

به نام خدا

پروژه درس ساختار و زبان کامپیوتر

1403-1متر

سعید نوفرستی 403106838

- هدف این پروژه پیاده‌سازی و مقایسه‌ی *کانولوشن دو بعدی تصویر (2D Convolution) به دو روش است:

۱. پیاده‌سازی ساده با C (Baseline)
۲. پیاده‌سازی بسیار سریع با Assembly + AVX SIMD

سپس:

- مقایسه زمان اجرا (Benchmark)
- محاسبه Speedup
- Zero Padding
- تشخیص لبه با Sobel
- تشخیص شی (Object Recognition)
- اجرای خودکار روی صدها تصویر

این برنامه روی تصاویر pgm کار میکند پس ابتدا نیاز داریم
این دستور را در ترمینال لینوکس وارد کنیم

sudo apt update

sudo apt install imagemagick

سپس هر تصویر دلخواه را با دستور

convert input.jpg -colorspace Gray input.pgm

به تصویر pgm تبدیل میکنیم

ساختار کلی برنامه به اینصورت است

فایل اصلی برنامه main.c —|

نسخه C کانولوشن conv.c —|

Assembly + AVX # نسخه conv.asm —|

هدر conv.h —|

اندازه‌گیری زمان timer.c —|

پردازش 100/400 تصویر object_recognition.c —|

تصاویر ورودی /inputs —|

پیکربندی اجرایی Makefile —|

حالا به سراغ اجرای برنامه روی تصویر دلخواه میرویم
این برنامه چهار مد کاری دارد

۱. blur (تک تصویره)

۲. sharpen (تک تصویره)

۳. Edge_Detection (تک تصویره)

۴. object recognition (روی تصویر دارای شی یا بدون شی)

برای شروع یک تصویر(pgm) را انتخاب میکنم برای مثال تصویر زیر:



1. اجرای فیلتر blur

make clean

make

./main Blur input30.pgm

خروجی C و ASM :



که به وضوح نسبت به تصویر اول blur شده است
خروجی کد در ترمینال به این شکل بود:

```
saeed@saeed-Vivobook-ASUSLaptop-K3605VU-K3605VU:~/me/Convolution$ ./main Blur input30.pgm
C time = 0.005142
ASM time = 0.000719
speedup : 7.153068
error : 0.000000
```

که نشان میدهد سرعت کار با اسملی 7.15 برابر کد C است
و ارور صفر نشان دهنده این است که خروجی asm و C دقیقا
مشابه هم هست با این تفاوت که سرعت asm بسیار بیشتر
بوده است

2. اجرای فیلتر Sharpen
حالا تصویر blur شده در قسمت قبل با نام input.pgm ذخیره کرده و روی آن فیلتر sharpen را اجرا میکنیم:

```
make clean
make
./main Sharpen input.pgm
```

اینبار خروجی در ترمینال به این شکل بود

```
saeed@saeed-Vivobook-ASUSLaptop-K3605VU-K3605VU:~/me/Convolution$ ./main Sharpen input.pgm
C time = 0.005701
ASM time = 0.000936
speedup : 6.088460
error : 0.000000
```

و خروجی تصویر به شکل زیر است که به وضوح از تصویر blur شده قبل شارپ تر شده است

ASM



C



3. اجرای فیلتر Edge_Detection

make clean

make

./main Edge_Detection input30.pgm

خروجی کد

```
saeed@saeed-Vivobook-ASUSLaptop-K3605VU-K3605VU:~/me/Convolution$ ./main Edge_Detection input30.pgm
C time = 0.004440
ASM time = 0.000866
speedup : 5.127548
error : 0.000000
```

تصویر خروجی

ASM



C



لبه های تصویر مشخص شده هست

-حالا سراغ کار نهایی میرویم که تشخیص شی در ۴۰۰ تصویر مختلف هست اینبار در پوشه inputs تصاویر مختلف که ۲۰۰ تصویر دارای شی و ۲۰۰ تصویر تصاویر سفید و خاکستری و مشکی و... بدون شی هستند را قرار میدهیم تا روی این تصاویر این ازمایش را انجام دهیم

make clean

make

./main object_recognition

خروجی کد به این صورت بود

```
REPORT
C total time : 2.702299 sec
ASM total time : 0.765157 sec
Speedup : 3.53x
Objects (C) : 199 / 400
Objects (ASM) : 199 / 400
```

حتی با وجود اینکه در تایم اسملی تایم zero padding هم محاسبه شده باز هم حدود ۳.۵ برابر سرعت بیشتری به ما میدهد که شگفت انگیز است
اما میبینیم که خروجی از لحاظ تشخیص تعداد تصویر برای هر دو کد C, asm یکسان است
دقت محاسبه به این صورت است که :

۱ تصویر که دارای شی بوده به اشتباه بدون شی تشخیص داده شده و ۳۹۹ تصویر درست تشخیص داده شده است:
پس با دقت $399/400 * 100$ درصد محاسبه انجام شده

یعنی دقت برابر بوده با :
% 99.75

این پروژه در لینک زیر در گیت هاب هم در دسترس است

<https://github.com/SSN4444/Convolution.git>

با تشکر از استاد گرامی