1.

3116050IL

揚曜淳

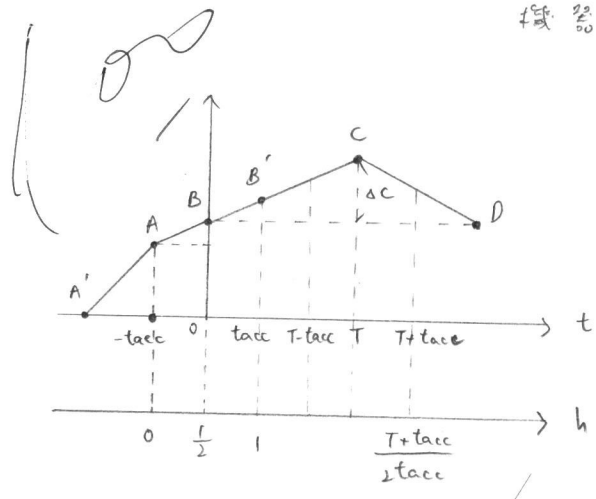
機器人碩一

Six Boundary Conditions

$q_A(t^+) = q_A(t^-)$, $q_{B'}(t^+) = q_{B'}(t^-)$

$\dot{q}_A(t^+) = \dot{q}_A(t^-)$, $\dot{q}_{B'}(t^+) = \dot{q}_{B'}(t^-)$

$\ddot{q}_A(t^+) = \ddot{q}_A(t^-)$, $\ddot{q}_{B'}(t^+) = \ddot{q}_{B'}(t^-)$



$h(t) = \frac{t+t_{acc}}{2t_{acc}}$, $-t_{acc} \le t \le t_{acc}$

$$\begin{cases} q(h) = a_4h^4 + a_3h^3 + a_2h^2 + a_1h + a_0 \\ \dot{q}(h) = 4a_4h^3 + 3a_3h^2 + 2a_2h + a_1 \\ \ddot{q}(h) = 12a_4h + 6a_3h + 2a_2 \end{cases}$$

$$\begin{cases} q(0) = a_0 = A \\ \dot{q}(0) = a_1 = -\Delta B / \frac{1}{2} \\ \ddot{q}(0) = 2a_2 = 0 \end{cases} \Rightarrow \begin{array}{l} a_0 = B + \Delta B \\ a_1 = -2\Delta B \\ a_2 = 0 \end{array}$$

$$\begin{cases} \dot{q}(1) = 4a_4 + 3a_3 + 2a_2 + a_1 = \frac{\Delta C}{\frac{T+t_{acc}}{2t_{acc}} - \frac{1}{2}} \\ \ddot{q}(1) = 12a_4 + 6a_3 = 0 \end{cases} \Rightarrow \begin{cases} 4a_4 + 3a_3 + a_1 = \frac{2\Delta C\, t_{acc}}{T} \\ 12a_4 + 6a_3 = 0 \end{cases} \Rightarrow \begin{cases} a_3 = 2\left(\Delta C \frac{t_{acc}}{T} + \Delta B\right) \\ a_4 = -\left(\Delta C \frac{t_{acc}}{T} + \Delta B\right) \end{cases}$$

position :

$$q(t) = q(h(t)) = \left[\left(\Delta C \cdot \frac{t_{acc}}{T} + \Delta B\right)(2-h)h^2 - 2\Delta B\right]h + B + \Delta B$$

velocity :

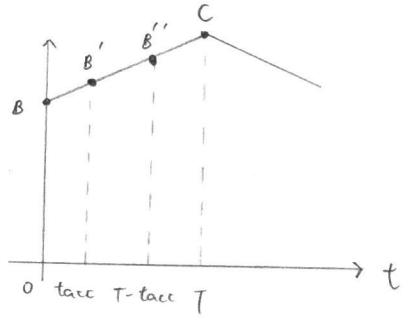
$$\dot{q}(t) = \dot{q}(h)\frac{dh}{dt} = \left[\left(\Delta C \cdot \frac{t_{acc}}{T} + \Delta B\right)(1.5-h)2h^2 - \Delta B\right] \cdot \frac{1}{t_{acc}}$$

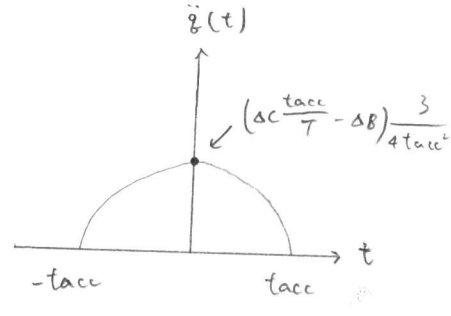
acceleration :

$$\ddot{q}(t) = \ddot{q}(h)\cdot\left(\frac{dh}{dt}\right)^2 + \dot{q}(h)\frac{dh^2}{dt^2}$$

$$= \left[\left(\Delta C \frac{t_{acc}}{T} + \Delta B\right)(1-h)\right] \cdot \frac{3h}{t_{acc}^2}$$

acceleration profile :





From B' to B'' , set $h = \frac{t}{T}$

then

$q(t) = \Delta C \cdot h + B$ ... position

$\dot{q}(t) = \frac{\Delta C}{T}$ ..... velocity

$\ddot{q}(t) = 0$ ... acceleration

2. 論文提出簡單的演算法，在解決機械手臂 C-space 路徑規劃與避障問題上，能有效率的進行運算，同時也透過 recursively slice projections 的方式提高效率與可控性

① 方法簡單，易於應用

② 在自由度較少的情況下，運算十分快速

③ 即使擁有多自由度，也能使用此方法

④ 即使在環境混亂或有凹面物體的情況下也能使用

How to formulate obstacles in the robot space:

⇒ 將 robot's workspace 上的障礙物 mapping 到 configuration space 上，以上在路徑移動時會造成碰撞的地方即為 C-space obstacles，既然知道會碰撞的路徑，剩餘部分即為 free space，代表機械手臂可有效移動的地方。

令 C 為擁有 n 個 joints 的 C-space obstacles

C 會近似於 (n-1) 維 slice projections 的聯集結果

每個 (n-1) 維的 slice projections 的近似結果，可透過聯集 (n-2) 維的 slice projections 得到，持續往下計算，直到 1 維 volumes。

⇒① 每次只考慮一個軸，並找到 the ranges of legal values of link,

② Sample the legal range of link at the specified resolution

③ 每往下探索一個軸，將先前的 legal values 察看一次，直到完成最後一軸（持續進行 ① ② 步驟）

How to simplify the search process :

① 對 C-space obstacle 做 slice projection，透過 (n-1) 維到 1 維的疊合結果得到 C-space 的解

② 只考慮前 3 軸，因為移動主要由此控制，在運算上較為容易

③ 量化 link 的 free space，並計算出 slice projections，以擴大該區域，並以長度較長的 link 來找路徑

How to organize the database for the collision-free space

將 C-space 以 recursive 的方式來表示

實作部分則以深度為 (n-1) 的 tree 來進行，其中 n 為手臂之 joint 數目，分支由可變動範圍分段數量來決定，葉子則是軸的移動範圍(可達 or 不可達)，可能為

樹中許多內部節點將沒有後一代，目為會導致 join 的衝突發生

the Algorithm for finding a feasible path.

C space 中有許多 joints，可從中找到多組 legal region

透過將相對的 joint 對應的 region join 在一塊，可以得出一個二維 region

白色是 free region，斜線是 collision space，透過 region 的重疊來確認軌跡的連通性

kernels

為了能移動機械手臂，將沒使用的 O_2 O_3 region 透過 O_1 進行聯結，產生出 region graph，增加 kernel 相連的可能性，從中找出可行路徑

知道 region graph 的起點與終點，來尋找出二、三維 kernels 可行解

若成功連接一條線，即可規劃出路徑