Gebze Technical University

Computer Engineering

CSE 244

2017 Spring

MIDTERM REPORT

BURAK DEMİRCİ

141044091

Bu projede üç adet program yazmamız gerekiyor.Bu programlar; **timeServer**, **seeWhat** ve **showResult**.

Program 1 – timerServer

Bu program bizim ana server'ımız olacak.
Çalıştırlama şekli:
./timeServer <ticks in miliseconds> <n> <mainpipename>
<ticks in miliseconds> : Serveriın kaç milisaniyede bir clientdan gelen requestleri kontrol edeceği değer.
<n> : İnteger bir değer. Server her client için 2n x 2n invertible matris oluşturacak.
<mainpipename> : Server ile clientin haberleşeceği ana fifonun adı.

Bu programın yapması gerekenler.
-- Her client için bir proses oluşturacak ve o prosesler her client için 2n x 2n
invertable random bir matris oluşturacak ve clienta gönderecek. Client request
sinyalini ana fifo üzerinden gönderecek ve server client için oluşturduğu matrisi
başka bir fifo üzerinden gönderecek.
-- **Bu programı oluşturması gereken log dosyaları ve log dosyasına yazılması gereken bilgiler.**
Bu programın 1 adet log dosyası oluşturması gerekiyor.Log dosyasına
Matrisin oluşturulduğu zaman(milisaniya olarak)
Clientın pid'si
oluşturulan matrisin determinantı.

timerServer programı her bir client için bir proses acip bu clientin bütün isteklerini karşılamaktadır.

Oluşan her clientle arasında Clientpidsi kullanılarak bir fifo oluşturulup bilgiler bu kanalla aktarılmaktadır nedeni ise bilgilerin karışmaması ve severi meşgul etmemesi

Program 2 – seeWhat

Bu program bizim clientımız olacak.Birden fazla çalıştırılabilir.
Çalıştırılma şekli:
./seeWhat <mainpipename>
Bu programın yapması gerekenler
Bu program serverdan veri(2n x 2n invertible matris) alabilmek için servera
reqest sinyali gönderecek. Aynı zamanda ana fifo üzerinden pid sini de göndermesi gerekiyor.Serverdan matrisi alacak ve bu matrisle ilgili bazı işlemler yapacak.Bu işlemleri yapmak için en az 2 tane proses oluşturmalı. Oluşturacağı proseslerden biri n x n lik shifted inverse matrisi bulacak ve **result1** i hesaplayacak.
Result1 = det(orjinal matris) – det(shifted inverse matris)
Diğer proses 2d convolution matris oluşturak ve **result2** yi hesaplayacak.
Result2 = det(orjinsl matris) – det(2d conconvolution)
Program result1 ve result2 nin ne kadar zamanda(time elapsed) hesaplandığını
bulaması gerekiyor.
Shifted Inverse matris

nxn	**nxn**
nxn	**nxn**

Yukardaki şekildeki gibi ana matrisi 4'e bölüp her bir birimin tersini alıp yerine koyarak yeni bir matris oluşturma işlemidir.

2d convolution Matris

Bu matrisin nasıl oluşturulduğunu öğren. Raporda 2d convolutionun nerde kullanıldığı yazılacak.
Convolutionda kullanılacak matrisler
Kernel matrix Skew-symetric matrix1 Skew-symetric matrix2

2DConvolution matrix asıl kullanım alanı resim işleme (image proccesing)

Belli başlı kernel sistemlerinde resimleri bulanıklaştırma siyah beyaz yapma resim ayarlarıyla oynamak amaçlı kullanılıyor. Kullanılmasının en büyük nedeni resimler bilgisayarda ve ekranda martix olarak ifade edilmesidir aşağıda bununla alakalı birkaç görsel bulunmaktadır

2D Convolution

$$\begin{pmatrix} 17 & 24 & 1 & 8 & 15 \\ 23 & 5 & 7 & 14 & 16 \\ 4 & 6 & 13 & 20 & 22 \\ 10 & 12 & 19 & 21 & 3 \\ 11 & 18 & 25 & 2 & 9 \end{pmatrix} * \begin{pmatrix} 1 & 3 & 1 \\ 0 & 5 & 0 \\ 2 & 1 & 2 \end{pmatrix}$$

full same valid

17	75	90	35	40	53	15
23	159	165	45	105	137	16
38	198	120	165	205	197	52
56	95	160	200	245	184	35
19	117	190	255	235	106	53
20	89	160	210	75	90	6
22	47	90	65	70	13	18

(1) Valid region

Make the image big enough so that there is a guard ring around the region of interest. That is, use only the *valid* portion of the convolution. The resulting image will be smaller by the width of the convolution operator, as suggested by the outlined region below:

(2) Zero surround

Assume the image is embedded in a sea of zeros, as shown below. This is the assumption made by the *full* and *same* methods of conv2.

(3) Nearest neighbor

Assume that the pixels just outside the image are the same as their nearest neighbor inside the image. This is the situation portrayed below (obtained with add_border(rgb, 'dup',width))

(4) Circular or cyclic convolution

Consider the extended image to be a tiled version of the original, and then convolve the central image using portions of the adjacent tiles at the borders.

The following matlab code constructs a tiled image:

```
>> out = repmat(rgb,3,3);
```

A cyclic border can also be constructed from add_border(rgb, 'circ',width).

(5) Fancy Extrapolation

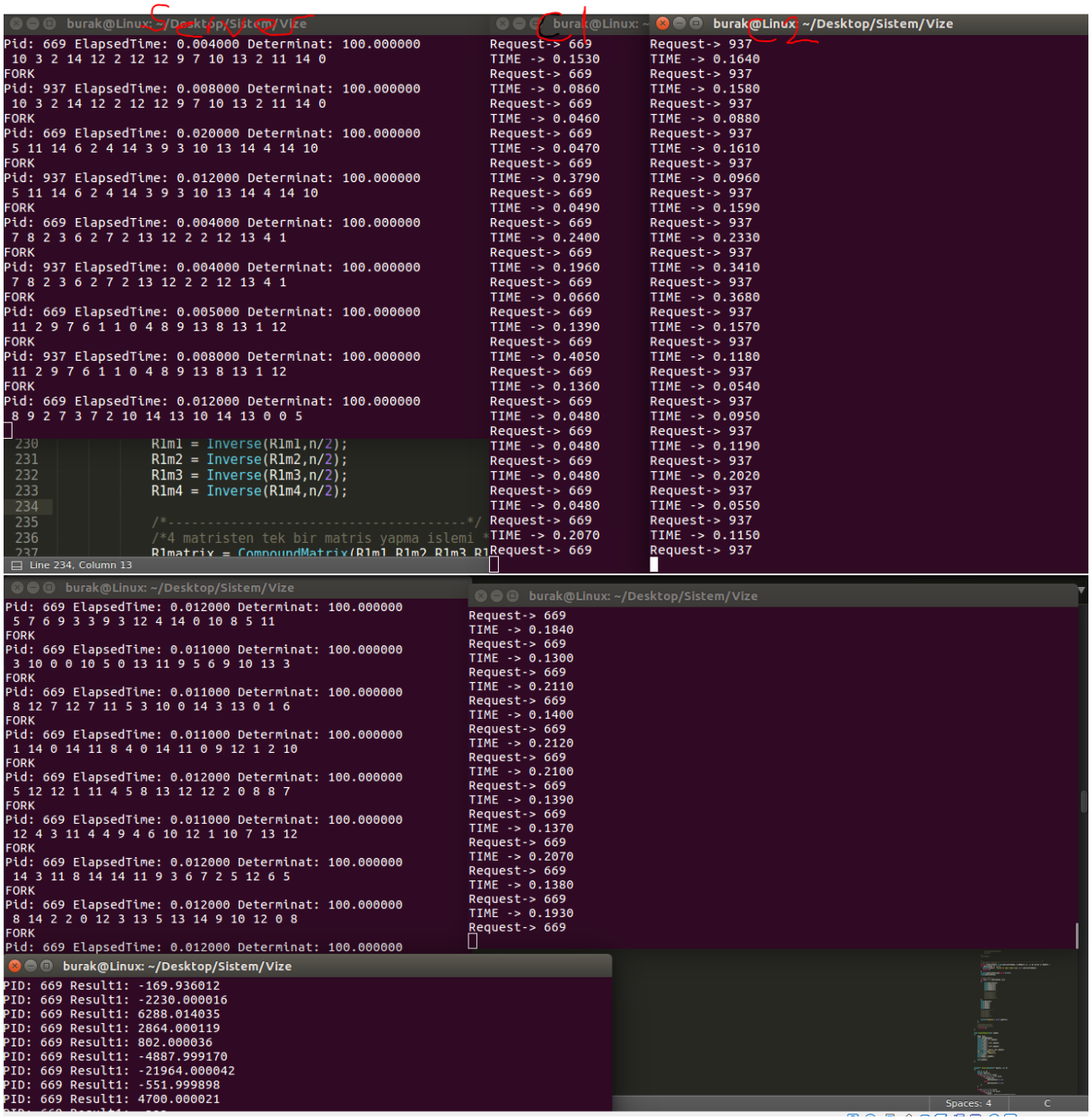
One could contemplate using a fancy extrapolation scheme, perhaps based on cubic splines, to obtain pixels outside the original image. This really has no more justification than the other methods described above, and is no substitute for making the original image much larger than the region of interest.

Bu programın oluşturması gereken log dosyaları ve içerikleri
Bu program her işlem için bir tane log dosyası oluşturacak.
Her log dosyasına işlemler sonucu oluşan matrisler matlab formatında yazılacak
(Orijinal matris, shifted inverse matris, 2d convolution matris

Program 3 – showResult

Programın çalıştırılma şekli
./ShowResult (argümansız)
Bu program sonuçları yazacak programımız olacak
Her clientdan result1, result2 ve clientın pidsini alacak ve onları dosyasına ve
ekrana aşağıdaki gibi yazdıracak.

Ekrana yazılacak bilgiler
pid Result1 Result2
. . .
. . .


burak@Linux: ~/Desktop/Sistem/Vize
Pid: 669 ElapsedTime: 0.004000 Determinat: 100.000000
10 3 2 14 12 2 12 12 9 7 10 13 2 11 14 0
FORK
Pid: 937 ElapsedTime: 0.008000 Determinat: 100.000000
10 3 2 14 12 2 12 12 9 7 10 13 2 11 14 0
FORK
Pid: 669 ElapsedTime: 0.020000 Determinat: 100.000000
5 11 14 6 2 4 14 3 9 3 10 13 14 4 14 10
FORK
Pid: 937 ElapsedTime: 0.012000 Determinat: 100.000000
5 11 14 6 2 4 14 3 9 3 10 13 14 4 14 10
FORK
Pid: 669 ElapsedTime: 0.004000 Determinat: 100.000000
7 8 2 3 6 2 7 2 13 12 2 2 12 13 4 1
FORK
Pid: 937 ElapsedTime: 0.004000 Determinat: 100.000000
7 8 2 3 6 2 7 2 13 12 2 2 12 13 4 1
FORK
Pid: 669 ElapsedTime: 0.005000 Determinat: 100.000000
11 2 9 7 6 1 1 0 4 8 9 13 8 13 1 12
FORK
Pid: 937 ElapsedTime: 0.008000 Determinat: 100.000000
11 2 9 7 6 1 1 0 4 8 9 13 8 13 1 12
FORK
Pid: 669 ElapsedTime: 0.012000 Determinat: 100.000000
8 9 2 7 3 7 2 10 14 13 10 14 13 0 0 5
230 R1m1 = Inverse(R1m1,n/2);
231 R1m2 = Inverse(R1m2,n/2);
232 R1m3 = Inverse(R1m3,n/2);
233 R1m4 = Inverse(R1m4,n/2);
234
235 /*-----------------------------------*/
236 /*4 matristen tek bir matris yapma islemi *
237 R1matrix = CompoundMatrix(R1m1,R1m2,R1m3,R1
Line 234, Column 13
Request-> 669
TIME -> 0.1530
Request-> 669
TIME -> 0.0860
Request-> 669
TIME -> 0.0460
Request-> 669
TIME -> 0.0470
Request-> 669
TIME -> 0.3790
Request-> 669
TIME -> 0.0490
Request-> 669
TIME -> 0.2400
Request-> 669
TIME -> 0.1960
Request-> 669
TIME -> 0.0660
Request-> 669
TIME -> 0.1390
Request-> 669
TIME -> 0.4050
Request-> 669
TIME -> 0.1360
Request-> 669
TIME -> 0.0480
Request-> 669
TIME -> 0.0480
Request-> 669
TIME -> 0.0480
Request-> 669
TIME -> 0.0480
Request-> 669
TIME -> 0.2070
Request-> 669
burak@Linux: ~/Desktop/Sistem/Vize
Request-> 937
TIME -> 0.1640
Request-> 937
TIME -> 0.1580
Request-> 937
TIME -> 0.0880
Request-> 937
TIME -> 0.1610
Request-> 937
TIME -> 0.0960
Request-> 937
TIME -> 0.1590
Request-> 937
TIME -> 0.2330
Request-> 937
TIME -> 0.3410
Request-> 937
TIME -> 0.3680
Request-> 937
TIME -> 0.1570
Request-> 937
TIME -> 0.1180
Request-> 937
TIME -> 0.0540
Request-> 937
TIME -> 0.0950
Request-> 937
TIME -> 0.1190
Request-> 937
TIME -> 0.2020
Request-> 937
TIME -> 0.0550
Request-> 937
TIME -> 0.1150
Request-> 937
burak@Linux: ~/Desktop/Sistem/Vize
Pid: 669 ElapsedTime: 0.012000 Determinat: 100.000000
5 7 6 9 3 3 9 3 12 4 14 0 10 8 5 11
FORK
Pid: 669 ElapsedTime: 0.011000 Determinat: 100.000000
3 10 0 0 10 5 0 13 11 9 5 6 9 10 13 3
FORK
Pid: 669 ElapsedTime: 0.011000 Determinat: 100.000000
8 12 7 12 7 11 5 3 10 0 14 3 13 0 1 6
FORK
Pid: 669 ElapsedTime: 0.011000 Determinat: 100.000000
1 14 0 14 11 8 4 0 14 11 0 9 12 1 2 10
FORK
Pid: 669 ElapsedTime: 0.012000 Determinat: 100.000000
5 12 12 1 11 4 5 8 13 12 12 2 0 8 8 7
FORK
Pid: 669 ElapsedTime: 0.011000 Determinat: 100.000000
12 4 3 11 4 4 9 4 6 10 12 1 10 7 13 12
FORK
Pid: 669 ElapsedTime: 0.012000 Determinat: 100.000000
14 3 11 8 14 14 11 9 3 6 7 2 5 12 6 5
FORK
Pid: 669 ElapsedTime: 0.012000 Determinat: 100.000000
8 14 2 2 0 12 3 13 5 13 14 9 10 12 0 8
FORK
Pid: 669 ElapsedTime: 0.012000 Determinat: 100.000000
burak@Linux: ~/Desktop/Sistem/Vize
Request-> 669
TIME -> 0.1840
Request-> 669
TIME -> 0.1300
Request-> 669
TIME -> 0.2110
Request-> 669
TIME -> 0.1400
Request-> 669
TIME -> 0.2120
Request-> 669
TIME -> 0.2100
Request-> 669
TIME -> 0.1390
Request-> 669
TIME -> 0.1370
Request-> 669
TIME -> 0.2070
Request-> 669
TIME -> 0.1380
Request-> 669
TIME -> 0.1930
Request-> 669
burak@Linux: ~/Desktop/Sistem/Vize
PID: 669 Result1: -169.936012
PID: 669 Result1: -2230.000016
PID: 669 Result1: 6288.014035
PID: 669 Result1: 2864.000119
PID: 669 Result1: 802.000036
PID: 669 Result1: -4887.999170
PID: 669 Result1: -21964.000042
PID: 669 Result1: -551.999898
PID: 669 Result1: 4700.000021
Spaces: 4
C