Tutorial 4b - Introduction to Image Processing

Author: Peter Tse
Contact: hntse@ust.hk

This section of the note is dedicated to the racer robot in the competition. Since it is mandatory for it to be fully automated, here we will give you a quick and brief guide to it. These are the only information we will provide, and seniors shall not give further hints to any of the groups.

Nested For Loop

This section is meant to remind you how to construct a nested for loop for 2D arrays. If we have the following codes,
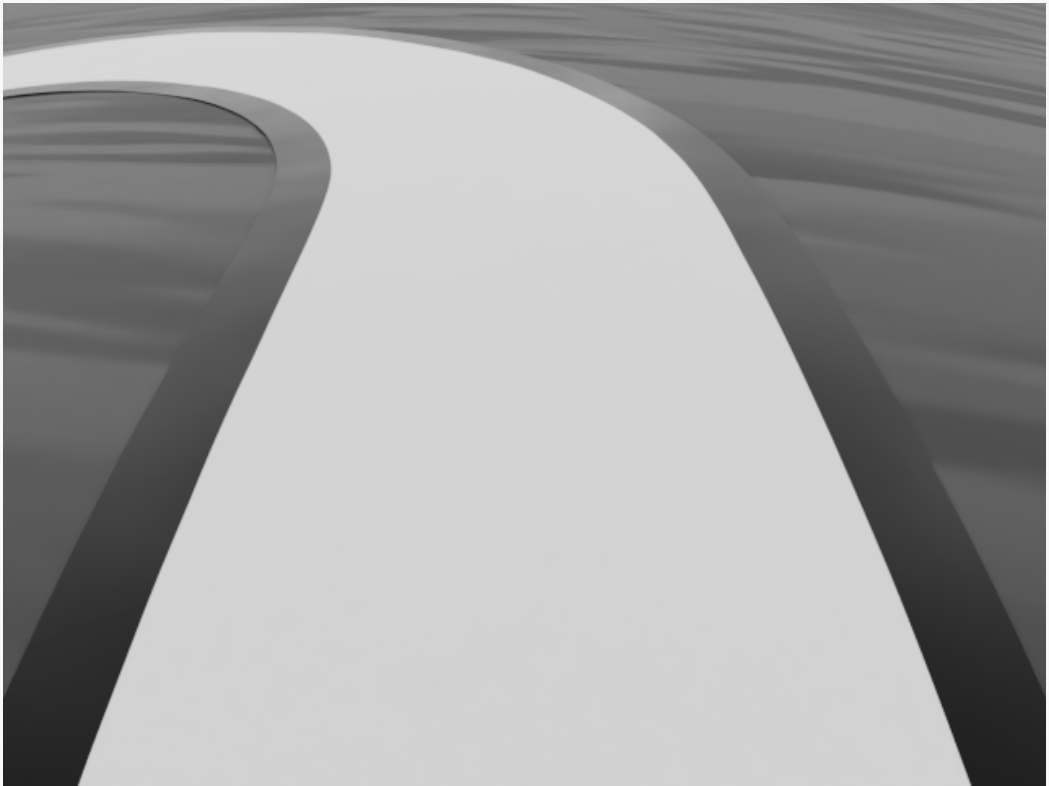
```c
#include <stdio.h>

int main() {
  const int WIDTH = 3;
  const int HEIGHT = 5;
  int a[HEIGHT][WIDTH]; // note the order

  for (int i = 0; i < HEIGHT; i++)
    for (int j = 0; j < WIDTH; j++)
      a[i][j] = i+j;

  for (int i = 0; i < HEIGHT; i++) {
    for (int j = 0; j < WIDTH; j++)
      printf("%d ", a[i][j]);
    printf("\n");
  }

  return 0;
}
```

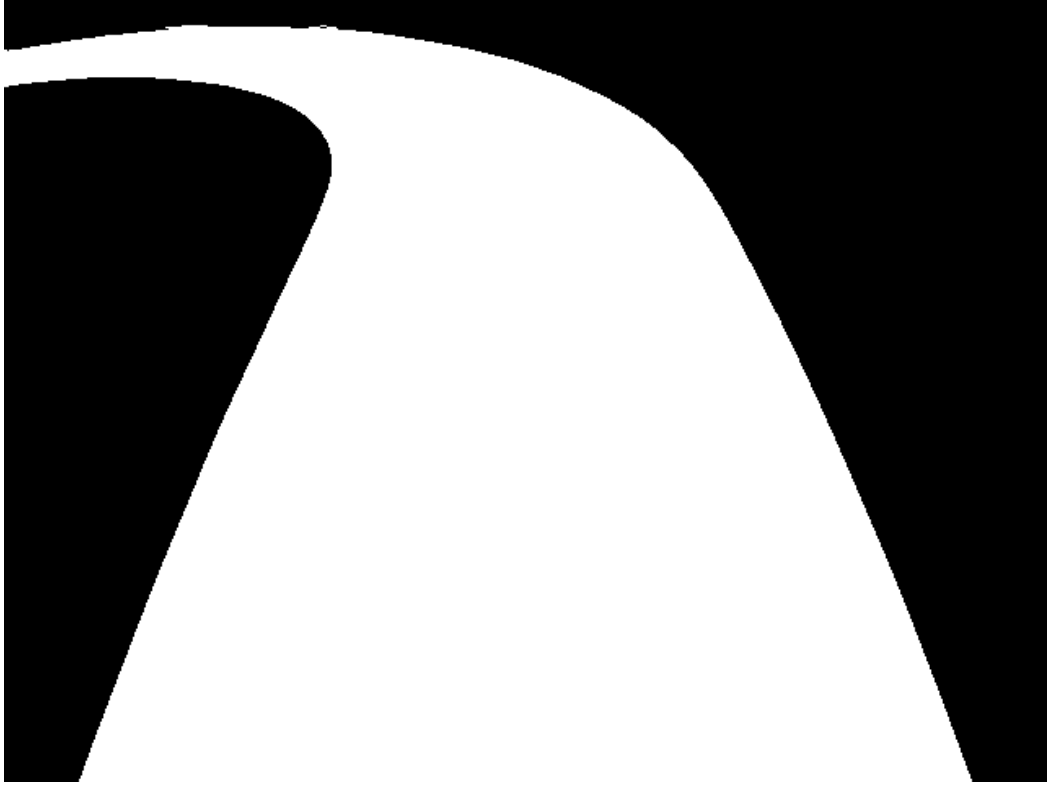We will have the following output.

```
0 1 2
1 2 3
2 3 4
3 4 5
4 5 6
```

Colored, Grayscale or Black&White?

Recall that we can either have a fully colored image, a grayscale image or a black&white image in Ov7725.

Mode	Image Sample (From Rendering in SmartCar)
Fully colored	
Grayscale	

Mode	Image Sample (From Rendering in SmartCar)
Black&White	

Please note that the images are rendered instead of taking in reality, so the image shown here may not fully represent the real image that you will be getting.
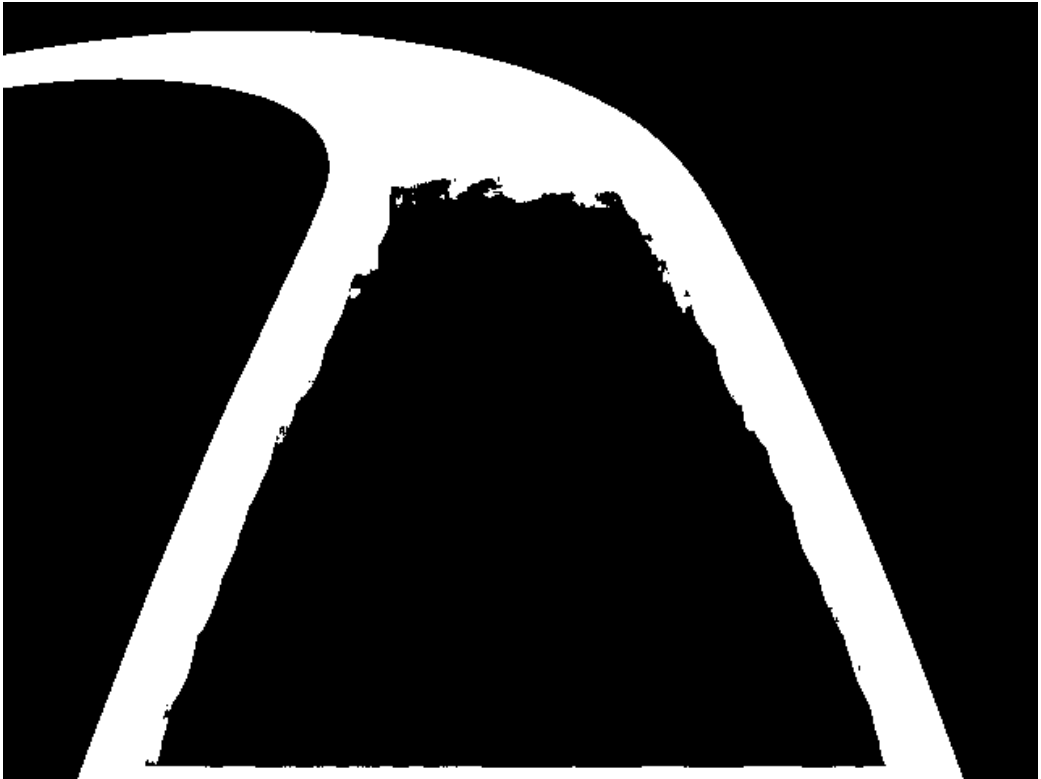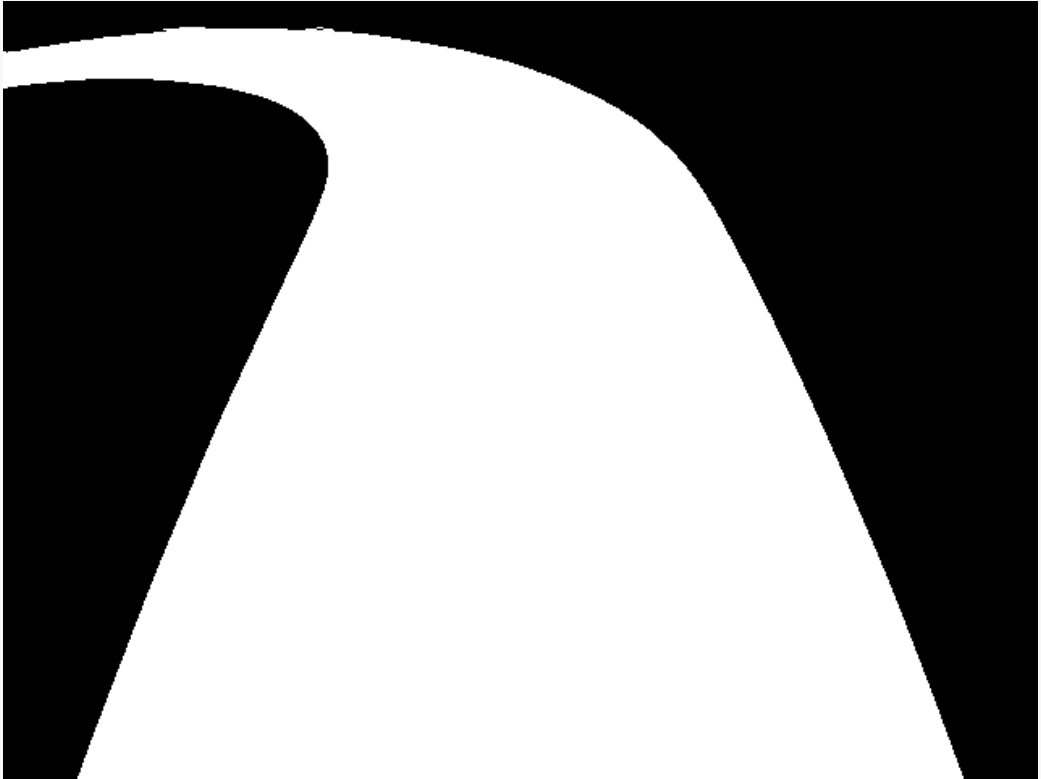
One should notice the following:

- Fully coloured pictures contain too much details, notice that colours are stored as RGB in the MCU, and it is kind of difficult to handle these many data. You will need to have a good threshold function for you to distinguish from the track and the surrounding, for example.
- Greyscale pictures contain less information than fully coloured ones, but can still keep a lot of the details. Greyscale is usally represented by integer 0 to 255, which makes it much easier to handle. It is useful when details are necessary.
- Black&white pictures contain much less information than the other ones that some details are lost. However, since black&white images are only represented by 1 and 0, it is very easy to handle them. It is useful when details are not that necessary.

It is general that for image processing, we would choose greyscale or black&white images to analyze since they are easy to handle. In this competition, it is advised to use either of them.

Brightness & Intensity

Apart from colour choice, the brightness and the intensity of the image captured also affect the quality of the image you have, which in term affect the ability to extract good data from it.

Here we are using black&white images as example, since the effects here are more significant.

Effect	Image Sample (from Rendering in SmartCar)
Low Intensity	
Medium Intensity	

Effect	Image Sample (from Rendering in SmartCar)
High Intensity	

If you do not have the right capture intensity, your image might contain noise that you cannot handle properly. Note that the intensity you needed is dependent on the surroundings of the camera, therefore you should have a method to tune your intensity of the camera on site right before the competition!

Median Filter

After picking the colour mode and tuning your brightness/intensity, there might still be some noise in the image captured (might be due to noise in connection wires, dust on lens, etc). You might want to reduce the noise such that your algorithm can produce a better result. Here we introduce a method 'Median Filter', which is simple to implement, and is effective on removing noise.

Here is the algorithm in psuedocode.

```
1   let k be the square window length
2   let a be the input array
3   let b be the output array
4   for x from k/2 to width-k/2 // the range is to prevent out-of-boundary array access
5     for y from k/2 to height-k/2
6       m = the median of the colour value of a(x, y) and its surrounding k*k square
7       b(x, y) = m
```

To illustrate, here we have a 1D version of median filter in action.

For example, we have a 1D array `x = {1, 80, 6, 2}`, and the window size is `3`.

The iterations of the filter is as follows.

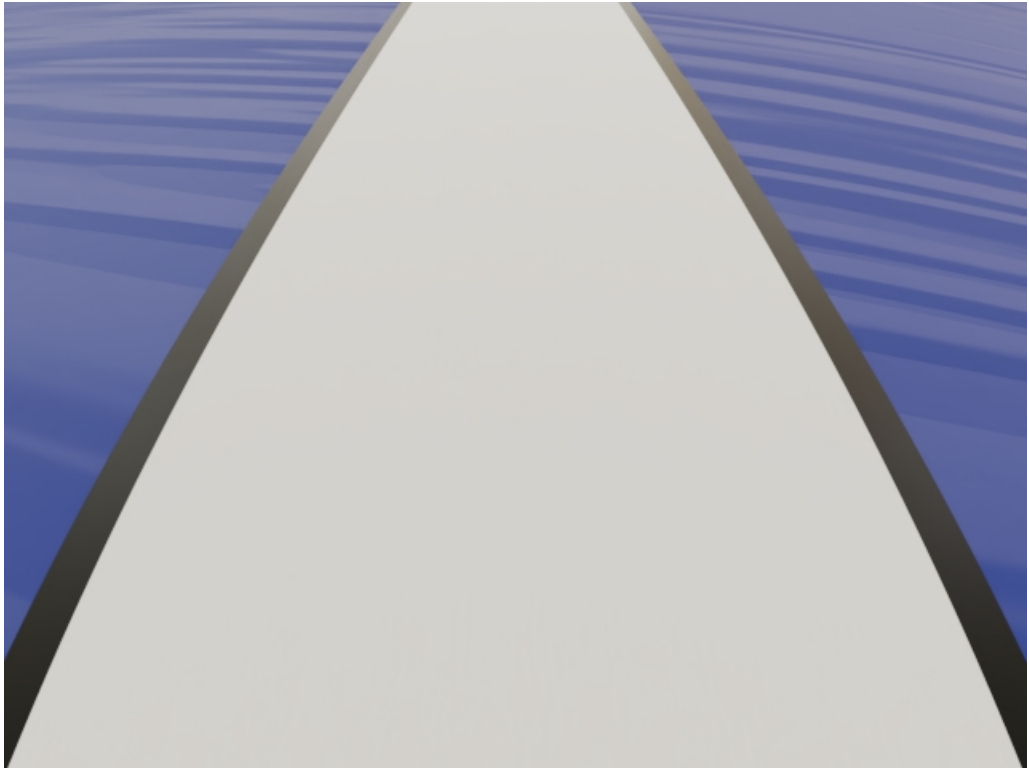1. `median(1, 80, 6) = 6`

2. `median(80, 6, 2) = 6`

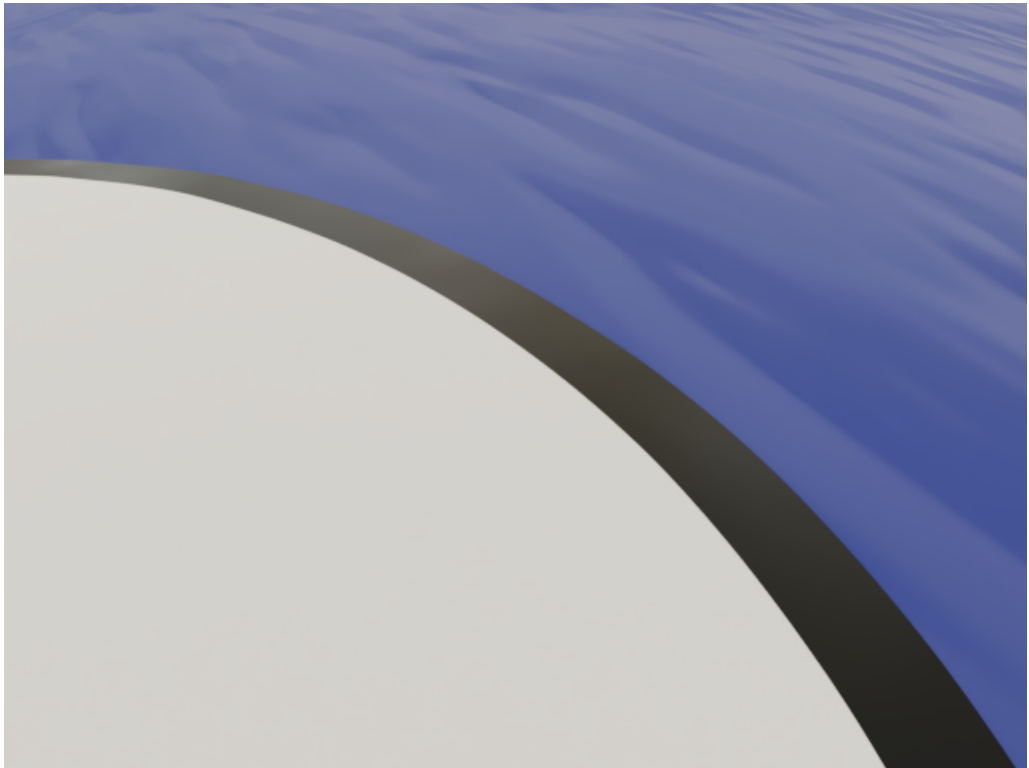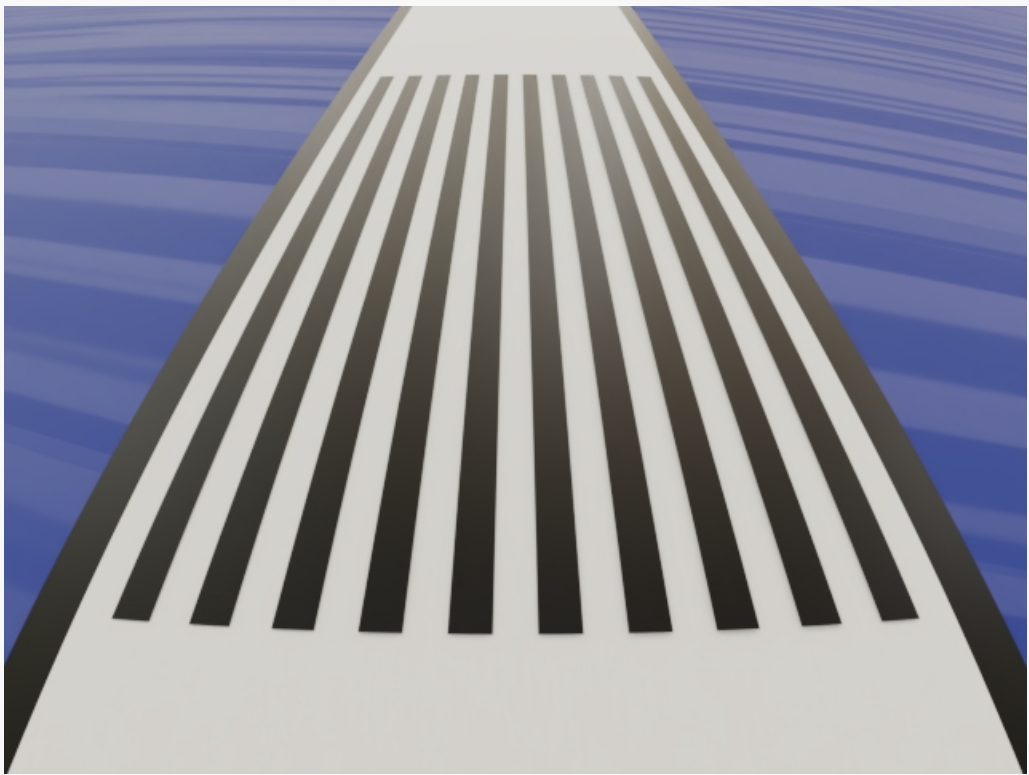Therefore, the new array would be `x' = {1, 6, 6, 2}`.

You can see the median filter in action to a real photo here.

Hint to Competition

1. Have a way to tune intensity on site.

2. Your algorithm to the racer robot should have (1) direction tracking, (2) SZ detection and (3) control system.

   - For (1) and (2), it might be useful to use the distribution of colours in the image as a reference.
   - For (3), by 'control system', it is that there should be a link between the way to control the motor and servo, and the analysis the car has made.

3. You may refer to these rendered images when you design the algorithm. Note that, again, these images may not represent fully what you actually would have got.

Location	Sample Image (from Rendering)
Straight	
Turn 1	

Location	Sample Image (from Rendering)
Turn 2	
SZ1	