

## Hermite (quỹ đạo 2D) từ robot đến look-ahead pose

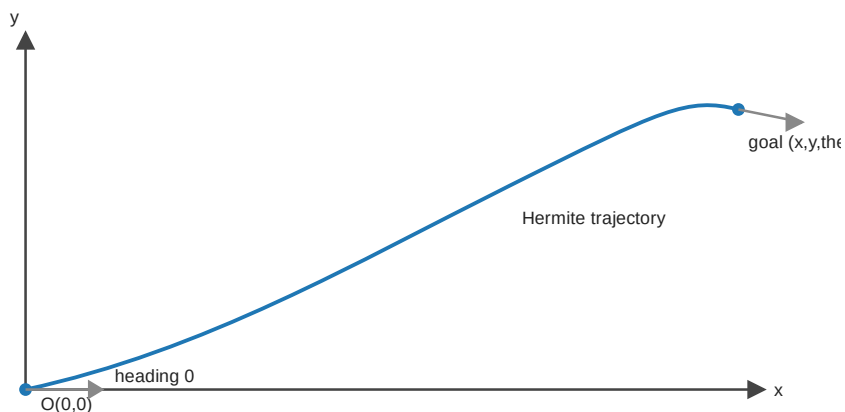


Figure 1: Hermite trajectory

### Bài toán

Cho pose đích trong hệ tọa độ robot:  $\text{goal} = (x, y, \theta)$ , với robot ở gốc  $(0, 0, 0)$  (hướng theo trục  $+x$ ). Cần sinh quỹ đạo từ robot đến pose đích.

### Ý tưởng

Dùng đường cong Hermite bậc 3 để đảm bảo: - Đi qua đúng điểm đầu và cuối.  
- Đúng hướng đầu và cuối.

Đường cong tham số  $t$  từ 0 đến 1:

$$p(t) = (x(t), y(t))$$

Ràng buộc: -  $p(0) = (0, 0)$ , hướng 0 rad. -  $p(1) = (x, y)$ , hướng  $\theta$  rad.

### Hệ số Hermite

$$h_{00} = 2t^3 - 3t^2 + 1$$

$$h_{10} = t^3 - 2t^2 + t$$

$$h_{01} = -2t^3 + 3t^2$$

$$h_{11} = t^3 - t^2$$

### Đạo hàm đầu/cuối

Chọn độ dài đặc trưng  $L$  (thường  $L = \sqrt{x^2 + y^2}$ ) và hướng  $\text{dir}$ :

```
dir = +1  (đi xuôi)
dir = -1  (đi ngược 180° ở cả đầu và cuối)
```

Khi đó:

```
p'(0) = (dir * L, 0)
p'(1) = (dir * L*cos(theta), dir * L*sin(theta))
```

### Công thức quỹ đạo

```
x(t) = h10*(dir*L) + h01*x + h11*(dir*L*cos(theta))
y(t) = h01*y + h11*(dir*L*sin(theta))
```

### Hướng (heading) trên đường cong

Đạo hàm:

```
h00' = 6t^2 - 6t
h10' = 3t^2 - 4t + 1
h01' = -6t^2 + 6t
h11' = 3t^2 - 2t
```

```
dx = h10'*(dir*L) + h01'*x + h11'*(dir*L*cos(theta))
dy = h01'*y + h11'*(dir*L*sin(theta))
```

Hướng:

```
heading(t) = atan2(dy, dx)
```

### Cách sử dụng

Lấy mẫu  $t$  đều (ví dụ 50-200 điểm) để tạo danh sách điểm quỹ đạo. Nếu cần độ cong, có thể tính thêm từ đạo hàm bậc 2.

### Ví dụ code C++

```
#include <vector>
#include <cmath>

struct Pose2D {
    double x;
    double y;
    double theta;
};

std::vector<Pose2D> generateHermiteTrajectory(
    double x, double y, double theta,
    int samples = 100)
{
```

```

std::vector<Pose2D> path;
path.reserve(samples + 1);

double L = std::sqrt(x * x + y * y);
if (L < 1e-6) {
    path.push_back({0.0, 0.0, 0.0});
    return path;
}

for (int i = 0; i <= samples; ++i) {
    double t = static_cast<double>(i) / samples;
    double t2 = t * t;
    double t3 = t2 * t;

    double h00 = 2 * t3 - 3 * t2 + 1;
    double h10 = t3 - 2 * t2 + t;
    double h01 = -2 * t3 + 3 * t2;
    double h11 = t3 - t2;

    double px = h10 * L + h01 * x + h11 * (L * std::cos(theta));
    double py = h01 * y + h11 * (L * std::sin(theta));

    double dh00 = 6 * t2 - 6 * t;
    double dh10 = 3 * t2 - 4 * t + 1;
    double dh01 = -6 * t2 + 6 * t;
    double dh11 = 3 * t2 - 2 * t;

    double dx = dh10 * L + dh01 * x + dh11 * (L * std::cos(theta));
    double dy = dh01 * y + dh11 * (L * std::sin(theta));

    double heading = std::atan2(dy, dx);

    path.push_back({px, py, heading});
}

return path;
}

```

### Ghi chú

- Nếu  $(x, y)$  rất gần 0 thì quỹ đạo suy biến, chỉ trả về điểm gốc.
- $L$  có thể điều chỉnh lớn hơn để quỹ đạo mềm hơn.