Introdução a Técnicas de Programação - 2015.2

Descrição de projeto

Processamento de Imagem

Introdução

Processamento de imagem é qualquer forma de processamento de dados no qual a entrada e saída são imagens tais como fotografias ou quadros de vídeo. Ao contrário do tratamento de imagens, que preocupa-se somente na manipulação de figuras para sua representação final, o processamento de imagens é um estágio para novos processamentos de dados tais como aprendizagem de máquina (inteligência artificial) ou reconhecimento de padrões (sua câmera detecta um rosto para a foto, por exemplo). A maioria das técnicas envolve o tratamento da imagem como um sinal bidimensional, no qual são aplicados padrões de processamento de sinal.

Métodos de processamento de imagem

O termo **resolução** é freqüentemente usado como uma contagem de pixels em imagens digitais. Mas, quando a contagem de pixels é referenciada como resolução, a convenção é descrever a *resolução em pixels* como o conjunto de dois números positivos inteiros, em que o primeiro número é a quantidade de colunas (largura) de pixels e o segundo é número de linhas (altura) de pixels; algo como *640 X 480*, por exemplo.

A **segmentação** de imagem consiste em dividir a imagem em regiões que dizem respeito ao mesmo conteúdo e aplicação. Existem objetos de interesse em uma imagem e podemos isolar aqueles pixels que não fazem parte desses objetos. Tradicionalmente a segmentação de imagem tem sido vista como um estágio prévio de processamento para reconhecimento ou análise.

A **filtragem no domínio espacial** é uma técnica de processamento de imagens que normalmente manipula os valores de uma vizinhança para modificar a imagem digital. A imagem digital é uma matriz de pixels, sendo que a filtragem no domínio espacial trabalha com uma máscara, uma janela, que percorre toda a imagem realizando as operações de filtragem desejadas.

A filtragem linear baseia-se em dois conceitos, o da **convolução** e o da **correlação**, onde correlação é a forma como a mascara é alternada dentro da imagem; este processo visa calcular a soma dos pontos de cada posição, ou seja correlação é o processo de deslocamento do filtro. Já a convolução é praticamente igual à correlação, sendo que a única diferença é o fato de rotacionar em 180° o primeiro filtro. Esses conceitos são essenciais para realizar *blurring*, *sharpening*, deteccão de bordas, etc.

A **máscara (kernel ou matriz de convolução)** pode ser representada através da função: $p(i, j) = T[o(x, y)]$, onde $p(i, j)$ é a imagem processada, $o(x, y)$ é a imagem original e T é um operador em o, definido em uma certa vizinhança de (x, y). A manipulação da imagem utilizando as técnicas de filtragem no domínio espacial, removendo determinada característica ou ainda destacando/suavizando determinado detalhe, somente é possível se conhecer a posição de cada pixel de uma imagem.

Tresholding é um método simples de segmentação de imagens, utilizada na criação de imagens binárias, por níveis de cinza. Vários *pixels* são selecionados e tratados como objetos, atribuindo-se valores para cada de acordo com os níveis de cinza, separando o que é o 'fundo' e o objeto *pixel* de interesse, ou seja, binarizando a imagem (preto e branco). Existem várias maneiras de inferir um valor T (Tresholding) que significará o limiar, ou seja, o valor limite que segmentará a imagem em uma imagem binária: a simples aplicação manual de um valor aleatório e fixo de T; um método iterativo - a aplicação de valores aleatórios de T, percorrendo várias vezes a imagem, segmentando-a e armazenando tais valores de T, que por fim será tirada uma média para um novo valor T melhor.

Detecção de borda é uma técnica de processamento de imagem para determinar pontos de uma imagem digital em que a intensidade luminosa muda repentinamente. Mudanças repentinas em imagens geralmente refletem eventos importantes no cenário, como a

descontinuação da profundidade (transição entre o objeto e o fundo), descontinuação da orientação da superfície, mudança das propriedades do material ou variações na iluminação da cena.

$$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Transformações de imagens são possíveis como rotações, ampliar, reduzir, mover imagem**. Sharpening** é um filtro que aumenta a clareza dos detalhes e das bordas da imagem e o **blurring** é o filtro que borra os detalhes e as bordas da imagem.

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Sharpening

Blurring

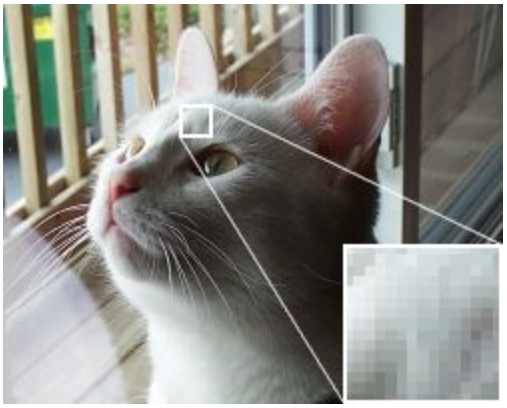
Formato de imagem PPM

O formato de imagem PPM (NetPBM) armazena os valores de cada pixel em um arquivo ASCII. Cada pixel é composto pela combinação das cores vermelho, verde e azul, sempre nessa ordem. Para entender como funciona essa combinação, você pode acessar http://www.rapidtables.com/web/color/RGB_Color.htm e variar os campos R (Red), G (Green) e B (Blue). A cor preta, por exemplo, é representada por R = 0, G = 0 e B = 0. Já o amarelo é representado por R = 255, G = 255 e B = 0. Antes dos valores dos pixels em si, uma imagem PPM contém um pequeno cabeçalho, onde a função desse cabeçalho é identificar que se trata de uma imagem PPM, as dimensões da imagem e o valor máximo de cada cor. A imagem a seguir (que foi ampliada para ser melhor exibida), possui dimensões de 3x2.

A respectiva representação dessa imagem no formato PPM se dá como a seguir

```
P3
3 2
255
255 0 0
0 255 0
0 0 255
255 255 0
255 255 255
0 0 0
```

A primeira linha contém P3, que identifica que se trata de uma imagem PPM. Em seguida, vem os valores 3 e 2, separados por espaço, indicando que a imagem tem 3 colunas e 2 linhas. Na terceira linha está o valor 255. Esse valor é usado para indicar, de certa forma, a qualidade da imagem. Valores baixos representam imagens com pouca qualidade e valores maiores representam maior qualidade. A primeira figura abaixo usa o valor padrão de qualidade 255, enquanto que a segunda figura usa como valor 16. Compare a diferença nas qualidades entre as figuras abaixo para notar a diferença. As demais linhas se referem aos valores RGB dos pixels da imagem. **Neste trabalho você pode sempre deixar o valor fixo em 255.**



Alto valor de qualidade

Baixo valor de qualidade

Descrição do projeto

O projeto de Processamento de Imagem deve ser desenvolvido em linguagem C, a ser executado, em sua versão mais simples, através de linha de comando (entrada e saída em um console/terminal). Além dos filtros básicos para o processamento de imagens, o projeto desenvolvido deve implementar os seguintes requisitos funcionais:

- Será desenvolvido um programa que lê um arquivo PPM e realiza uma ou mais operações de transformação da imagem digitadas pelo usuário.
- Os comandos possíveis são:
  - ‘thr’: binarização da imagem usando thresholding
  - ‘blu’: executa blurring
  - ‘sha’: executa sharpening
  - 'rot': rotação da imagem, dado o ângulo
  - 'amp’: ampliar a imagem, dado o zoom.
  - ‘red’: reduzir a imagem, dado o zoom.
- O término do programa ocorre em uma das seguintes situações:
  - a segmentação foi feita completamente
  - aconteceu um erro na segmentação da imagem
  - o usuário interrompeu o programa.

O projeto deve atender também os seguintes critérios de programação:

1. **Uso de arranjos / matrizes;**
2. **Uso de registros (`struct`);**
3. **Uso de recursão;**
4. **Definição de novos tipos de dados através de `typedef`;**
5. **Modularização do programa em diferentes arquivos (uso de diferentes arquivos .c e .h, cada um com sua funcionalidade);**
6. **Definição de um padrão de indentação do código fonte e de nomenclatura das sub-rotinas e variáveis, e a adequação a este padrão;**
7. **Documentação adequada do código-fonte.**

Observações:

- O projeto deve ser desenvolvido individualmente ou em duplas (grupos de dois alunos). Não serão permitidos grupos com três ou mais alunos.
- Para facilitar o acompanhamento do projeto, cada dupla deve estar associada a uma única turma de PTP. Ou seja, não serão permitidas duplas formadas por alunos matriculados em turmas de PTP diferentes.
- Cada dupla deve desenvolver sua solução de forma independente das demais. Soluções idênticas serão consideradas plágios e, portanto, sanções serão devidamente aplicadas em todas as duplas com soluções similares.
- Códigos e algoritmos podem ser utilizados da web desde que devidamente referenciados. Caso sejam encontrados trechos de código na web equivalentes aos apresentados pelo grupo sem a devida citação, o código será igualmente considerado plágio e sanções serão aplicadas ao grupo. Vale salientar que a avaliação será realizada unicamente sobre o código produzido pela dupla. Códigos retirados da web, apesar de permitidos com a devida citação, serão desconsiderados dos critérios de pontuação.

Avaliação no componente curricular PTP (IMD0012.1)

Esta avaliação compreende a execução e desenvolvimento do projeto em 4 semanas. A cada semana, o grupo deve apresentar uma etapa do projeto desenvolvido, seguindo o calendário abaixo:

ETAPA 1 - semana de 03 e 05 de novembro de 2015

1. Tipos de dados necessários (`typedef`, `structs` e `enums`);
2. Modularização do programa (quais os arquivos .c e .h)
3. Leitura de imagens de arquivo;
4. Subrotina para binarizacão da imagem.

ETAPA 2 - semana de 10 e 12 de novembro de 2015

5. Transformacões com as imagens: ampliar, reduzir, rotacionar.

ETAPA 3 - semana de 17 e 19 de novembro de 2015

6. Subrotinas de *blurring* e *sharpening*;
7. Detecção de bordas;

ETAPA 4 - semana de 24 e 26 de novembro de 2015

8. Implementação de elementos extras, definidos pelo próprio grupo.

Sobre as duplas

Os alunos têm ATÉ o dia **26 de outubro de 2015** para comunicar aos professores de ITP e PTP se farão o trabalho em dupla (e a composição da mesma) ou se farão individualmente.

Critérios de pontuação

O desenvolvimento do projeto aqui descrito vale **100%** da nota da terceira unidade de PTP.

A pontuação da avaliação seguirá os critérios e distribuição abaixo:

- **Atendimento dos requisitos funcionais: 50%**
  a imagem está representada corretamente? os filtros estão implementados corretamente? os filtros podem ser aplicados a qualquer imagem? etc.
- **Uso dos recursos da linguagem C: 20%**
  a dupla demonstrou saber usar de forma adequada os recursos da linguagem C (arranjos, structs, typedefs, recursividade, etc)?
- **Organização do código e documentação: 10%**
  o código está documentado? a identação e uso de { } seguem um padrão **(identacão)**? o programa está devidamente modularizado em diferentes arquivos?
- **Funcionalidades extras: 20%**
  as funcionalidades extras desenvolvidas pela dupla foram suficientemente complexas?

A pontuação a ser dada pelas funcionalidades extras não é definida *a priori*. Cada caso será avaliado em função da complexidade envolvida. Itens extras de baixa complexidade serão desconsiderados na pontuação.

Entrega do projeto

O projeto deve ser submetido pelo SIGAA até a data **26 de novembro de 2015** em um **arquivo comprimido (.zip)** contendo os arquivos fontes do projeto (.c e .h) e um arquivo README.TXT. Este arquivo deve ter as seguintes informações:

- O que foi feito (o básico e as funcionalidades extras implementadas);
- O que não foi feito (caso não tenha sido possível implementar alguma funcionalidade);
- O que seria feito diferentemente (quando aprendemos à medida que vamos implementando, por vezes vemos que podíamos ter implementado algo de forma diferente. Assim, esse tópico é para vocês refletirem sobre o que vocês aprenderam durante o processo que, se fossem fazer o mesmo projeto novamente, fariam diferente);
- Como compilar o projeto, deixando explícito se foi utilizada alguma biblioteca externa que precise ser instalada, que versão e quais parâmetros extras são necessários para o compilador.
- Em caso de duplas:
  - Identificação dos autores;
  - Contribuição de cada integrante no desenvolvimento do projeto (quem fez o quê).

Recomendações diversas:

1. Solução de back-up: não será tolerada a desculpa de que um disco rígido falhou nas avaliações. Assim, é importante manter várias cópias do código-fonte desenvolvido ou usar um sistema de backup automático como o Dropbox, Google Drive, Box ou similares. Uma solução melhor ainda é fazer uso de um sistema de controle de versões como git, e um repositório externo como Github ou Bitbucket.
2. Especificar precisamente a interface e o comportamento de cada sub-rotina. Usar esta informação para guiar o desenvolvimento e documentar o código desenvolvido.

Extras

- Interface gráfica.
- Implementacão de filtros mais avancados: Fourier, Gauss, deteccão de objetos específicos na imagem, etc.
- Trabalhar com desenhos de linhas e curvas bezier.
- Desenho de fontes. A ideia é usar curvas de bezier para desenhar fontes como TTF, por exemplo. Um exemplo de chamada seria plot_string(x, y, "string");
- Compressão de imagens: usar um algoritmo como o RLE para criar imagens compactadas simples. (https://en.wikipedia.org/wiki/Run-length_encoding)
- Desenho de gráficos de funções. Ler uma função e plotar a mesma. Seria usado structs para representar as funções. Uso de recursão para implementar um parser simples.