0]}", 22,
WINDOW_WIDTH / 2 + 67, WINDOW_HEIGHT / 2 - 30,   GOLD)
draw_text3(screen, f"{sorted(score_list, reverse=True)[1]}", 22,
WINDOW_WIDTH / 2 + 67, WINDOW_HEIGHT / 2,      SILVER)
draw_text3(screen, f"{sorted(score_list, reverse=True)[2]}", 22,
WINDOW_WIDTH / 2 + 67, WINDOW_HEIGHT / 2 + 30, BRONZE)
draw_text3(screen, f"{sorted(score_list, reverse=True)[3]}", 22,
WINDOW_WIDTH / 2 + 67, WINDOW_HEIGHT / 2 + 60,  BLACK)
draw_text3(screen, f"{sorted(score_list, reverse=True)[4]}", 22,
WINDOW_WIDTH / 2 + 67, WINDOW_HEIGHT / 2 + 90,  BLACK)
```

첫 플레이 때 (tries == 0) 조작 방법을 알려주고,

2회차 부터는 이전까지의 플레이 점수 중 상위 5개를 보여준다. 0 리스트 score_list의 tries 인덱스 값을 이전에 측정한 점수로 바꾸고 sorted 사용.

Gameplay Screenshots

-마우스 이동과 클릭만으로 게임이 가능-