

**ĐẠI HỌC QUỐC GIA HÀ NỘI**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ**

-----🙞🙜🕮🙞🙜-----

****

**BÁO CÁO THỰC HÀNH MÔN HỌC**

**TƯƠNG TÁC NGƯỜI – ROBOT**

**BÀI THỰC HÀNH SỐ 4**

**Sinh viên thực hiện**

**Nguyễn Huy Thắng - 22027545**

**Hà Nội, ngày 18 tháng 10 năm 2025**

**Tóm tắt nội dung**

Dự án thực hành tập trung vào việc phát triển và triển khai các thuật toán Lập kế hoạch Chuyển động Xã hội cho Robot di động Pioneer 3-DX hoặc TIAGo trong môi trường mô phỏng Webots. Mục tiêu là giúp robot điều hướng trong một môi trường có người, đặc biệt là các tình huống yêu cầu robot phải duy trì khoảng cách xã hội và tránh va chạm chủ động với các tác nhân di chuyển khác. Pipeline chính bao gồm:

* Xác định mục tiêu động: Tính toán trọng tâm của một nhóm người để xác định điểm đến của robot, mô phỏng hành vi tiếp cận nhóm.
* Lập kế hoạch đường đi cấp thấp: Sử dụng dữ liệu từ GPS, Compass, và Lidar để tính toán vận tốc bánh xe.

1. **Mục tiêu**

* Mục tiêu tổng thể: Xây dựng thành công một bộ điều khiển robot trong Webots có khả năng thực hiện nhiệm vụ điều hướng phức tạp trong môi trường có người.
* Triển khai hành vi xã hội: Thể hiện khả năng của robot trong việc phát hiện và phản ứng một cách chủ động và an toàn đối với vật cản tĩnh và tác nhân di chuyển.

1. **Thực hành**
   1. **Bài tập 1: Tiếp cận nhóm người đứng yên**

**Yêu cầu:** Robot di chuyển đến trọng tâm của một nhóm 3 người đứng yên.

1. **Khởi tạo thiết bị và động cơ**

# Khởi tạo thiết bị

gps = robot.getDevice('gps')

gps.enable(TIME\_STEP)

compass = robot.getDevice('compass')

compass.enable(TIME\_STEP)

lidar = robot.getDevice('Sick LMS 291')

)

# Khởi tạo Motor

left\_motor = robot.getDevice('left wheel')

right\_motor = robot.getDevice('right wheel')

left\_motor.setPosition(float('inf'))

right\_motor.setPosition(float('inf'))

left\_motor.setVelocity(0)

right\_motor.setVelocity(0)

}

**Giải thích:**

* Sử dụng lớp Supervisor để truy cập và điều khiển các thành phần trong thế giới mô phỏng.
* GPS và Compass được dùng để xác định Vị trí và Góc hướng của robot.
* Lidar được kích hoạt để đảm bảo robot có khả năng dừng lại khi gặp vật cản rất gần.
* Motor được đặt ở chế độ điều khiển vận tốc setPosition(float('inf')) với vận tốc ban đầu bằng 0.

1. **Hàm tính toán mục tiêu (trọng tâm)**

  def compute\_goal():

    valid = [p for p in people\_nodes if p]

    if len(valid) < 3:

        print(f"⚠️ Lỗi: Chỉ tìm thấy {len(valid)}/3 người đi bộ. Kiểm tra DEF names.")

        return None

    p = [pos(v) for v in valid]

    cx = sum(pi[0] for pi in p) / 3  # x

    cy = sum(pi[1] for pi in p) / 3  # y

    return cx, cy, cx, cy

**Giải thích:**

* robot.getFromDef(name) của lớp Supervisor để truy xuất các đối tượng người đi bộ thông qua DEF names.
* Vị trí trọng tâm (Centroid) được tính bằng cách lấy trung bình cộng tọa độ và của 3 người:
* Vị trí này trở thành mục tiêu di chuyển.

1. **Lập kế hoạch chuyển động**

# ... trong vòng lặp chính

dx = gx - x; dy = gy - y

dist = math.hypot(dx, dy)

heading = math.atan2(dy, dx)

err\_angle = heading - yaw # Lỗi góc

# Bộ điều khiển Tỉ lệ (P-Controller)

Kp\_angle = 3.0

Kp\_speed = 3.0

w = Kp\_angle \* err\_angle # Vận tốc góc

v = max(-0.5, min(0.5, Kp\_speed \* dist)) # Vận tốc tuyến tính

# Chuyển đổi vận tốc robot sang vận tốc bánh xe

v\_left = (v - w \* WHEEL\_DIST / 2.0) / WHEEL\_RADIUS

v\_right = (v + w \* WHEEL\_DIST / 2.0) / WHEEL\_RADIUS

left\_motor.setVelocity(v\_left); right\_motor.setVelocity(v\_right)

**Giải thích:**

* Sử dụng bộ điều khiển Tỉ lệ (P) cho cả góc lái và vận tốc tuyến tính.
* Điều khiển Góc: Vận tốc góc tỉ lệ với lỗi góc err\_angle.
* Điều khiển Tốc độ: Vận tốc tuyến tỉ lệ với khoảng cách đến mục tiêu dist, giúp robot di chuyển chậm lại khi đến gần.
* Công thức vận tốc vi sai được áp dụng để chuyển đổi vận tốc tuyến tính và vận tốc góc của robot thành vận tốc riêng lẻ cho bánh xe trái và phải..

**Kết quả**

[Video kết quả](https://drive.google.com/file/d/1c1I7BDYTILMh9axb4mg-xkL9V4LQRr8O/view?usp=sharing)

* Video thấy kết quả chạy đúng chương trình theo yêu cầu bài toán.
* Robot xác định được trọng tâm của nhóm người, tính toán đường đi, và di chuyển mượt mà đến vị trí đó cho đến khi khoảng cách đến mục tiêu dist đạt giá trị nhỏ nhất, sau đó dừng lại..
* Chuẩn bị cho phần 2 khi có thêm đối tượng người di chuyển.
  1. **Bài Tập 2: Tiếp cận nhóm và tránh người đi ngang**

**Yêu cầu:** Thêm một người đi bộ thứ 4 di chuyển cắt ngang. Robot phải dừng lại nếu người đó ở trong phạm vi nguy hiểm.

1. **Tích hợp cơ chế xã hội**

# Lấy node của người đi bộ thứ 4

fourth\_person = robot.getFromDef('pedestrian(3)')

# ... trong vòng lặp chính

if fourth\_person:

    fourth\_pos = pos(fourth\_person)

    dist\_to\_fourth = math.hypot(fourth\_pos[0] - x, fourth\_pos[1] - y)

    PEDESTRIAN\_STOP\_DIST = 0.5 # Ngưỡng dừng

    if dist\_to\_fourth < PEDESTRIAN\_STOP\_DIST:

        print(f"🛑 Phát hiện người đi bộ thứ 4 trong khoảng cách nguy hiểm, robot dừng lại.")

        left\_motor.setVelocity(0)

        right\_motor.setVelocity(0)

        continue # Bỏ qua các bước tính toán di chuyển

**Giải thích:**

* Sử dụng cùng cơ chế truy cập robot.getFromDef() để lấy vị trí của người đi bộ thứ 4 – tác nhân gây nhiễu.
* dist\_to\_fourth: Mỗi bước thời gian, robot tính toán khoảng cách Euclide đến người thứ tư.
* Hành vi ưu tiên cao: Nếu khoảng cách này nhỏ hơn ngưỡng an toàn - PEDESTRIAN\_STOP\_DIST = 0.5m, robot thực hiện hành vi dừng ngay lập tức và sử dụng continue để bỏ qua mọi tính toán điều khiển di chuyển khác trong bước lặp đó.
* Hành vi này mô phỏng sự nhận biết ngữ cảnh xã hội và nhường đường cho tác nhân khác đang di chuyển trong không gian chung..

1. **Lidar tránh va chạm**

if lidar:

    scan = lidar.getRangeImage()

    # Lấy khoảng cách nhỏ nhất ở phía trước

    min\_dist = min(scan[len(scan)//4:3\*len(scan)//4])

    if min\_dist < STOP\_DIST:

        print(f"🛑 Lidar phát hiện vật cản trong khoảng cách {STOP\_DIST}m, robot dừng.")

        left\_motor.setVelocity(0)

        right\_motor.setVelocity(0)

        continue

**Giải thích**

* scan = lidar.getRangeImage(): trả về một mảng chứa khoảng cách đo được từ Lidar đến vật cản tại mỗi góc quét.
* if min\_dist < STOP\_DIST: kiểm tra điều kiện dừng của robot
* left\_motor.setVelocity(0) và right\_motor.setVelocity(0): khi phát hiện vật cản đặt vận tốc hai bánh bằng 0 thực hiện dừng lại
* Cơ chế tránh va chạm tĩnh vẫn được duy trì, hoạt động như một lớp an toàn thứ hai.

**Kết quả**

[Video kết quả](https://drive.google.com/file/d/160X6W_r_w0AV5nkkjSCzguu3Z-SIu1VM/view?usp=sharing)

* Robot ban đầu tiến về trọng tâm nhóm người. Khi người đi bộ thứ 4 cắt ngang và đi vào phạm vi 0.5m, robot ngay lập tức dừng lại. Sau khi người thứ 4 đi qua và rời khỏi phạm vi an toàn, robot tự động tiếp tục nhiệm vụ tiếp cận nhóm.

**Kết luận**

* Thực hành này đã cung cấp kinh nghiệm thực tế trong việc thiết kế các bộ điều khiển cho Robot tương tác xã hội trong môi trường mô phỏng.
* Điều hướng thành công: Robot Pioneer 3-DX đã được lập trình thành công để xác định vị trí, tính toán mục tiêu trọng tâm, và di chuyển đến đó bằng bộ điều khiển P đơn giản, đây là nền tảng cho mọi nhiệm vụ điều hướng phức tạp.
* Tích hợp hành vi xã hội: Quan trọng hơn, robot đã thể hiện khả năng nhận biết và phản ứng với một tác nhân động. Cơ chế dừng lại dựa trên khoảng cách xã hội với người đi bộ thứ 4 mô phỏng hành vi nhường đường an toàn cần thiết cho robot hoạt động trong môi trường chia sẻ với con người.