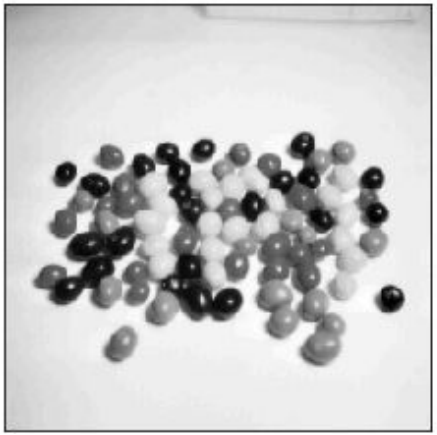
Trabalho II - Processamento Gráfico

Programa de Pós graduação em Modelagem Computacional

Gisele Goulart
Marvelúcia Almeida
Professor Rafael Bonfim - 2020/2

Banco de imagens

- O banco de dados **USC-SIPI** é um conjunto de imagens para apoiar pesquisas em processamento e análise de imagens e visão computacional.
- As imagens coloridas são de 24 bits e as na escala de cinza possuem 8 bits. Com um total de 44 imagens (6 imagens utilizadas).



Questão 1

TRANSFORMADAS DISCRETAS DE IMAGENS

Transformadas Discretas de Imagens

- Transformada Discreta de Fourier (DFT)

$$X[k,l] = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} [\frac{1}{\sqrt{M}} \sum_{m=0}^{M-1} x[m,n] e^{-j2\pi \frac{mk}{M}}] e^{-j2\pi \frac{nl}{N}}$$

- Transformada Discreta de Cosseno (DCT)

$$X_{k_1,k_2} = \sum_{n_1=0}^{N_1-1} \left(\sum_{n_2=0}^{N_2-1} x_{n_1,n_2} \cos\left[\frac{\pi}{N_2} \left(n_2 + \frac{1}{2} \right) k_2 \right] \right) \cos\left[\frac{\pi}{N_1} \left(n_1 + \frac{1}{2} \right) k_1 \right]$$

- Transformada Discreta de Seno (DST)

$$F\{u,v\} = \frac{2c(u)\,c(v)}{N} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} f(m,n)\, sin\left(\frac{2m+1}{2N} u\pi \right) sin\left(\frac{2n+1}{N} v\pi \right)$$

Transformadas Discretas de Imagens

Objetivo:

- Aplicar as transformadas nas imagens com e sem ruído;
- Para cada imagem calcular:
  - Módulo da amplitude das intensidades da imagem transformada;
  - Logaritmo do módulo da amplitude das intensidades da imagem transformada;
  - Porcentagem de intensidades nulas da imagem transformada em ambos cenários investigados;
  - Ressaltar a geração da imagem ruidosa e os parâmetros adotados para aplicação das transformadas.

Implementação DFT, DCT, DST

- Linguagem de programação: Python.
- Bibliotecas utilizadas: *glob, PIL(Image), numpy, matplotlib, scipy*.
  - ➔ Carregamento das imagens em preto e branco (*PIL*);
  - ➔ Após isso, as imagens foram convertidas em uma matriz *float* de intensidades (*numpy*);
  - ➔ Geração da **matriz de ruídos gaussiana** (função *np.random.normal,* **média:128 e desvio padrão:20**) (*numpy*);
  - ➔ Adição da matriz de ruído à matriz da imagem;
  - ➔ Emprego da relação que garante com que os valores das intensidades da matriz com ruído possua intensidades entre 0 e 255 (8 bits);

$$f_m = f - \min(f)$$

$$f_s = K \left[\frac{f_m}{\max(f_m)} \right]$$

Implementação DFT, DCT, DST

- ➔ Aplicação da **transformada às imagens** com e sem ruído:

  DCT e **DST** (*scipy*)

  DFT (*numpy*)

- ➔ Cálculo do **módulo** da amplitude das intensidades da imagem transformada (*numpy*);
- ➔ Cálculo do **logaritmo** do módulo da amplitude das intensidades da imagem transformada (*numpy*);
- ➔ Cálculo do valor **máximo**, **mínimo** e **médio do módulo** das amplitudes (*numpy*);
- ➔ Cálculo da **porcentagem das intensidades nulas** da imagem transformada (*numpy*);
- ➔ Reconstrução da imagem (*PIL*, *Image*);
- ➔ Geração dos resultados gráficos (*matplotlib*).

Resultados DFT

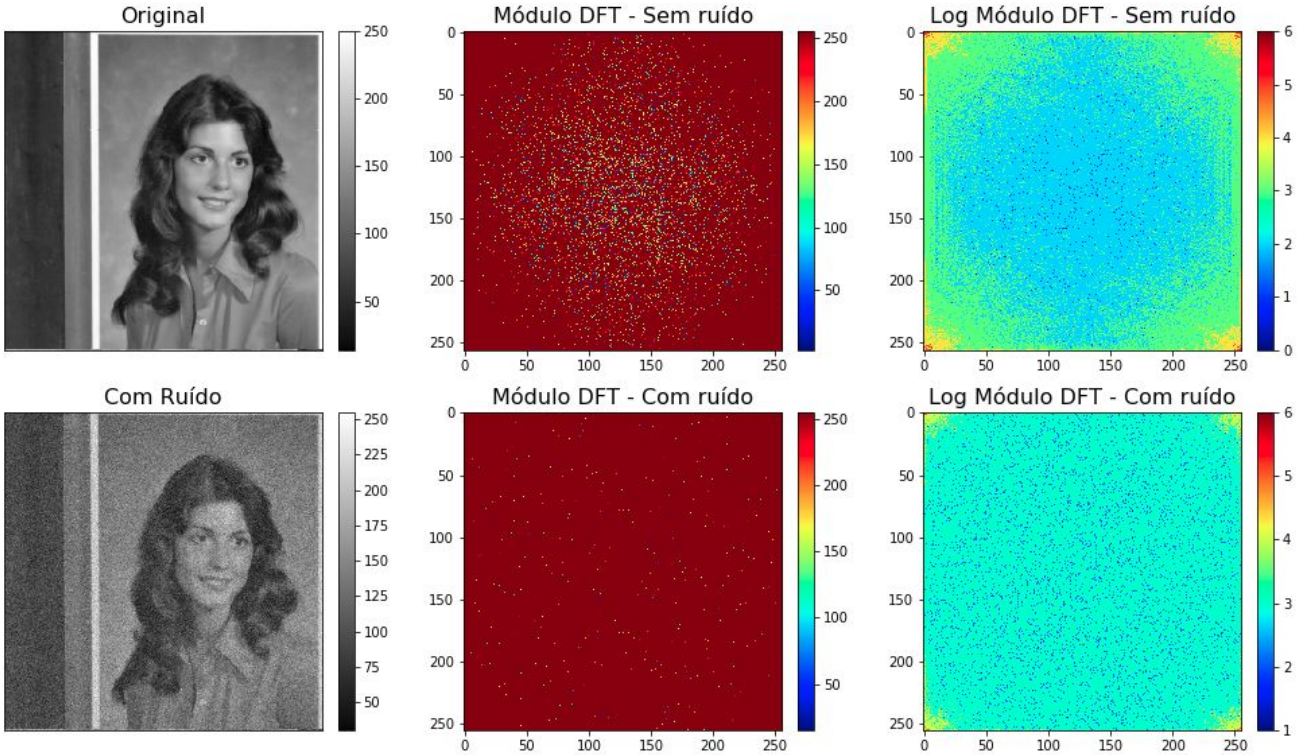


Figura 1

Resultados DFT

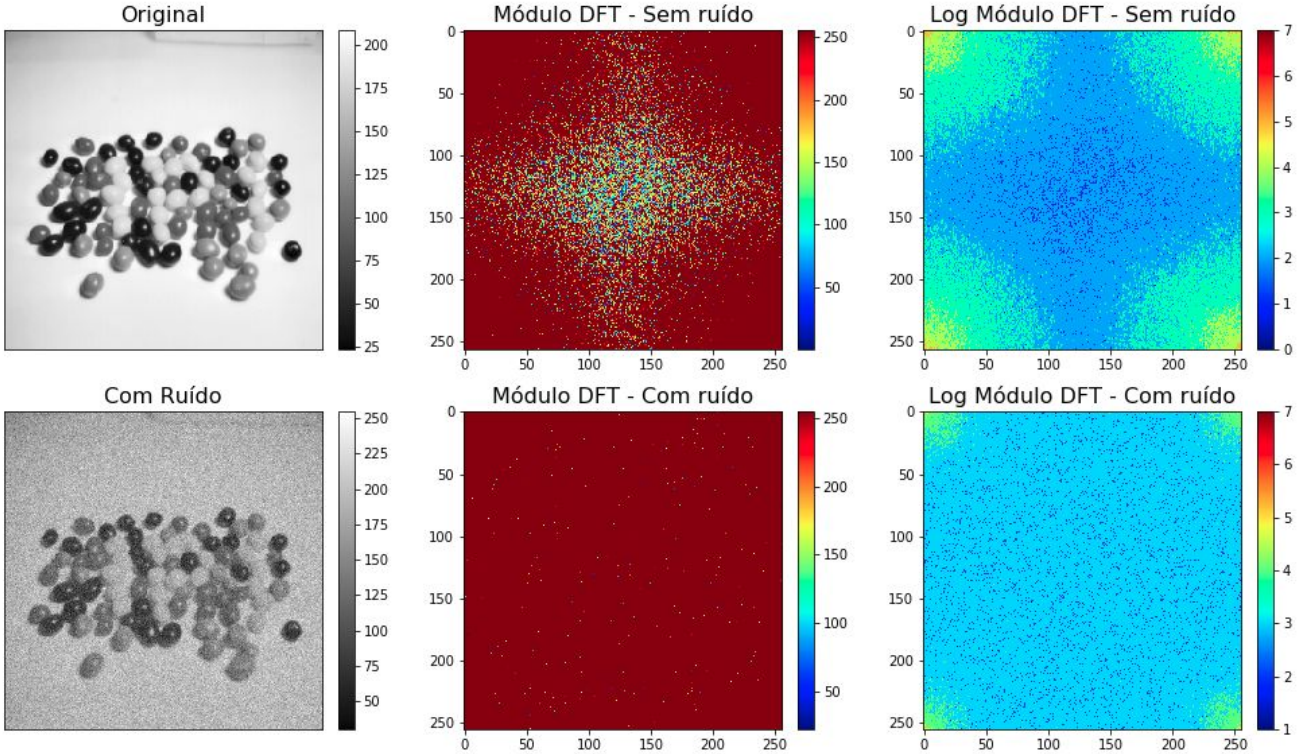


Figura 2

Resultados DFT

DFT - Figura 1													
Sem ruído							Com ruído						
Módulo das amplitudes			Log do módulo				Módulo das amplitudes			Log do módulo			
MAX	MIN	MED	MAX	MIN	MED	ZEROS	MAX	MIN	MED	MAX	MIN	MED	ZEROS
7.286.171,0	3,23	2.598,14	6,86	0,51	2,98	0%	8.229.899,0	15,09	3.855,76	6,91	1,18	3,44	0%

DFT - Figura 2													
Sem ruído							Com ruído						
Módulo das amplitudes			Log do módulo				Módulo das amplitudes			Log do módulo			
MAX	MIN	MED	MAX	MIN	MED	ZEROS	MAX	MIN	MED	MAX	MIN	MED	ZEROS
10.959.374,0	1,25	2.667,47	7,04	0,09	2,87	0%	11.504.537,0	22,67	4.389,54	7,06	1,35	3,49	0%

Resultados DCT

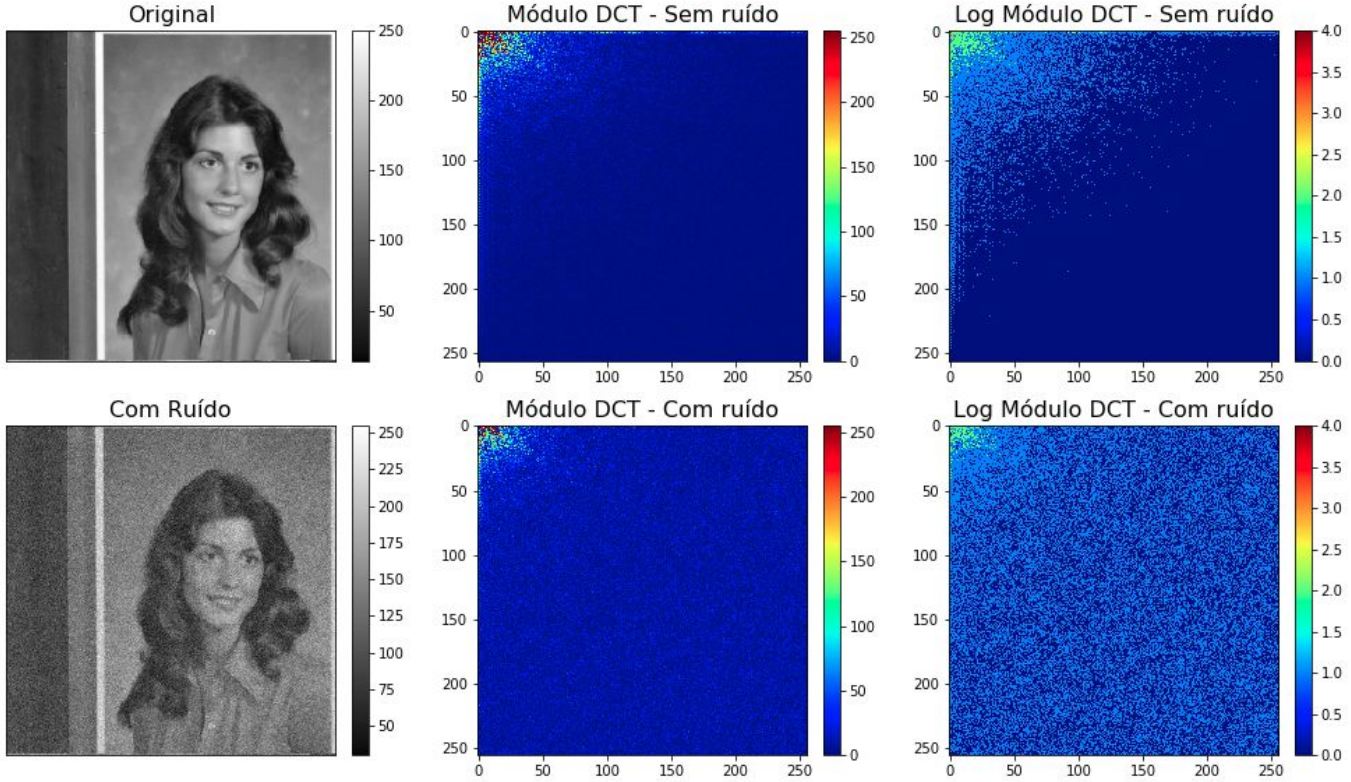


Figura 1

Resultados DCT

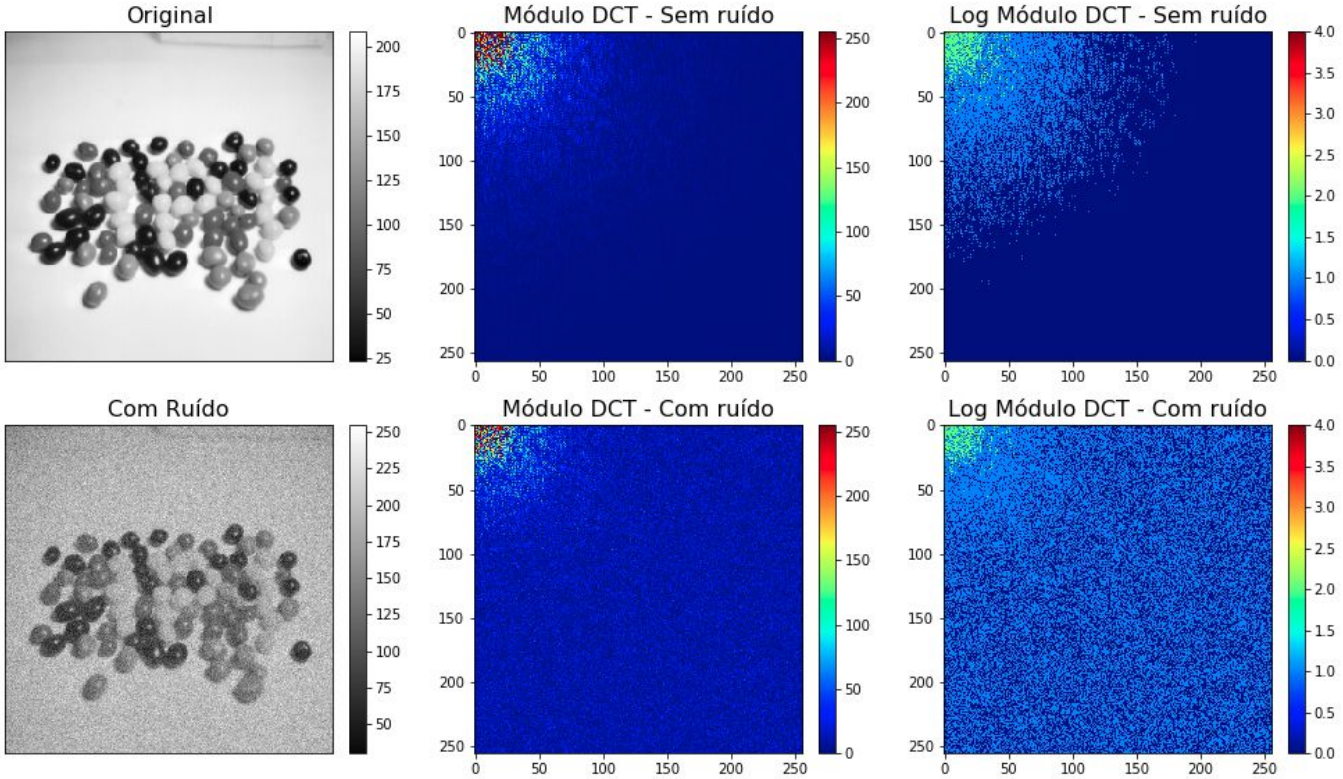


Figura 2

Resultados DCT

DCT - Figura 1													
Sem ruído							Com ruído						
Módulo das amplitudes			Log do módulo				Módulo das amplitudes			Log do módulo			
MAX	MIN	MED	MAX	MIN	MED	ZEROS	MAX	MIN	MED	MAX	MIN	MED	ZEROS
28.461,60	0	8,52	4,45	-3,62	0,39	22,10%	32.148,04	$4{,}70\times10^{-5}$	13,35	4,51	-4,33	0,87	5,93%

DCT - Figura 2													
Sem ruído							Com ruído						
Módulo das amplitudes			Log do módulo				Módulo das amplitudes			Log do módulo			
MAX	MIN	MED	MAX	MIN	MED	ZEROS	MAX	MIN	MED	MAX	MIN	MED	ZEROS
42.810,05	$7{,}67\times10^{-5}$	9,40	4,63	-4,11	0,31	30,40%	44.939,60	$7{,}79\times10^{-07}$	15,49	4,65	-6,11	0,93	5,14%

Resultados DST

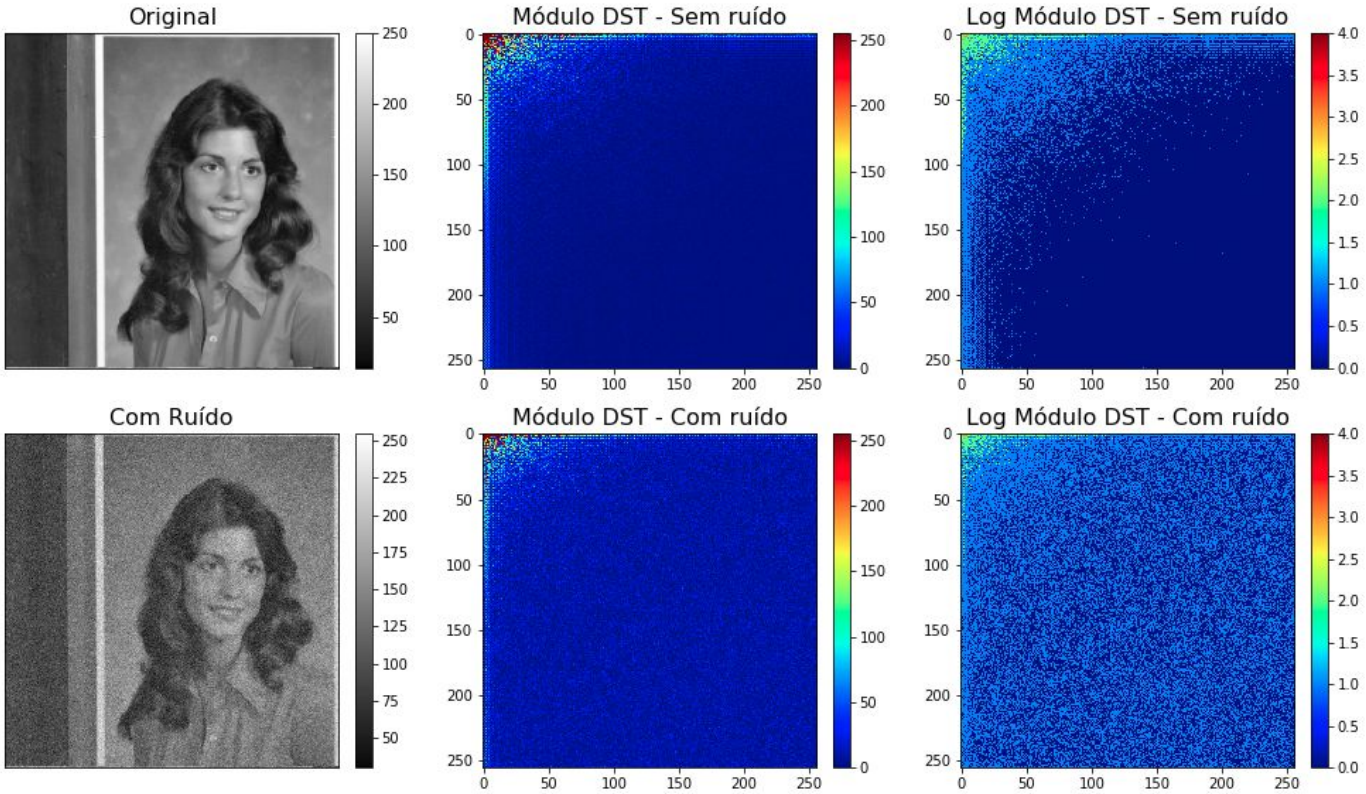


Figura 1

Resultados DST

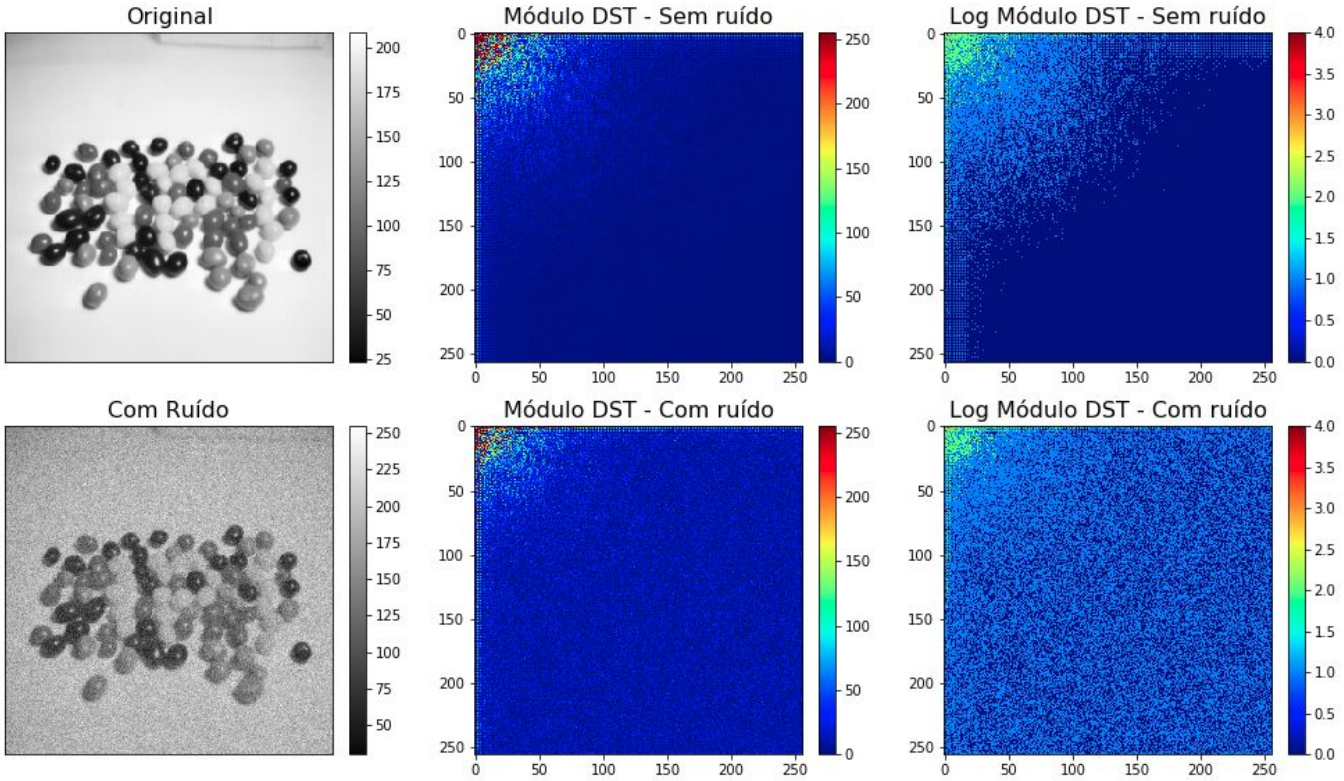


Figura 2

Resultados DST

DST - Figura 1													
Sem ruído							Com ruído						
Módulo das amplitudes			Log do módulo				Módulo das amplitudes			Log do módulo			
MAX	MIN	MED	MAX	MIN	MED	ZEROS	MAX	MIN	MED	MAX	MIN	MED	ZEROS
11.416,29	$1{,}72\times10^{-5}$	11,38	4,06	-4,76	0,48	18,67%	12.953,64	0	16,04	4,11	-3,48	0,91	5,49%

DST - Figura 2													
Sem ruído							Com ruído						
Módulo das amplitudes			Log do módulo				Módulo das amplitudes			Log do módulo			
MAX	MIN	MED	MAX	MIN	MED	ZEROS	MAX	MIN	MED	MAX	MIN	MED	ZEROS
15.973,98	$2{,}89\times10^{-6}$	12,41	4,20	-5,54	0,40	26%	17.247,25	$1{,}64\times10^{-5}$	18,60	4,24	-4,78	0,95	5%

Questão 2

FLUXO ÓTICO EM SEQUÊNCIA TEMPORAL DE IMAGENS

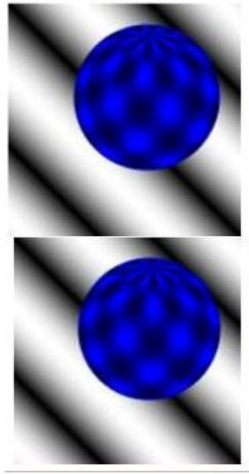
Fluxo ótico em imagens sequenciais

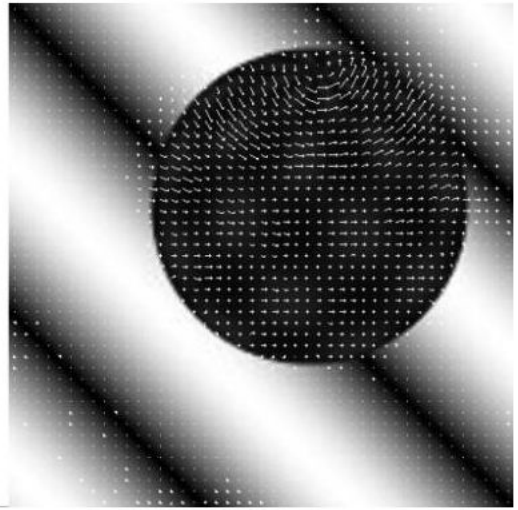
Objetivo:

- Aplicar os métodos Horn e Schunck (HS) e Lucas e Kanade (LK) em uma sequência temporal de imagens;
- Identificar os parâmetros adotados em cada método;
- Mostrar o campo vetorial (fluxo ótico) obtido através dos métodos aplicados;
- Apresentar a derivada material;

Fluxo Ótico

- É a **distribuição da velocidade** aparente do movimento dos **padrões de intensidade** no plano da imagem (Horn e Schunck, 1981).
- Apresenta informações sobre o **arranjo espacial** dos objetos e a **taxa de variação** desse arranjo.
- É computado em um **pixel** da imagem considerando os **seus vizinhos** e **restrições adicionais**.





Fluxo Ótico

- Métodos **diferenciais** utilizados para determinação do fluxo ótico: **Horn e Schunck e Lucas e Kanade.**
- Esses dois métodos consideram que a **intensidade** entre uma imagem e outra é aproximadamente **constante** em um **intervalo de tempo pequeno.**

$$I(x, y, t) = I(x + dx, y + dy, t + dt)$$

- Assim, a derivada material do fluxo ótico resultará na **Equação de Restrição** do Fluxo Ótico:

$$\frac{D\mathrm{I}}{Dt} = \nabla \mathrm{I} \cdot \vec{\mathbf{v}} + \mathrm{I}_t$$

$$\nabla \mathrm{I} \cdot \vec{\mathbf{v}} + \mathrm{I}_t = 0 \rightarrow \mathrm{I}_x u + \mathrm{I}_y v + \mathrm{I}_t = 0$$

Método Horn e Schunck (HS)

- Utiliza uma forma de regularização aplicada à equação anterior chamada de **restrição de suavização**, ou seja, considera que o **fluxo de vetores** de uma imagem para outra **varia de forma suave**;
  - **Restrição de iluminação**: iluminação constante nas duas imagens.
  - **Restrição de suavização**: pontos vizinhos apresentam velocidades semelhantes.
- As derivadas parciais são estimadas pela média das quatro primeiras regiões adjacentes (vizinhas) da imagem;
- É um **método iterativo** que minimiza a equação da restrição de suavização pelo **método dos mínimos quadrados**, obtendo o campo de velocidades (campo vetorial/**fluxo ótico**) para cada pixel da imagem;

Implementação do Horn e Schunck (HS)

- Linguagem de programação: Python.
- Bibliotecas utilizadas: *OpenCV, numpy, matplotlib.*
  - ➔ Carregamento das imagens (512x512) (*OpenCV, cv2.imread*);
  - ➔ Chama o método HS, passando seus **parâmetros**: as duas imagens sequenciais e o **fator de suavização** (alpha);
  - ➔ Cálculo das **derivadas parciais** (Ix, Iy, It);
  - ➔ Cálculo das **componentes** do fluxo ótico (u, v);
  - ➔ Geração do **campo de fluxo ótico** (*CV2, cv2.arrowedLine*);
  - ➔ Cálculo da **derivada material** da imagem (DI/Dt)**;**
  - ➔ Geração dos resultados gráficos (*matplotlib*);
  - ➔ Cálculo da norma de Frobenius da matriz da derivada material (*numpy*);

$$\|A\|_{\mathrm{F}} = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2}$$

Horn e Schunck (HS) - Testes Alpha

f1

f2

Alpha	Norma da Derivada
4	3273.0550
1	1987.2742
0.6	1532.5471
0.1	871.1743

Horn e Schunck (HS) - Testes Alpha

f2

f3

Alpha	Norma da Derivada
4	3473.6680
1	2059.1113
0.6	1545.2078
0.1	752.0276

Horn e Schunck (HS) - Testes Alpha

f3

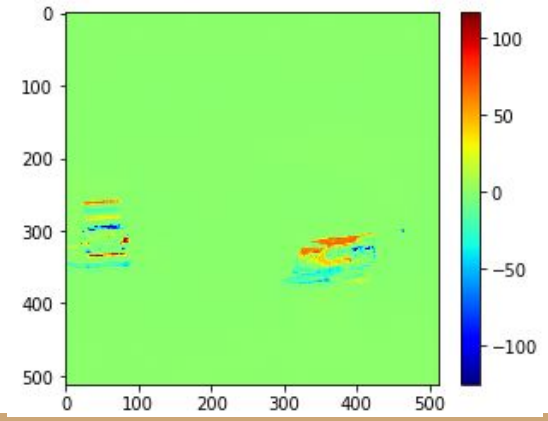
f4

Alpha	Norma da Derivada
4	3954.6210
1	2487.3780
0.6	1922.2987
0.1	1071.9418

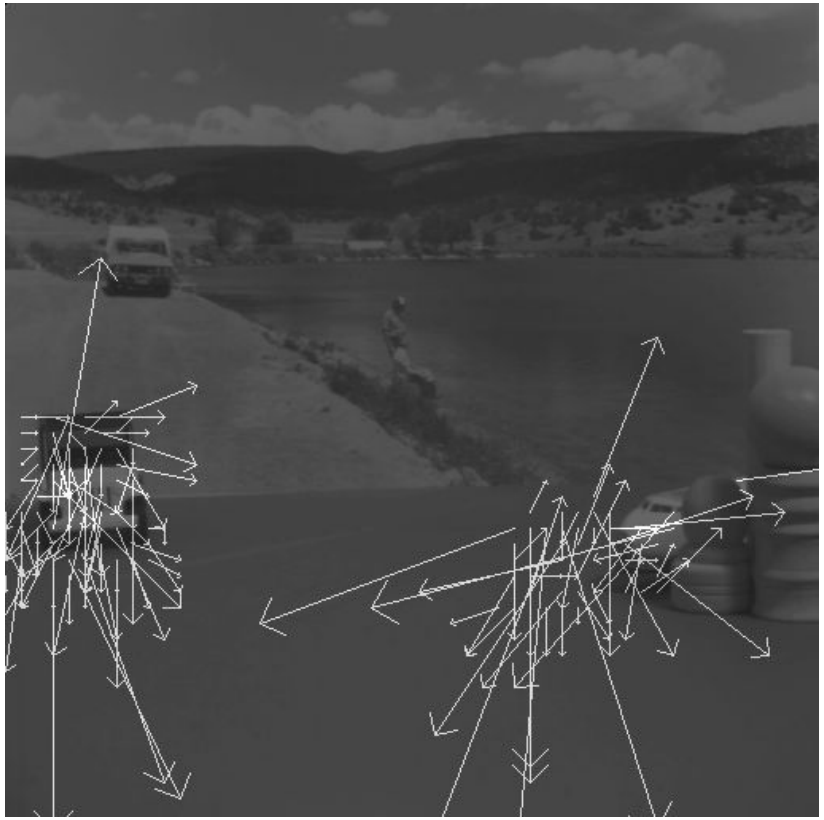
Resultados Horn e Schunck (HS)

f1

f2



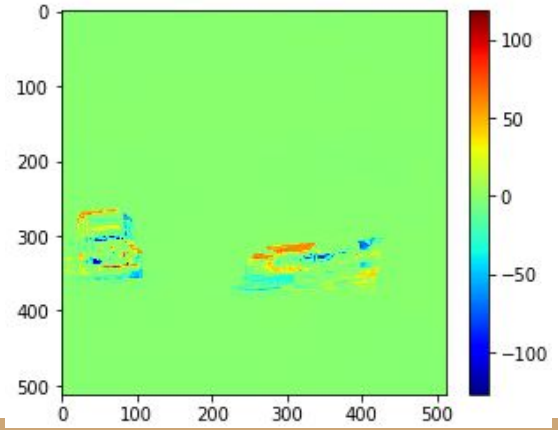
Alpha= 4



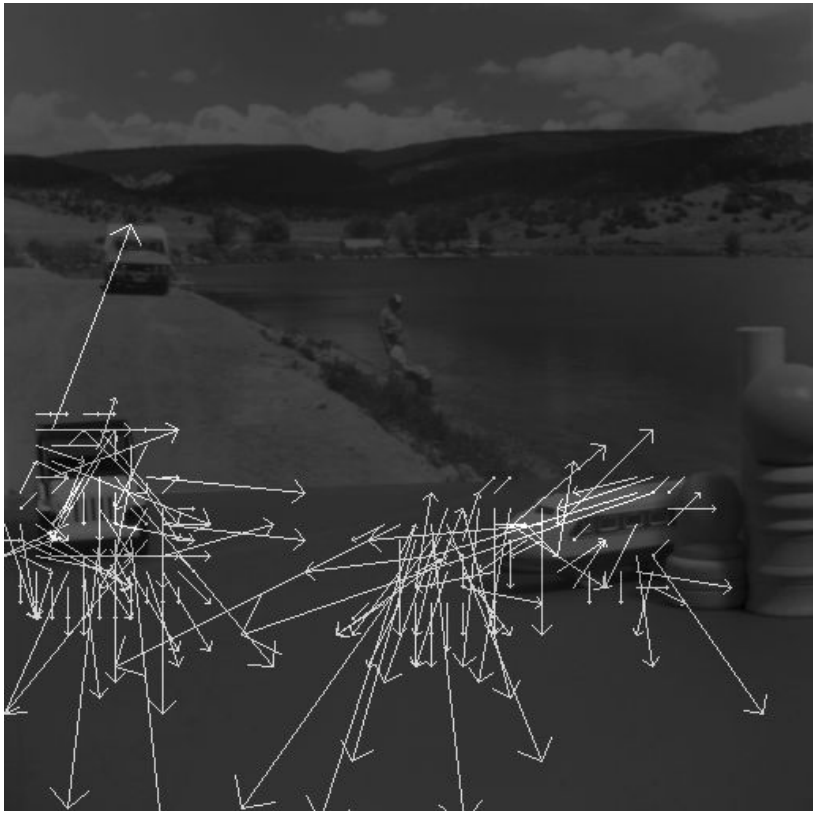
Resultados Horn e Schunck (HS)

f2

f3



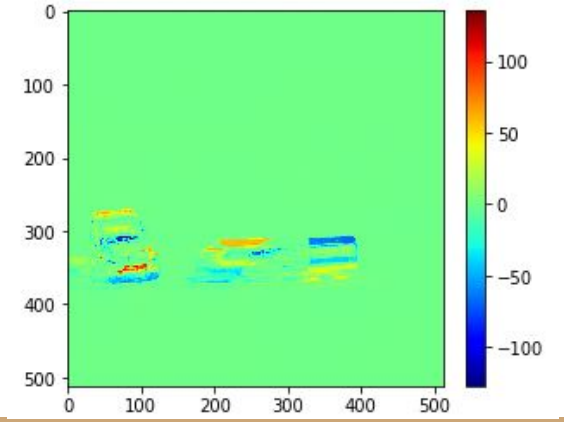
Alpha= 4



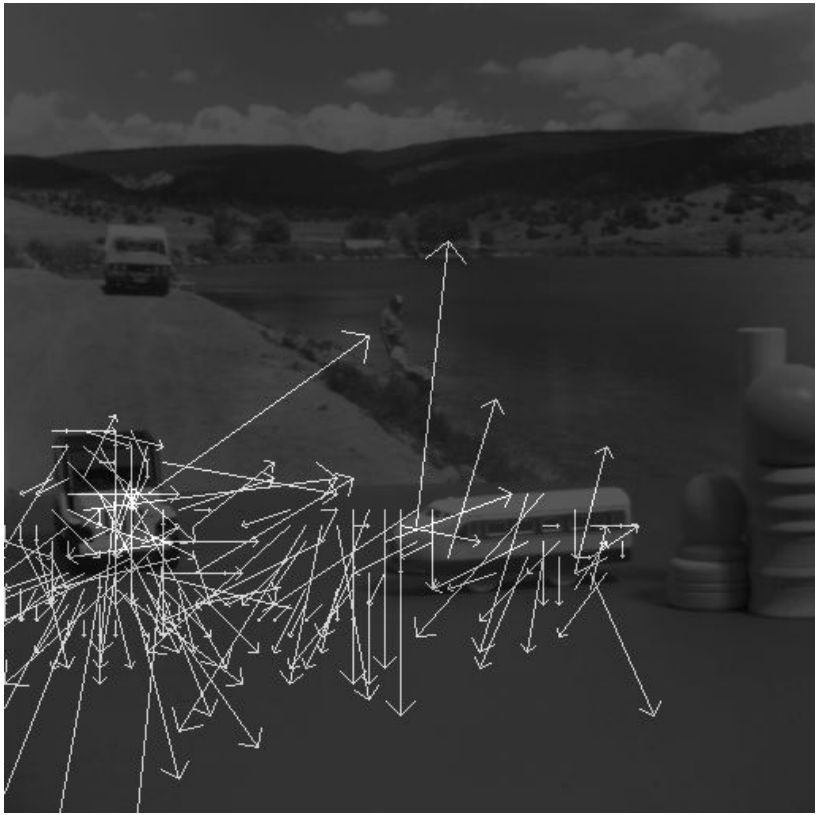
Resultados Horn e Schunck (HS)

f3

f4



Alpha= 4



Método Lucas e Kanade (LK)

- É um **método não-iterativo**;
- Assume um **fluxo ótico constante local** em pequenas **janelas** de tamanhos (nxn) e com n>1, **centradas em um único pixel**;
- Resolve as equações do sistema formado a partir do problema de minimização da restrição do fluxo óptico através de um **ajuste ponderado de mínimos quadrados**;
- Apresenta robustez contra ruídos, mas a malha de pontos do campo de velocidade não é tão densa;

Implementação do Lucas e Kanade (LK)

- Linguagem de programação: Python.
- Bibliotecas utilizadas: *OpenCV, numpy, matplotlib.*
  - Carregamento das imagens (512x512) (*OpenCV, cv2.imread*);
  - Chama o método LK, passando seus **parâmetros**: as duas imagens sequenciais e o **tamanho da janela**;
  - Cálculo das **derivadas parciais** (Ix, Iy, It);
  - Cálculo das **componentes** do fluxo ótico (u, v);
  - No qual, a matriz de pesos é filtro gaussiano de tamanho **5x5** e **desvio padrão de 3** (OpenCV,*cv2.GaussianBlur*).
  - Geração do **campo de fluxo ótico** (*CV2, cv2.arrowedLine*);
  - Cálculo da **derivada material** da imagem (DI/Dt);
  - Geração dos resultados gráficos (*matplotlib*);
  - Cálculo da norma de Frobenius da matriz da derivada material (*numpy*);

$$\|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2}$$

Lucas e Kanade (LK) - Testes Janela

f1

f2

Janela	Norma da Derivada
3	104931.2733
5	103219.2757
7	102256.4848
9	101498.8131

Lucas e Kanade (LK) - Testes Janela

f2

f3

Janela	Norma da Derivada
3	55750.6050
5	51155.1222
7	53672.6159
9	53102.8614

Lucas e Kanade (LK) - Testes Janela

f3

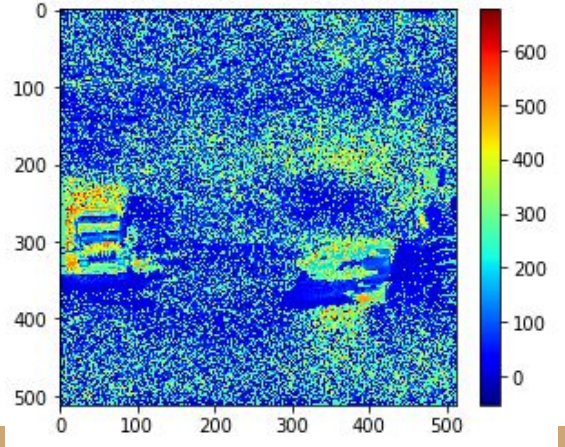
f4

Janela	Norma da Derivada
3	95737.5797
5	94095.5993
7	93181.0213
9	92525.5137

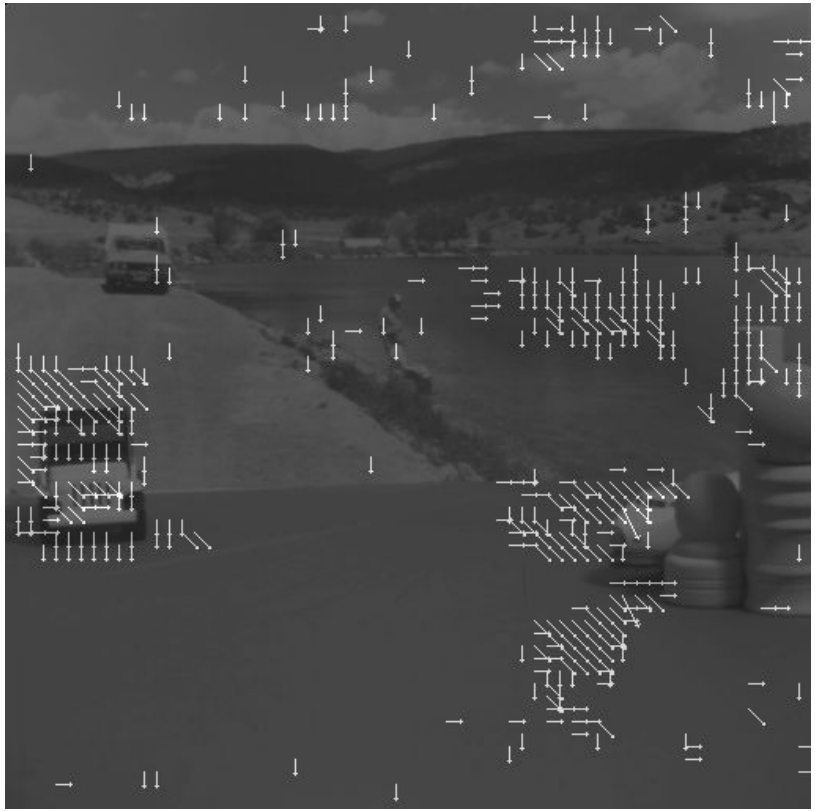
Resultados Lucas e Kanade (LK)

f1

f2



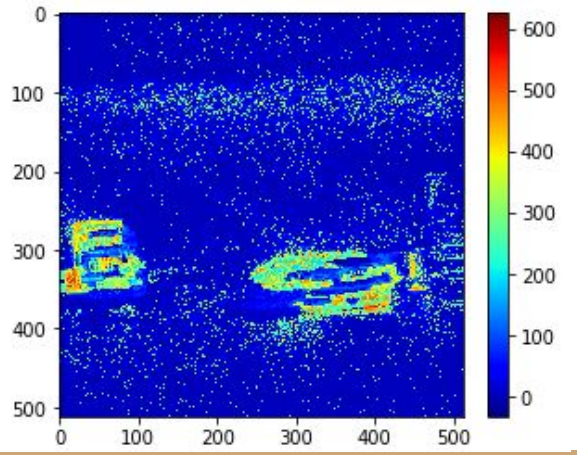
Janela= 5



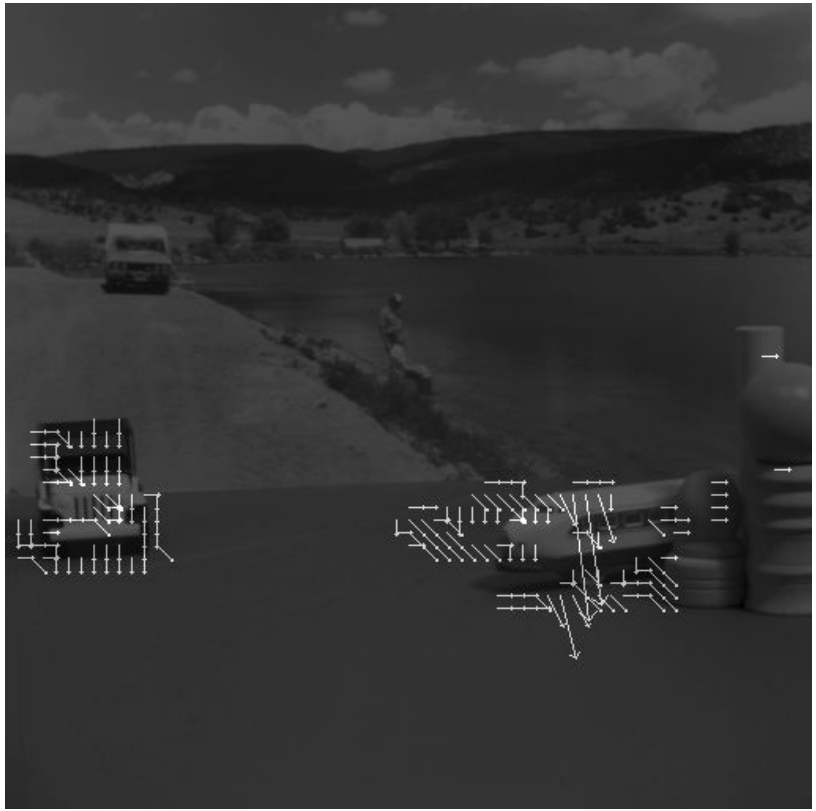
Resultados Lucas e Kanade (LK)

f2

f3



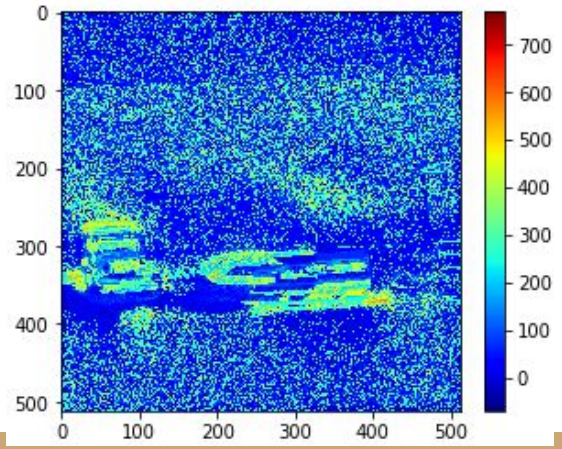
Janela= 5



Resultados Lucas e Kanade (LK)

f3

f4



Janela= 5

Referências

- Banco de imagens:
  - http://sipi.usc.edu/database/database.php?volume=misc
- Bibliotecas Python:
  - https://pillow.readthedocs.io/en/stable/reference/Image.html
  - https://numpy.org/
  - https://matplotlib.org/
  - https://docs.python.org/3/library/glob.html
  - https://docs.scipy.org/doc/scipy/reference/generated/scipy.fftpack.fft2.html#scipy.fftpack.fft2
- Ferraz, T. V. D., Rodrigues, G. F., 2015. Rastreamento labial utilizando fluxo óptico para reconhecimento de fala em imagens de vídeo. Universidade Federal de São João del-Rei.
- Implementações LK e HS:
  - https://github.com/dmarkatia/LucasKanade/blob/master/LK.py
  - https://stackoverflow.com/questions/27904217/horn-schunck-optical-flow-implementation-issue
- GitHub com a implementação: https://github.com/giselegoulart/graphic-processing