Yıldız Teknik Üniversitesi

Bilgisayar Mühendisliği Bölümü

BLM2021 – Alt Seviye Programlama 2. Ödev

İbrahim Alperen Kürüm – 20011052 – Grup 1

15.01.2023

DILATION

KOD TASLAĞI

NOT: Tüm indexler asm koduna çevrilirken byte olarak alınağı için 2 katına çıkarılmıştır

```
Int f = filter_size
int d  = filter_size/2        n = 512*512
for(i = 513 * d; range(513-f); i = i+512)
|       for(j = i; range(513-f); j = j+1)
|       |       max = 0;
|       |       for(k = j- (513*d); range(f); j = j+512)
|       |       |       for(t = k ; range(f); t = t+1)
|       |       |       |       if(resim_org[t] > max)
|       |       |       |               max = resim_org[t]
|       |       |       |       |
|       |       PUSH max
for(i = (n-1) – 513*d ; range(513-f); i = i+512)
|       for(j = i; range(513-f); j = j+1)
|               POP max
|               resim_org[j] = max
|       |
```

Satır satır açıklama için sonraki sayfaya gidiniz.

Satır Açıklamaları

NOT: EROSION işlemi DILATION işleminin sadece 2 satır değiştirilmiş halidir AX register'ına atanan değerler ve CMP den sonraki jump komutları farklıdır.

```
64  void Dilation(int n, int filter_size, short* resim_org) {
65      __asm {
66          MOV EDI,resim_org
67          MOV EBX, filter_size
68          SHR EBX, 1
69          MOV EAX, 1026
70          XOR EDX, EDX
71          MUL EBX
72          MOV EBX, EAX
73          MOV ECX, 513
74          SUB ECX, filter_size
75      L1: MOV ESI,EBX
76          MOV EDX,ECX
77          MOV ECX, 513
78          SUB ECX, filter_size
79
80      L2: PUSH EDX
81          PUSH EBX
82          PUSH ECX
83          MOV ECX, filter_size
84          SHR ECX, 1
85          MOV EAX, 1026
86          XOR EDX, EDX
87          MUL ECX
88          SUB EBX,EAX
89          MOV ECX, filter_size
90          XOR AX,AX
91
92      L3: PUSH EBX
93          PUSH ECX
94          MOV ECX,filter_size
95
96      L4: CMP WORD PTR[EBX+EDI],AX
97          JBE F
98          MOV AX,WORD PTR[EBX+EDI]
99
100     F:  ADD EBX,2
101         LOOP L4
102         POP ECX
103         POP EBX
104         ADD EBX, 1024
105         LOOP L3
106         POP ECX
107         POP EBX
108         ADD EBX, 2
109         POP EDX
110         PUSH AX
111         LOOP L2
112         MOV ECX,EDX
113         MOV EBX,ESI
114         ADD EBX, 1024
115         LOOP L1
116
117
118         MOV EBX, filter_size
119         SHR EBX, 1
120         MOV EAX, 1026
121         XOR EDX, EDX
122         MUL EBX
123         MOV EBX,n
124         DEC EBX
125         SHL EBX,1
126         SUB EBX,EAX
127
128         MOV ECX, 513
129         SUB ECX, filter_size
130
131     L5: MOV ESI,EBX
132         MOV EDX,ECX
133         MOV ECX, 513
134         SUB ECX, filter_size
135     L6:
136         POP AX
137         MOV WORD PTR[ESI+EDI],AX
138
139         SUB ESI,2
140         LOOP L6
141         SUB EBX, 1024
142         MOV ECX,EDX
143         LOOP L5
144     }
145     printf("\nDilation islemi sonucunda resim \"dilated.pgm\" ismiyle olusturuldu
```

a) 66-72>>> Resmin olduğu adresi EDI registerına koydum koydum. Sonra filtre sayısının yarısını resmin satır sayısının byte cinsinden değeri (513*2 byte) ile çarpıyorum ve EBX registerına aktarıyorum çünkü resmin ilk pikselden başlanırsa resmin dışına erişmeye çalışır bundan kurtulmak için de padding işlemini gerçekleştirmem gerekir, padding işlemi beklenmediği için kenarlarda 1, 2 veya 3 piksel atlanarak işleme başlanıyor.

b) 73-74>>> ECX registerına da ilk döngünün dönme sayısı olan değeri ekliyorum.

c) 75-79>>> İlk döngümün başlangıç sınırları burasıdır. Buranın içerisinde index değerim ve döngünün dönme değeri kullanılmayan registerlar içerisine atılıyor. Sonra diğer döngü için olan dönme değeri ayalanıyor. Bir sonraki index değerimiz bu döngüdeki index değeri ile aynı olduğu için o değiştirilmiyor.

d) 80-91>>> 2. Döngüde ilk olarak döngülerdeki index ve döngü dönme değerleri stacke atılıyor. Sonra bir sonraki döngü için index ve döngü değerleri hesaplanıp registerlara konuyor. Ve max değişkenimin tutulduğu register (AX) sıfırlanıyor.

e) 92-95>>>3. Döngüde sadece index ve dönme değerleri stack'e atılıp yeni dönme değeri belirleniyor. Bir sonraki index değerimiz bu döngüdeki index değeri ile aynı olduğu için o değiştirilmiyor.

f) 96-99>>>4. Döngü yani son dögüde sadece karşılaştırma işlemi yapılıyor. Daha önceden 0 tanımlanan max(EAX) değeri ile yapısal elemanımızın içindeki değerler tek tek burada karşılaştırılıyor. En son 3. Ve 4. Döngüden çıktığında max değerinin(AX) içinde yapısal eleman içerisindeki en büyük değer bulunuyor.

g) 100-101>>> 4. Döngüdeki index değerinin artım miktarı ayarlanıyor.

h) 102-105>>> 3. Döngüdeki index ve dönme değerleri stackden çekilip index değerinin artma miktarı ayarlanıyor.

i) 106-111>>> 2. Döngüdeki index ve dönme değerleri stackden çekilip index değerinin artma miktarı ayarlanıyor. Sonra yapısal elemanda bulunan max değeri(AX) push işlemi ile stack'e atılıyor.

j) 112-115>>> 1. Döngüdeki index ve dönme değerleri ni tuttuğum registerlardan geri yerlerine koyuyor.

k) 118-143>>> Bu arada kullanılan ilk 2 döngünün tersten gerçekleştirilmesi yapılıyor. Gerekli değerler yerleştirildikten sonra stackten değerleri çekerek resmimizde gerekli piksele o değerin ataması yapılıyor.

RESİMLER

Orijinal hali

Filtre boyutu = 3

Filtre boyutu =5

Filtre boyutu = 7

EROSION

KOD TASLAĞI

NOT: Tüm indexler asm koduna çevrilirken byte olarak alınağı için 2 katına çıkarılmıştır

```
Int f = filter_size
int d  = filter_size/2                  n = 512*512
for(i = 513 * d; range(513-f); i = i+512)
|       for(j = i; range(513-f); j = j+1)
|       |       max = 255;
|       |       for(k = j- (513*d); range(f); j = j+512)
|       |       |       for(t = k ; range(f); t = t+1)
|       |       |       |       if(resim_org[t] < max)
|       |       |       |               min = resim_org[t]
|       PUSH min       |       |


for(i = (n-1) – 513*d ; range(513-f); i = i+512)
|       for(j = i; range(513-f); j = j+1)
|               POP min
|               resim_org[j] = min
|       |
```

Satır satır açıklama için sonraki sayfaya gidiniz.

Satır Açıklamaları

```
148  void Erosion(int n, int filter_size, short* resim_org) {
149
150      __asm {
151          MOV EDI,resim_org
152          MOV EBX, filter_size
153          SHR EBX, 1
154          MOV EAX, 1026
155          XOR EDX, EDX
156          MUL EBX
157          MOV EBX, EAX
158          MOV ECX, 513
159          SUB ECX, filter_size
160  L1:     MOV ESI,EBX
161          MOV EDX,ECX
162          MOV ECX, 513
163          SUB ECX, filter_size
164
165  L2:     PUSH EDX
166          PUSH EBX
167          PUSH ECX
168          MOV ECX, filter_size
169          SHR ECX, 1
170          MOV EAX, 1026
171          XOR EDX, EDX
172          MUL ECX
173          SUB EBX,EAX
174          MOV ECX, filter_size
175          MOV AX, 255
176
177  L3:     PUSH EBX
178          PUSH ECX
179          MOV ECX,filter_size
180
181  L4:     CMP WORD PTR[EBX+EDI],AX
182          JA F
183          MOV AX,WORD PTR[EBX+EDI]
184
185  F:      ADD EBX,2
186          LOOP L4
187          POP ECX
188          POP EBX
189          ADD EBX, 1024
190          LOOP L3
191          POP ECX
192          POP EBX
193          ADD EBX, 2
194          POP EDX
195          PUSH AX
196          LOOP L2
197          MOV ECX,EDX
198          MOV EBX,ESI
199          ADD EBX, 1024
200          LOOP L1
201
202
203          MOV EBX, filter_size
204          SHR EBX, 1
205          MOV EAX, 1026
206          XOR EDX, EDX
207          MUL EBX
208          MOV EBX,n
209          DEC EBX
210          SHL EBX,1
211          SUB EBX,EAX
212
213          MOV ECX, 513
214          SUB ECX, filter_size
215
216  L5:     MOV ESI,EBX
217          MOV EDX,ECX
218          MOV ECX, 513
219          SUB ECX, filter_size
220  L6:
221          POP AX
222          MOV WORD PTR[ESI+EDI],AX
223
224          SUB ESI,2
225          LOOP L6
226          SUB EBX, 1024
227          MOV ECX,EDX
228          LOOP L5
229
230      }
231      printf("\nErosion islemi sonucunda resim \"eroded.pgm\" ismiyle olusturuldu.
232  }
```

a) 151-157>>> Resmin olduğu adresi EDI registerına koydum koydum. Sonra filtre sayısının yarısını resmin satır sayısının byte cinsinden değeri (513*2 byte) ile çarpıyorum ve EBX registerına aktarıyorum çünkü resmin ilk pikselden başlanırsa resmin dışına erişmeye çalışır bundan kurtulmak için de padding işlemini gerçekleştirmem gerekir, padding işlemi beklenmediği için kenarlarda 1, 2 veya 3 piksel atlanarak işleme başlanıyor.

b) 158-159>>> ECX registerına da ilk döngünün dönme sayısı olan değeri ekliyorum.

c) 160-164>>> İlk döngümün başlangıç sınırları burasıdır. Buranın içerisinde index değerim ve döngünün dönme değeri kullanılmayan registerlar içerisine atılıyor. Sonra diğer döngü için olan dönme değeri ayalanıyor. Bir sonraki index değerimiz bu döngüdeki index değeri ile aynı olduğu için o değiştirilmiyor.

d) 165-176>>> 2. Döngüde ilk olarak döngülerdeki index ve döngü dönme değerleri stacke atılıyor. Sonra bir sonraki döngü için index ve döngü değerleri hesaplanıp registerlara konuyor. Ve min değişkenimin tutulduğu register (AX) grey scale'de en yüksek değer olan 255 konuyor.

e) 177-180>>>3. Döngüde sadece index ve dönme değerleri stack'e atılıp yeni dönme değeri belirleniyor. Bir sonraki index değerimiz bu döngüdeki index değeri ile aynı olduğu için o değiştirilmiyor.

f) 181-184>>>4. Döngü yani son dögüde sadece karşılaştırma işlemi yapılıyor. Daha önceden 0 tanımlanan max(EAX) değeri ile yapısal elemanımızın içindeki değerler tek tek burada karşılaştırılıyor. En son 3. Ve 4. Döngüden çıktığında min değerinin(AX) içinde yapısal eleman içerisindeki en küçük değer bulunuyor.

g) 185-186>>> 4. Döngüdeki index değerinin artım miktarı ayarlanıyor.

h) 187-190>>> 3. Döngüdeki index ve dönme değerleri stackden çekilip index değerinin artma miktarı ayarlanıyor.

i) 191-196>>> 2. Döngüdeki index ve dönme değerleri stackden çekilip index değerinin artma miktarı ayarlanıyor. Sonra yapısal elemanda bulunan en küçük değer(AX) push işlemi ile stack'e atılıyor.

j) 197-200>>> 1. Döngüdeki index ve dönme değerleri ni tuttuğum registerlardan geri yerlerine koyuyor.

k) 200-228>>> Bu arada kullanılan ilk 2 döngünün tersten gerçekleştirilmesi yapılıyor. Gerekli değerler yerleştirildikten sonra stackten değerleri çekerek resmimizde gerekli piksele o değerin ataması yapılıyor.

RESİMLER

Orijinal hali

Filtre boyutu = 3

Filtre boyutu =5

Filtre boyutu = 7