Xenobot technical report

Shengwen Cheng

January 2017

1 Introduction

Xenobot is a computer vision based self-driving system inspired by MIT duckietown project. We hope to do the real-time robotics research or add new customized functions after re-implementing the system.

The algorithm we are using is a modified version of MIT duckietown, so you can find many similarities between two projects. This report is focus on how our lane following algorithm works.

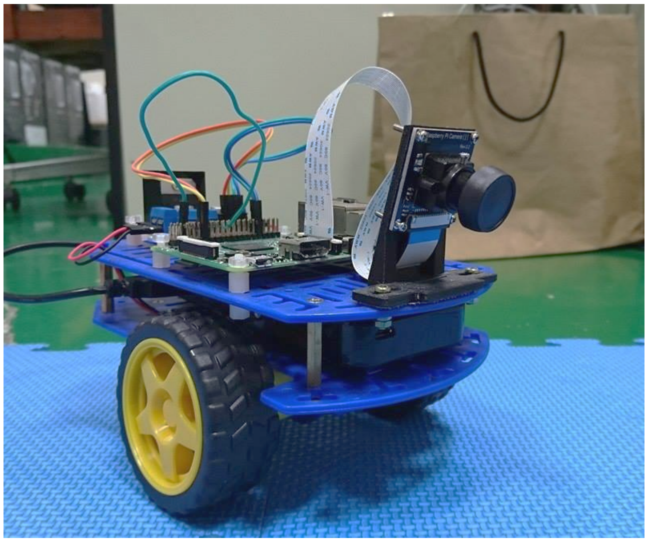

Figure 1: Xenobot

2 Lane pose estimation

2.1 Assumptions of the environment

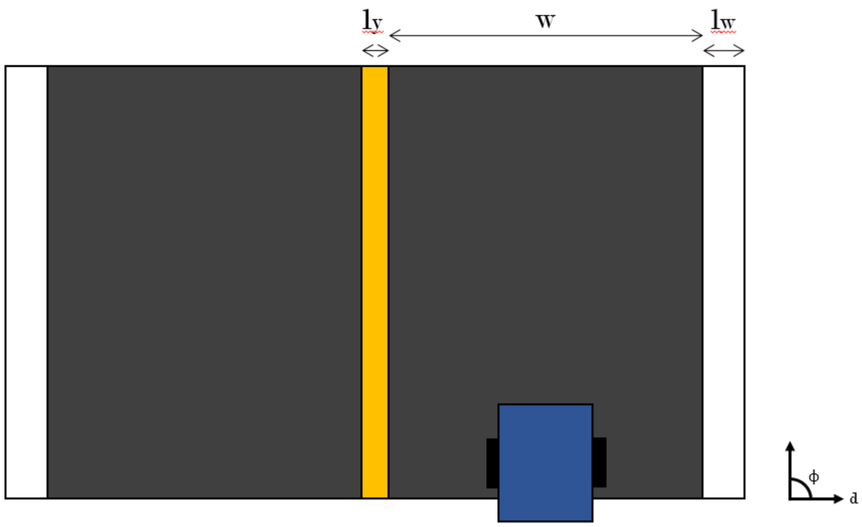

Figure 2: lane

2.2 Segements detection

2.3 Coordinate transformation

2.4 Segment side recognition

We obtained the lane segments during the lane detection process. Next step to do is to figure out the side of the segment on the lane mark. We can determine it by reading multiple pixel values in the direction of the segment normal vector on the color thresholding image.

2.5 Lane pose estimation for single segment

3 PID controller

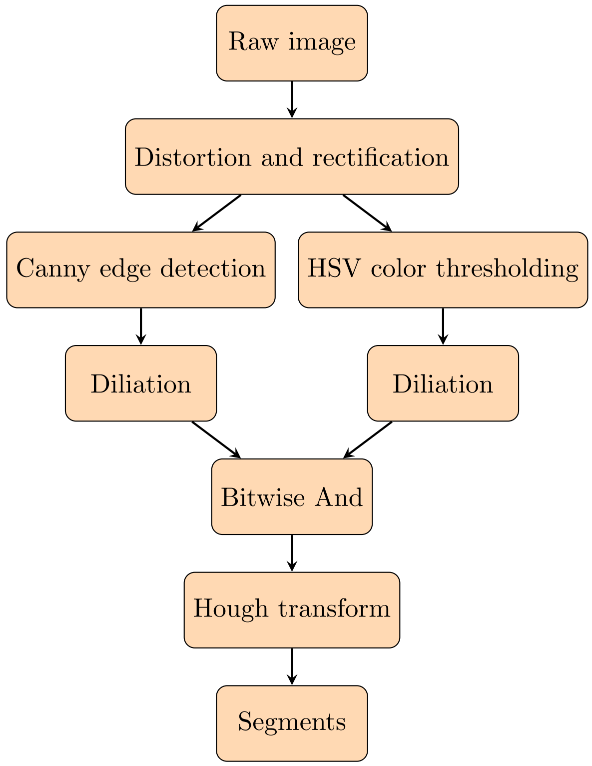

Figure 3: Work flow of lane detector

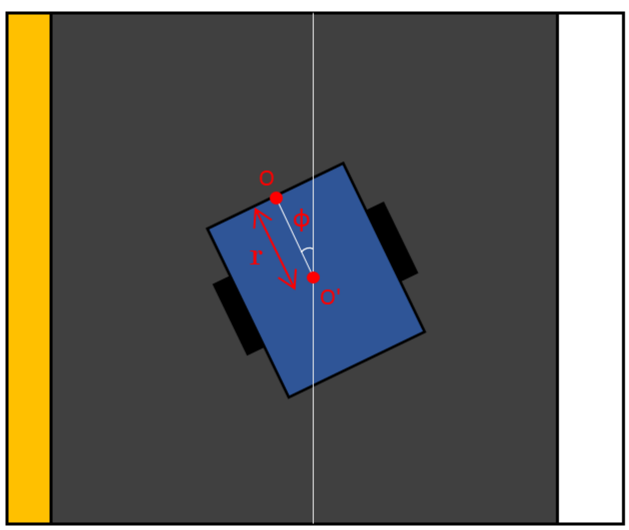

Figure 4: Center of camera and the car

Figure 5: segment side

Algorithm 1: Segment side recognition

Data: segment, accumulator threshold, color binarization image
Result: side (left or right)

$\vec{P_1} = (x_1, y_1)$
$\vec{P_2} = (x_2, y_2)$
$\vec{P} = \frac{\vec{P_1}+\vec{P_2}}{2}$
$\vec{t} = \frac{\vec{P_2}-\vec{P_1}}{\left\|\vec{P_2}-\vec{P_1}\right\|}$
$\vec{n} = (-y_t, x_t)$
for $i < pixel\ count$ **do**
    $x \leftarrow \lceil x_p + x_n \cdot i \rceil$
    $y \leftarrow \lceil y_p + y_n \cdot i \rceil$
    if $I(x, y) = I_{max}$ **then**
        $left \leftarrow left + 1$
    $x \leftarrow \lfloor x_p - x_n \cdot i \rfloor$
    $y \leftarrow \lfloor y_p - y_n \cdot i \rfloor$
    if $I(x, y) = I_{max}$ **then**
        $right \leftarrow right + 1$
end
if $left > threshold$ & $right < threshold$ **then**
    return is left
else if $right > threshold$ & $left < threshold$ **then**
    return is right
else
    return unknown side

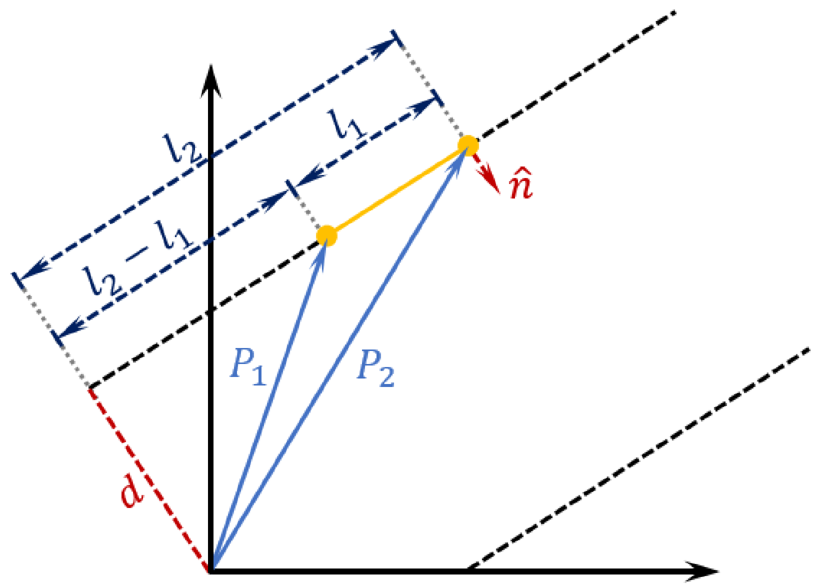

Figure 6: Geometry relationships

Algorithm 2: generate vote

Data: segment

Result: pose d and ϕ

$\vec{P_1} = (x_1, y_1)$

$\vec{P_2} = (x_2, y_2)$

$\vec{t} = \frac{\vec{P_2} - \vec{P_1}}{\left\| \vec{P_2} - \vec{P_1} \right\|}$

$\vec{n} = (-y_t, x_t)$

$\phi = \arctan(\frac{y_t}{x_t}) - \frac{\pi}{2}$

if *segment color = white* **then**

    if *edge side = right* **then**

        $\vec{k} = (\frac{w}{2} + l_w) \cdot \vec{n}$

    else

        $\vec{k} = (\frac{w}{2}) \cdot \vec{n}$

    end

else if *segment color = yellow* **then**

    if *edge side = left* **then**

        $\vec{k} = (-\frac{w}{2} - l_y) \cdot \vec{n}$

    else

        $\vec{k} = (-\frac{w}{2}) \cdot \vec{n}$

    end

$\vec{j} = (r \cdot \sin\phi, r \cdot \cos\phi)$

$P_1' = \vec{P_1} + \vec{k} - \vec{j}$

$P_2' = \vec{P_2} + \vec{k} - \vec{j}$

$d_1 = \vec{P_1} \cdot \vec{n}$

$d_2 = \vec{P_2} \cdot \vec{n}$

$d = \frac{d_1 + d_2}{2}$

Figure 7: Vote generation