Project 2 Writeup (Histogram Match)

Instructions

- Describe any interesting decisions you made to write your algorithm.
- Show and discuss the results of your algorithm.
- Feel free to include code snippets, images, and equations.
- Use as many pages as you need, but err... on the short side. If you feel you only need to write a short amount to meet the brief, then pizza.
- **Please make this document anonymous.**

Histogram Matching

O presente trabalho foca na implementação do *simulated annealing* para a realização da aproximação da distribuição da intensidade dos pixels de duas imagens. Dessa forma, foram utilizados conceitos relacionados à modificação e aplicação do estado dos coeficientes de um polinômio de segundo grau para gerar novas intensidades e diminuir a distância entre a imagem original e a final.

Detalhes de Implementação

Esta seção apresenta a explicação dos detalhes de implementação dos principais módulos do programa.

O Código 1 apresenta o cálculo da distância entre dois histogramas.

- *Linhas 2 a 5*: normalização dos histogramas.
- *Linhas 6 a 8*: cálculo da diferença entre os histogramas.

Código 1.

```
1 function summation = D(h1, h2)
2   pixels1 = sum(h1);
3   pixels2 = sum(h2);
4   h1 = h1 ./ pixels1;
5   h2 = h2 ./ pixels2;
6   res = h1 .- h2;
7   res = res .^ 2;
8   summation = sum(res);
9 endfunction
```

O Código 2 apresenta a solução do sistema linear e retorna os três coeficientes referentes ao polinômio de segundo grau.

Código 2.

```
function res = LS(p1, p2, p3)
  a = [p1(1)^2, p1(1), 1; p2(1)^2, p2(1), 1; p3(1)^2, p3
     (1), 1];
  b = [p1(2); p2(2); p3(2)];
  res = a\b;
endfunction
```

O Código 3, por sua vez, escolhe um ponto de forma aleatória e realiza uma das operações (adição ou subtração) no ponto em questão, considerando os limites de 0 a 255.

Código 3.

```
function pontos = vizinhoAleatorio(s)
  pontos = s;
  flag = 1;
  while (flag == 1)
    column = randi([1, 2]);
    row = randi([1, 3]);
    op = randi([1,2]);
    if (op == 1)
      if (pontos(row, column) + 1 < 256)
        pontos(row, column) = pontos(row, column) + 1;
        flag = 0;
      endif
    else
      if (pontos(row, column) - 1 > -1)
        pontos(row, column) = pontos(row, column) - 1;
        flag = 0;
      endif
    endif
  endwhile
endfunction
```

O trecho referente ao Código 4 gera um novo histograma a partir da tabela de correlação encontrada no Código 5.

Código 4.

```
function mapped = map(correlacao, Ha)
  mapped = zeros(256, 1);
  for i = 1:256
      value = correlacao(i);
      mapped(i) = Ha(value+1);
```

```
  endfor
endfunction
```

O Código 5 gera uma tabela de correlação referente aos novos coeficientes obtidos com a função *LS* (*Linha 3*), respeitando os limites da intensidade de 0 a 255.

Código 5.

```
function tabela = calculaTabelaCorrecao(Sn)
  tabela = zeros(256, 1);
  ls = LS([Sn(1,1); Sn(1,2)], [Sn(2,1); Sn(2,2)], [Sn
     (3,1); Sn(3,2)]);
  for i = 1:256
    resp = (ls(1) * ((i-1) ^ 2)) + (ls(2) * (i-1)) + (ls
       (3));
    if (resp > 255)
      tabela(i) = 255;
    else
      if (resp < 0)
        tabela(i) = 0;
      else
        tabela(i) = resp;
      endif
    endif
  endfor
  tabela = floor(tabela);
endfunction
```

Por último, o Código 6 invoca as funções anteriores para gerar e avaliar a qualidade dos novos histogramas a partir do anterior, definindo a probabilidade de aceitação p e q. Tal função retorna a tabela de correlação final a ser aplicada diretamente nos canais da imagem original.

Código 6.

```
function L = AS(s0, t0, e, k, Ha, Hb)
  i = 0;
  S = s0;
  T = t0;
  accepted = 0;
  L_S = calculaTabelaCorrecao(S);
  map_s = map(L_S, Ha);
  while (T > e)
    if (i == k)
      T = atualizaTemperatura(T);
      i = 0;
    endif
```

```
13      Sn = vizinhoAleatorio(S);
14      L_Sn = calculaTabelaCorrecao(Sn);
15      map_sn = map(L_Sn, Ha);
16      E_S = D(map_s, Hb);
17      E_Sn = D(map_sn, Hb);
18      delta = E_Sn - E_S;
19      p = 0;
20      if (delta < 0)
21        p = 1.0;
22      else
23        p = exp(-delta / T);
24      endif
25      q = geraProbabilidade();
26      if q < p
27        map_s = map_sn;
28        S = Sn;
29        accepted = accepted + 1;
30      endif
31      i = i + 1;
32    endwhile
33    accepted
34    L = calculaTabelaCorrecao(S);
35  endfunction
```

Resultados

Resultados não foram obtidos, considerando que o comportamento do código ao aplicar tentativas de modificações nas variáveis e nas funções geradoras de tabelas de correlação, cálculo do sistema linear, mapeamento da tabela de correlação e distância do histograma, não foram relevantes para a obtenção de um resultado aceitável.