機器導航與探索 HW2

110033225 黃興佑

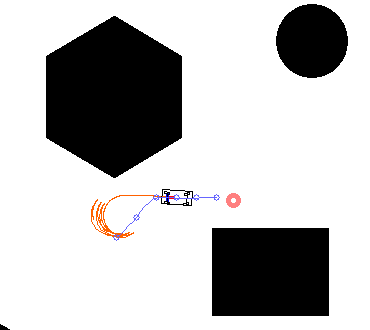
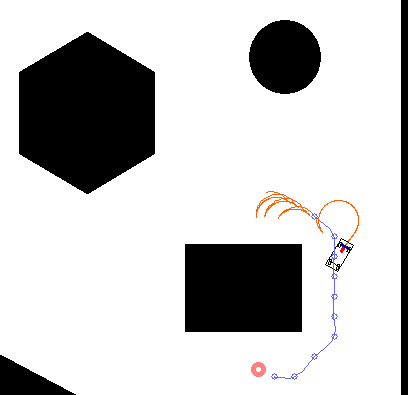
1. **kinematic models**: basic / differential drive

|  |  |  |
| --- | --- | --- |
| pid |  | PID算法中最重要的是把P gain, I gain 和 D gain調到合理值，最科學的作法是用Matlab調PID。  但現在就看圖手調參數。  調整後，某些pathtracking的震幅還是會很大，但大部分是可以接受的範圍。  另外只對角速度做了PID，直線速度只給定值。 |
| pure pursuit |  | Pure Pursuit 很明顯地改善PID偶爾會有比較大steady state error問題，但伴隨而來的問題是 tracking oscillation，即小幅震盪和急遽抖動/轉向。  雖然pathtracking的目標本身達成，但現實層面會讓機器控制難以穩定、乘客體感十分不舒服。 |
| lqr |  |  |

1. **kinematic models**: bicycle

|  |  |  |
| --- | --- | --- |
| pid |  | bicycle model 放大了在 differential drive demo中的問題，總是會有個不容忽視的穩態誤差。  但也可能是PID參數調得太爛了。  另外因為 bicycle model 是控制 a 和 delta，delta 由 PID 給出，a 則是自己設的速度上限減去當前的車速。 |
| pure pursuit |  | 在左圖中，可以看到第一個目標的path tracking的穩態誤差很小；但是第二個目標點卻甚至比PID還大。在少數情況下還會原地打轉。  我有嘗試不同的lookahead distance，但似乎沒有太多的改變。  想了很久但還是個未解之謎。 |
| stanley |  | Stanley 得出的 tracking path 就漂亮且穩定很多。但也花了不少時間算數學… |
| lqr |  |  |

1. collision

**邏輯**：撞牆後先倒退，重新嘗試 pathtracking，如此反覆超過一定次數後就做 path replanning，re-plan的方法很樸素，用第一個作業的cubic spline對當前路徑做圓滑。

**感想**: 大部分情況下都不會到 re-plan 的環節，通常倒退幾次後就能闖出去。

用 bicycle model x stanley control 比較好觀察到 collision handling 的邏輯。

備註: 運行環境為 docker  
