

Universidade Federal de Minas Gerais  
Instituto de Ciências Exatas  
Departamento de Ciência da Computação  
Algoritmos e Estruturas de Dados II

# Trabalho Prático 0

## Documentação

Julia Tiemi Alkmim Morishita  
2015004240  
Abril de 2016

## Introdução

Existem diversos métodos para reconhecer um sinal dentro de um sinal maior, sendo um deles a Correlação Cruzada. Nela, comparamos duas séries baseado no deslocamento relativo entre elas.

Imagens do formato PMG são imagens B&W. Elas possuem em sua estrutura um identificador do formato “P2”, seguido por sua largura  $c$  e altura  $l$  em pixels. O próximo valor apresentado é o valor máximo que cada pixel pode ter em questão da intensidade da cor. Abaixo desses dados, segue uma matriz  $lxc$ , onde cada elemento é o valor do pixel em questão.

Para realizar a detecção de objetos nessa imagem, portanto, basta realizar a Correlação Cruzada, onde o sinal maior é a imagem completa e o sinal menor a parte da matriz que desejamos encontrar nela.

## Implementação

A primeira tarefa a ser realizada era a leitura das imagens PGM para armazenar seus dados, isso se aplica para a cena e para o objeto que se desejava encontrar.

### 1) Estrutura *PGM*:

Foi criado uma estrutura para armazenar todos os dados de uma determinada imagem. A estrutura *PGM* consiste em dois int,  $c$  e  $l$ , que representam as colunas (largura) e linha (altura) da imagem, portanto, da matriz. Além disso ela possui um unsigned char *maximo*, que armazena a intensidade maxima que um pixel pode assumir e um ponteiro de ponteiro para um unsigned char *dados*, que virá a armazenar a matriz.

### 2) Função *PGM LePGM(char\* entrada)*:

Para registrar os dados de uma imagem nessa estrutura, foi criada a função *LePGM*. Essa função recebe como parâmetro um dos argumentos de entrada. Isso significa que na linha de comando da execução do programa:

***./exec imagem\_cena.pgm imagem\_objeto.pgm imagem\_txt.pgm***

O primeiro argumento *argv[0]* é *exec*, *argv[1]* é *imagem\_cena.pgm*, *argv[2]* é *imagem\_objeto.pgm* e *argv[3]* é *imagem.txt*.

Para esse função, estamos interessados apenas em *argv[1]* e *argv[2]* como argumentos.

A saída dessa função é um ponteiro para uma estrutura *PGM* com os dados de uma certa imagem registrados.

A função abre o arquivo da imagem e aloca dinamicamente uma estrutura *PGM* para o armazenamento dos dados. Ao abrir o arquivo, foi escolhido o mode *rt* para leitura de um arquivo de texto, apenas como garantia que a imagem seria aberta como tal.

Após ter aberto o arquivo, ela lê as primeiras duas linhas do arquivo, salvando o que é de interesse na estrutura e usa esses dados pra alocar dinamicamente a matriz *PGM->dados*, que posteriormente é preenchida com os valores do arquivo original.

Terminado esse processo, o arquivo é fechado e a estrutura *PGM* é retornada para a *main*.

Isso é feito para a imagem da cena e do objeto.

Tendo em mãos esse dados, podemos começar a utilizar o algoritmo da Correlação Cruzada.

É necessário aplicar o algoritmo em todos os pixels possíveis da imagem. Para isso, foi realizada uma função *JanelaDeslizante*, que faz o deslocamento da “janela” que determina quais pixels estão sendo analisados.

3) Função *Ponto JanelaDeslizante(PGM \*cena, PGM \*obj)*:

Essa função recebe os dados de ambas imagens, de forma a determinar qual conjunto de pixels da cena está sendo comparada com o objeto de desejo.

Ela vai deslocando a janela até cobrir todos os pixels, chamando sempre a função que aplica a Correlação Cruzada para deslocamento.

Ela identifica a posição da janela pelo ponto “a11” da matriz da janela. Se a função da Correlação Cruzada retornar um valor maior que o anterior, este vem a ser substituído. No final do algoritmo, temos como saída o ponto que determina onde está o objeto na imagem.

4) Estrutura *Ponto*:

Estrutura simples para armazenamentos de dois inteiros que representam as coordenadas de um ponto na matriz.

5) *int CorrelacaoCruzada(PGM \*cena, PGM \*obj, Ponto p)*:

Dentro da função *JanelaDeslizante*, recorreremos ao algoritmo da Correlação Cruzada.

Essa função recebe os dados de ambas imagens, além de uma estrutura ponto que determina que parte da cena está sob análise.

É aplicada a Correlação Cruzada, definida por:

$$\sum_{j=0}^{m-1} \sum_{i=0}^{n-1} f[j, i] \cdot g[x + j, y + i]$$

E o valor dado por essa expressão é retornada para *JanelaDeslizante*.

Quando a cena for completamente analisada, *JanelaDeslizante* retorna uma estrutura *Ponto* que determina a localização do objeto em questão na cena.

6) Função *void Saida(char\* entrada, Ponto p)*:

Foi determinado que o ponto que determinasse o local do objeto na imagem fosse impresso num arquivo de texto, passado como argumento na execução do programa.

Essa função recebe esse argumento *imagem\_saida.txt* e o *Ponto p*.

O arquivo é aberto para escrita e o ponto *p* é impresso na forma “*y x*”.

Terminado a função do programa, temos vários leaks de memória, pois não era possível desalocar as matrizes das imagens na função *LePGM*, onde foram alocadas. Portanto, foi criada uma função para realizar tal tarefa.

7) Função *void Desaloca(PGM \*imagem)*:

Ela recebe a estrutura *PGM* e libera a matriz que foi alocada dinamicamente.

## Resultados

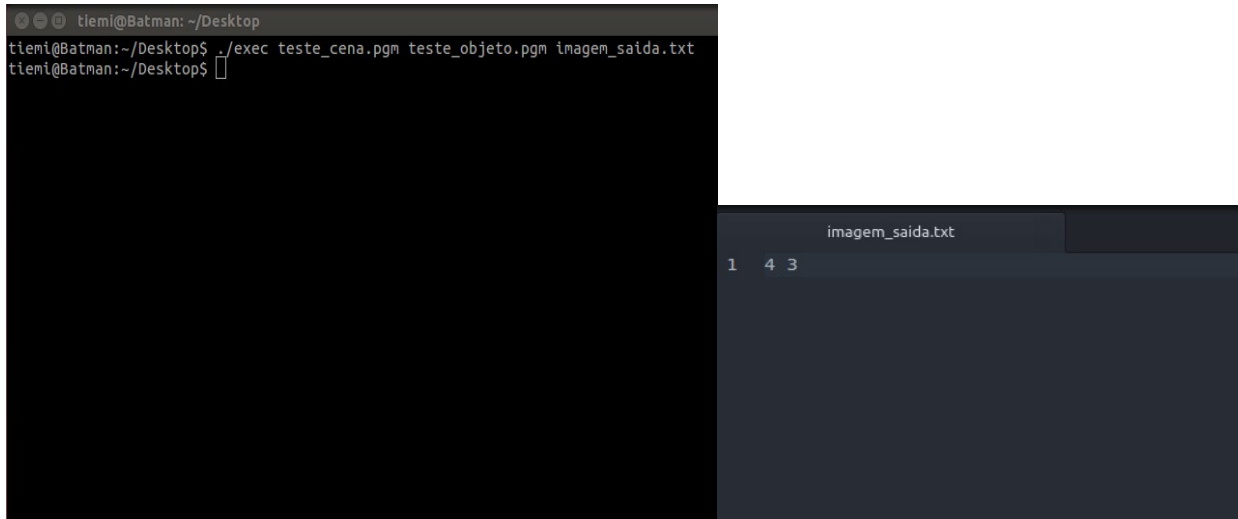
Foram realizados três testes com o programa, cada um com um par de cena e objeto.

1) exemplo\_cena.pgm e exemplo\_objeto.pgm

```
tiemi@Batman: ~/Desktop
tiemi@Batman:~/Desktop$ make
gcc -g -Wall -c tp0.c -o tp0.o
gcc -g -Wall -c TAD.c -o TAD.o
gcc -g -Wall tp0.o TAD.o -o exec
tiemi@Batman:~/Desktop$ ./exec exemplo_cena.pgm exemplo_objeto.pgm imagem_saida.
txt
tiemi@Batman:~/Desktop$
```

```
imagem_saida.txt
1 2 1
```

## 2) teste\_cena.pgm e teste\_objeto.pgm



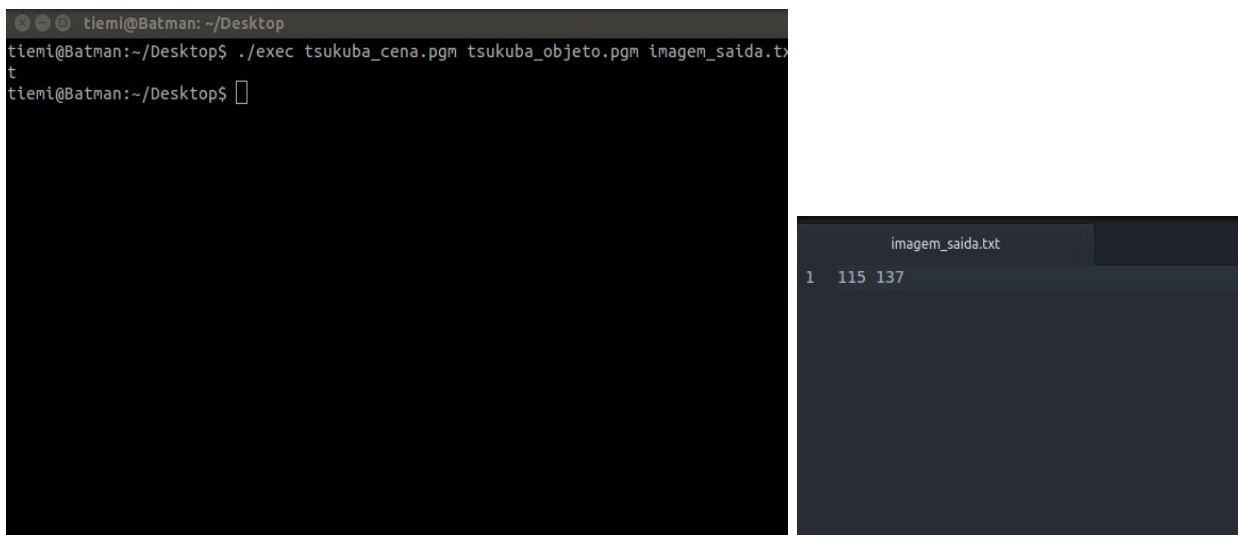
The image shows a terminal window and a text file. The terminal window has a title bar with three icons and the text 'tiemi@Batman: ~/Desktop'. The command prompt shows the execution of './exec teste\_cena.pgm teste\_objeto.pgm imagem\_saida.txt'. The text file 'imagem\_saida.txt' contains the output '1 4 3'.

```
tiemi@Batman: ~/Desktop
tiemi@Batman:~/Desktop$ ./exec teste_cena.pgm teste_objeto.pgm imagem_saida.txt
tiemi@Batman:~/Desktop$
```

imagem\_saida.txt

```
1 4 3
```

## 3) tsukuba\_cena.pgm e tsukuba\_objeto.pgm



The image shows a terminal window and a text file. The terminal window has a title bar with three icons and the text 'tiemi@Batman: ~/Desktop'. The command prompt shows the execution of './exec tsukuba\_cena.pgm tsukuba\_objeto.pgm imagem\_saida.txt'. The text file 'imagem\_saida.txt' contains the output '1 115 137'.

```
tiemi@Batman: ~/Desktop
tiemi@Batman:~/Desktop$ ./exec tsukuba_cena.pgm tsukuba_objeto.pgm imagem_saida.txt
tiemi@Batman:~/Desktop$
```

imagem\_saida.txt

```
1 115 137
```

Todos os teste foram bem sucedidos.

## Conclusão

O Trabalho Prático 0 foi realizado sem maiores dificuldades, tendo uma implementação bem simples.

A maior dificuldade encontrada foi na leitura do valor máximo do pixel como unsigned char. Esse problema foi resolvido com um inteiro auxiliar *aux* que foi utilizada na leitura. Esse valor foi mais tarde convertido em um unsigned char.

## Referências

<https://en.wikipedia.org/wiki/Cross-correlation>