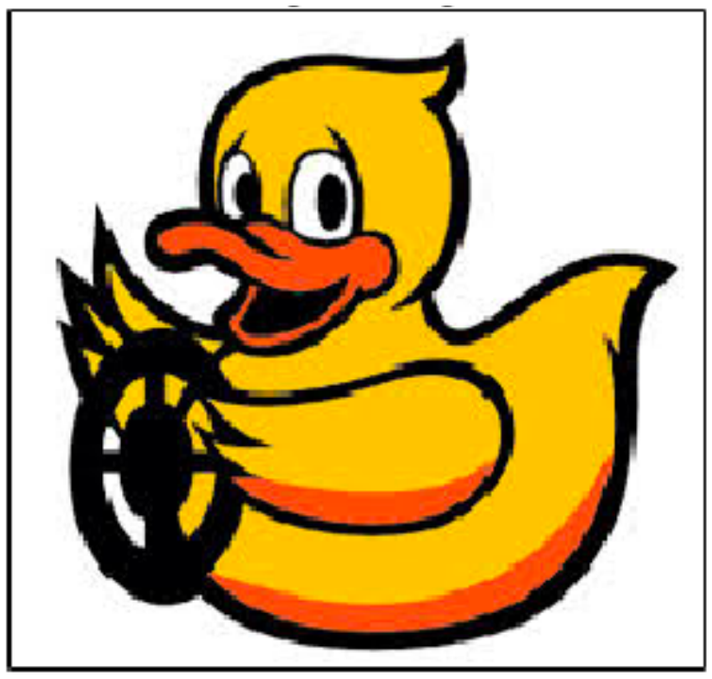
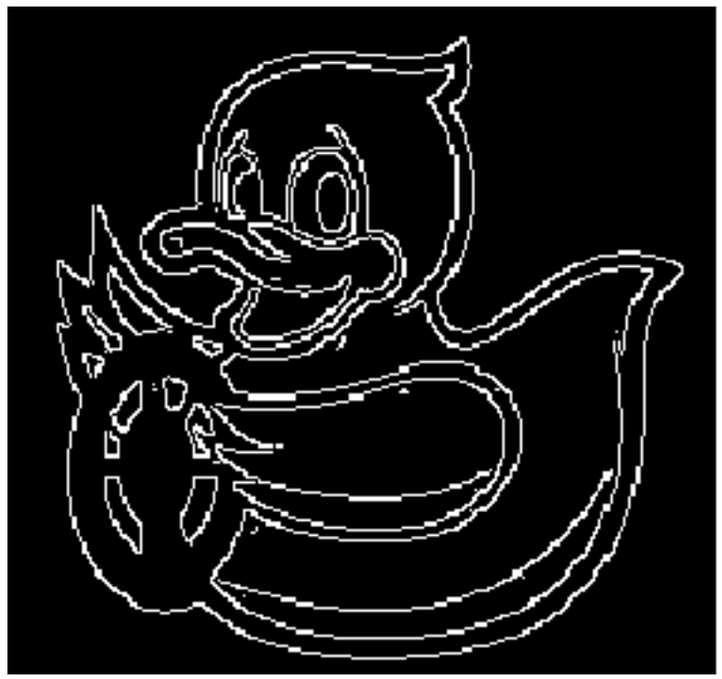
Lecture 4:
Python and OpenCV

呂承龍 Chen-Lung Eric Lu
高熙鈞 KaoCG
李宗儒 ZR,Li

Table of Content

- Python & Jupyter Notebook
- Line Detector & OpenCV
- Vehicle Detector, Face detector, Duckie Detector





Python & Jupyter Notebook

If I'm Jupyter

then you are my Sun

being the center of my world till the end of the time

by Jupyter Notebook

Jupyter

- A widely used high-level language
- Emphasize code readability (use whitespace or tab to delimit the code block)
- An interpreted language (no need to compile)
  直譯式
- Widely used in data science nowadays


python™

V.S.

V.S.

python™ v.s.

example.py

```python
import rospy
import numpy as np

print "This is a python code"

class test_class(object):
    """docstring for test_class"""
    def __init__(self):

        self.param_1 = 1
        self.param_2 = []

        res = self.demontration(5)
        print "param_1 = ", self.param_1
        print "res = ", res

    def demontration(self, n_loop):

        for i in range(0, n_loop):
            self.param_1 += i

        return i


if __name__=='__main__':
    obj = test_class()
```

example.cpp

```cpp
#include <stdio.h>
#include <iostream>
#include <ros/ros.h>

class test_class
{
public:
    test_class();
    ~test_class();
private:
    int16_t param_1;
    int16_t param_2[3] = {1, 2, 3};
};

test_class::test_class(){
    param_1 = 1;

    res = this.demonstration(5);
    printf("param_1 = %d\n", param_1);
    printf("res = %d\n", res);
}

uint8_t test_class::demonstration(uint8_t n_loop){

    uint8_t i = 0;

    for(i=0; i<n_loop; i++){
        param_1 += i;
    }

    return i
}

int main(){
    test_class testing();
    return 0;
}
```

python™ v.s.

import/include library

example.py

```python
import rospy
import numpy as np

print "This is a python code"

class test_class(object):
    """docstring for test_class"""
    def __init__(self):

        self.param_1 = 1
        self.param_2 = []

        res = self.demontration(5)
        print "param_1 = ", self.param_1
        print "res = ", res

    def demontration(self, n_loop):

        for i in range(0, n_loop):
            self.param_1 += i

        return i


if __name__=='__main__':
    obj = test_class()
```

example.cpp

```cpp
#include <stdio.h>
#include <iostream>
#include <ros/ros.h>

class test_class
{
public:
    test_class();
    ~test_class();
private:
    int16_t param_1;
    int16_t param_2[3] = {1, 2, 3};
};

test_class::test_class(){
    param_1 = 1;

    res = this.demonstration(5);
    printf("param_1 = %d\n", param_1);
    printf("res = %d\n", res);
}

uint8_t test_class::demonstration(uint8_t n_loop){

    uint8_t i = 0;

    for(i=0; i<n_loop; i++){
        param_1 += i;
    }

    return i
}

int main(){
    test_class testing();
    return 0;
}
```

python™ v.s. C++

import/include library

main function

example.py

```python
import rospy
import numpy as np

print "This is a python code"

class test_class(object):
    """docstring for test_class"""
    def __init__(self):

        self.param_1 = 1
        self.param_2 = []

        res = self.demontration(5)
        print "param_1 = ", self.param_1
        print "res = ", res

    def demontration(self, n_loop):

        for i in range(0, n_loop):
            self.param_1 += i

        return i


if __name__=='__main__':
    obj = test_class()
```

example.cpp

```cpp
#include <stdio.h>
#include <iostream>
#include <ros/ros.h>

class test_class
{
public:
    test_class();
    ~test_class();
private:
    int16_t param_1;
    int16_t param_2[3] = {1, 2, 3};
};

test_class::test_class(){
    param_1 = 1;

    res = this.demonstration(5);
    printf("param_1 = %d\n", param_1);
    printf("res = %d\n", res);
}

uint8_t test_class::demonstration(uint8_t n_loop){

    uint8_t i = 0;

    for(i=0; i<n_loop; i++){
        param_1 += i;
    }

    return i
}

int main(){
    test_class testing();
    return 0;
}
```

python™ v.s. C++

import/include library

class

main function

example.py

```python
import rospy
import numpy as np

print "This is a python code"

class test_class(object):
    """docstring for test_class"""
    def __init__(self):

        self.param_1 = 1
        self.param_2 = []

        res = self.demontration(5)
        print "param_1 = ", self.param_1
        print "res = ", res

    def demontration(self, n_loop):

        for i in range(0, n_loop):
            self.param_1 += i

        return i


if __name__=='__main__':
    obj = test_class()
```

example.cpp

```cpp
#include <stdio.h>
#include <iostream>
#include <ros/ros.h>

class test_class
{
public:
    test_class();
    ~test_class();
private:
    int16_t param_1;
    int16_t param_2[3] = {1, 2, 3};
};

test_class::test_class(){
    param_1 = 1;

    res = this.demonstration(5);
    printf("param_1 = %d\n", param_1);
    printf("res = %d\n", res);
}

uint8_t test_class::demonstration(uint8_t n_loop){

    uint8_t i = 0;

    for(i=0; i<n_loop; i++){
        param_1 += i;
    }

    return i
}

int main(){
    test_class testing();
    return 0;
}
```

python™ v.s. C++

example.py

```python
import rospy
import numpy as np

print "This is a python code"

class test_class(object):
    """docstring for test_class"""
    def __init__(self):

        self.param_1 = 1
        self.param_2 = []

        res = self.demontration(5)
        print "param_1 = ", self.param_1
        print "res = ", res

    def demontration(self, n_loop):

        for i in range(0, n_loop):
            self.param_1 += i

        return i


if __name__=='__main__':
    obj = test_class()
```

import/include library

class

class declare

main function

example.cpp

```cpp
#include <stdio.h>
#include <iostream>
#include <ros/ros.h>

class test_class
{
public:
    test_class();
    ~test_class();
private:
    int16_t param_1;
    int16_t param_2[3] = {1, 2, 3};
};

test_class::test_class(){
    param_1 = 1;

    res = this.demonstration(5);
    printf("param_1 = %d\n", param_1);
    printf("res = %d\n", res);
}

uint8_t test_class::demonstration(uint8_t n_loop){

    uint8_t i = 0;

    for(i=0; i<n_loop; i++){
        param_1 += i;
    }

    return i
}

int main(){
    test_class testing();
    return 0;
}
```

python™ v.s. C++

import/include library

class

class declare

class define

main function

example.py

```python
import rospy
import numpy as np

print "This is a python code"

class test_class(object):
    """docstring for test_class"""
    def __init__(self):

        self.param_1 = 1
        self.param_2 = []

        res = self.demontration(5)
        print "param_1 = ", self.param_1
        print "res = ", res

    def demontration(self, n_loop):

        for i in range(0, n_loop):
            self.param_1 += i

        return i


if __name__=='__main__':
    obj = test_class()
```

example.cpp

```cpp
#include <stdio.h>
#include <iostream>
#include <ros/ros.h>

class test_class
{
public:
    test_class();
    ~test_class();
private:
    int16_t param_1;
    int16_t param_2[3] = {1, 2, 3};
};

test_class::test_class(){
    param_1 = 1;

    res = this.demonstration(5);
    printf("param_1 = %d\n", param_1);
    printf("res = %d\n", res);
}

uint8_t test_class::demonstration(uint8_t n_loop){

    uint8_t i = 0;

    for(i=0; i<n_loop; i++){
        param_1 += i;
    }

    return i
}

int main(){
    test_class testing();
    return 0;
}
```

python™ v.s. C++

import/include library

class

class declare

class define

main function

example.py

```python
import rospy
import numpy as np

print "This is a python code"

class test_class(object):
    """docstring for test_class"""
    def __init__(self):

        self.param_1 = 1
        self.param_2 = []

        res = self.demontration(5)
        print "param_1 = ", self.param_1
        print "res = ", res

    def demontration(self, n_loop):

        for i in range(0, n_loop):
            self.param_1 += i

        return i


if __name__=='__main__':
    obj = test_class()
```

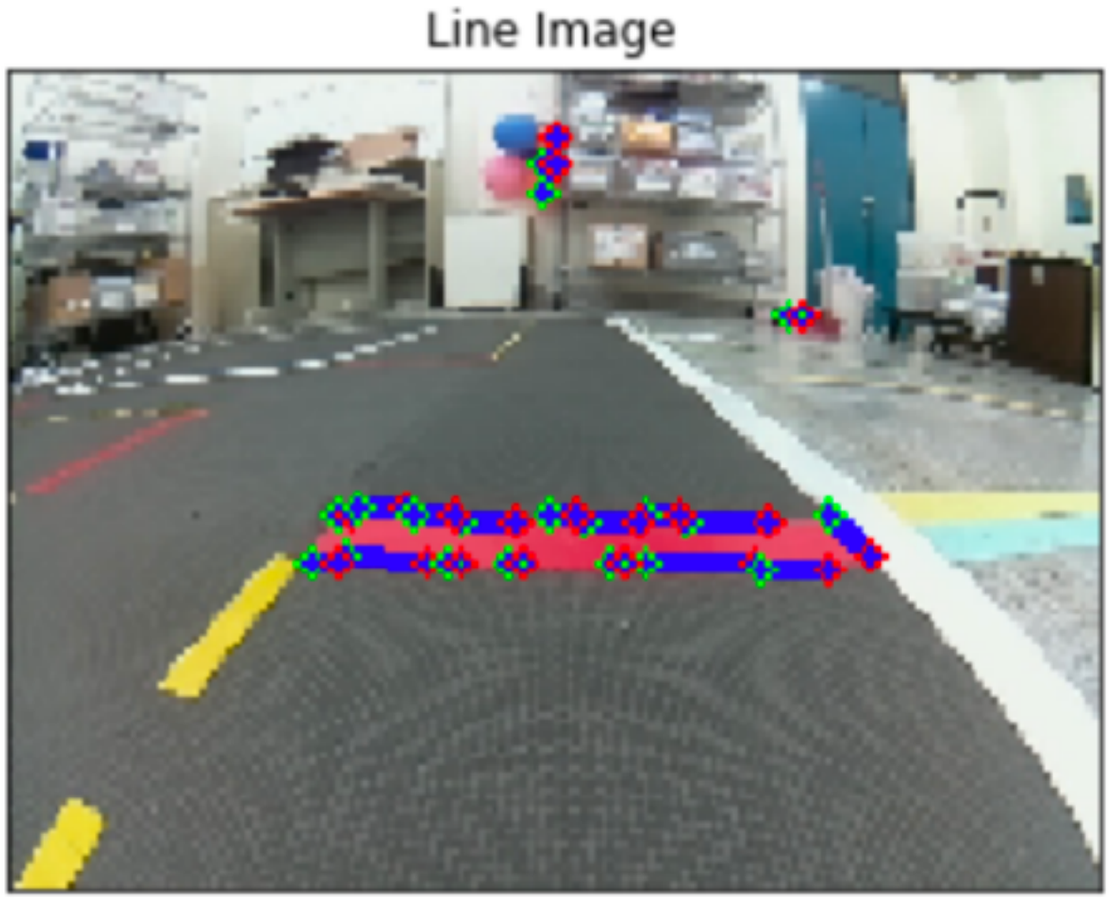
example.cpp

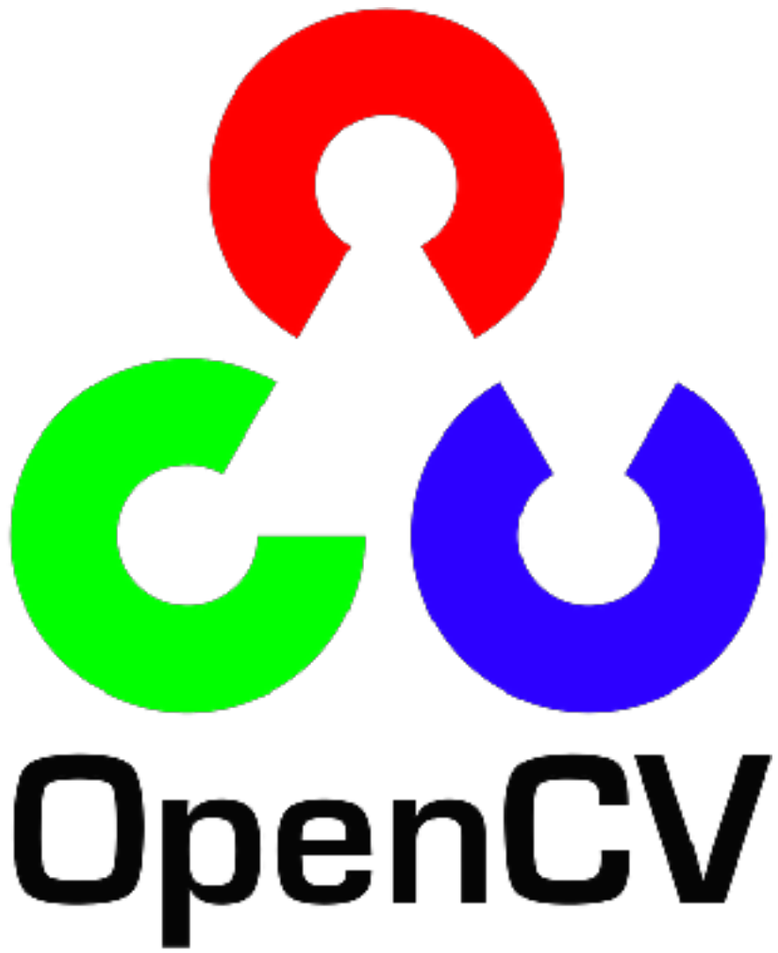
```cpp
#include <stdio.h>
#include <iostream>
#include <ros/ros.h>

class test_class
{
public:
    test_class();
    ~test_class();
private:
    int16_t param_1;
    int16_t param_2[3] = {1, 2, 3};
};

test_class::test_class(){
    param_1 = 1;

    res = this.demonstration(5);
    printf("param_1 = %d\n", param_1);
    printf("res = %d\n", res);
}

uint8_t test_class::demonstration(uint8_t n_loop){

    uint8_t i = 0;

    for(i=0; i<n_loop; i++){
        param_1 += i;
    }

    return i
}

int main(){
    test_class testing();
    return 0;
}
```

- A web application to create and share document that contain live code.
- Show the code output directly
- Only interpreted code (like python)

Let’s play some Python code with Jupiter Notebook!

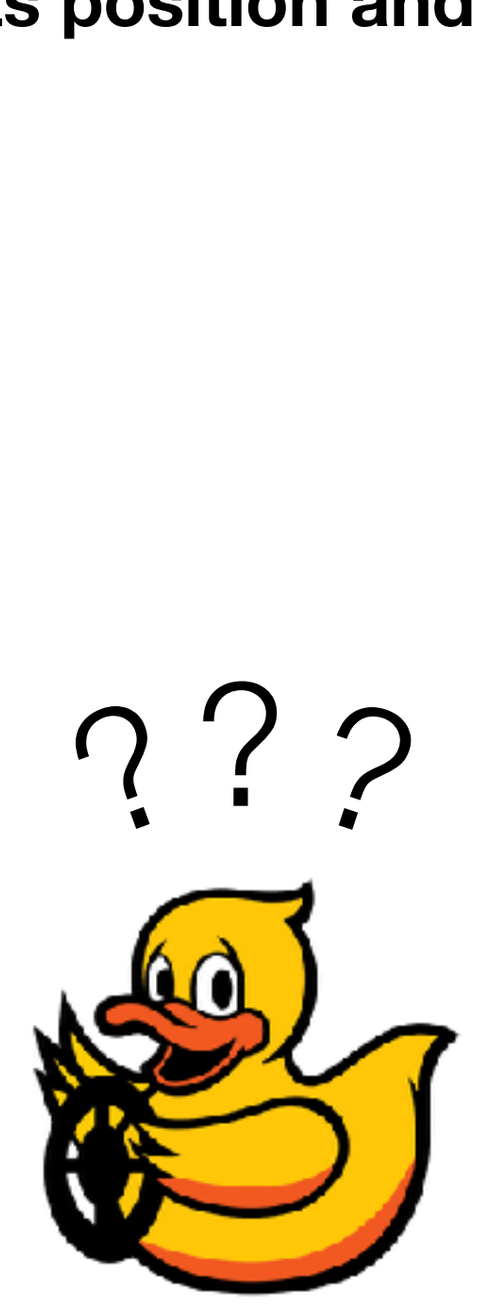
Line Detector & OpenCV

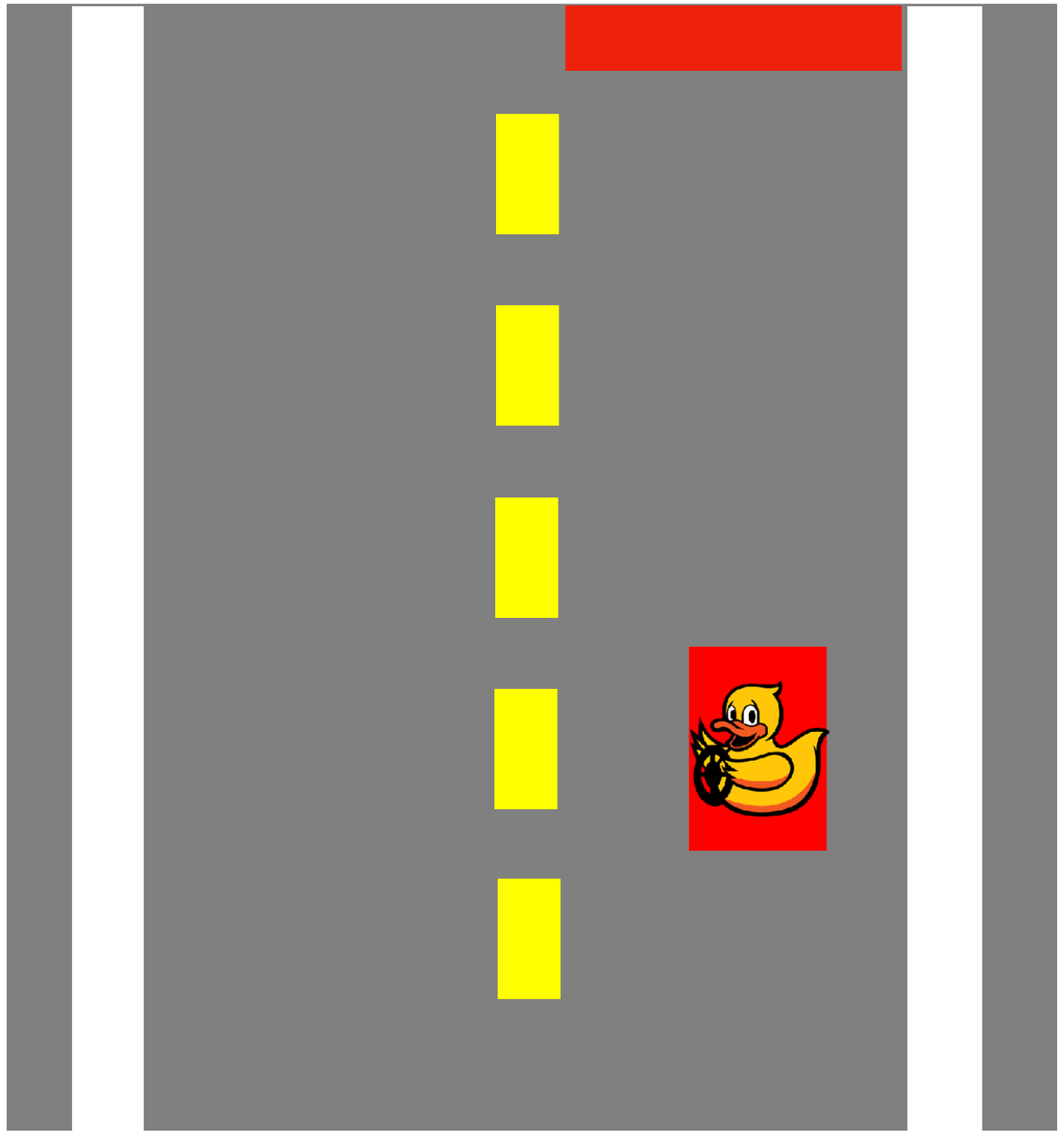




Q: What does Duckiebot detect so that it know its position and angle w.r.t to the road?

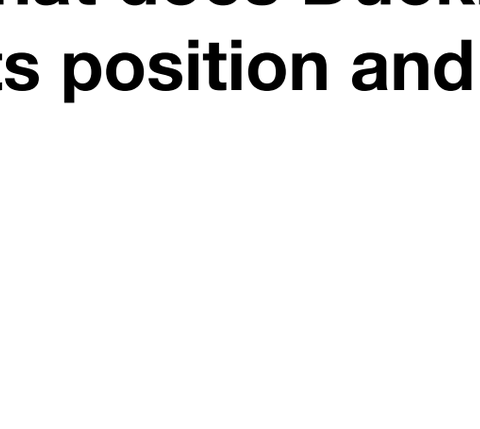
? ? ?

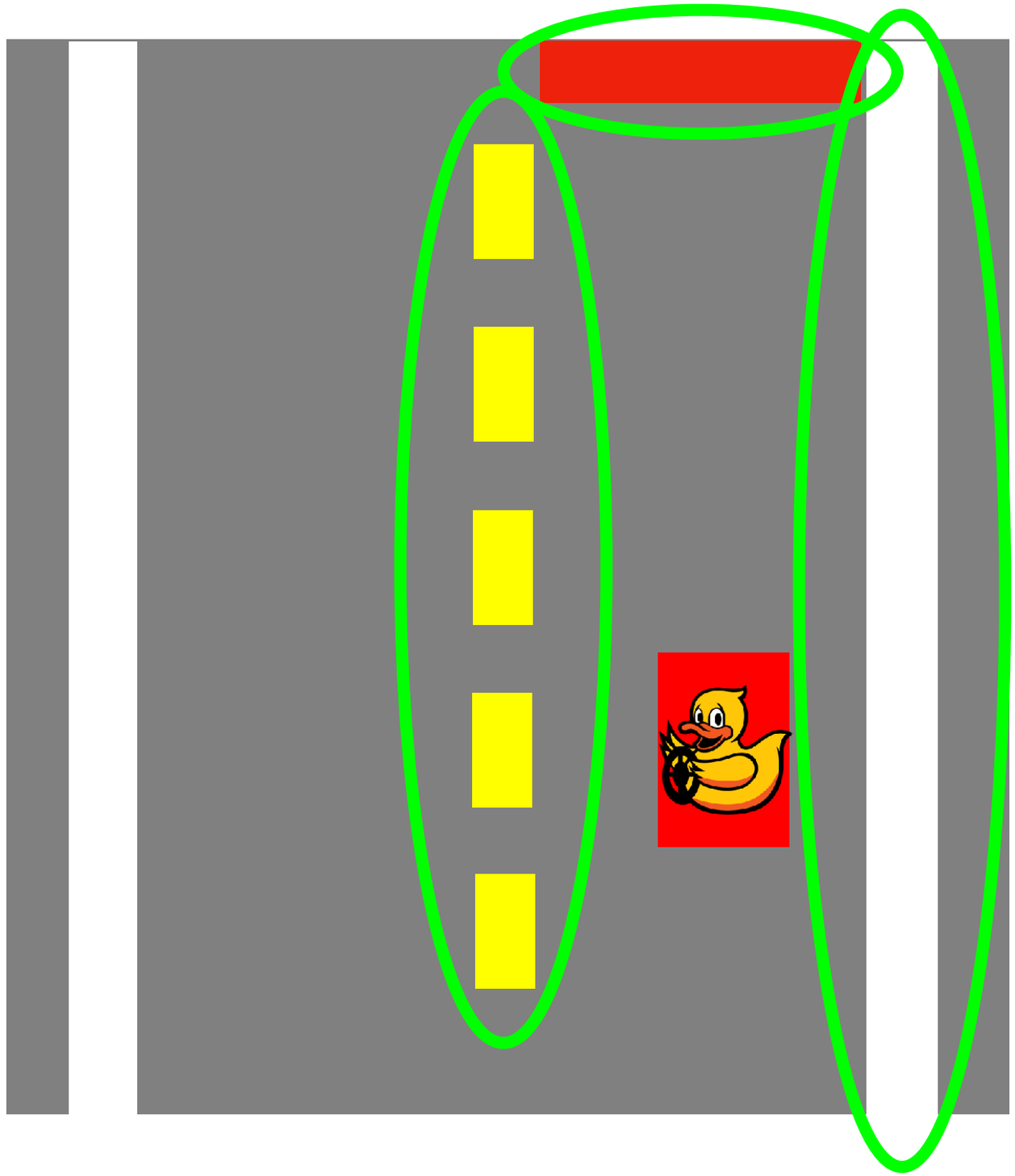




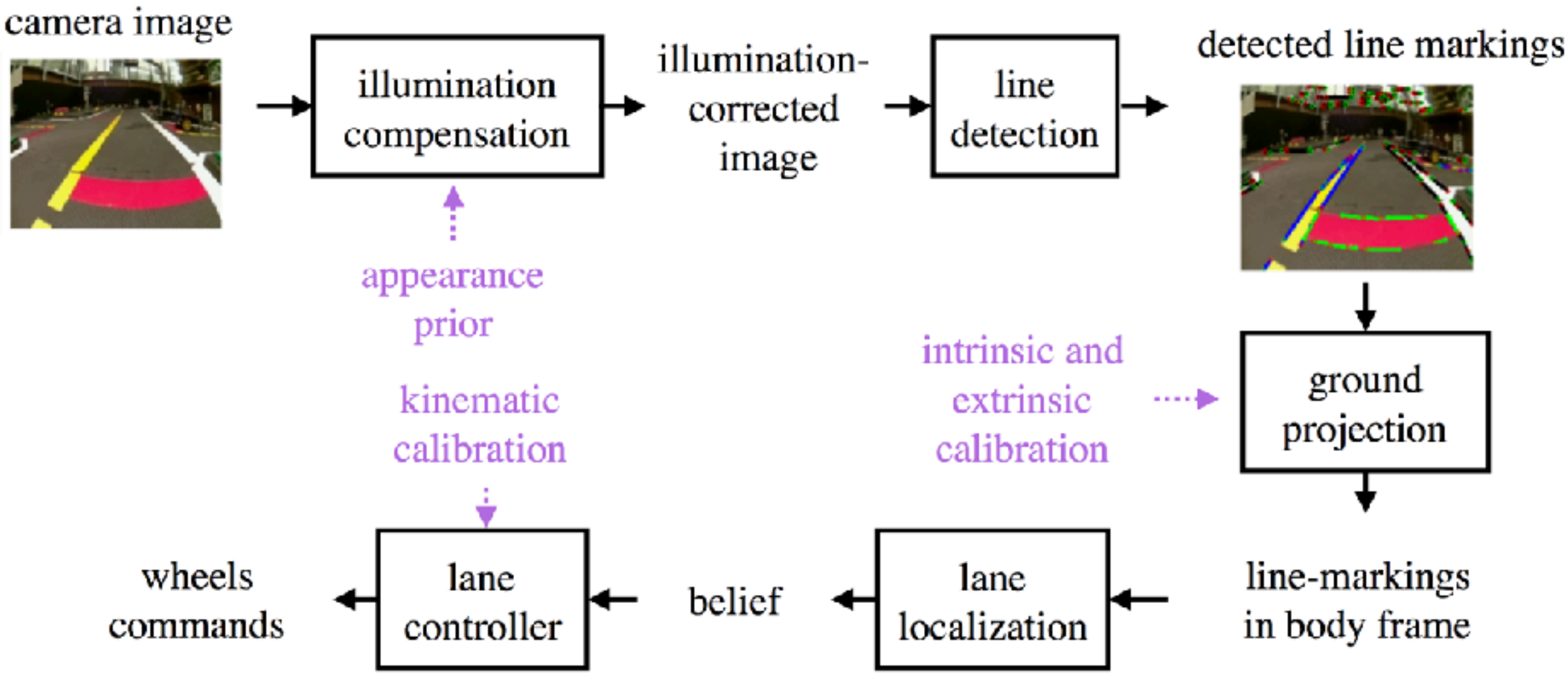
Q: What does Duckiebot detect so that it know its position and angle w.r.t to the road?

A: Lines!

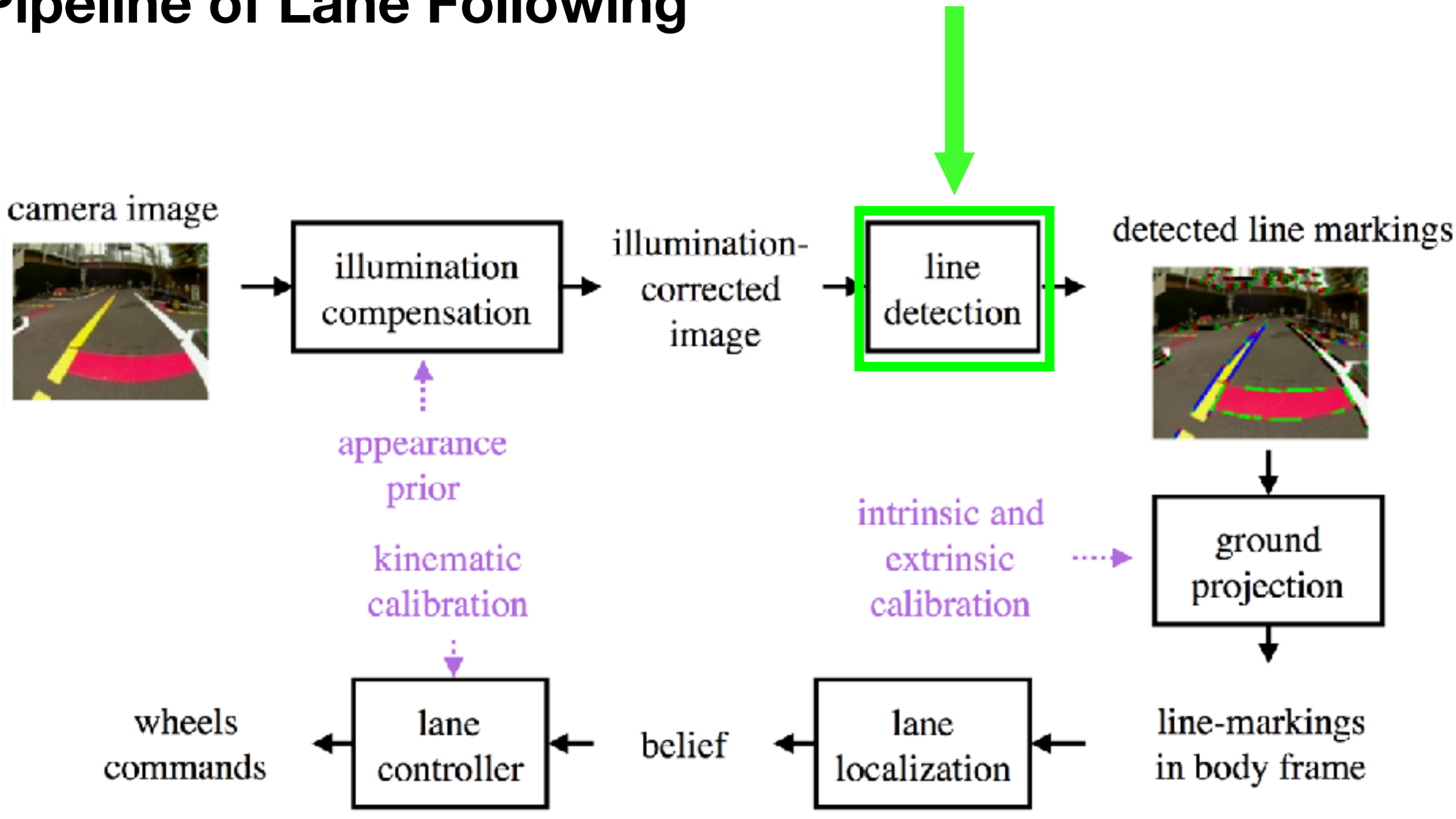




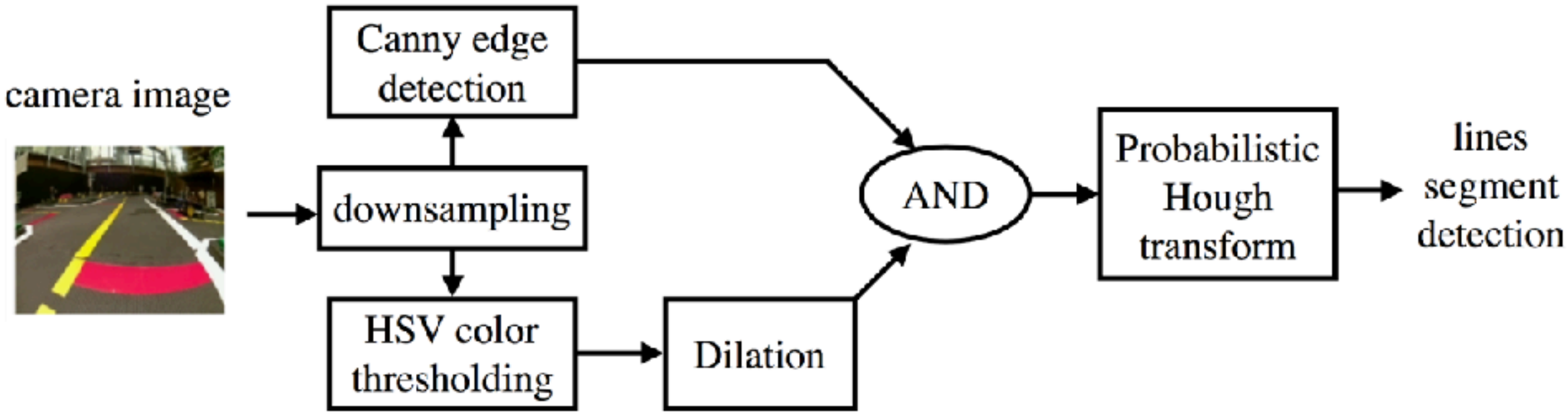
Pipeline of Lane Following



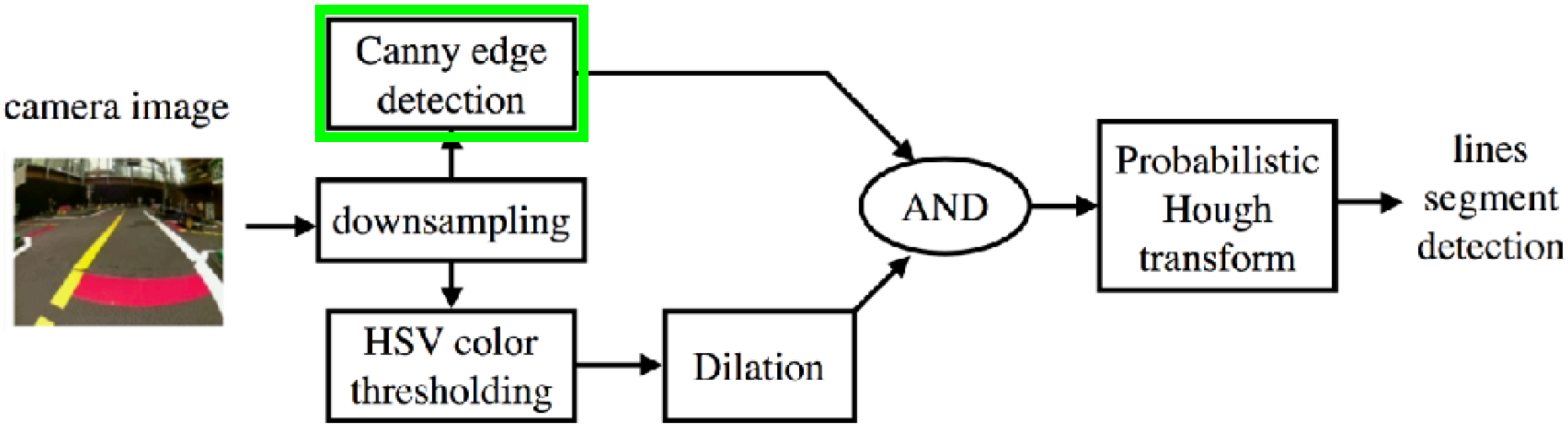
Pipeline of Lane Following



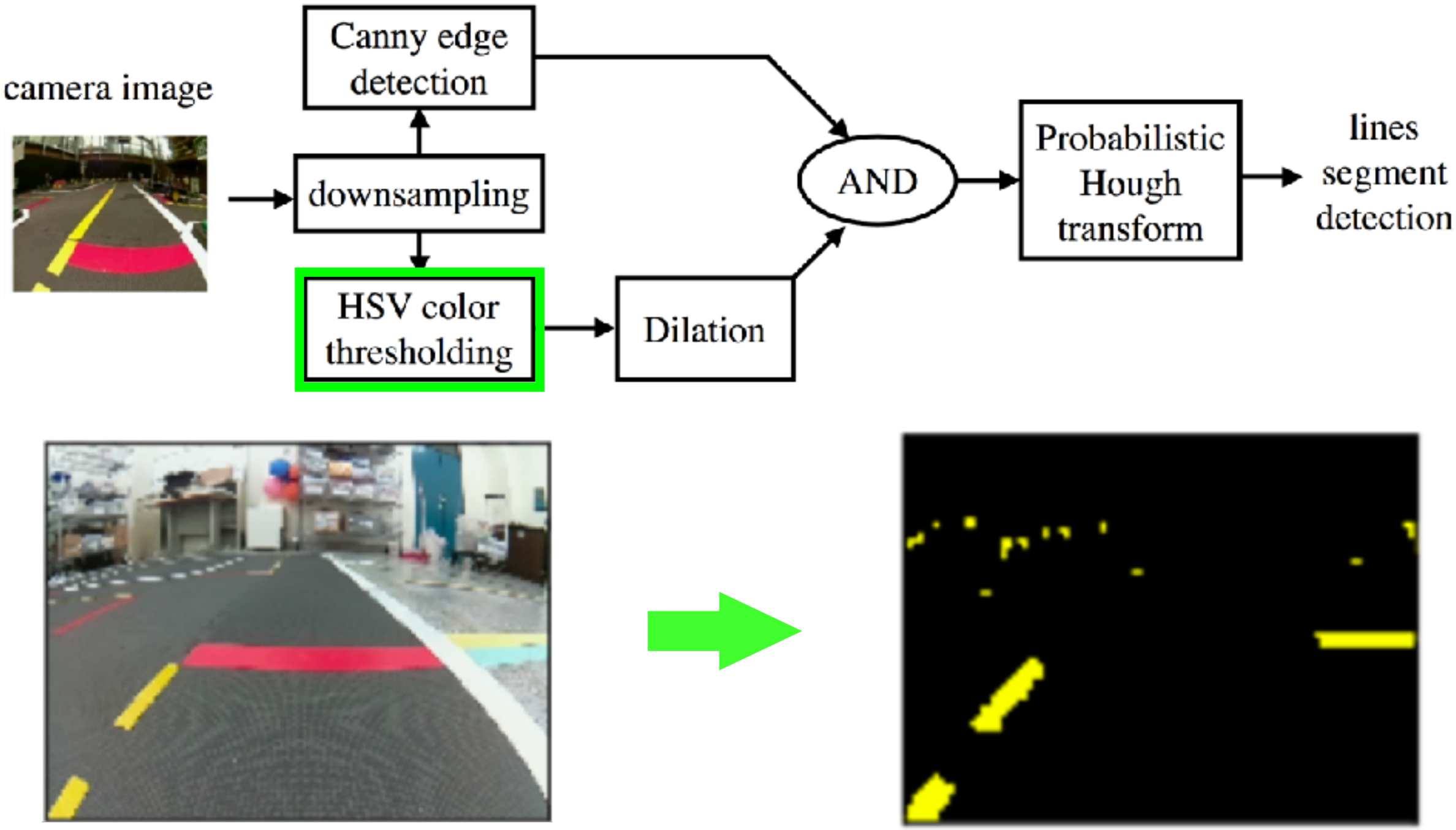
Theory behind Line Detection



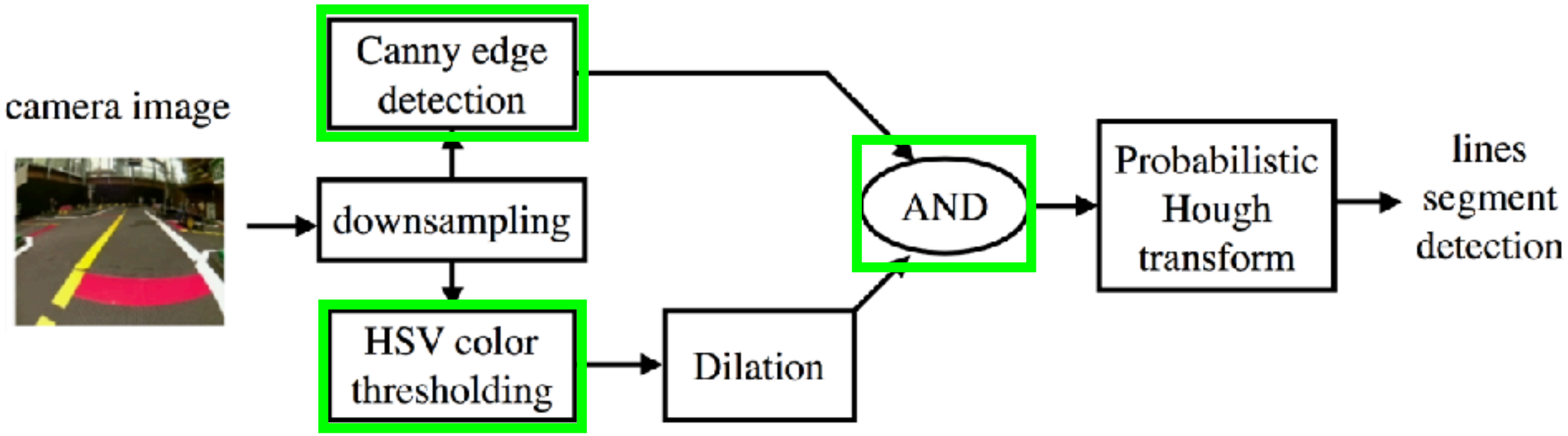
Theory behind Line Detection



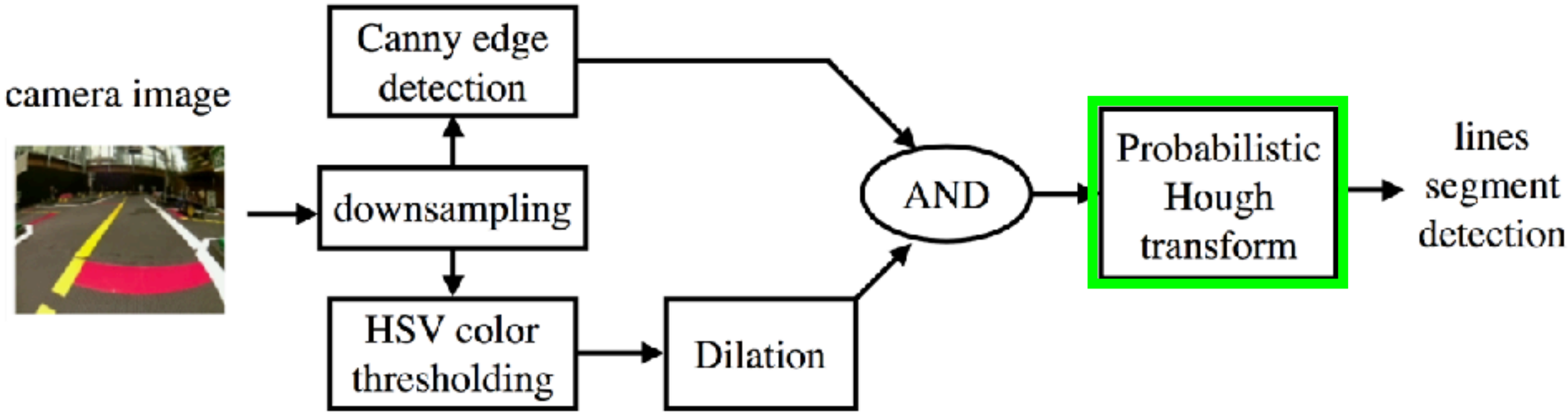
Theory behind Line Detection

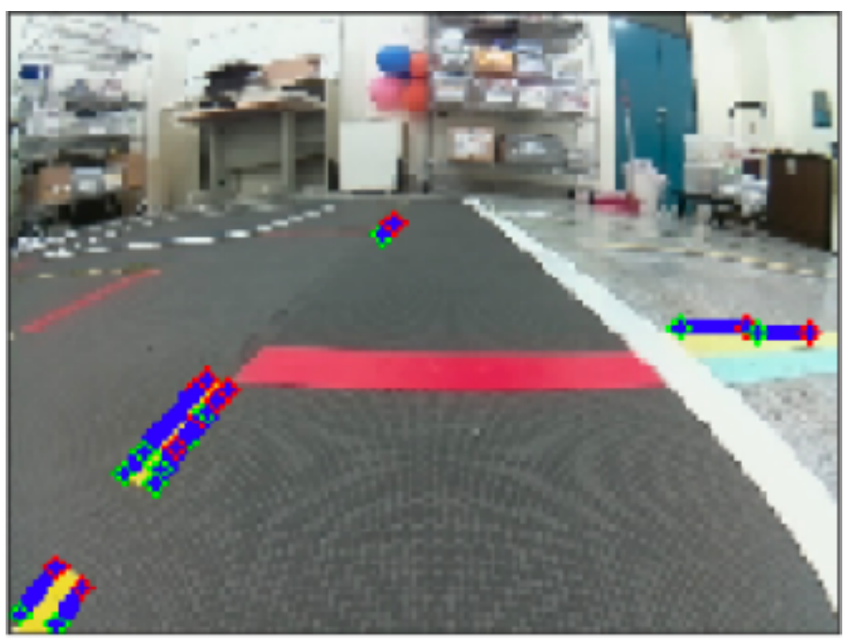


Theory behind Line Detection



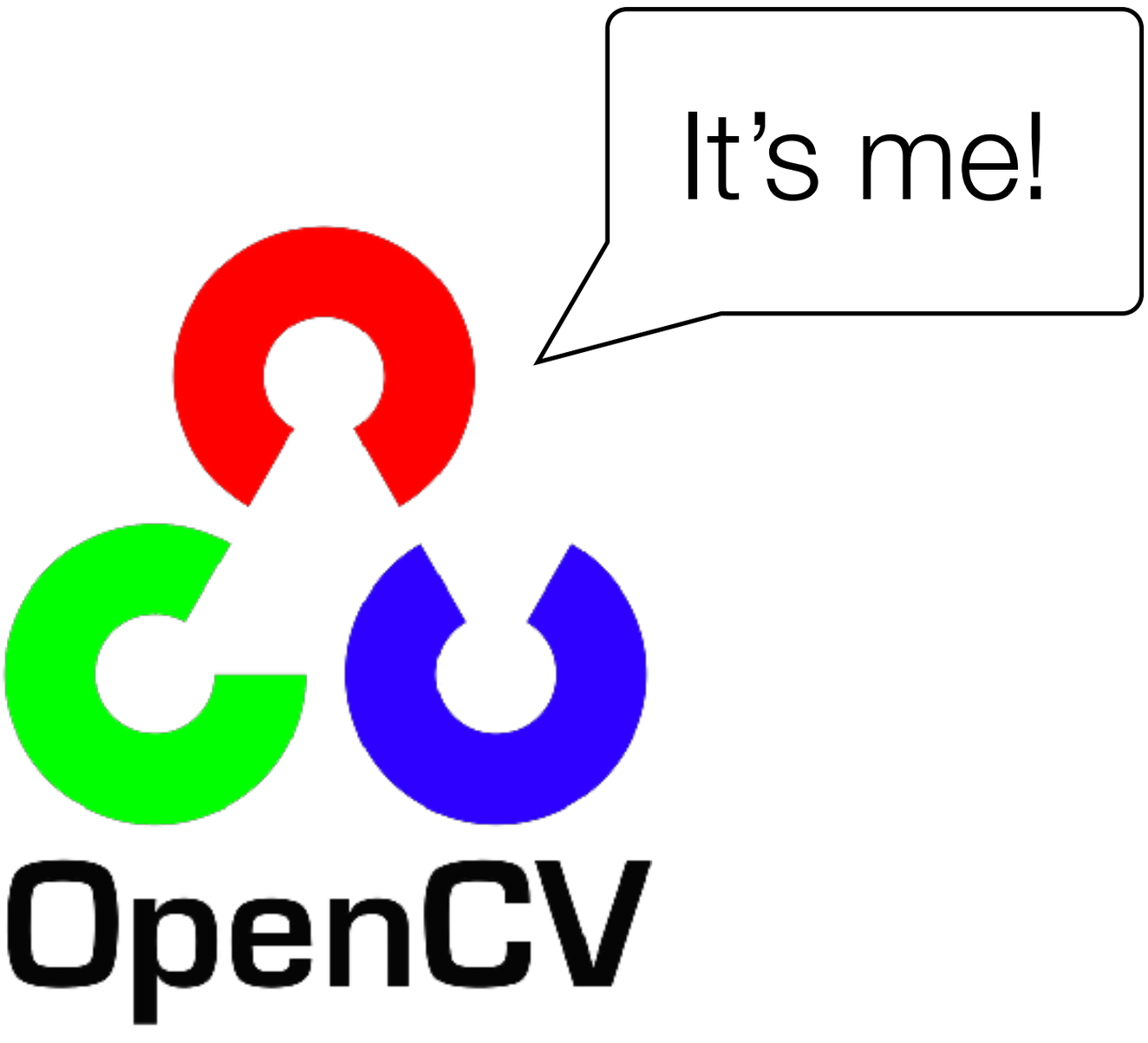
Theory behind Line Detection





How can we perform those magics?

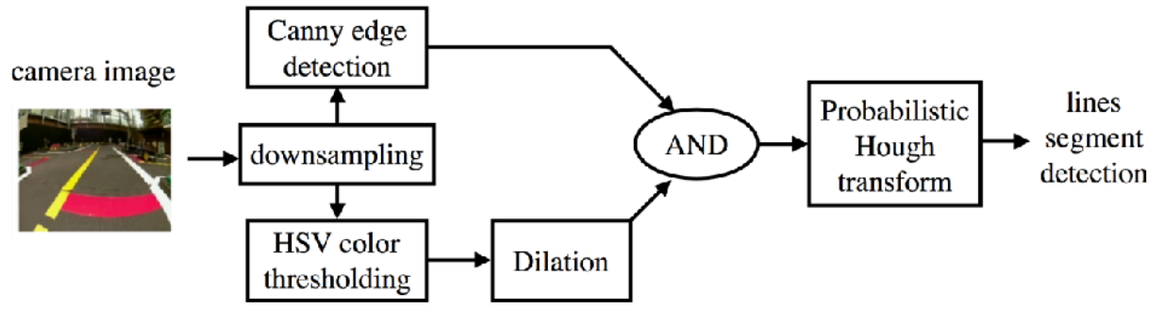
by **OpenCV**



- OpenCV (Open Source Computer Vision Library)
- Open source computer vision and machine learning software library
- Provide a common infrastructure for computer vision (machine learning) applications

What are “infrastructures”…?

What are “infrastructures”…?



Canny edge detector

cv2.Canny(img, minVal, maxVal)

Color space transform

result = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

Thresholding

result = cv2.inRange(img, minVal, maxVal)

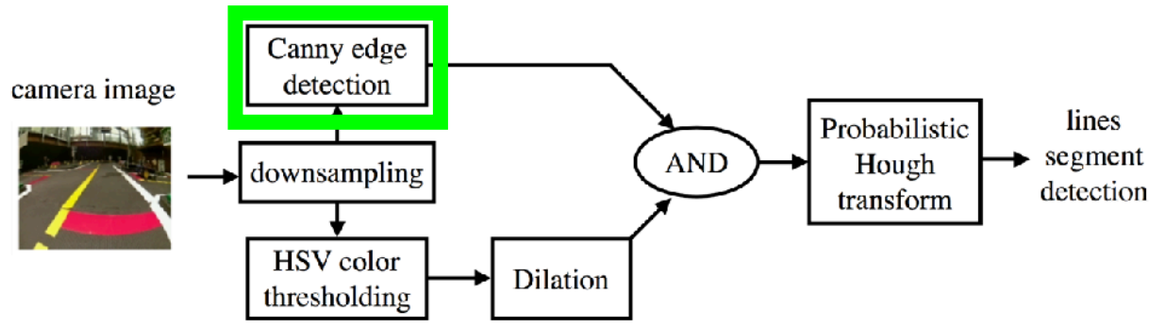
Probabilistic Hough Transform

result = cv2.HoughLinesP(img, 1, angle, 100, minLineLength, maxLineLength)

What are “infrastructures”…?

Canny edge detector

cv2.Canny(img, minVal, maxVal)



Color space transform

result = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

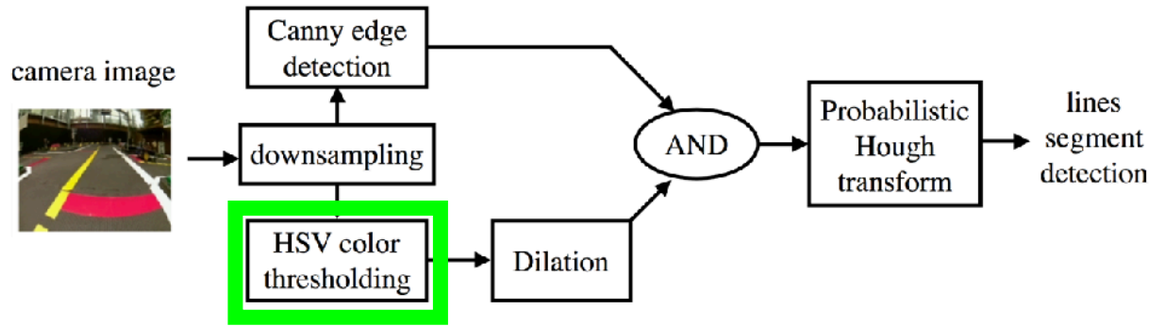
Thresholding

result = cv2.inRange(img, minVal, maxVal)

Probabilistic Hough Transform

result = cv2.HoughLinesP(img, 1, angle, 100, minLineLength, maxLineLength)

What are “infrastructures”…?



Canny edge detector

cv2.Canny(img, minVal, maxVal)

Color space transform

result = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

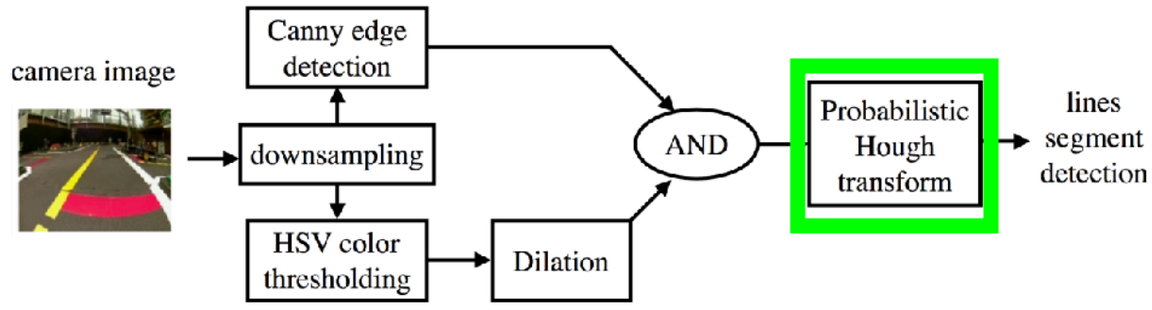
Thresholding

result = cv2.inRange(img, minVal, maxVal)

Probabilistic Hough Transform

result = cv2.HoughLinesP(img, 1, angle, 100, minLineLength, maxLineLength)

What are “infrastructures”…?



Canny edge detector

cv2.Canny(img, minVal, maxVal)

Color space transform

result = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

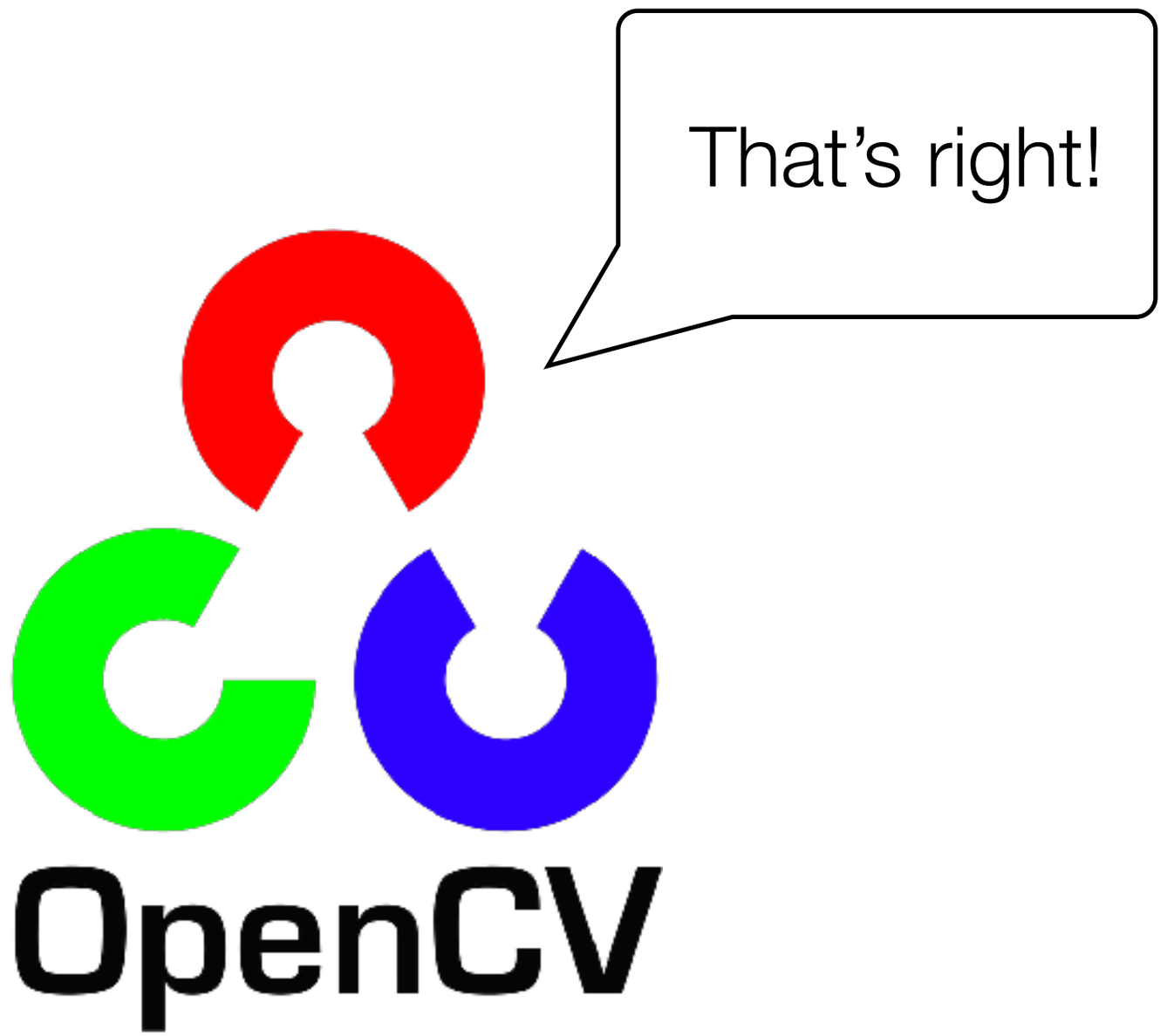
Thresholding

result = cv2.inRange(img, minVal, maxVal)

Probabilistic Hough Transform

result = cv2.HoughLinesP(img, 1, angle, 100, minLineLength, maxLineLength)

In conclusion, OpenCV helps us a lot in development



Last but not least…

Let’s go through some background knowledge of

Canny edge detector

HSV color thresholding

Canny edge detector

How do we detect edges?

- Convolution

Canny edge detector

Ex...

$$\frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	0	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Canny edge detector

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	0	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

	10								

Canny edge detector

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	0	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

	10								

Canny edge detector

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	0	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

	10	20							

Canny edge detector

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	0	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

	10	20							

Canny edge detector

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	0	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

	10	20	30						

Canny edge detector

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	0	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

	10	20	30						

Canny edge detector

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	0	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

	10	20	30						
			80						

Canny edge detector

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	0	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

	10	20	30	30	30	20	10		
	20	40	60	60	60	40	20		
	30	60	90	90	90	60	30		
	30	50	80	80	90	60	30		
	30	50	80	80	90	60	30		
	30	50	80	80	90	60	30		
	20	30	50	50	60	40	20		
	10	20	30	30	30	20	10		

Canny edge detector

So how convolution can help us?

Canny edge detector

Sobel Filters

- Sobel filters create images that emphasize edges.
- Two small convolution filters are used successively:

-1	0	1
-2	0	2
-1	0	1

Vertical

1	2	1
0	0	0
-1	-2	-1

Horizontal

Canny edge detector

$$\frac{1}{9}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Vertical

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	0	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Canny edge detector

$\frac{1}{9}$

-1	0	1
-2	0	2
-1	0	1

Vertical

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	0	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

		30							

Canny edge detector

$\frac{1}{9}$

-1	0	1
-2	0	2
-1	0	1

Vertical

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	0	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

		30							
		40							

Canny edge detector

$\frac{1}{9}$

-1	0	1
-2	0	2
-1	0	1

Vertical

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	0	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

		30							
		40							
		40							

Canny edge detector

$\frac{1}{9}$

-1	0	1
-2	0	2
-1	0	1

Vertical

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	90	0	90	90	90	0	0	0
0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

		30							
		40	0						
		40							

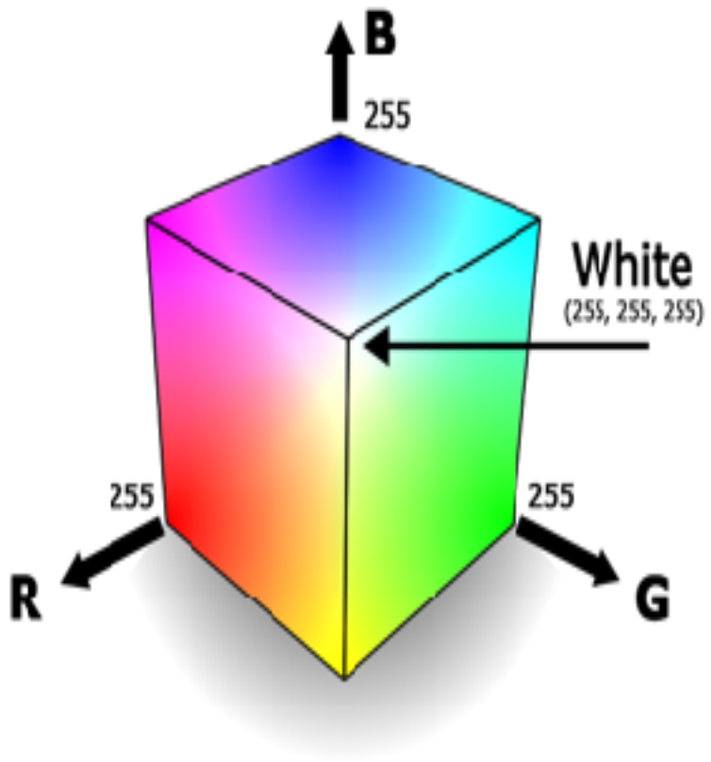
HSV color thresholding

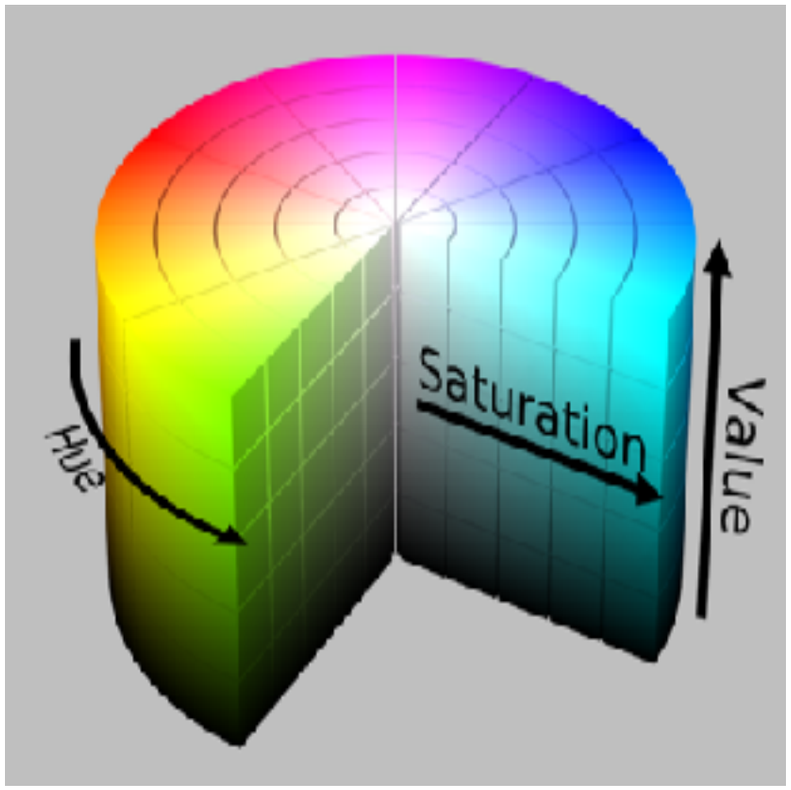
HSV color thresholding

HSV Color Space

HSV color thresholding

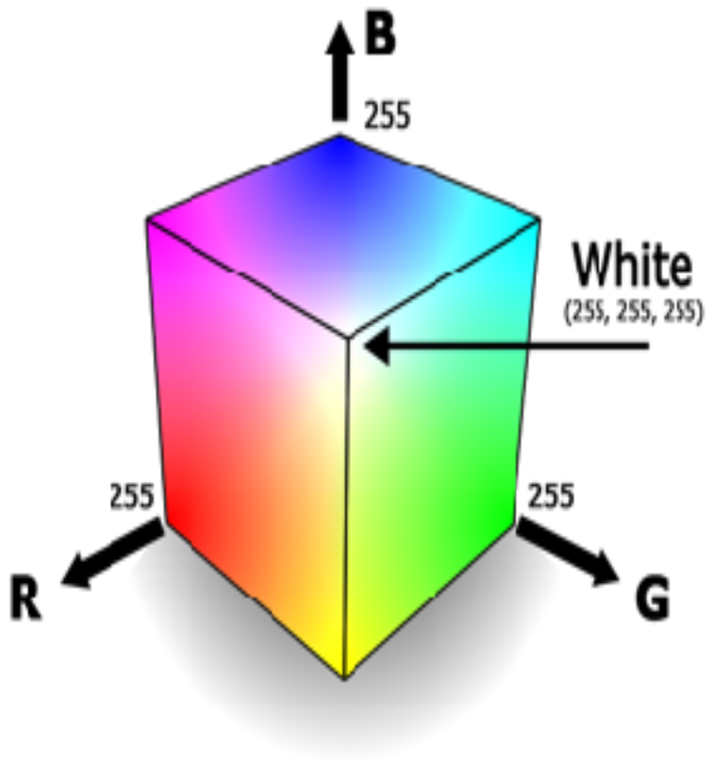
HSV Color Space

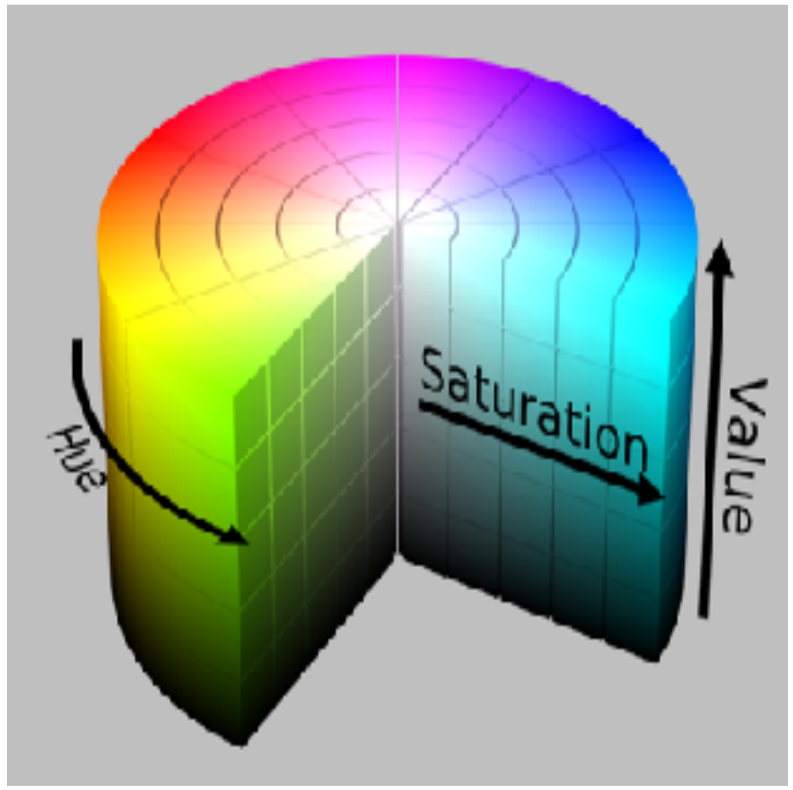




HSV color thresholding

HSV Color Space





(R, G, B)

(H, S, V)

Hue Saturation Value

Let’s go back to Jupiter Notebook!

Google is your friends <3