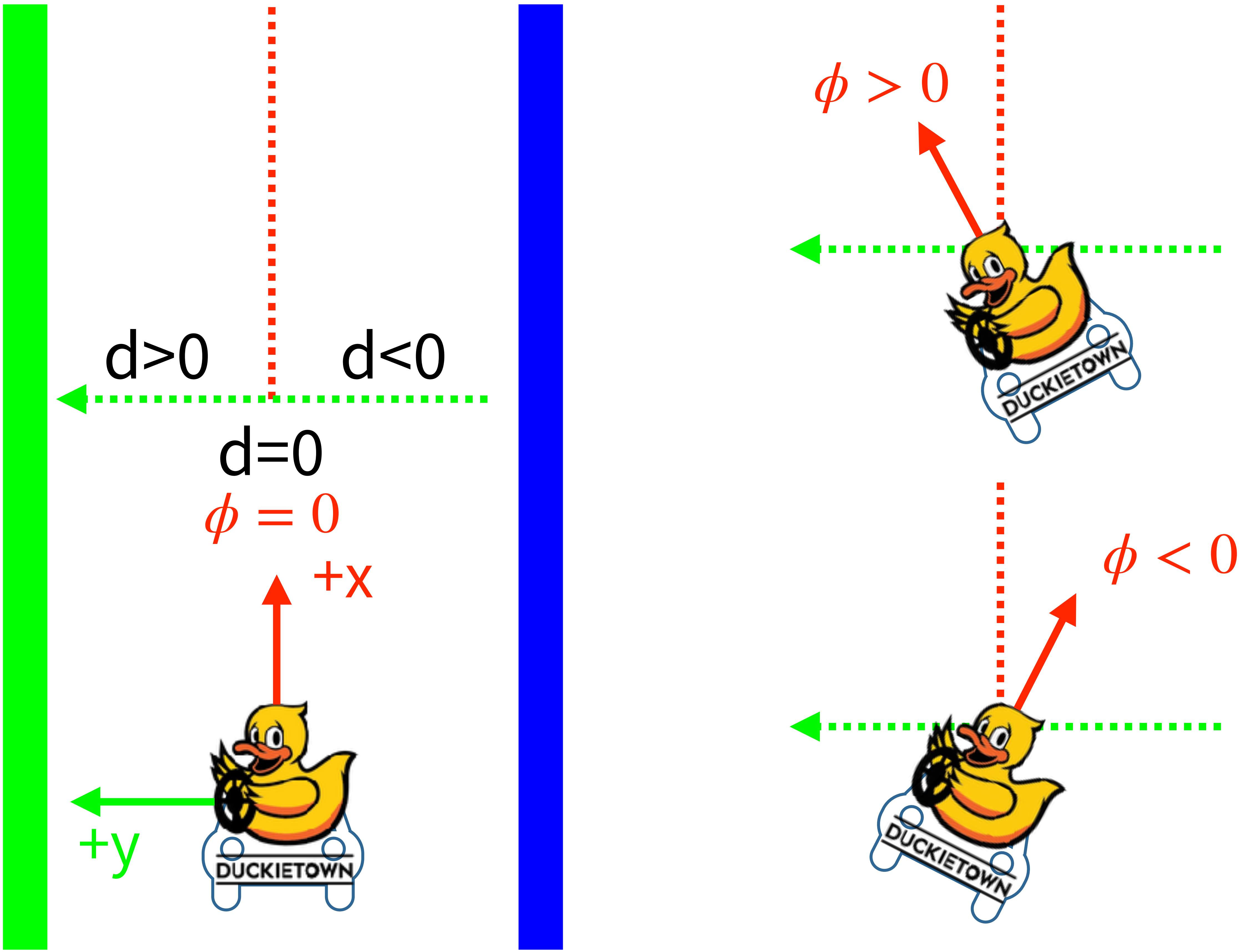
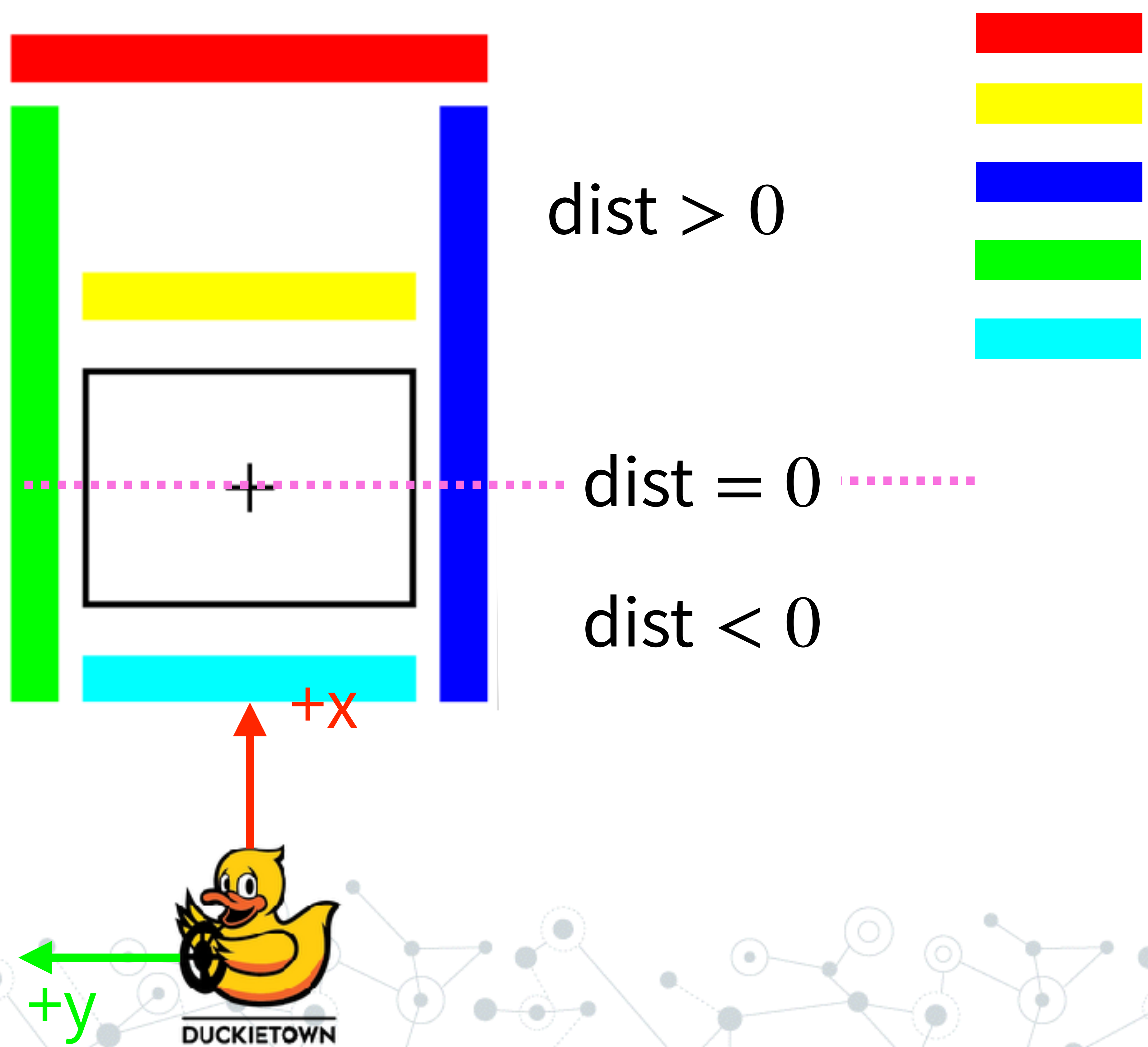
Lab11: 工人智慧與自駕車- Lane Filters

Stanley Chai

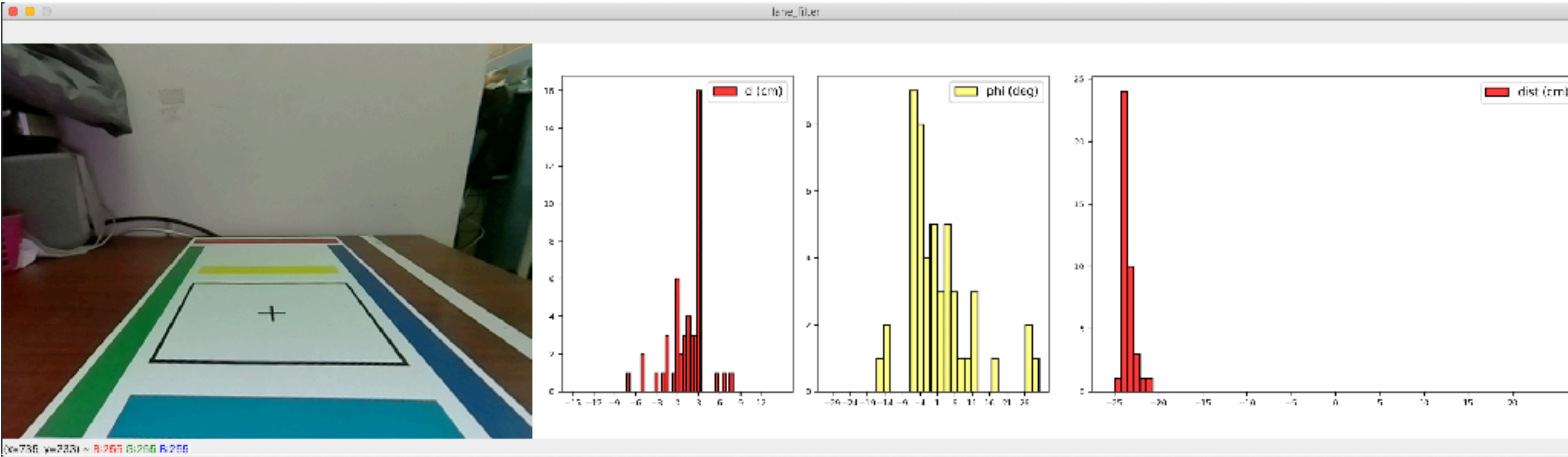
小鴨車座標系與路面座標系定義

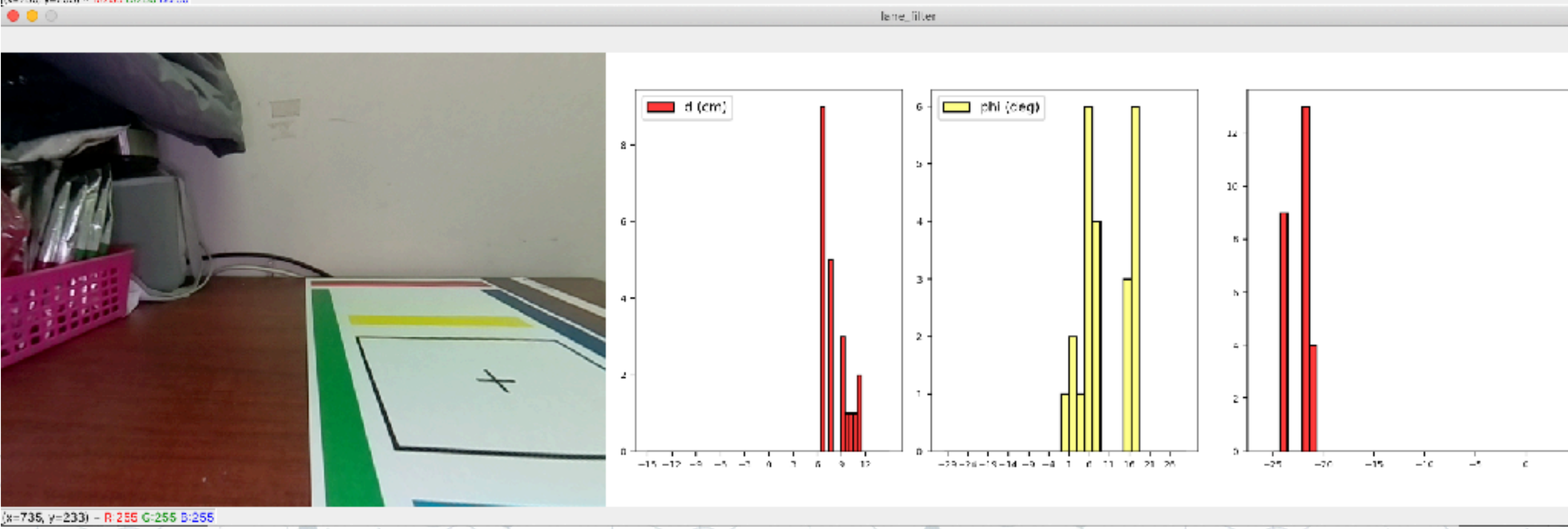


小鴨車座標系與停止線座標系定義

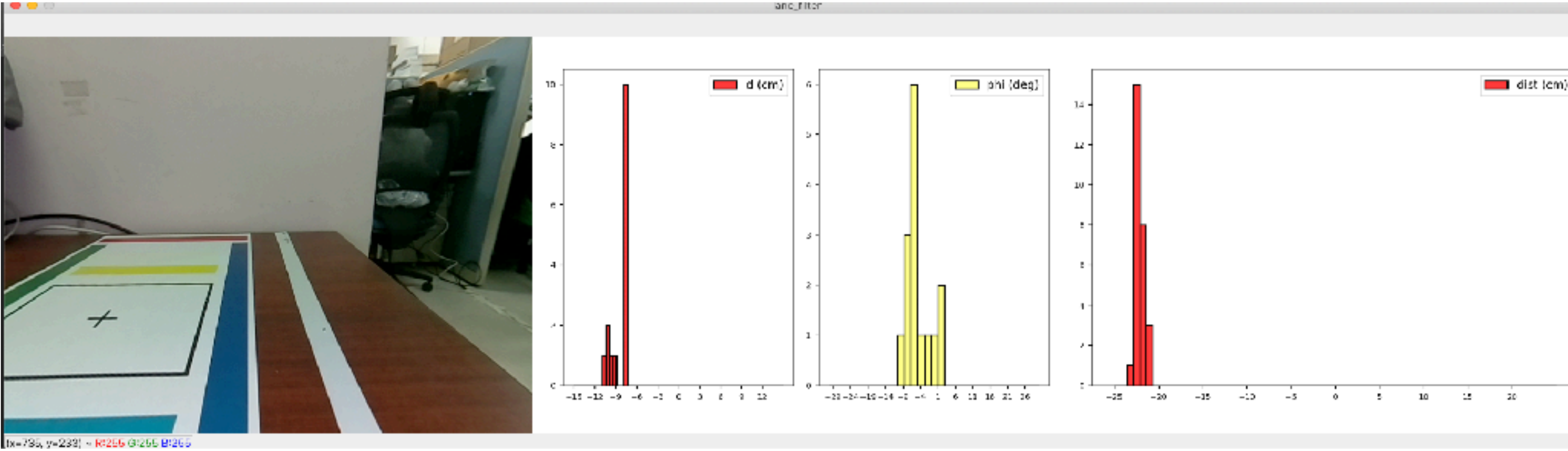


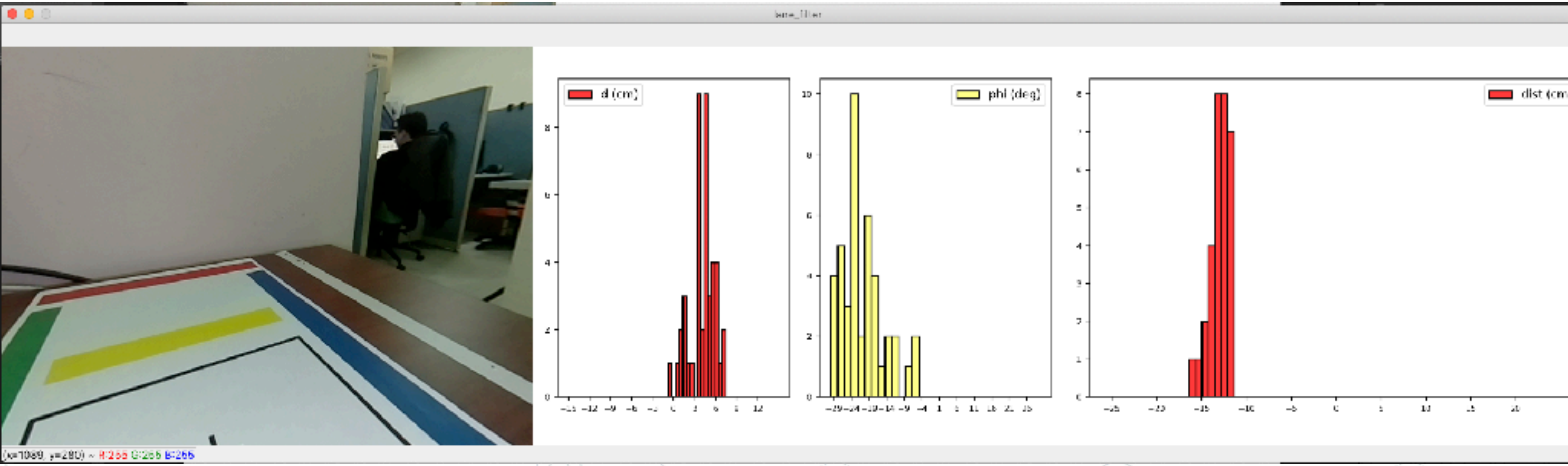
Lane Filter


lane_filter
d (cm)
phi (deg)
(x=735, y=233) ~ R:255 G:255 B:255

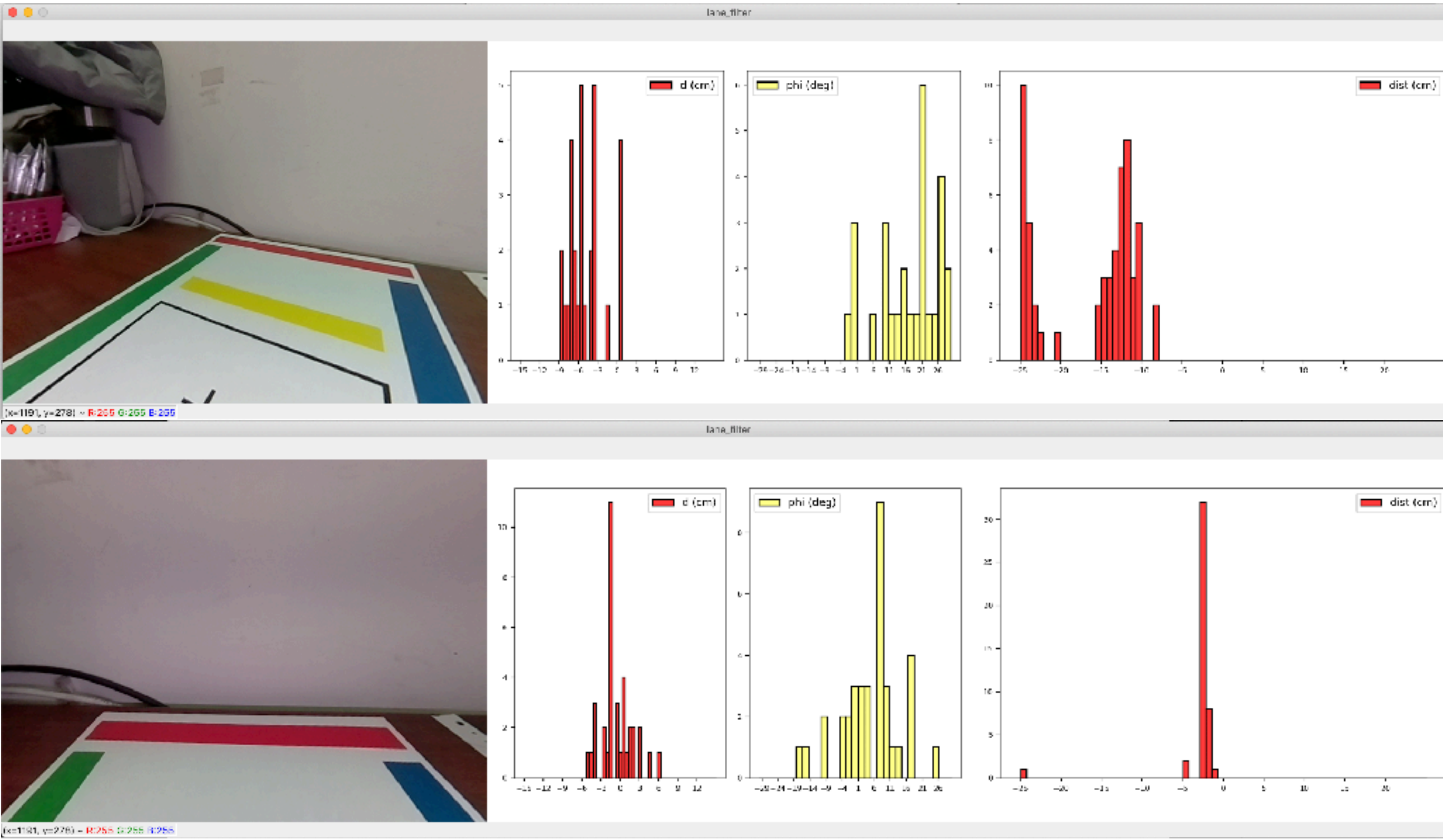

lane_filter
d (cm)
phi (deg)
(x=735, y=233) ~ R:255 G:255 B:255

Lane Filter



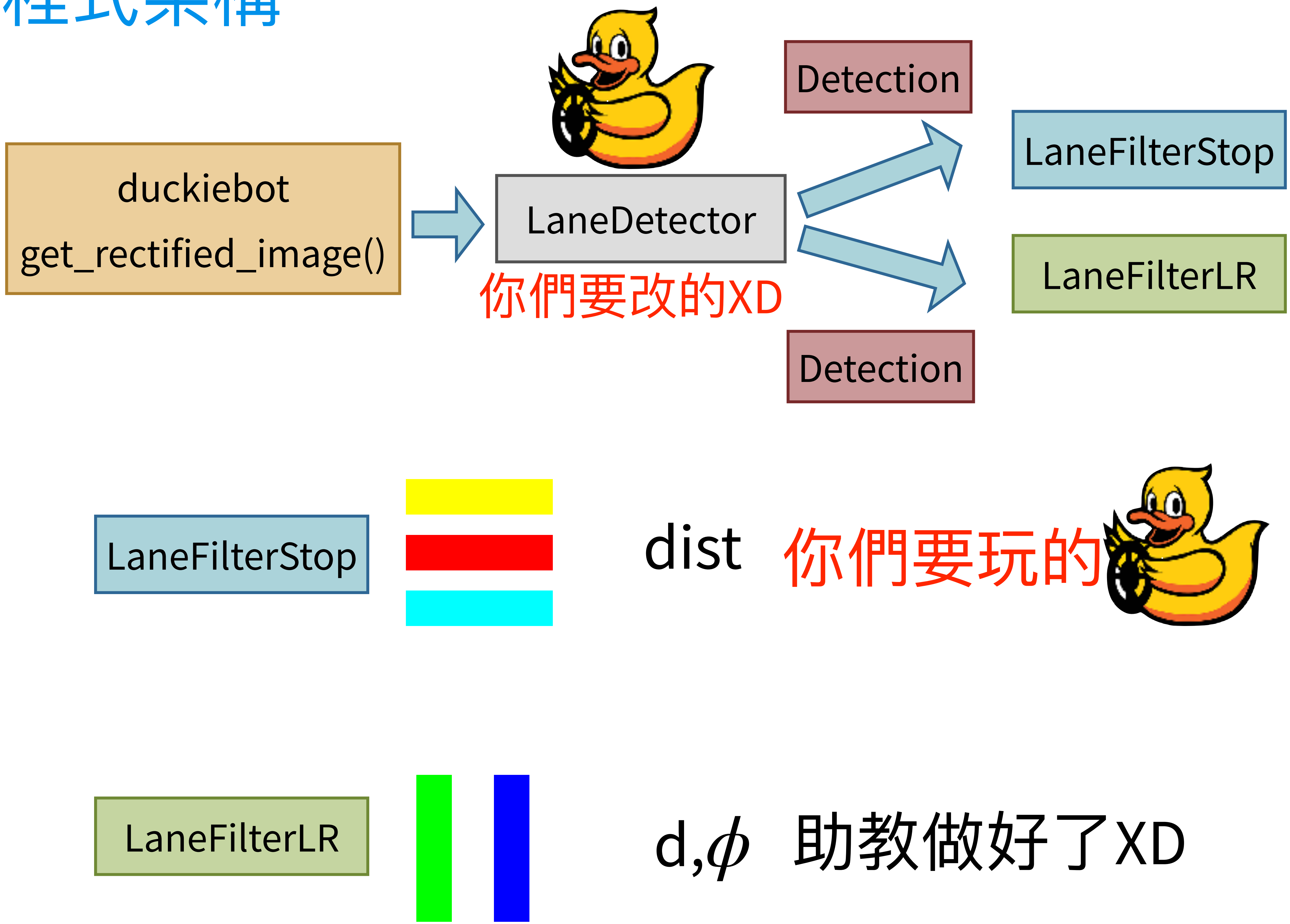


Lane Filter



lane_filter
d (cm)
phi (deg)
dist (cm)
lane_filter
d (cm)
phi (deg)
dist (cm)

程式架構

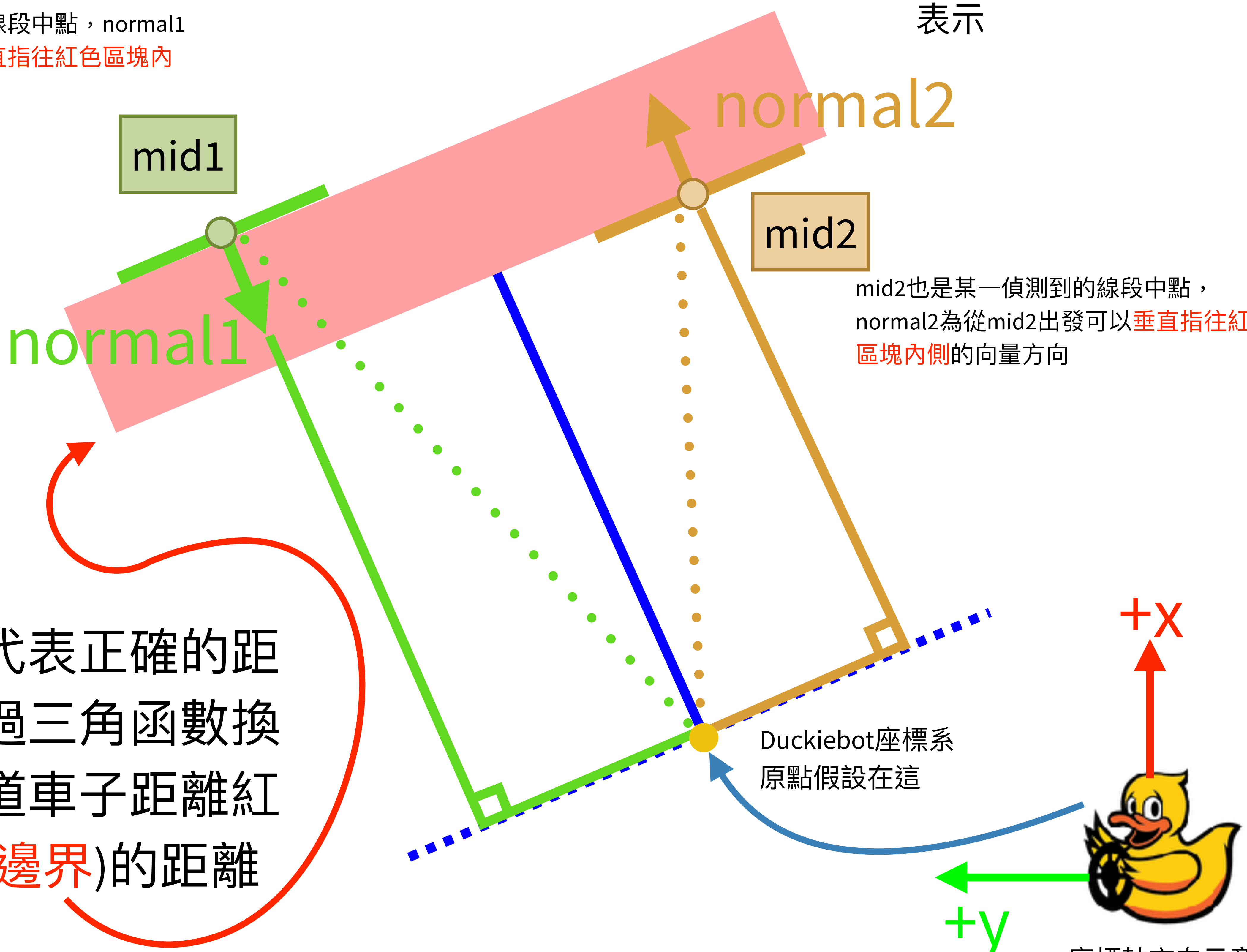


duckiebot
get_rectified_image()
LaneDetector
你們要改的XD
Detection
LaneFilterStop
LaneFilterLR
Detection
LaneFilterStop
dist
你們要玩的
LaneFilterLR
d,ϕ
助教做好了XD

Normal and Midpoint- Ideal

mid1/2 與normal 1/2 都是用duckiebot的X-Y座標系表示

mid1為某一偵測到的線段中點，normal1為從mid1出發可以垂直指往紅色區塊內側的向量方向



mid2也是某一偵測到的線段中點，normal2為從mid2出發可以垂直指往紅色區塊內側的向量方向

Duckiebot座標系原點假設在這

座標軸方向示意

實心線段代表正確的距離值，透過三角函數換算可以知道車子距離紅線(靠下側邊界)的距離

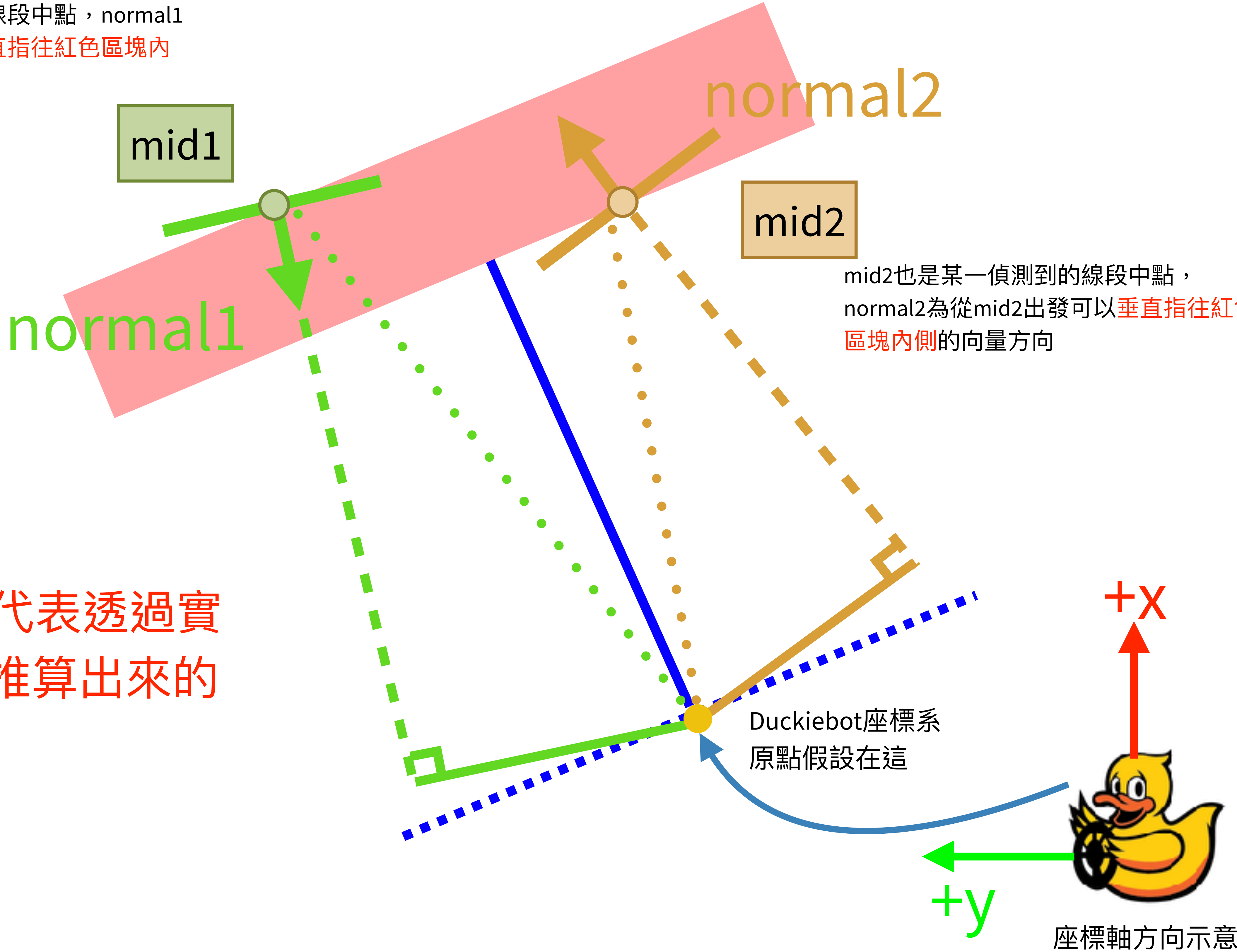
mid1+normal1投影的長度與mid2+normal2投影的長度**多一個紅線寬XD，記得減掉才會一致**

Normal and Midpoint- Reality

mid1為某一偵測到的線段中點，normal1
為從mid1出發可以垂直指往紅色區塊內
側的向量方向

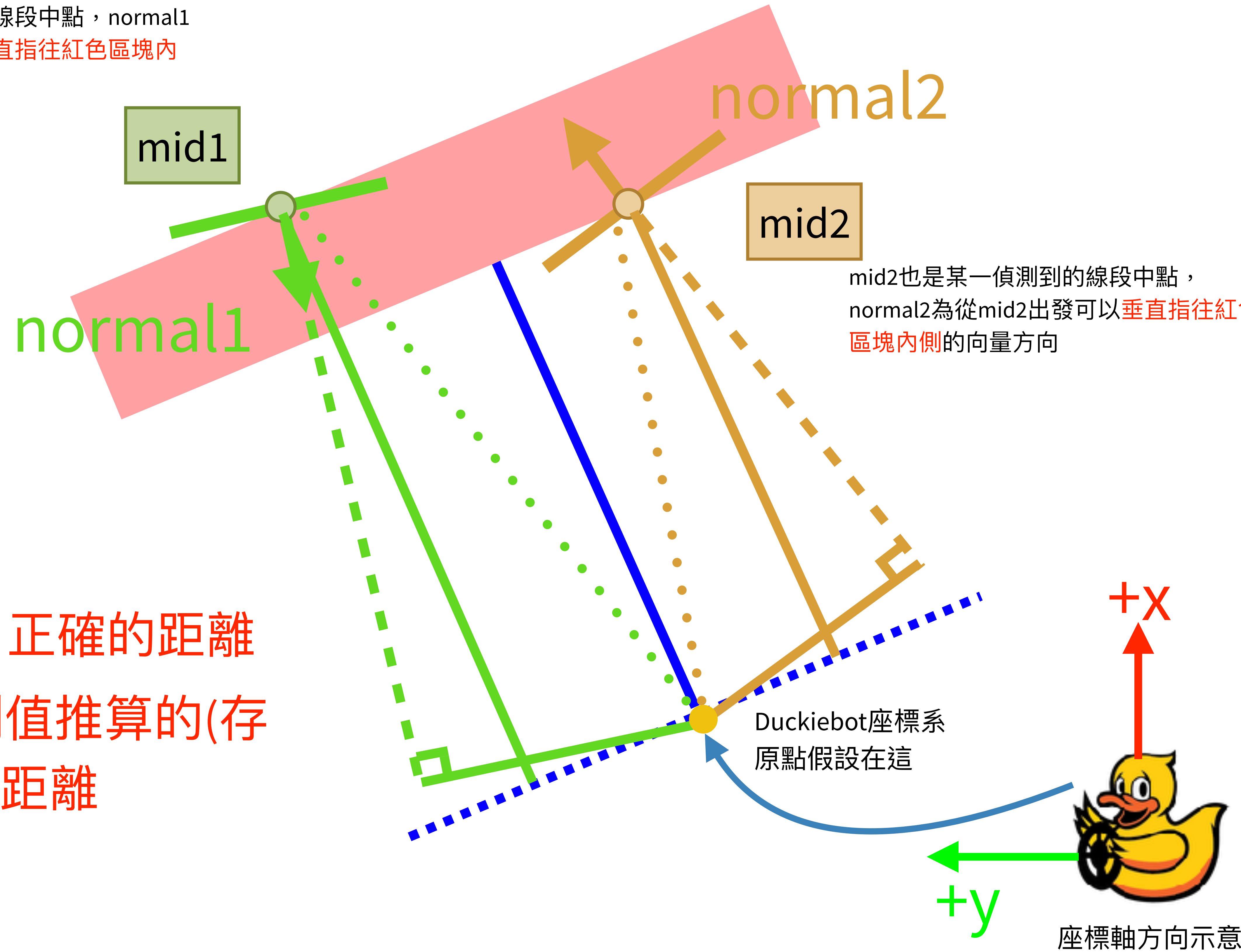
mid2也是某一偵測到的線段中點，
normal2為從mid2出發可以垂直指往紅色
區塊內側的向量方向

- - -線段代表透過實
際觀測值推算出來的
距離長度



Normal and Midpoint- Error

mid1為某一偵測到的線段中點，normal1為從mid1出發可以垂直指往紅色區塊內側的向量方向



mid2也是某一偵測到的線段中點，normal2為從mid2出發可以垂直指往紅色區塊內側的向量方向

實心線段: 正確的距離

- - - :觀測值推算的(存在誤差的)距離

mid1/2 與normal 1/2 都是用duckiebot的X-Y座標系表示

Voting的概念

- ◎透過觀察很多組線段，來投票選出**最佳的可能長度(例如每0.5公分當作一種可能來判斷)**
- ◎能避免受極端值的錯誤影響
  - 例如因為顏色判斷錯誤，導致少數的線段點算出來的距離值與大家差很多，如果單純用平均來算可能會影響最後的結果

lane_filter_stop.py

紅色和黃色的case

```python
if color in ['red', 'yellow']:
    """
    HW11 Hints: Your codes will only use the following variables
    mids_[top/bottom],normals_[top/bottom]: Both arrays have size Nx2, N represents the detected number of line segments
    self.params.marker_width[color]: width of the color chunks
    self.params.line_dist[color]: distance of the bottom edge to the parking center (BLACK CROSS ON THE MAP)

    Mids: Mid-points of the tangent vectors
    Normals: Normal vectors heading to the color chunk
    You are handling multiple detected lines at the same time, DON'T USE for-loop IF POSSIBLE (Try it)

    You need to calculate the distance to the parking center
    If the car center is BELOW the parking center, the distance value should be NEGATIVE
    If the car center is ABOVE the parking center, the distance value should be POSITIVE
    dist2parking is an array with length N with all the distance you estimated using mids and normals
    """

    if valid_top_select.sum() > 0:
        mids_top = mids[valid_top_select]
        normals_top = normals[valid_top_select]
        """
        YOUR CODE HERE, Calculate the correct dist2parking with the same length as the normals_top
        """
        # Start >>> Your code here
        dist2parking = np.zeros(len(normals_top))  # Fake result, write your own correct one
        # End <<< Your code here
        votes_dist.append(dist2parking)
    if valid_bottom_select.sum() > 0:
        mids_bottom = mids[valid_bottom_select]
        normals_bottom = normals[valid_bottom_select]
        """
        YOUR CODE HERE, Calculate the correct dist2parking with the same length as the normals_bottom
        """
        # Start >>> Your code here
        dist2parking = np.zeros(len(normals_bottom))  # Fake result, write your own correct one
        # End <<< Your code here
        votes_dist.append(dist2parking)
```

註解不是助教無聊寫爽的，請讀一下

mids 與normals都是用duckiebot的X-Y座標系表示

這段是mid1+Normal1的case

mids 與normals都是用duckiebot的X-Y座標系表示

這段是mid2+Normal2的case

lane_filter_stop.py

註解翻譯: 青色的case 一定有詐，不然為何分開處理?

```python
else:  # cyan, ... ide is closer to the center ?
    if valid_top_select.sum() > 0:
        mids_top = mids[valid_top_select]
        normals_top = normals[valid_top_select]
        """
        YOUR CODE HERE, Calculate the correct dist2parking with the same length as the normals_top
        """
        # Start >>> Your code here
        # Just a fake result, write your own correct one
        dist2parking = np.zeros(len(normals_top))
        # End <<< Your code here
        votes_dist.append(dist2parking)
    if valid_bottom_select.sum() > 0:
        mids_bottom = mids[valid_bottom_select]
        normals_bottom = normals[valid_bottom_select]
        """
        YOUR CODE HERE, Calculate the correct dist2parking with the same length as the normals_bottom
        """
        # Start >>> Your code here
        # Just a fake result, write your own correct one
        dist2parking = np.zeros(len(normals_bottom))
        # End <<< Your code here
        votes_dist.append(dist2parking)
```

mids 與normals都是用duckiebot的X-Y座標系表示

這段是mid1+Normal1的case

mids 與normals都是用duckiebot的X-Y座標系表示

這段是mid2+Normal2的case

助教碎碎念

◎紅/黃還有青色線段都不會直接在你的正前方出現，因此才會有角度問題需要計算

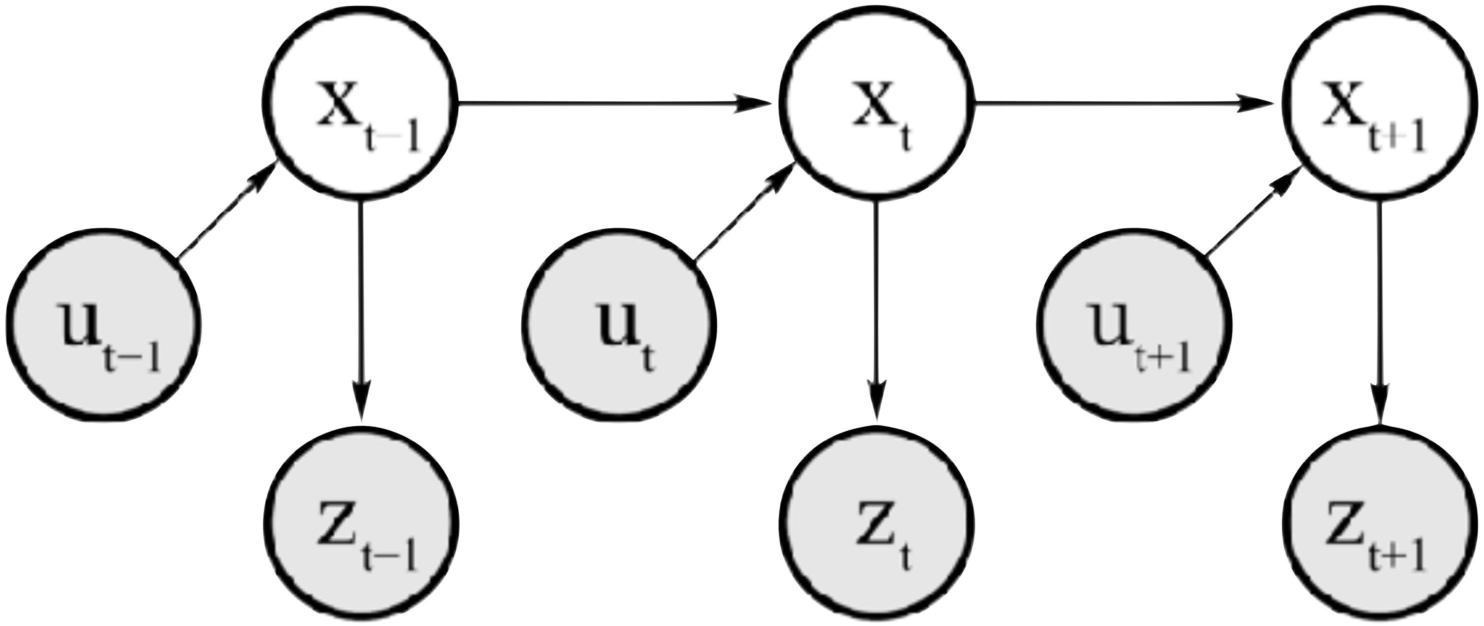
◎程式碼有點複雜，因為用到Bayes Filter的想法，一個簡單的範例與數學定義(不含推導)放在附錄中，晚上失眠時可以讀一下，就知道為什麼程式裡會有prediction與posterior_update這兩個function

Bayes Filter Notes

Random variables

- X is a random variable, x is a specific value that X may happen with probability $p(X = x)$
  - Upper case: random variable (r.v.) => A function
  - Lower case: specific value of a r.v.
  - E.g., A Coin with it's face X, $p(X = \text{head}) = 0.6$
    $p(X = \text{tail}) = 0.4$
- $$\sum_{x \in \{\text{head}, \text{tail}\}} p(X = x) = 1$$

State, Action and Observation



- Robot (internal) state: x_t (Not directly observable)
- Root action: u_t
- Sensor observation: z_t
- A robot at state x_t takes an action u_t and then gets an observation z_t afterwards.

Prior and Posterior

- x is a quantity that we want to infer from z
  - x is the robot position
  - z is the sensor measurement from ultrasonic sensor
- Prior probability distribution: $p(x)$
  - Summary for x before new measurement z
- Posterior probability distribution over x: $p(x \mid z)$
  - Summary for x after observing z
- Generative model: $p(z \mid x)$
  - How current state x causes sensor measurement z

Bayes Rule

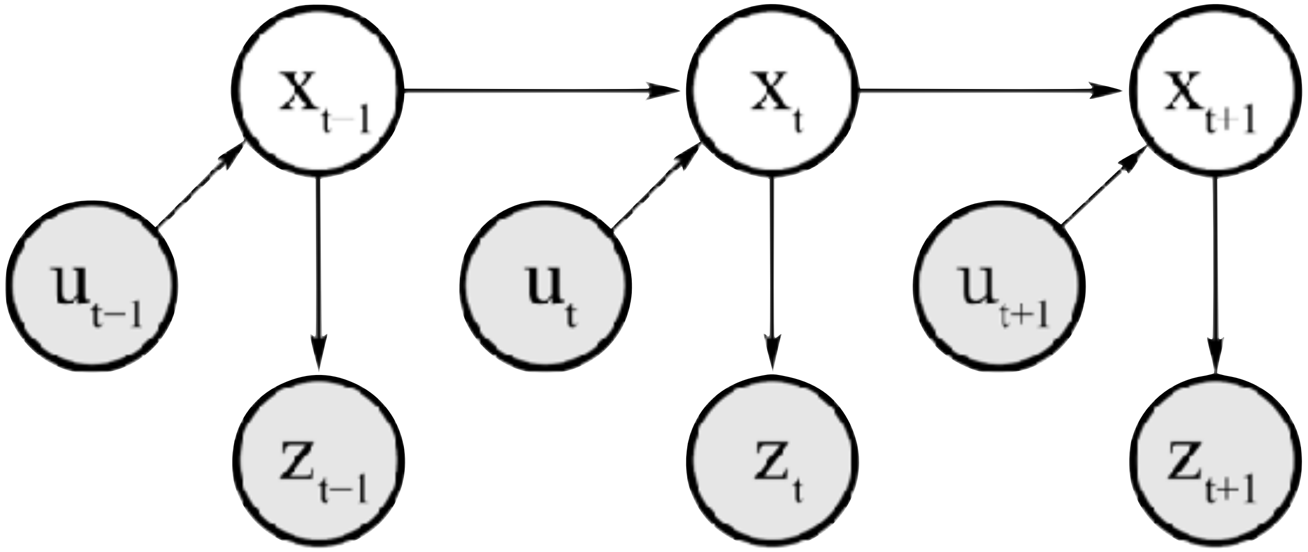
- $$p(x \mid z) = \frac{p(z \mid x)p(x)}{p(z)} = \eta p(z \mid x)p(x)$$
  - η is the normalizer, z is a given value
  - $p(z \mid x)p(x)$ can be calculated easily for all possibilities of x
  - η can be solved using the fact: $\sum_{x_i} p(x_i \mid z) = 1$
- *Condition on one more variable y
  - $$p(x \mid y, z) = \frac{p(x, y, z)}{p(y, z)} = \frac{p(x, y, z)/p(y)}{p(y, z)/p(y)}$$
    $$= \frac{p(x, y, z)/p(x, y)p(x \mid y)}{p(z \mid y)}$$
    $$= \frac{p(z \mid x, y)p(x \mid y)}{p(z \mid y)}, \quad \text{assume } p(z \mid y) > 0$$

State, Action and Observation

- u_t is a deterministic action
- State transition probability: $p(x_t \mid x_{t-1}, u_t)$
- Measurement probability: $p(z_t \mid x_t)$

Belief State

- A **belief** reflects the robot's knowledge about it's true state
- **True** state (not directly measurable) : x_t
- **Belief** state distribution
  - Prediction (after u_t): $\overline{bel}(x_t) \doteq p(x_t \mid z_{1:t-1}, u_{1:t})$
  - Posterior (after u_t, observe z_t): $bel(x_t) \doteq p(x_t \mid z_{1:t}, u_{1:t})$
- TODO: Assumptions for u and z



Discrete Bayes Filter

- Given $bel(x_{t-1}), u_t, z_t$
- **Prediction**:

$$\overline{bel}(x_t) = \sum_{x_{t-1}} p(x_t \mid u_t, x_{t-1})\, bel(x_{t-1})$$

- **Posterior update**:

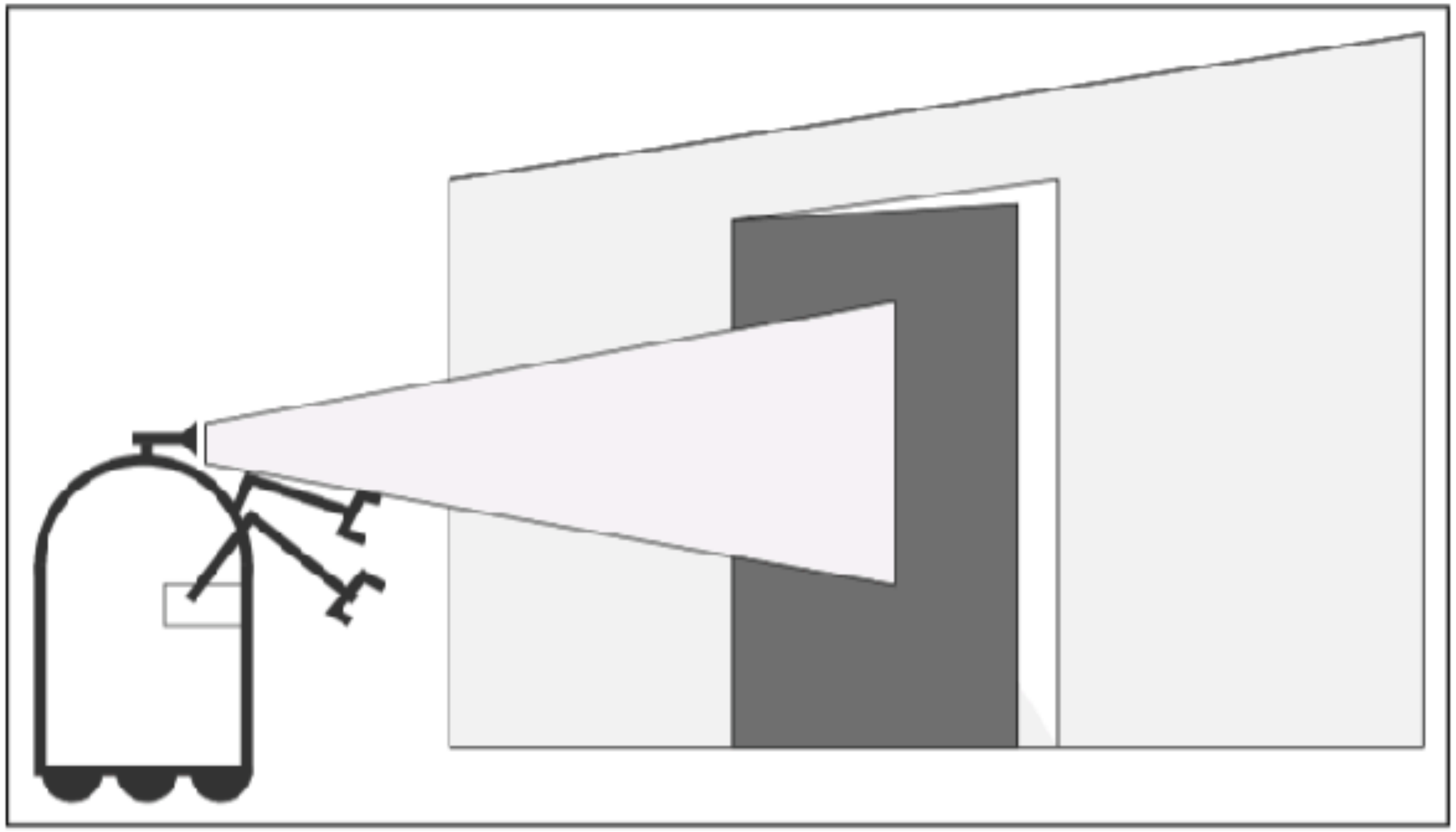
$$bel(x_t) = \eta\, p(z_t \mid x_t)\, \overline{bel}(x_t)$$

$$\sum_{x_t} bel(x_t) = 1 \rightarrow \text{for } \eta$$

- Initial condition: $bel(x_0) = p(x_0)$

Example: Robot Door Manipulation

- Door states:
  - is_open
  - is_closed
- Sensor results
  - sense_open
  - sense_closed
- Robotic actions
  - none
  - push



Example: Robot Door Manipulation

- Initial (True) state (we don't know): X_0
- Initial belief (what we guess):
  - $bel(X_0 = \text{is\_open}) = bel(X_0 = \text{is\_closed}) = 0.5$

- Sensor measurement probabilities
  - $p(Z_t = \text{sense\_open} \mid X_t = \text{is\_open}) = 0.6$
    $p(Z_t = \text{sense\_closed} \mid X_t = \text{is\_open}) = 0.4$
    $p(Z_t = \text{sense\_open} \mid X_t = \text{is\_closed}) = 0.2$
    $p(Z_t = \text{sense\_closed} \mid X_t = \text{is\_closed}) = 0.8$

Example: Robot Door Manipulation

- State transition probabilities (by actions u_t): Our model for robot-door interactions

$$p(X_t = \text{is\_open} \mid U_t = \text{push}, X_{t-1} = \text{is\_open}) = 1$$
$$p(X_t = \text{is\_closed} \mid U_t = \text{push}, X_{t-1} = \text{is\_open}) = 0$$
$$p(X_t = \text{is\_open} \mid U_t = \text{push}, X_{t-1} = \text{is\_closed}) = 0.8$$
$$p(X_t = \text{is\_closed} \mid U_t = \text{push}, X_{t-1} = \text{is\_closed}) = 0.2$$

$$p(X_t = \text{is\_open} \mid U_t = \text{none}, X_{t-1} = \text{is\_open}) = 1$$
$$p(X_t = \text{is\_closed} \mid U_t = \text{none}, X_{t-1} = \text{is\_open}) = 0$$
$$p(X_t = \text{is\_open} \mid U_t = \text{none}, X_{t-1} = \text{is\_closed}) = 0$$
$$p(X_t = \text{is\_closed} \mid U_t = \text{none}, X_{t-1} = \text{is\_closed}) = 1$$

T=1: $u_1 = \text{do\_nothing}, z_1 = \text{sense\_open}$

- **Prediction:** $\overline{bel}(x_1) = \sum_{x_0} p(x_1 \mid u_1, x_0)\, bel(x_0)$

$$\overline{bel}(X_1 = \text{is\_open})$$
$$= p(X_1 = \text{is\_open} \mid U_1 = \text{none}, X_0 = \text{is\_open})\, bel(X_0 = \text{is\_open})$$

- $$+p(X_1 = \text{is\_open} \mid U_1 = \text{none}, X_0 = \text{is\_closed})\, bel(X_0 = \text{is\_closed})$$

$$= 1 \times 0.5 + 0 \times 0.5 = 0.5$$

$$\overline{bel}(X_1 = \text{is\_closed})$$
$$= p(X_1 = \text{is\_closed} \mid U_1 = \text{none}, X_0 = \text{is\_open})\, bel(X_0 = \text{is\_open})$$

- $$+p(X_1 = \text{is\_closed} \mid U_1 = \text{none}, X_0 = \text{is\_closed})\, bel(X_0 = \text{is\_closed})$$

$$= 0 \times 0.5 + 1 \times 0.5 = 0.5$$

T=1: $u_1 = \text{do\_nothing}, z_1 = \text{sense\_open}$

- **Posterior update**: $bel(x_1) = \eta\, p(z_1 \mid x_1)\, \overline{bel}(x_1)$

  $bel(X_1 = \text{is\_open})$

- $= \eta\, p(Z_1 = \text{sense\_open} \mid X_1 = \text{is\_open})\, \overline{bel}(X_1 = \text{is\_open})$

  $= \eta\, 0.6 \times 0.5 = 0.3\,\eta$

  $bel(X_1 = \text{is\_closed})$

- $= \eta\, p(Z_1 = \text{sense\_open} \mid X_1 = \text{is\_closed})\, \overline{bel}(X_1 = \text{is\_closed})$

  $= \eta\, 0.2 \times 0.5 = 0.1\,\eta$

- $0.3\,\eta + 0.1\,\eta = 1, \eta = 2.5 \rightarrow$ $bel(X_1 = \text{is\_open}) = 0.75$, $bel(X_1 = \text{is\_closed}) = 0.25$

T=2: $u_2 = \text{push}, z_2 = \text{sense\_open}$

- **Prediction**
  - $\overline{bel}(X_2 = \text{is\_open}) = 1 \times 0.75 + 0.8 \times 0.25 = 0.95$
  - $\overline{bel}(X_2 = \text{is\_closed}) = 0 \times 0.75 + 0.2 \times 0.25 = 0.05$
- **Posterior update**
  - $bel(X_2 = \text{is\_open}) = \eta\, 0.6 \times 0.95$
  - $bel(X_2 = \text{is\_closed}) = \eta\, 0.2 \times 0.05$
  - $\eta \approx 1.724 \rightarrow \begin{array}{l} bel(X_2 = \text{is\_open}) \approx 0.983 \\ bel(X_2 = \text{is\_closed}) \approx 0.017 \end{array}$