Parallel & Distributed Programming

Assignment 1 – OpenMP

Deadline : 2013/03/19 23:59

- In assignment 1, you are asked to modify a **C++** program using OpenMP

- **Image Alignment**

This is a partial implementation of median threshold bitmap aliment algorithm. The algorithm is used to know the pixel offset of two unaligned images, assuming that the images taken have no rotation relation. The inputs are two images, the output is the x and y offset of the second image related to first image.

First, we have two color images as input. Then, we calculate the difference of two images and try to find the offset of x and y which make the difference minimum. In this algorithm, the color images are converted to grayscale images (which take a value from 0 to 255 to represent a gray pixel). And by using the median of the grayscale images, we convert grayscale images into bit images (which take a bit to represent a black or white pixel) with median taken as thresholds. The difference function is defined as the XOR of two bit images. We count the '1's in the XOR image to know the difference. Offset are tried with brute force style and all pixels are taken to calculate the difference.

Sample Input 1 (please refer to TA's Makefile):

```
> ./bin/align bmp_image1 bmp_image2
or
> make exec
```

test01a.bmp	test01a_half.bmp
test01b.bmp	test01b_dark.bmp
test01c.bmp	test01c_half.bmp
test01c_light.bmp	
test01d.bmp	test01d_dark.bmp

- Filename with 'a' is the original image
- Filename with 'b' is the image shifted on x direction by 1 px from the original image
- Filename with 'c' is the image shifted on x direction by 10 px from the original image
- Filename with 'd' is the image shifted on x direction by 5 px and on y direction by 5 px from the original image
- Filename with 'half' is the resized version of original image by half on each direction
- Filename with 'light' is a lighter version of the original image
- Filename with 'dark' is a darker version of the original image
- Test02* have the same structure as Test01*.

Sample Output 1:

```
image width = 256   image height = 256
alignment offset x = 5, y = 5
minDifference = 194
89.03user 0.03system 0:31.39elapsed 283%CPU (0avgtext+0avgdata 9936maxresident)
```

- **Submission**

You have to write a report including:

1. How you parallelize your program.
2. Which data structures are private and which are shared.
3. The overall speedup.

Speedup formula:

$$Speedup = \frac{ExecutionTime_{sequential}}{EcecutionTime_{parallel}}$$

Please submit a compressed folder named by your ID with following structure. For example :

```
R01922XXX.zip
    +---R01922XXX
        +---main.cpp
        +---REPORT.pdf
```

- **Grading Policy**

Image Alignment	70%	Correctness & Speedup	The output correctness of your program, and with reasonable speedup. If your output is incorrect, you will get no credit in this part. *yourScore(speedup)* *if (incorrect)* *return 0* *if (speedup > 80% of TA's)* *return 100* *if (speedup > 50% of TA's)* *return 60* *return 0*
	30%	Report	As mentioned above.

TA will compile your program **ON WORKSTATION** using provided Makefile by typing this :

```
make clean && make
```

TA will judge your program **ON WORKSTATION** using **different** images by typing this :

```
make exec
```

- **NOTE**

1. Check Makefile for pre-defined flags and compile targets.
2. Beware of the private or shared state of members in paralleled part. The program may crash without correct settings.
3. Plagiarism is strictly prohibited.
4. Late submission will result in 20 pts deduction per week.

- **References**

1. OpenMP tutorial: https://computing.llnl.gov/tutorials/openMP/
2. Fast, Robust Image Registration for Compositing High Dynamic Range Photographs from Handheld Exposures
http://pages.cs.wisc.edu/~lizhang/courses/cs766-2008f/projects/hdr/jgtpap2.pdf

(If you have any question about the specification of this assignment, please mail to r01922101@ntu.edu.tw, r01922135@ntu.edu.tw, r01944053@ntu.edu.tw)