Maciej Kumorek Computer Science mk0934@bristol.ac.uk
Kacper Sokol Mathematics & Computer Science ks1591@bristol.ac.uk

Report

Assignment 3: „Concurrent image filtering“

Functionality and Design

Implementation

We have designed basic farming system containing distributor and eight workers, each blurring 3 pixels of the image at time. We have also included two types of blur algorithms: average - described in the assignment and median - which sort each pixel and its neighbours according to value and chooses the median one.

Buttons A, B and C are functioning as in the description. The button D allows changing of filtering method and the LED indicates if median (on) or average (off) is chosen. Before processing image, system lights 4 LEDs to communicate user it's ready to start. While processing, progress is displayed on clock LEDs. System exit all threads either while finished processing or when button C is pressed.

Our system can process any image that does not exceed width of (max value of int)/3 due to limitations of stack. The height is in theory unlimited because we buffer three lines at time. We also measure the processing time and print it to the standard output. All the measurements and effects of blurring small, medium, large and same image of dimension 354x472 for 20 times are included later in the report. There is also a note on efficiency according to different system parameters.

CSP specification of distributor-buttonListener

Alphabet is the same:
$\alpha(DIST) = \alpha(DIST') = \alpha(BL) = \alpha(BL') = \{aPressed, bPressed, cPressed, dPresse$

Button listener:
$BL = aPressed \rightarrow BL' \mid dPressed \rightarrow BL \mid cPressed \rightarrow SKIP$
$BL' = bPressed \rightarrow BL' \mid cPressed \rightarrow SKIP \mid terminate \rightarrow SKIP$
Distributor:
$DIST = aPressed \rightarrow DIST' \mid dPressed \rightarrow DIST \mid cPressed \rightarrow SKIP$
$DIST' = bPressed \rightarrow DIST' \mid cPressed \rightarrow SKIP \mid terminate \rightarrow SKIP$

Interactions:
$(BL \| DIST) = \mu X \bullet (aPressed \rightarrow X \mid cPressed \rightarrow X \mid dPressed \rightarrow X)$
$(DIST' \| BL') = \mu X \bullet (terminate \rightarrow X)$
$(BL' \| DIST') = \mu X \bullet (bPressed \rightarrow X \mid cPressed \rightarrow X)$

Maciej Kumorek Computer Science mk0934@bristol.ac.uk
Kacper Sokol Mathematics & Computer Science ks1591@bristol.ac.uk

Tests and experiments

Changing the number of workers

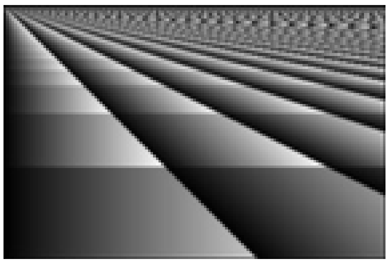
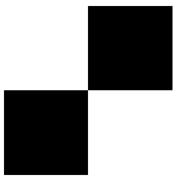
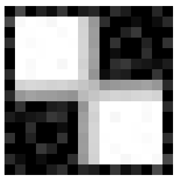
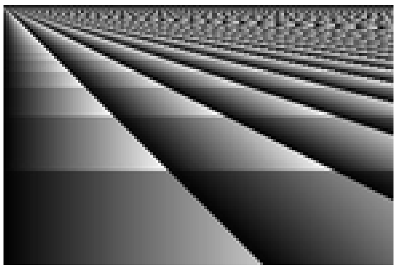
We noticed that having too little workers gives the same performance as having too many of them. We found in our experiments that having 8-9 workers gives the best times when processing images given (Cathedral example). We believe having too many workers slows the processing down due to overhead of channel communication between distributor and each worker.

Amount of pixels sent to each worker

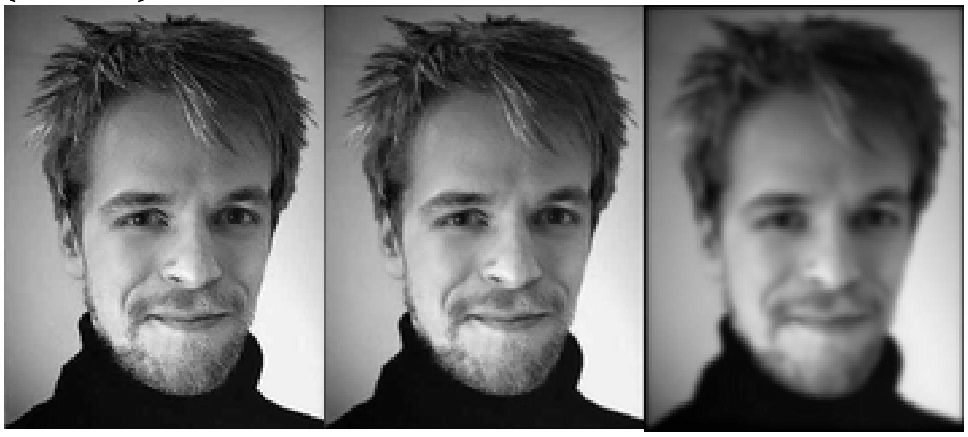
Initially each worker was processing only one pixel. However we noticed that workers seem to finish their task quickly and channel communication would take longer than actual time for computation. We thought we can increase number of pixel, however increasing this variable too much resulted in other workers waiting too long for new data-set and overall there would not be any improvement.

Type of channels used

We tried to use streaming channels between distributor and workers however due to hardware limitations we would have to decrease the number of workers to 4 only, therefore we decided not to use them in this place. We are using streaming channels to send results to collector as this is typical many-to-one scenario.
There are a few examples of blurred images (original vs blurred):





(300x200), (16x16)

(640x480)



(354x472) Blurred 0, 1 and 20 times.

In the following table there are example measurement taken:

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
447	430	445	512	492	499	439	452	445	471
11	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
456	431	481	498	501	440	438	463	451	459

Table 1. Times of 20 consecutive blurs in seconds of 354x472 image.

400x256	**16x16**	**354x472**
287	1	450

Table2. Times in seconds of blurring images of given resolution in seconds.

Critical analysis

One of the limitations we noticed is the fact that distributor has to synchronize with workers and then it has to read a new line into a buffer. Although it will never deadlock however there is an unnecessary delay and lines can be buffered when workers are being synchronized if another solution is chosen.

Due to hardware limitations, our buffer has to be contained withing certain limit therefore we cannot read images of the width greater than maximum value of INT divided by three. This is because we use nested for loops where we iterate over the three lines of a buffer.

A blur filter that adds black frame affects the final image when it is blurred multiple times as the black border changes the average/median values. Average filter would could affect the 'white balance' of small images when blurred multiple times.

One more issue we encountered is that we cannot pause the blurring process immediately. Sometimes we have to wait for workers to finish their computation and synchronize again so that a PAUSE signal is propagated to all threads.

The major advantage of our system is that it cannot deadlock because we synchronize all threads and there is mostly one-way communication along the system.