Robótica

Conceptos de visión por computador para robots

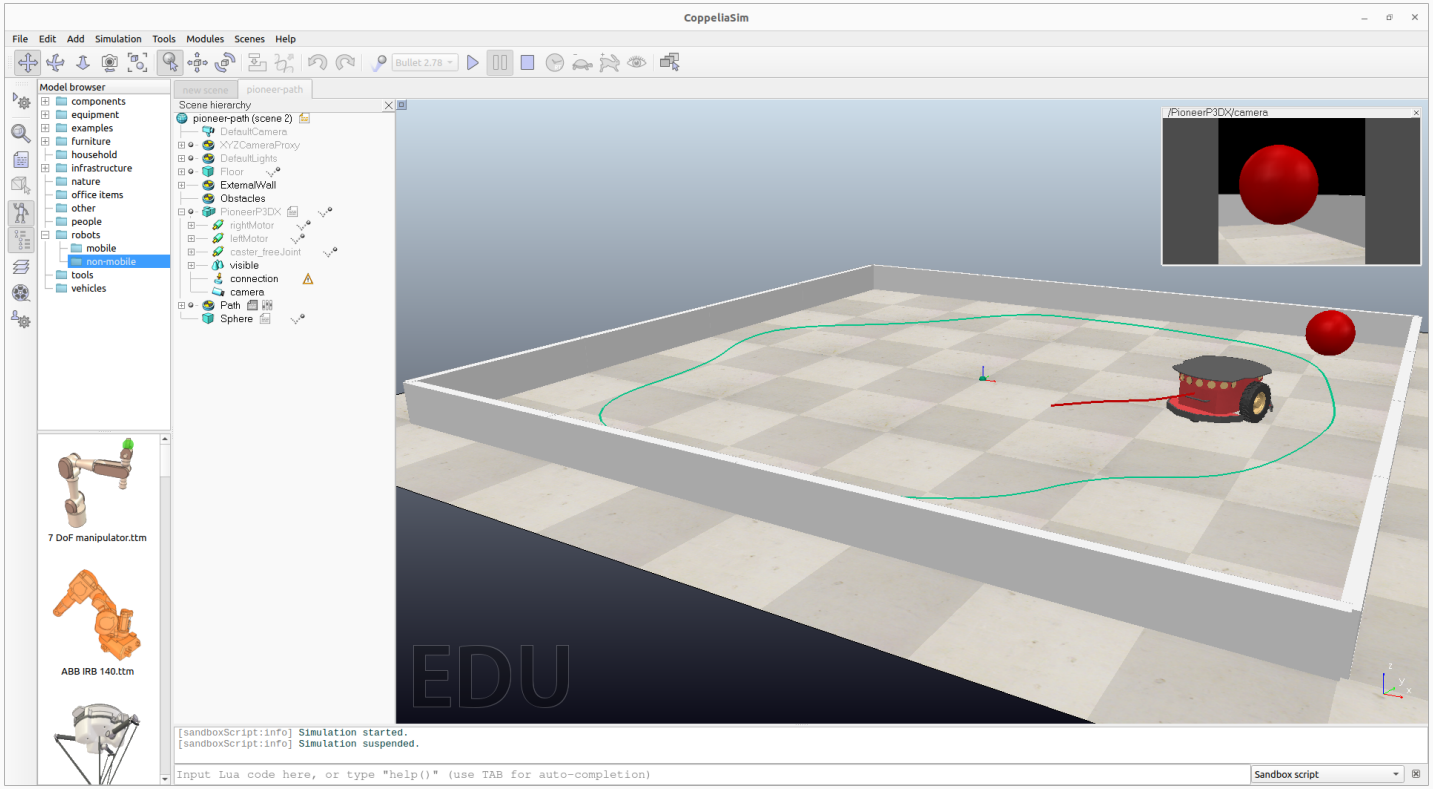
Javier de Lope Asiaín

Deaprtamento de Inteligencia Artificial
Escuela Técnica Superior de Ingenieros Informáticos
Universidad Politécnica de Madrid

Objetivo

El objetivo de la práctica es que el robot siga una bola roja que se encuentra en movimiento a velocidad constante en la escena.

La bola sigue un camino predefinido mediante un *path* en el simulador (la bola no es controlable).



Estrategia básica

Estrategia básica

1. Detectar la bola en la imagen

Estrategia básica

1. Detectar la bola en la imagen (segmentación por color)

Estrategia básica

1. Detectar la bola en la imagen (segmentación por color)
2. (Filtrado y umbralizado)

Estrategia básica

1. Detectar la bola en la imagen (segmentación por color)
2. (Filtrado y umbralizado)
3. Buscar *contornos* en la imagen binaria

Estrategia básica

1. Detectar la bola en la imagen (segmentación por color)
2. (Filtrado y umbralizado)
3. Buscar *contornos* en la imagen binaria
4. Obtener los momentos (espaciales) de la imagen

Estrategia básica

1. Detectar la bola en la imagen (segmentación por color)
2. (Filtrado y umbralizado)
3. Buscar *contornos* en la imagen binaria
4. Obtener los momentos (espaciales) de la imagen
    - El área que ocupa la bola es M_{00}
    - El centroide de la bola es $c_x = M_{10}/M_{00}$, $c_y = M_{01}/M_{00}$

Estrategia básica

1. Detectar la bola en la imagen (segmentación por color)
2. (Filtrado y umbralizado)
3. Buscar *contornos* en la imagen binaria
4. Obtener los momentos (espaciales) de la imagen
   - El área que ocupa la bola es M_{00}
   - El centroide de la bola es $c_x = M_{10}/M_{00}$, $c_y = M_{01}/M_{00}$
5. La consigna para el controlador es mantener $c_x = 128$ (la imagen que se recibe de la cámara es de 256×256)

Estrategia básica

1. Detectar la bola en la imagen (segmentación por color)
2. (Filtrado y umbralizado)
3. Buscar *contornos* en la imagen binaria
4. Obtener los momentos (espaciales) de la imagen
   - El área que ocupa la bola es M_{00}
   - El centroide de la bola es $c_x = M_{10}/M_{00}$, $c_y = M_{01}/M_{00}$
5. La consigna para el controlador es mantener $c_x = 128$ (la imagen que se recibe de la cámara es de 256×256)
6. El área da una idea de lo cerca que está el robot a la bola y puede utilizarse para determinar la velocidad del robot

Estrategia básica

1. Detectar la bola en la imagen (segmentación por color)
2. (Filtrado y umbralizado)
3. Buscar *contornos* en la imagen binaria
4. Obtener los momentos (espaciales) de la imagen
   - El área que ocupa la bola es M_{00}
   - El centroide de la bola es $c_x = M_{10}/M_{00}$, $c_y = M_{01}/M_{00}$
5. La consigna para el controlador es mantener $c_x = 128$ (la imagen que se recibe de la cámara es de 256×256)
6. El área da una idea de lo cerca que está el robot a la bola y puede utilizarse para determinar la velocidad del robot
7. También puede utilizarse c_y ya que la bola asciende en la imagen (disminuye el valor de esa coordenada) según se aleja

Procesamiento de la imagen de la cámara

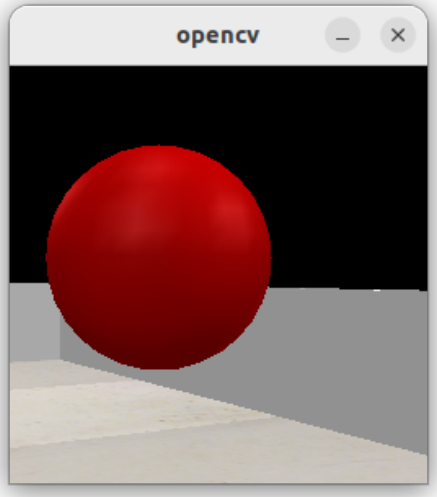


Imagen original

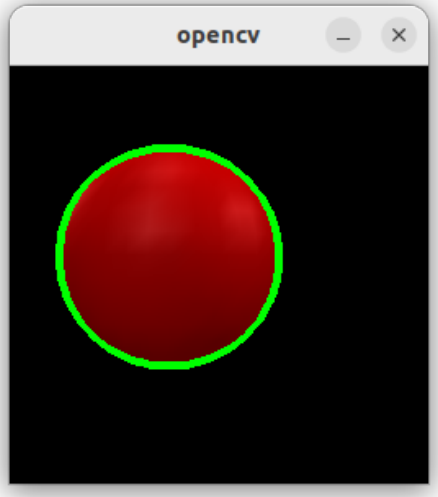
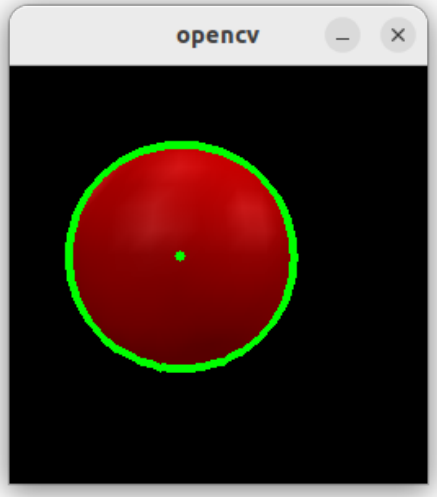


Imagen filtrada solo con objetos rojos



Centroide de los objetos rojos

Clase P3DX

```
class P3DX():

    ...

    def __init__(self, sim, robot_id, use_camera=False):
        self.sim = sim
        print('***_getting_handles', robot_id)
        self.left_motor = self.sim.getObject(f'/{robot_id}/leftMotor')
        self.right_motor = self.sim.getObject(f'/{robot_id}/rightMotor')
        self.sonar = []
        for i in range(self.num_sonar):
            self.sonar.append(self.sim.getObject(f'/{robot_id}/ultrasonicSensor[{i}]'))
        if use_camera:
            self.camera = self.sim.getObject(f'/{robot_id}/camera')

    ...

    def get_image(self):
        img, resX, resY = self.sim.getVisionSensorCharImage(self.camera)
        img = np.frombuffer(img, dtype=np.uint8).reshape(resY, resX, 3)
        img = cv2.flip(cv2.cvtColor(img, cv2.COLOR_BGR2RGB), 0)
        return img
```

Controlador

```
def follow_ball(image):
    ...


def main(args=None):
    coppelia = robotica.Coppelia()
    robot = robotica.P3DX(coppelia.sim, 'PioneerP3DX', True)
    coppelia.start_simulation()
    while coppelia.is_running():
        readings = robot.get_sonar()
        img = robot.get_image()
        ...
    coppelia.stop_simulation()
```

www.dia.fi.upm.es/~jdlope