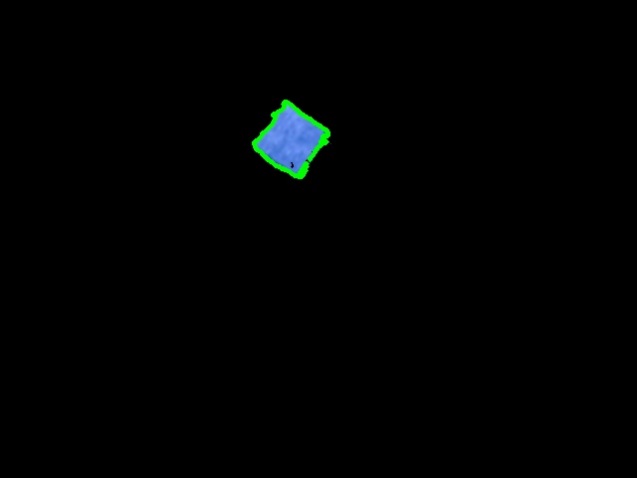
機械手臂專題結報:

我最基本的想法是以極座標來考慮所欲夾取物件所在的平面。很明顯任何一個在平面上的物體(方塊積木)只會有兩個自由度需要去考慮，所以我所實做的手臂夾取方法就是將夾取動作拆分成徑向長度(r)與角度(theta)兩個自由度的控制。

我所實做的方法主要可以拆分成三大塊:

**第一: 影像處理方式來做物件定位(edgeDetection.py)**

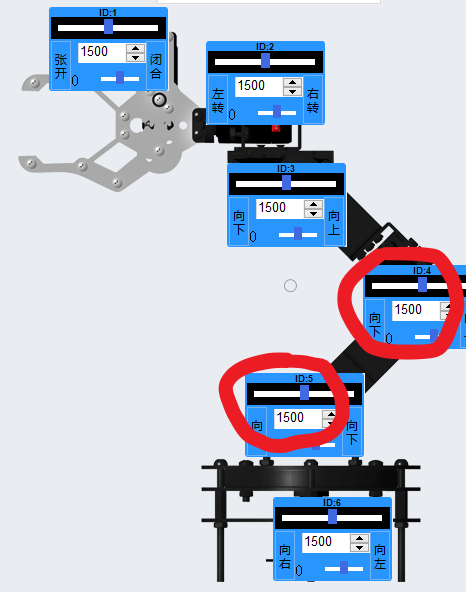
前提假設:相片沒有太多變形，直接利用等比例求物件真實座標



我所實做的方式是在物理環境中設定原點座標(底座圓盤中心)、俯瞰點座標(0,13cm)，並固定一切物理參數:鏡頭俯視位置、鏡頭距桌面高度。只要確定所有物理參數固定不變，我利用影響處理方式框出積木在像素圖片中的中心點座標，找出物件與俯瞰點(320, 240)之像素座標差，即可依照環境參數等資訊，等比例求物件實際座標。

由影像計算真實座標方法的侷限:物理環境(鏡頭俯視對準的位置、鏡頭高度)，決定了計算參數，進而影響物件座標的精確度。我目前只做手動校正。可能修正是利用深度相機來做更精確的物件與手臂底座座標的建構。我目前所實驗的結果是在edgeDetection\_rs.py中。實驗結果是必須修改我目前所使用的影像處理方式，因為由深度相機中擷取出來的色彩相片，是從影像串流中擷取，所以RGB值不是每一次都一樣。以我目前所使用的方式並不能保證每次都能定位出物件的像素坐標。

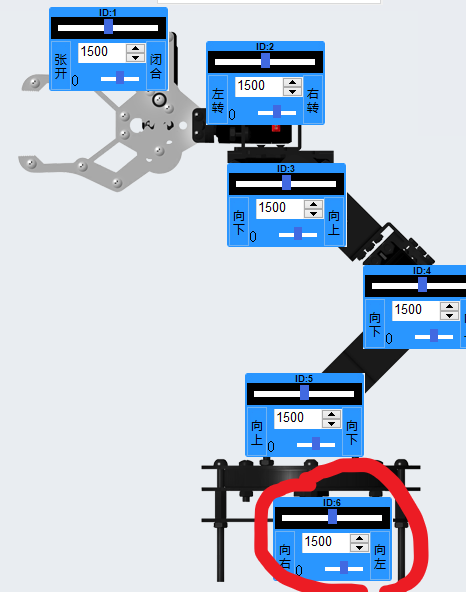
**第二: 極座標 r自由度的控制:使用兩個最近距離示範路徑的最終角度，以比例的方式給出當前距離所要的馬達角度。(averagePath.py)**

****

適用性:最遠(近)可夾取物件，不可超過(低於)示範路徑中所放物件之最長(短)徑向長度。

詳細地來說，findBasicPath(path\_list, r)、findNeighborPath(path\_list, r)分別用來找出最接近當前所欲夾取物件之徑向長度的兩條示範路徑。path\_list為所有示範路徑，r為欲夾取物件之徑向長度。

**第三: 極座標θ自由度的控制:根據影像計算出來的物件實際座標，計算手臂底座馬達應該的轉向角度。(averagePath.py)**

****

適用性:目前只實作夾取與徑向(radial)平行的物件。不平行的物件夾取必須再控制第二個手臂馬達。

詳細地說，directTheta(ser, path\_list, r, p)計算底座馬達應該轉向的角度，其中p為物件中心之像素座標。