HUST

ĐẠI HỌC BÁCH KHOA HÀ NỘI

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.

# Tính toán tiến hóa

ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

# Path planning for indoor Mobile robot based on deep learning

Nguyễn Nhật Minh - 20225043
Ngô Trí Cảnh - 20220015

ONE LOVE. ONE FUTURE.

# Vấn đề của đường đi tạo bởi RRT

- RRT hiệu quả trong việc nhanh chóng tìm đường đi khả thi xong không gian phức tạp.
- Nhưng:
  - RRT thường tạo ra các đường đi zig-zag, nhiều khúc cua, thay đổi hướng đột ngột
  - Không tối ưu
  - ⇒ Điều này là không tốt cho một robot thực tế bị ảnh hưởng bởi các quy tắc vật lý.

# Nội dung chính

I. Post-Smoothing

II. RRT*

# I. Post-Smoothing

- Post-Smoothing:
  - là quá trình làm mịn đường đi sau khi thuật toán lập kế hoạch đường đi chính (ví dụ: RRT) đã hoàn thành việc tạo ra một đường đi ban đầu.
- Mục tiêu:
  - Làm ngắn đường đi: Loại bỏ các đoạn đường thừa, tạo đường đi trực tiếp hơn.
  - Làm mượt đường đi: Giảm số lượng khúc cua và làm cho đường đi mềm mại hơn, phù hợp cho chuyển động thực tế.
  - Cải thiện tính thẩm mĩ: Tạo ra đường đi trực quan hơn và dễ chịu hơn.

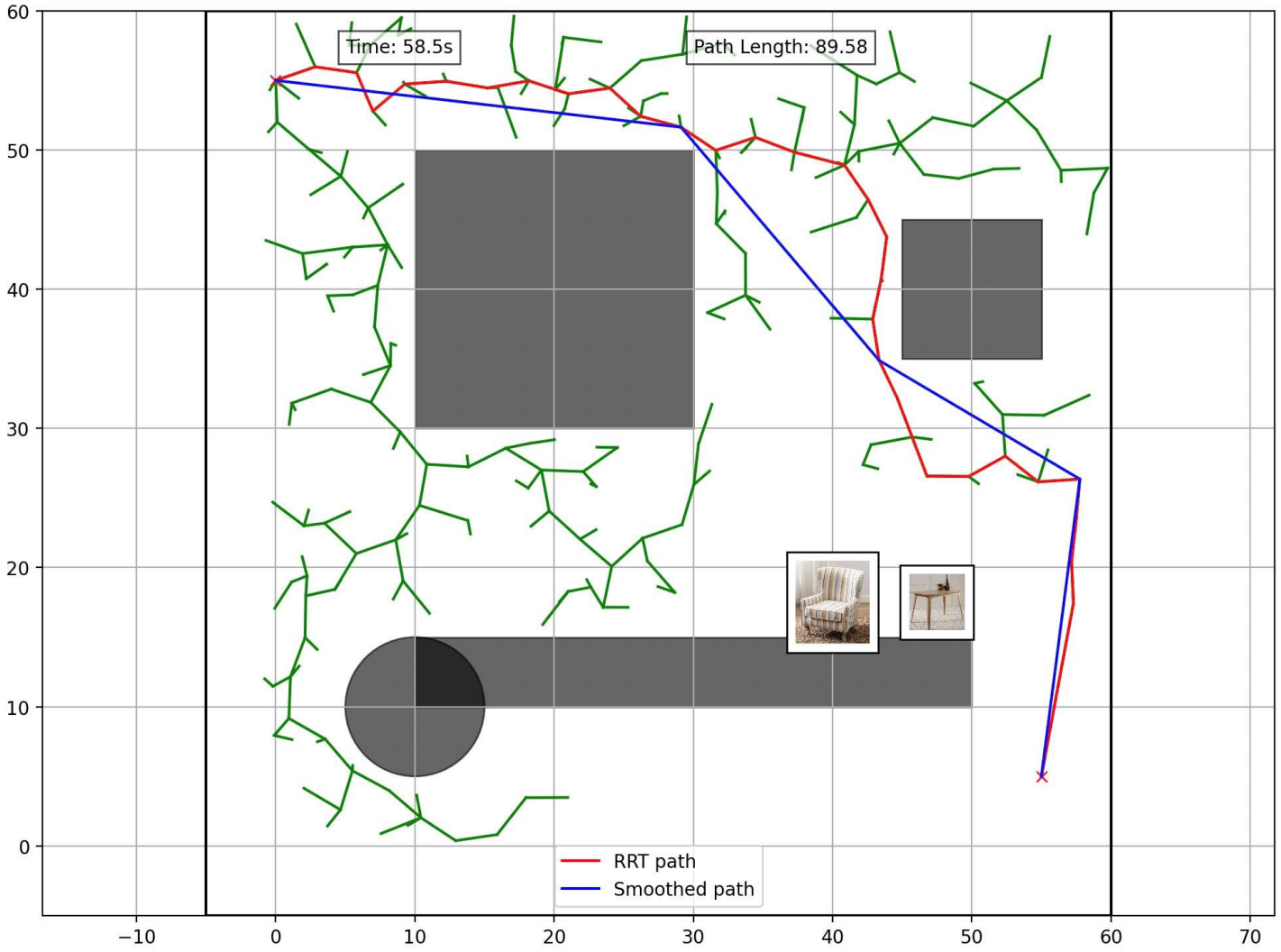
# I. Post-Smoothing

## Shortcut Smoothing

- Shortcut smoothing:
  - Nguyên lý: Thuật toán cố gắng nối 2 điểm bất kỳ không liền kế trong đường đi hiện tại để tạo ra "đường tắt" ngắn hơn.
- Ưu điểm:
  - Đơn giản, dễ cài đặt.
  - Hiệu quả trong việc làm ngắn và làm mượt đường đi.
  - Tính toán nhanh.
- Nhược điểm:
  - Có thể không tìm ra đường đi tối ưu tuyệt đối.
  - Kết quả phụ thuộc vào thứ tự duyệt điểm.

# I. Post-Smoothing

## Shortcut Smoothing- triển khai trong bài toán



# I. Post-Smoothing

## Spline Smoothing

- Spline smoothing:
  - Nguyên lý: Thay thế các đoạn thẳng trên đường đi RRT bằng các đường cong spline (B-spline, Bezier Spline, Cubic Spline, ...) mượt mà hơn.
- Ưu điểm:
  - Tạo ra đường đi rất mượt, liên tục về độ cong.
  - Thích hợp cho các ứng dụng yêu cầu đường đi mềm mại (ví dụ: robot di chuyển trong môi trường con người).
- Nhược điểm:
  - Phức tạp hơn shortcut smoothing.
  - Có thể làm cho đường đi dài hơn một chút so với shortcut smoothing (để đạt được độ mượt).
  - Tính toán có thể tốn thời gian hơn, đặc biệt với spline bậc cao.

# I. Post-Smoothing

## Spline Smoothing

- B-Spline (Basis spline):
  - là tổ hợp tuyến tính của các hàm cơ sở B-Spline và một tập hợp các điểm kiểm soát.

$$P(t) = \sum_{i=0}^{n} P_i * B_{i,p}(t)$$

    - P(t): Điểm trên đường cong B-Spline tại tham số t.
    - $P_i$: Là điểm kiểm soát thứ i.
    - k: Bậc của Spline
- Hàm cơ sở B-spline:
  - Knot vector: là một chuỗi các giá trị tham số không giảm, quyết định các điểm "nút" trên đường cong và ảnh hưởng đến hình dạng của spline.

$$B_{i,0}(t) := \begin{cases} 1 & \text{if } t_i \leq t < t_{i+1}, \\ 0 & \text{otherwise.} \end{cases}$$

$$B_{i,p}(t) := \frac{t - t_i}{t_{i+p} - t_i} B_{i,p-1}(t) + \frac{t_{i+p+1} - t}{t_{i+p+1} - t_{i+1}} B_{i+1,p-1}(t)$$

$t_i$: Là các knot vector.

# I. Post-Smoothing

## Spline Smoothing

- Bezier Spline (Bezier Curve):
  - là tổ hợp tuyến tính của đa thức Bernstein và một tập hợp các điểm kiểm soát.

$$\mathbf{B}(t) = \sum_{i=0}^{n} b_{i,n}(t)\mathbf{P}_i, \quad 0 \le t \le 1$$

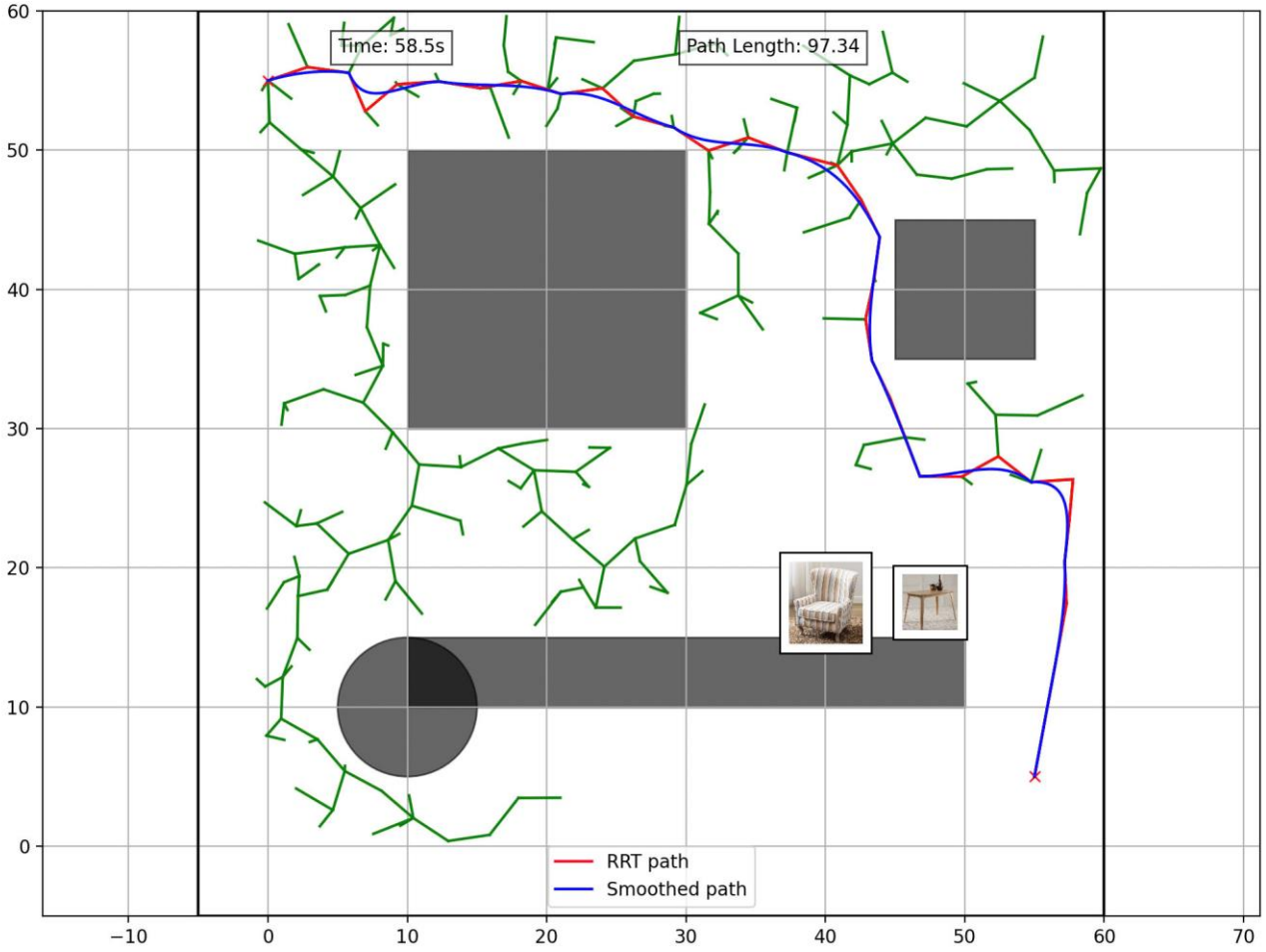
    - B(t): Điểm trên đường cong B-Spline tại tham số t .
    - $P_i$: Là điểm kiểm soát thứ i.
    - n: Bậc của curve
- Đa thức Bernstein:

$$b_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}, \quad i = 0, \ldots, n$$

$$\binom{n}{i} = \frac{n!}{i!(n-i)!}$$

# I. Post-Smoothing

## Spline Smoothing – Triển khai trong bài toán



# II. RRT*

RRT

1. **Khởi tạo**: Bắt đầu với một cây chỉ chứa nút gốc (điểm xuất phát).
2. **Lấy mẫu ngẫu nhiên**: Chọn một điểm ngẫu nhiên qrand trong không gian làm việc.
3. **Tìm nút gần nhất**: Xác định nút qnear trong cây gần qrand nhất.
4. **Mở rộng cây**: Từ qnear di chuyển 1 khoảng tối đa e về phía qrand để tạo điểm mới qnew.
5. **Kiểm tra va chạm**: Nếu đường nối qnear và qnew không va chạm, thêm qnew vào cây.
6. **Lặp lại**: Tiếp tục cho đến khi qnew đủ gần đích hoặc số lần lặp tối đa.

RRT*

1. **Lặp lại các bước 1–5 của RRT** để tạo qnew.
2. **Chọn cha tối ưu:**
   - Tìm tập các nút Qnear trong bán kính r qnew
   - Chọn nút qnear trong Qnear sao cho chi phí từ gốc đến qnear
3. **Thêm qnew vào cây** với cha là qnear tối ưu
4. **Viết lại cây:**
   - Với mỗi nút qnear trong Qnear, kiểm tra xem nếu có kết nối qnew làm cha của qnear có giảm chi phí từ gốc đến qnear không
   - Nếu có, cập nhật cha của qnear thành qnew

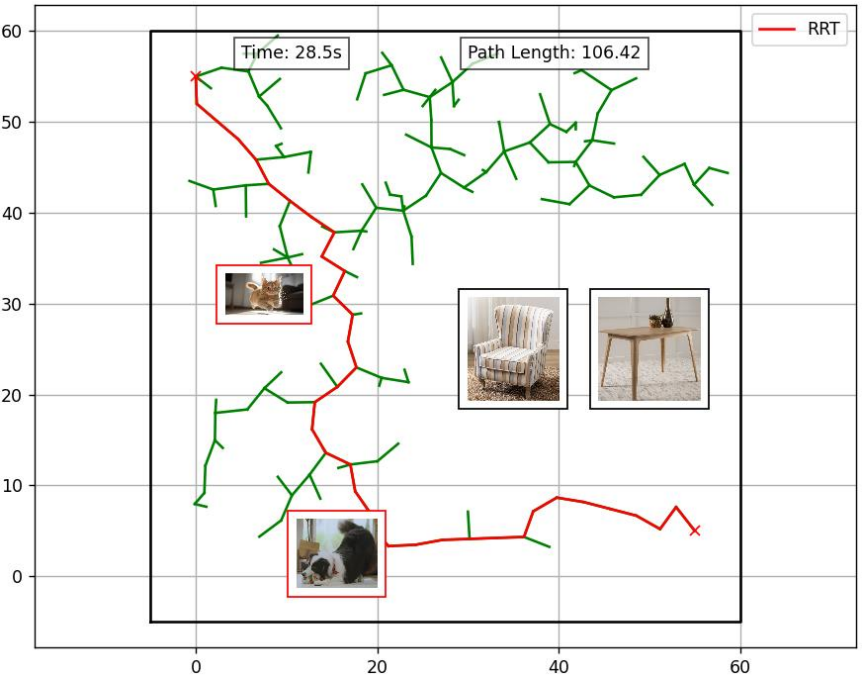
# II. RRT*

| Tiêu chí | RRT | RRT* |
|---|---|---|
| **Tính tối ưu** | Không tối ưu, chỉ tìm đường đi khả thi. | Tối ưu tiệm cận khi số mẫu đủ lớn. |
| **Chất lượng đường đi** | Đường đi thường dài, không trơn. | Đường đi ngắn và mượt hơn theo thời gian. |
| **Cơ chế tối ưu** | Không có. | Chọn cha tối ưu + Rewiring. |
| **Tính toán** | Nhanh, độ phức tạp O(n)O(n). | Chậm hơn, độ phức tạp O(nlogn)O(nlogn). |
| **Ứng dụng** | Phù hợp yêu cầu tốc độ (real-time). | Phù hợp khi cần đường đi chất lượng cao. |

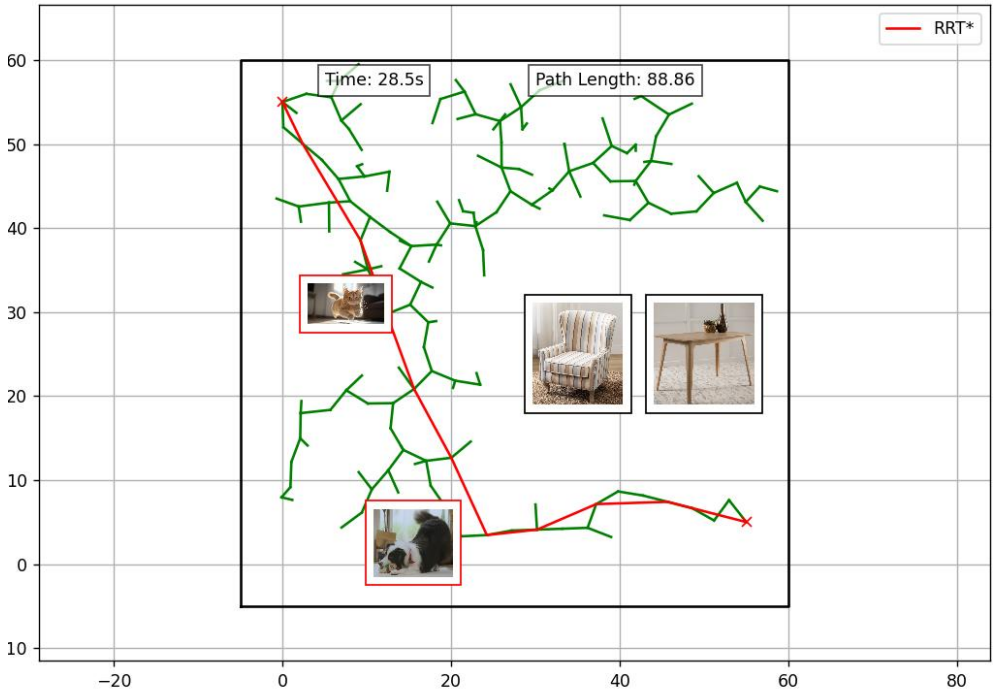
# II. RRT*

## Triển khai trong bài toán

RRT




RRT*



HUST

hust.edu.vn fb.com/dhbkhn

# THANK YOU !