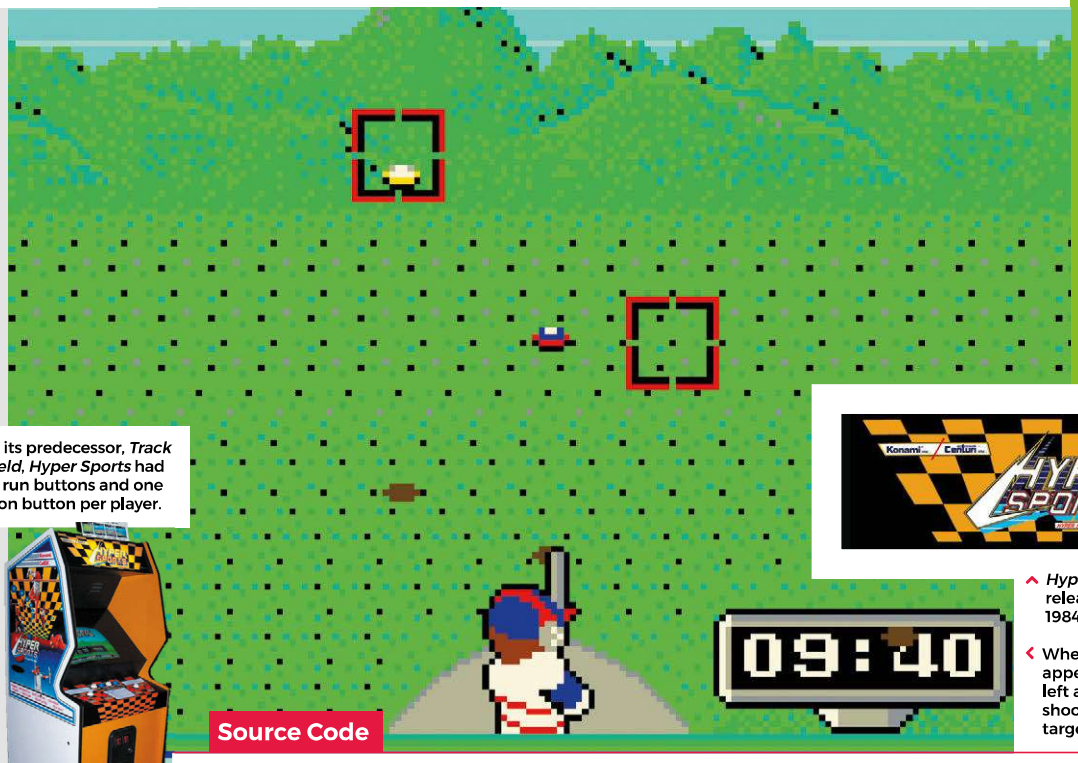

Like its predecessor, *Track & Field*, *Hyper Sports* had two run buttons and one action button per player.

Hyper Sports' Japanese release was tied in with the 1984 Summer Olympics.

When the clay targets appear, the player uses the left and right buttons to shoot either the left or right target respectively.

Source Code

Code Hyper Sports' shooting minigame

Gun down the clay pigeons in our re-creation of a classic minigame from Konami's Hyper Sports

AUTHOR
MARK VANSTONE

Konami's sequel to its 1983 arcade hit, *Track & Field*, *Hyper Sports* offered seven games – or events – in which up to four players could participate. Skeet shooting was perhaps the most memorable game in the collection, and required just two buttons: fire left and fire right. The display showed two target sights, and each moved up and down to come into line with the next clay disc's trajectory. When the disc was inside the red target square, the player pressed the fire button, and if their timing was correct, the clay disc exploded. Points were awarded for being on target, and every now and then, a parrot flew across the screen, which could be gunned down for a bonus.

To make a skeet shooting game with Pygame Zero, we need a few graphical elements. First, a static background of hills and grass, with two clay disc throwers each side of the screen, and a semicircle where our shooter stands – this can be displayed first, every time our `draw()` function is called. We can then draw our shooter (created as an Actor) in the centre near the bottom of the screen. The shooter has three images: one central while no keys are pressed, and two for the directions left and right when the player presses the left or right keys. We also need to have two square target sights to the left and right above the shooter, which we can create as Actors.

To make the clay targets, we create an array to hold disc Actor objects. In our `update()` function we can trigger the creation of a new disc based on a random number, and once created, start an animation to move it across the screen in front of the shooter. We can add a shadow to the discs by tracking a path diagonally across the screen so that the shadow appears at the correct Y coordinate regardless of the disc's height – this is a simple way of giving our game the illusion of depth. While we're in the `update()` function, looping around our disc object list, we can calculate the distance of the disc to the nearest target sight frame, and from that, work out which is the closest.

When we've calculated which disc is closest to the right-hand sight, we want to move the sight towards the disc so that their paths intersect. All we need to do is take the difference of the Y coordinates, divide by two, and apply that offset to the target sight. We also do the same for the left-hand sight. If the correct key (left or right arrows) is pressed at the moment a disc crosses the path of the sight frame, we register a hit and cycle the disc through a sequence of exploding frames. We can keep a score and display this with an overlay graphic so that the player knows how well they've done.

And that's it! You may want to add multiple players and perhaps a parrot bonus, but we'll leave that up to you.

Skeet shooting in Python

Here's Mark's code snippet, which creates a skeet shooting game in Python. To get it running on your system, you'll need to install Pygame Zero – you can find full instructions at **wfmag.cc/pgzero**.

Download the code from GitHub: **wfmag.cc/wfmag35**

```
from random import randint
gameState = shootTimer = score = 0
shooter = Actor('shooter', center=(400, 450))
frameLeft = Actor('frame', center=(320, 350))
frameRight = Actor('frame', center=(480, 350))
skeets = []
def draw():
  screen.blit("background", (0, 0))
  if gameState == 0:
    for s in range(len(skeets)):
        if skeets[s].x > 0 and skeets[s].x < 800 and skeets[s].
frame < 4:
        skeets[s].draw()
        screen.blit("shadow", (skeets[s].x-20, 400-(skeets[s].
life/2)))
    shooter.draw()
    frameLeft.draw()
    frameRight.draw()
  else:
    screen.draw.text("ROUND OVER", center = (400, 300),
owidth=0.5, ocolor=(255,255,255), color=(0,255,0) , fontsize=80)
  screen.blit("overlay", (0, 0))
  screen.draw.text("SCORE:"+str(score), center = (400, 550),
owidth=0.5, ocolor=(255,255,255), color=(0,0,255) , fontsize=80)
  screen.draw.text("PYGAME ZERO SKEET SHOOT", center = (400, 55),
owidth=0.5, ocolor=(255,255,255), color=(255,0,0) , fontsize=60)
def update():
  global shootTimer, gameState
  if gameState == 0:
    if len(skeets) == 100: gameState = 1
    if randint(0,100) == 1: makeSkeet(700)
    if randint(0,100) == 2: makeSkeet(100)
    if shootTimer == 0:
      shooter.image = "shooter"
    else: shootTimer -= 1
    for s in range(len(skeets)):
      skeets[s].life += 1
      if skeets[s].frame > 0 and skeets[s].frame < 4:
        skeets[s].image = "skeet"+str(skeets[s].frame)
        skeets[s].frame += 1
      if skeets[s].x < 320 and skeets[s].dir == "right":
        skeets[s].distToLeftTarget = 320 - skeets[s].x
      else: skeets[s].distToLeftTarget = 999
      if skeets[s].x > 480 and skeets[s].dir == "left":
        skeets[s].distToRightTarget = skeets[s].x - 480
      else: skeets[s].distToRightTarget = 999
    targetLeft = getNearestSkeetY("left")
    if targetLeft > 0: frameLeft.y += (targetLeft-
frameLeft.y)/2
    targetRight = getNearestSkeetY("right")
    if targetRight > 0: frameRight.y += (targetRight-
frameRight.y)/2
```


```
def on_key_down(key):
  global shootTimer
  if (shootTimer == 0):
    if key.name == "LEFT":
      shooter.image = "shooter_l"
      shootTimer = 10
      checkShot("left")
    if key.name == "RIGHT":
      shooter.image = "shooter_r"
      shootTimer = 10
      checkShot("right")
def makeSkeet(st):
  skeets.append(Actor('skeet', center=(st, 370)))
  s = len(skeets)-1
  skeets[s].frame = 0
  skeets[s].life = 0
  skeets[s].distToLeftTarget = 999
  skeets[s].distToRightTarget = 999
  endpoint = 800
  skeets[s].dir = "right"
  if st > 400:
    endpoint = 0
    skeets[s].dir = "left"
  animate(skeets[len(skeets)-1], duration=3, pos=(endpoint,
randint(-200,250)))
def getNearestSkeetY(leftorright):
  y = 0
  dist = 999
  for s in range(len(skeets)):
    if leftorright == "right":
      if(skeets[s].distToRightTarget < dist):
        dist = skeets[s].distToRightTarget
        y = skeets[s].y
    if leftorright == "left":
      if(skeets[s].distToLeftTarget < dist):
        dist = skeets[s].distToLeftTarget
        y = skeets[s].y
    return y
def checkShot(leftorright):
  global score
  sounds.shot.play()
  for s in range(len(skeets)):
    if leftorright == "right":
      if skeets[s].collidepoint((frameRight.x, frameRight.y))
and skeets[s].frame == 0:
        score += 1000
        skeets[s].frame = 1
    if leftorright == "left":
      if skeets[s].collidepoint((frameLeft.x, frameLeft.y)) and
skeets[s].frame == 0:
    score += 1000
      skeets[s].frame = 1
```