

Processamento de Imagens Utilizando GPU

Wedeueis Braz RA: 11004813

Implementação do algoritmo de rotulação em imagens binárias

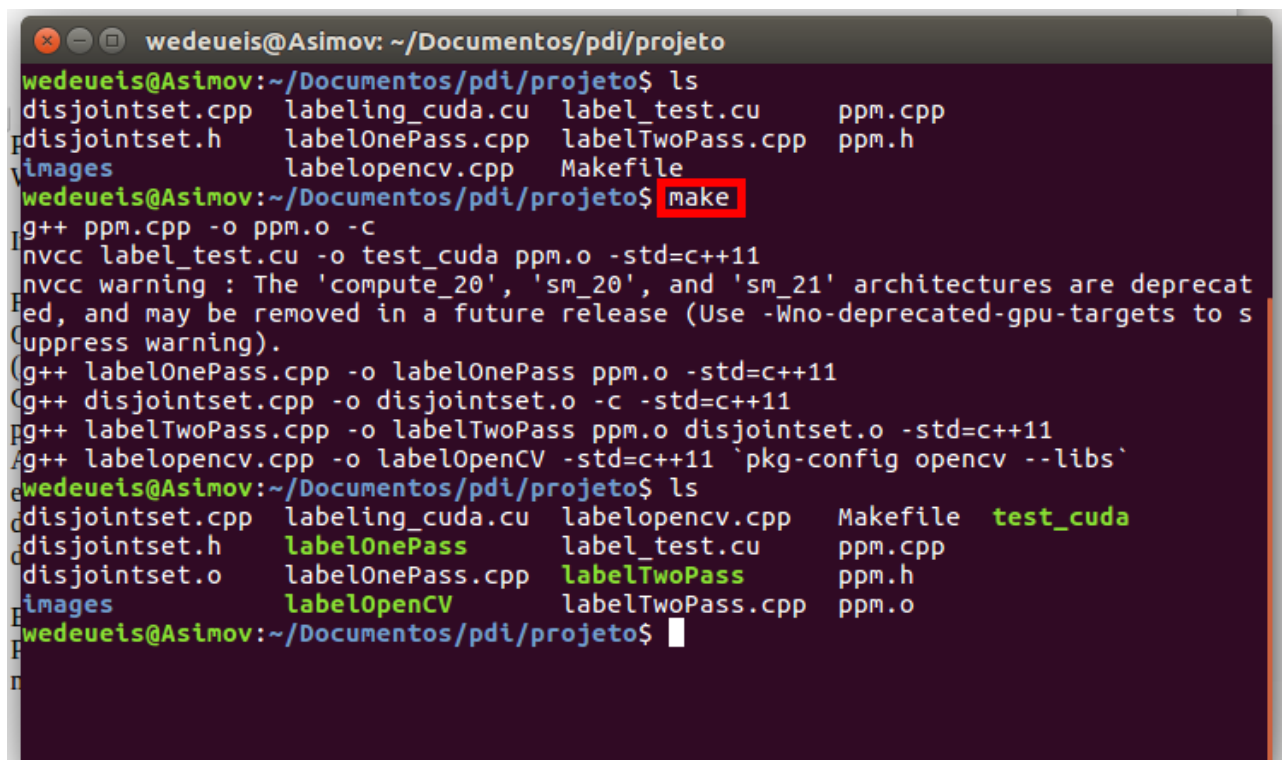
Foram implementados quatro algoritmos para fins de comparação.

Os programas que implementam as técnicas de rotulação de uma passada em CPU (labelOnePass.cpp), duas passadas em CPU (labelTwoPass.cpp e disjointset.cpp) e propagação em GPU (labeling_cuda.cu e label_test.cu) usam o objeto de imagem definido em ppm.cpp, obtido nos projetos da vision disponibilizados em aula. O programa que usa a implementação do OpenCV usa o formato de imagem Mat da própria biblioteca.

As imagens de teste estão na pasta imagens e apresentam 6 imagens com resoluções decrescentes entre 2048x2048 até 35x25 com objetos similares ocupando aproximadamente a mesma proporção de pixels e 3 imagens com a mesma resolução de 100x100 pixels porém com número de objetos diferentes.

Execução dos algoritmos:

Para compilar os programas basta entrar na pasta do projeto no terminal e executar o comando make:



```
wedeueis@Asimov: ~/Documentos/pdi/projeto
wedeueis@Asimov:~/Documentos/pdi/projeto$ ls
disjointset.cpp  labeling_cuda.cu  label_test.cu    ppm.cpp
disjointset.h    labelOnePass.cpp  labelTwoPass.cpp ppm.h
images          labelopencv.cpp  Makefile
wedeueis@Asimov:~/Documentos/pdi/projeto$ make
g++ ppm.cpp -o ppm.o -c
nvcc label_test.cu -o test_cuda ppm.o -std=c++11
nvcc warning : The 'compute_20', 'sm_20', and 'sm_21' architectures are deprecated, and may be removed in a future release (Use -Wno-deprecated-gpu-targets to suppress warning).
g++ labelOnePass.cpp -o labelOnePass ppm.o -std=c++11
g++ disjointset.cpp -o disjointset.o -c -std=c++11
g++ labelTwoPass.cpp -o labelTwoPass ppm.o disjointset.o -std=c++11
g++ labelopencv.cpp -o labelOpenCV -std=c++11 `pkg-config opencv --libs`
wedeueis@Asimov:~/Documentos/pdi/projeto$ ls
disjointset.cpp  labeling_cuda.cu  labelopencv.cpp  Makefile  test_cuda
disjointset.h    labelOnePass      label_test.cu    ppm.cpp
disjointset.o    labelOpenCV       labelTwoPass     ppm.h
images          labelTwoPass.cpp  ppm.o
wedeueis@Asimov:~/Documentos/pdi/projeto$
```

São quatro programas executáveis, 3 com algoritmos implementados e 1 que faz uso do algoritmo da biblioteca OpenCV.

Resultados:

```
wedeueis@Asimov:~/Documentos/pdi/projeto$ ./labelOpenCV
14 0 0 0 0 0 0 0 0
wedeueis@Asimov:~/Documentos/pdi/projeto$ ./labelOnePass
1692 26 18 4 1 0 0 2 5
wedeueis@Asimov:~/Documentos/pdi/projeto$ ./labelTwoPass
2489 37 26 6 1 0 1 4 8
wedeueis@Asimov:~/Documentos/pdi/projeto$ ./test_cuda
14944.1 66.3471 34.712 9.96736 4.11318 2.20669 6.96602 7.02355 7.03914
```

Dos programas implementados o mais eficiente como esperado foi o da biblioteca OpenCV, levando 14ms apenas na resolução mais alta de 2048x2048, e menos que 1ms para todo o restante.

Em segundo lugar o algoritmo de uma passada, em terceiro o de duas passadas e por último o implementado em cuda.

Como podemos ver a implementação em cuda se mostrou muito ineficiente, provavelmente devido a técnica simples empregada de propagação e a GPU de teste do modelo 930M de baixo desempenho para laptops, o que mostra a importância do estudo das técnicas de implementação paralela em Cuda para obter bons resultados.