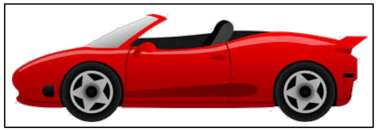
Localização simples de objetos, pirâmides de imagens e interpolação

PROF. CESAR HENRIQUE COMIN

Localização de objetos

Suponha que temos uma imagem de um carro, e queremos encontrar esse carro em uma outra imagem (uma cena).

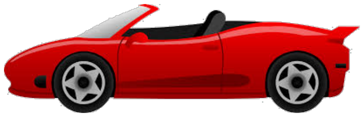


Para encontrar o carro, podemos “deslizar” a imagem do carro ao longo da imagem maior, e calcular a diferença quadrática entre os pixels da imagem do carro e a imagem maior.

Localização de objetos

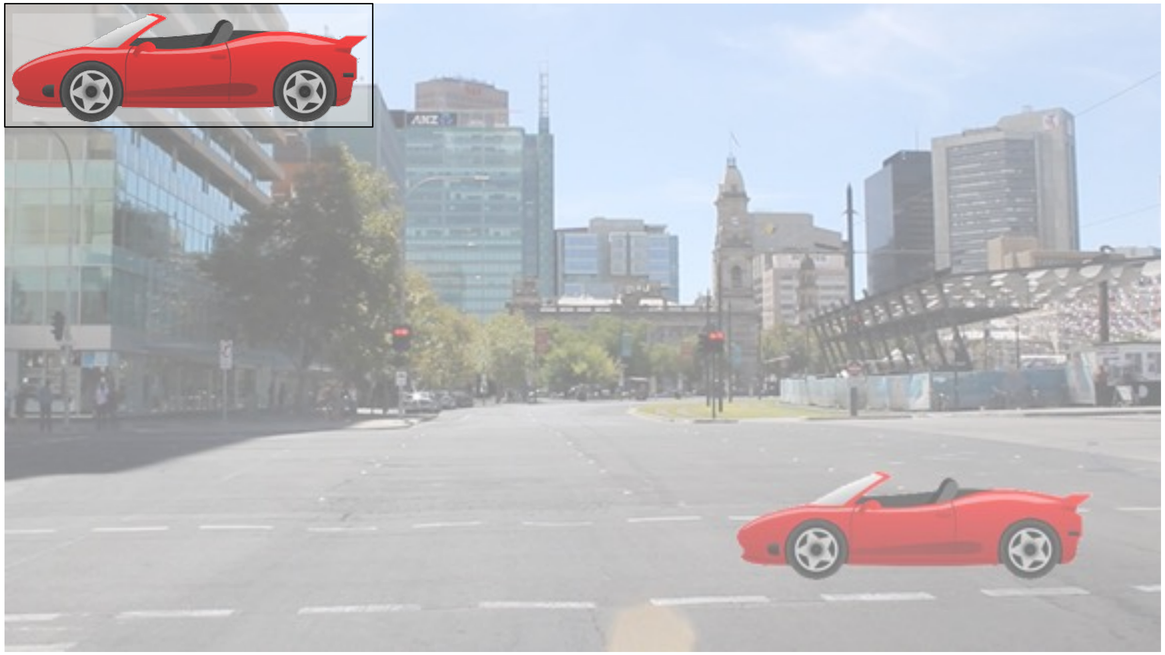
Imagem qualquer:

Imagem do objeto:



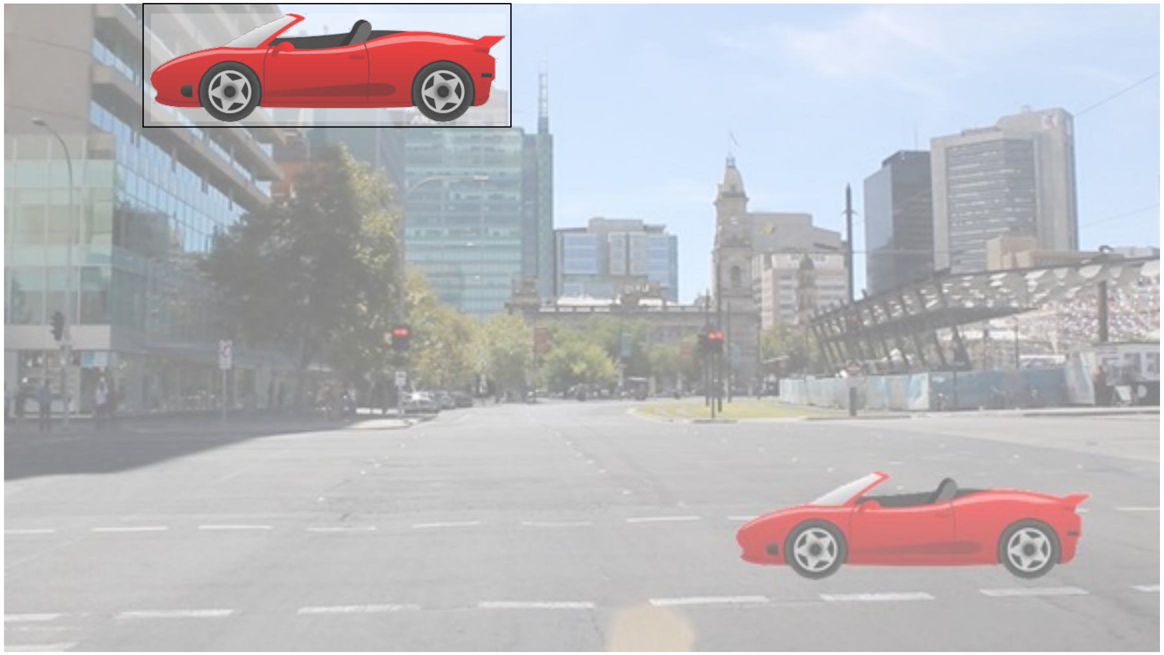
Localização de objetos

Objeto está aqui? Não.



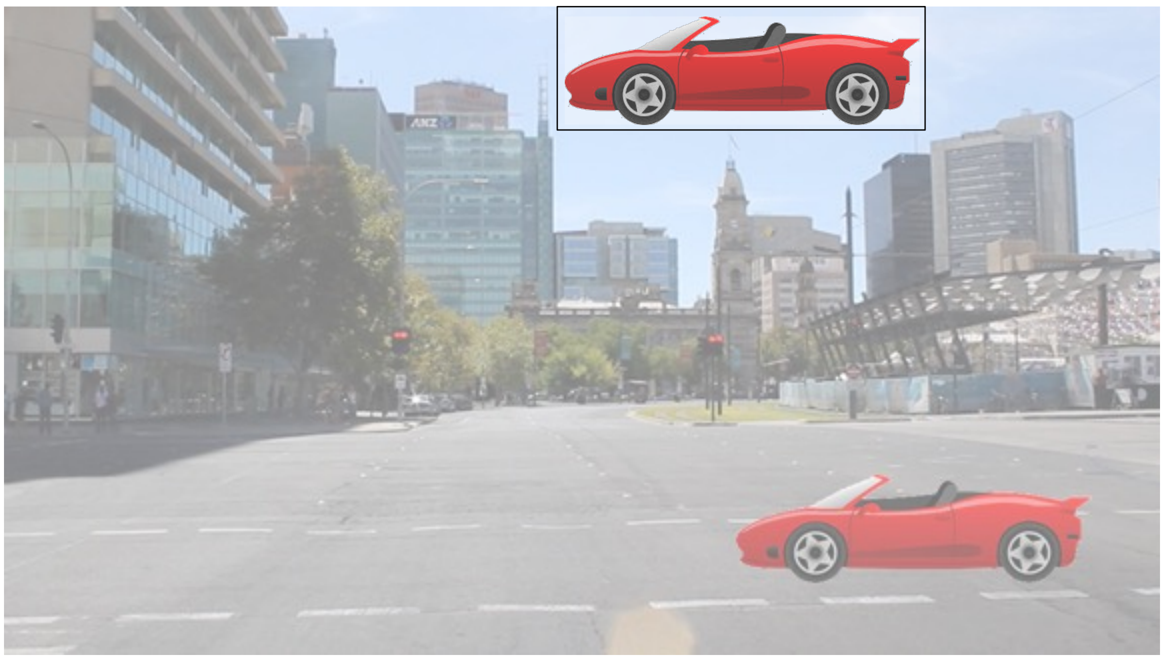
Localização de objetos

Objeto está aqui? Não.



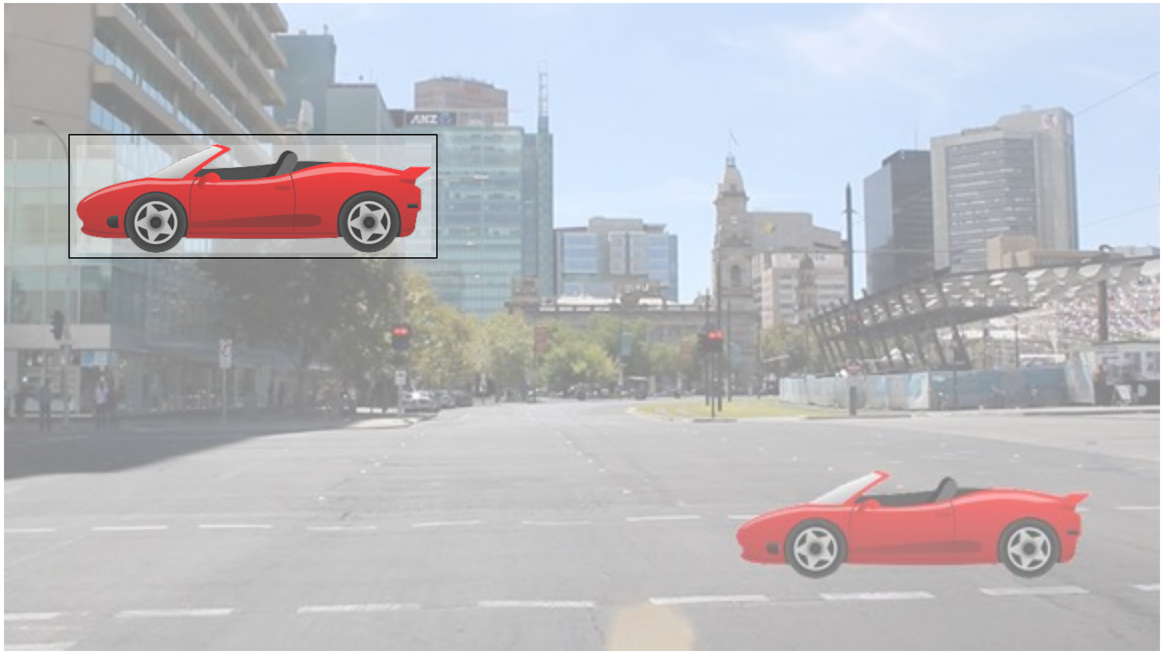
Localização de objetos

Objeto está aqui? Não.



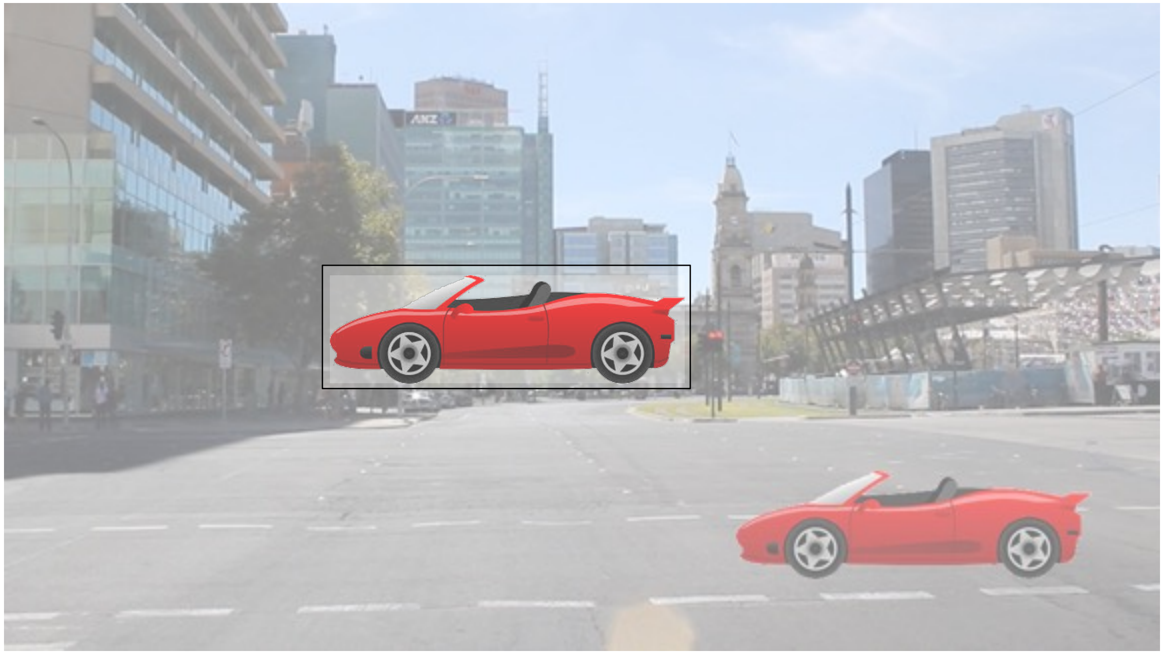
Localização de objetos

Objeto está aqui? Não.



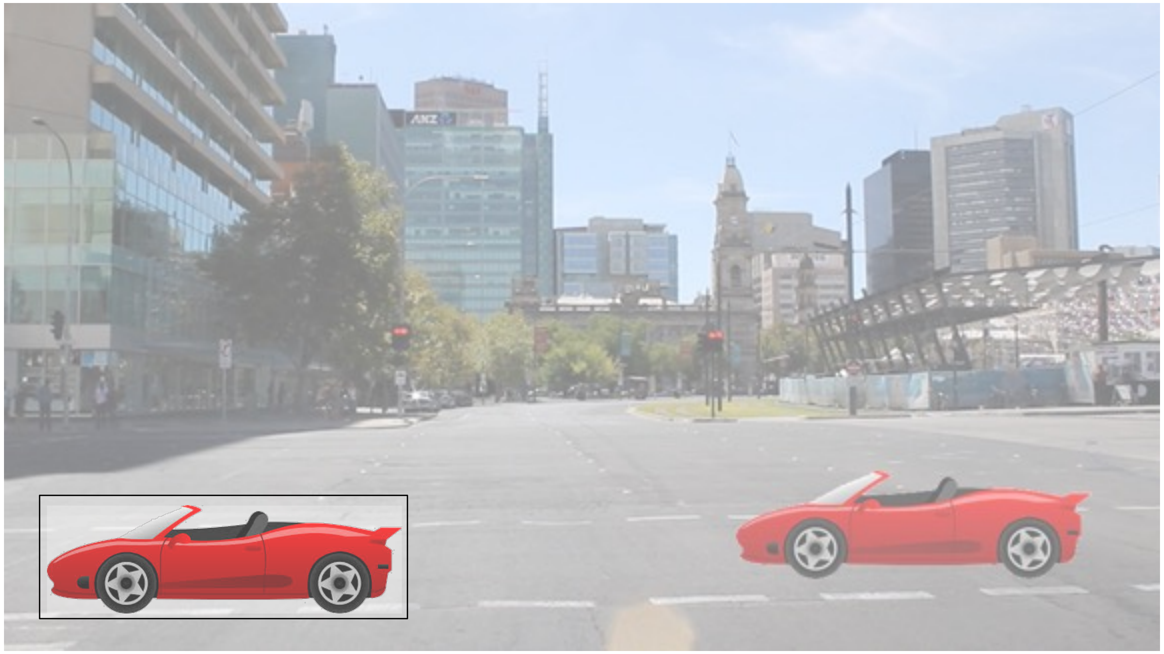
Localização de objetos

Objeto está aqui? Não.



Localização de objetos

Objeto está aqui? Não.



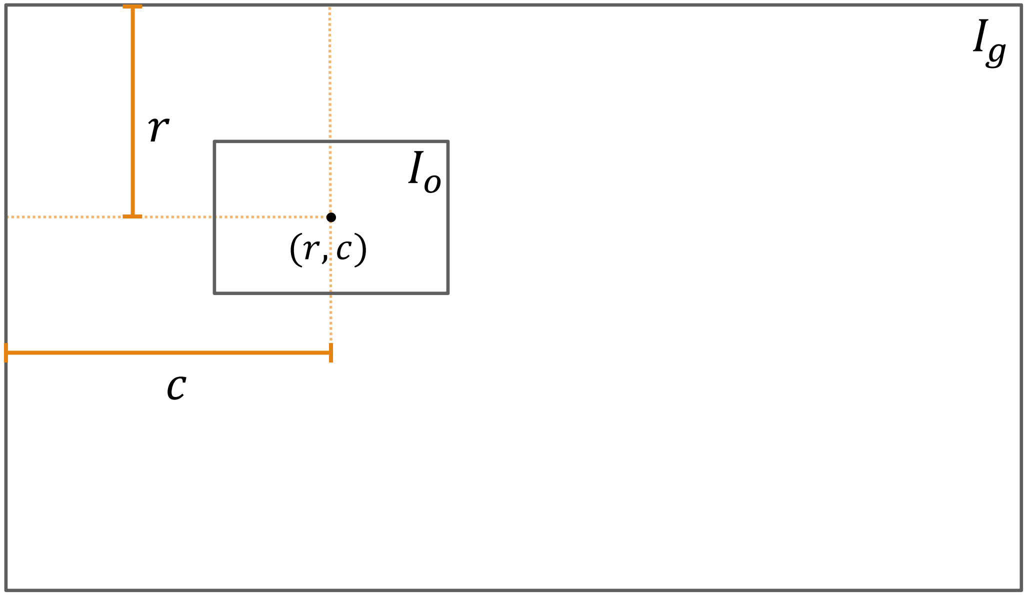
Localização de objetos

Objeto está aqui!

Localização de objetos

Seja I_o a imagem do carro, de tamanho $\mathrm{R}\times C$, e I_g a imagem a ser analisada, a diferença quadrática entre os pixels das imagens I_o e I_g na posição (r, c) é dada por

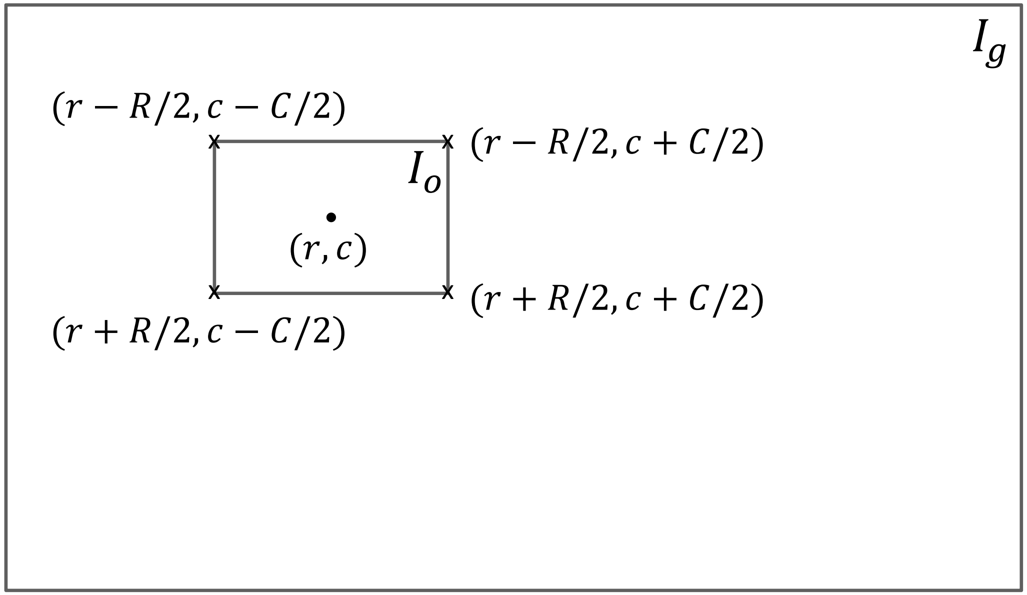
$$d(r,c) = \sum_{s=0}^{R-1} \sum_{t=0}^{C-1} [I_g(r+s-R/2, c+t-C/2) - I_o(s,t)]^2$$



Localização de objetos

Seja I_o a imagem do carro, de tamanho $\mathrm{R}\times C$, e I_g a imagem a ser analisada, a diferença quadrática entre os pixels das imagens I_o e I_g na posição (r,c) é dada por

$$d(r,c) = \sum_{s=0}^{R-1}\sum_{t=0}^{C-1}[I_g(r+s-R/2, c+t-C/2) - I_o(s,t)]^2$$



Localização de objetos

Exemplo

Imagem

5	2	0	1	4
2	4	1	3	2
3	2	3	2	4
0	2	3	4	5
2	1	0	2	3

Imagem objeto

2	3	2
2	3	4
1	0	2

Localização de objetos

Exemplo

Imagem

5	2	0	1	4
2	4	1	3	2
3	2	3	2	4
0	2	3	4	5
2	1	0	2	3

Imagem diferença

		?		

Imagem objeto

2	3	2
2	3	4
1	0	2

Para o pixel (1,2):

$$d(1,2) = (2-2)^2 + (0-3)^2 + (1-2)^2 + (4-2)^2 + (1-3)^2 + (3-4)^2 + (2-1)^2 + (3-0)^2 + (2-2)^2$$

Localização de objetos

Exemplo

Imagem

5	2	0	1	4
2	4	1	3	2
3	2	3	2	4
0	2	3	4	5
2	1	0	2	3

Imagem diferença

		29		

Imagem objeto

2	3	2
2	3	4
1	0	2

Para o pixel (1,2):

$$d(1,2) = (2-2)^2 + (0-3)^2 + (1-2)^2 + (4-2)^2 + (1-3)^2 + (3-4)^2 + (2-1)^2 + (3-0)^2 + (2-2)^2$$

$$d(1,2) = 29$$

Localização de objetos

Exemplo

Imagem

5	2	0	1	4
2	4	1	3	2
3	2	3	2	4
0	2	3	4	5
2	1	0	2	3

Imagem diferença

38	61	46	34	47
23	33	29	29	45
18	11	27	32	61
27	15	0	15	43
32	31	23	22	33

Imagem objeto

2	3	2
2	3	4
1	0	2

Localização de objetos

Exemplo

Imagem

5	2	0	1	4
2	4	1	3	2
3	2	3	2	4
0	2	3	4	5
2	1	0	2	3

Imagem diferença

38	61	46	34	47
23	33	29	29	45
18	11	27	32	61
27	15	0	15	43
32	31	23	22	33

Imagem objeto

2	3	2
2	3	4
1	0	2

Menor valor de diferença

Localização de objetos

Seja I_o a imagem do carro, de tamanho $\mathrm{R}\times C$, e I_g a imagem a ser analisada, a diferença quadrática entre I_o e I_g na posição (r, c) é dada por

$$d(r,c) = \sum_{s=0}^{R-1} \sum_{t=0}^{C-1} [I_g(r + s - R/2, c + t - C/2) - I_o(s,t)]^2$$

- O pixel (r, c) para o qual $d(r, c)$ é mínimo indica a posição do objeto na imagem.
- Essa técnica é chamada de *template matching.*

Localização de objetos

A diferença quadrática pode ser calculada utilizando correlação-cruzada!

$$d(r,c) = \sum_{s=0}^{R-1} \sum_{t=0}^{C-1} [I_g(r+s-R/2, c+t-C/2) - I_o(s,t)]^2$$

Localização de objetos

A diferença quadrática pode ser calculada utilizando correlação-cruzada!

$$d(r,c) = \sum_{s=0}^{R-1}\sum_{t=0}^{C-1}[I_g(r+s-R/2, c+t-C/2) - I_o(s,t)]^2$$

$$d(r,c) = \sum_{s=0}^{R-1}\sum_{t=0}^{C-1} I_g(r+s-R/2, c+t-C/2)^2 + I_o(s,t)^2 \\ -2I_g(r+s-R/2, c+t-C/2)I_o(s,t)$$

$$d(r,c) = I_g(r,c)^2 \circ w + S_{I_o} - 2I_g(r,c) \circ I_o$$

Onde w é um filtro de mesmo tamanho que o objeto, possuindo valor 1 em todas as posições e S_{I_o} o quadrado da soma dos valores do objeto.

Localização de objetos

Notebook “**Localização de objeto**”

Localização de objetos

Mas o que ocorre se a imagem do objeto for menor do que o objeto presente na imagem maior?

Localização de objetos

Objeto:

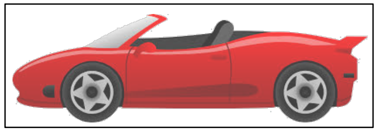


Imagem:

Localização de objetos

Está aqui? Não

Localização de objetos

Está aqui? Não

Localização de objetos

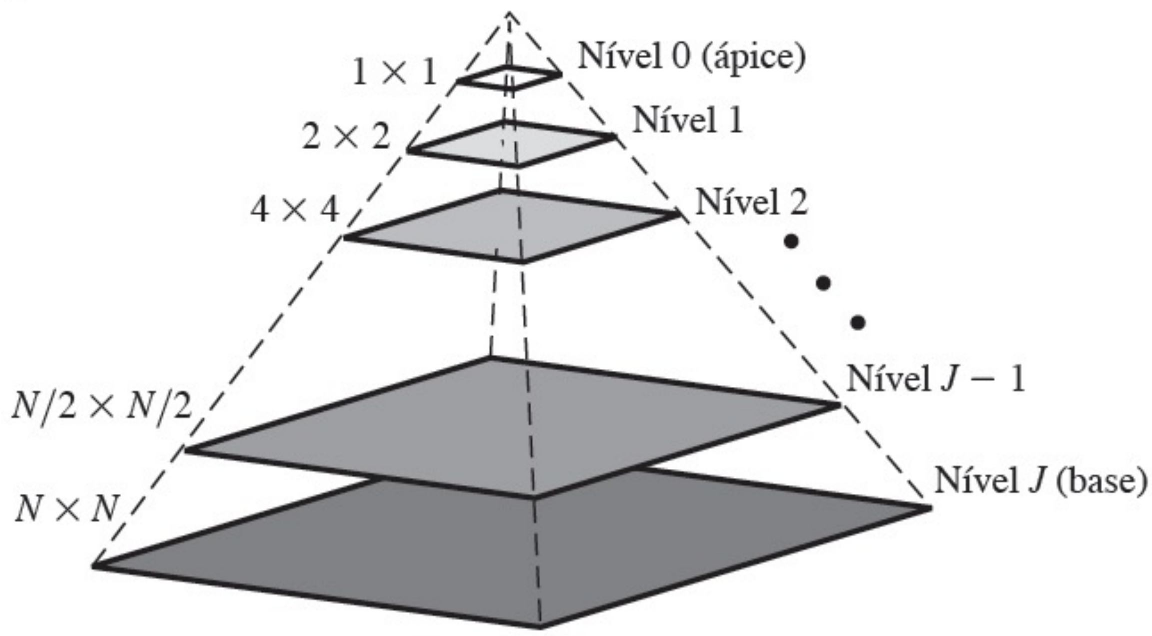
Está aqui? Não

Localização de objetos

Está aqui? Também não! A imagem do objeto é menor que o objeto contido na cena

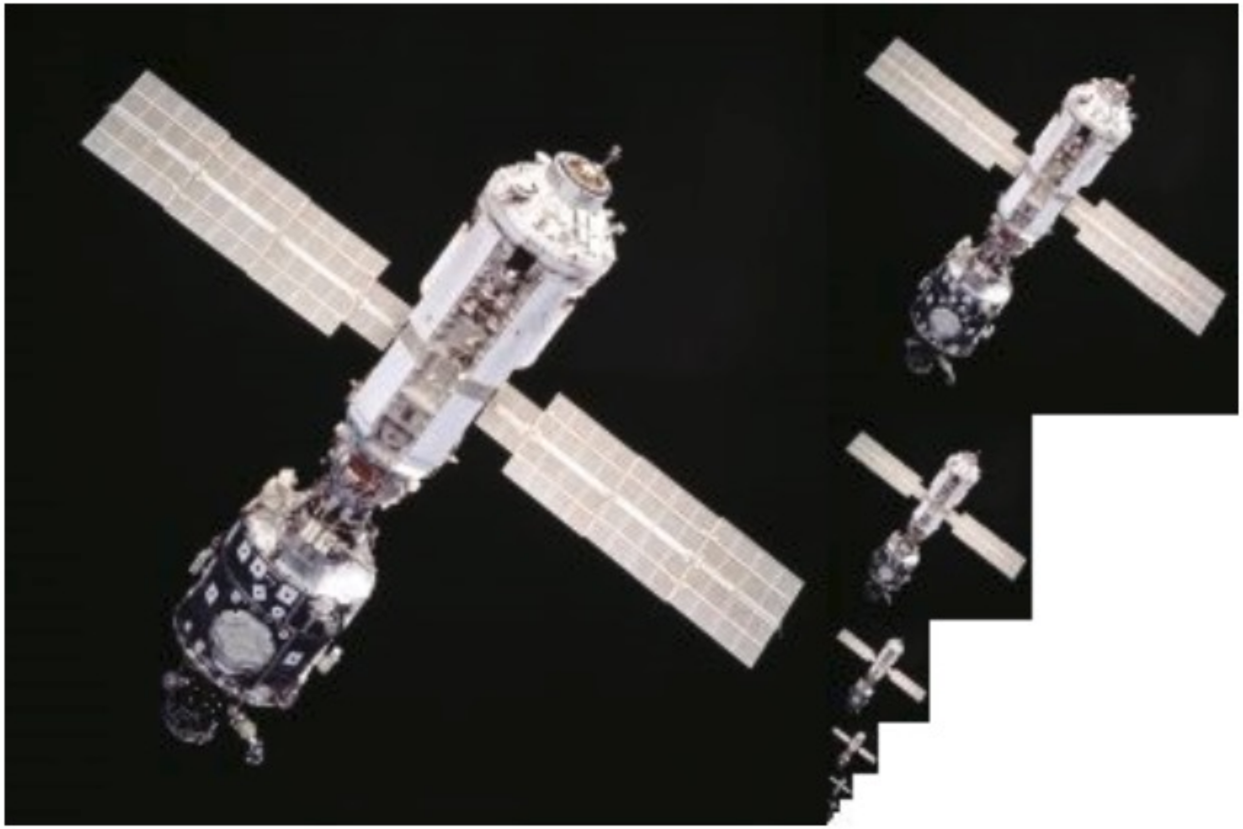
Pirâmides de imagens

- Pirâmides de imagens é uma forma de analisarmos imagens em diferentes escalas.
- É construída uma pirâmide da imagem, cada nível possui uma fração do tamanho do nível anterior.
- É muito comum considerar que cada nível possui metade do tamanho do anterior



Pirâmides de imagens

Exemplo:



Pirâmides de imagens

No nosso caso:

Pirâmides de imagens

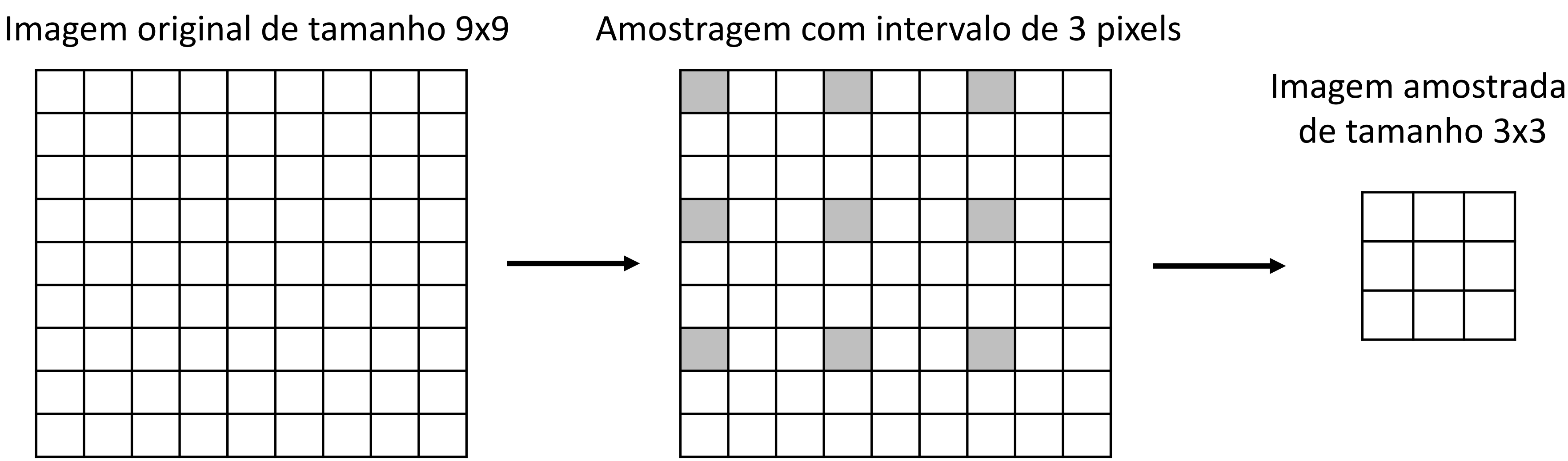
Se buscarmos o objeto em cada imagem (da menor para a maior) encontramos ele em uma das imagens.

Pirâmides de imagens

- Como diminuímos o tamanho da imagem?
- A abordagem mais simples é eliminarmos algumas linhas e colunas da imagem.
- Por exemplo, se queremos uma imagem com 1/3 do tamanho, eliminamos as linhas e colunas de índice [1, 2, 4, 5, 7, 8, ...]
    - Em outras palavras, mantemos apenas as linhas e colunas de índice [0, 3, 6, 9, ...]

Pirâmides de imagens

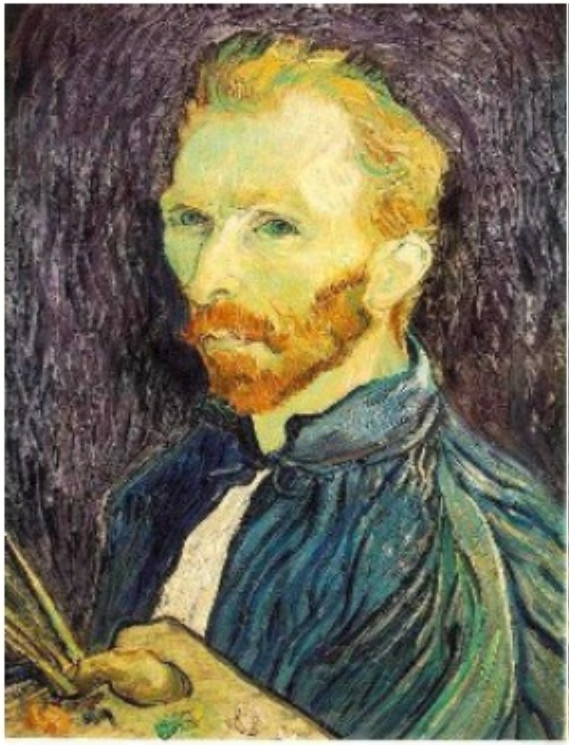
- Como diminuímos o tamanho da imagem?
- A abordagem mais simples é eliminarmos algumas linhas e colunas da imagem.
- Por exemplo, se queremos uma imagem com 1/3 do tamanho, eliminamos as linhas e colunas de índice [1, 2, 4, 5, 7, 8, ...]
  - Em outras palavras, mantemos apenas as linhas e colunas de índice [0, 3, 6, 9, ...]



Pirâmides de imagens

O problema dessa abordagem é que a imagem amostrada terá uma taxa de amostragem menor que a da imagem original, o que leva a aliasing.

Pirâmides de imagens



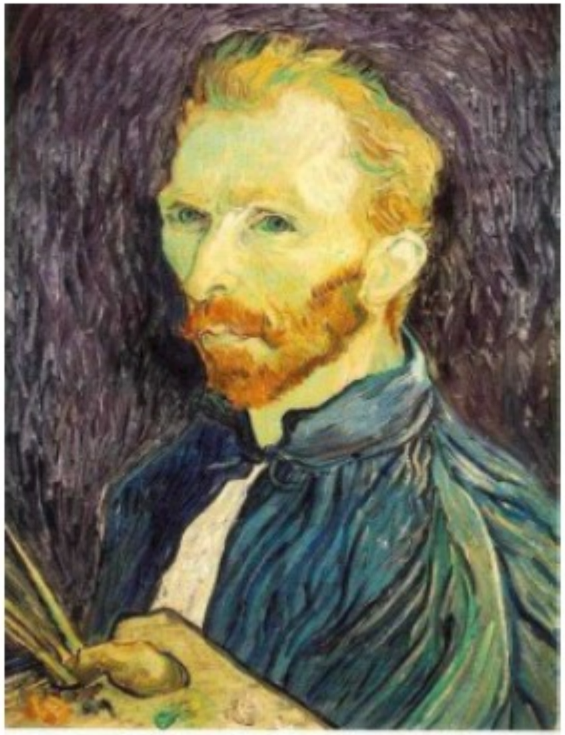
1/2

1/4

Pirâmides de imagens

- A solução é suavizar a imagem antes de diminuir a resolução (downsampling).
- Lembrete: suavização elimina altas frequências.
- Uma abordagem muito comum é a aplicação de suavização gaussiana utilizando um valor de σ adequado. Nesse caso, obtemos uma pirâmide gaussiana.

Pirâmides de imagens



Gaussiana 1/2

Gaussiana 1/4

1/4

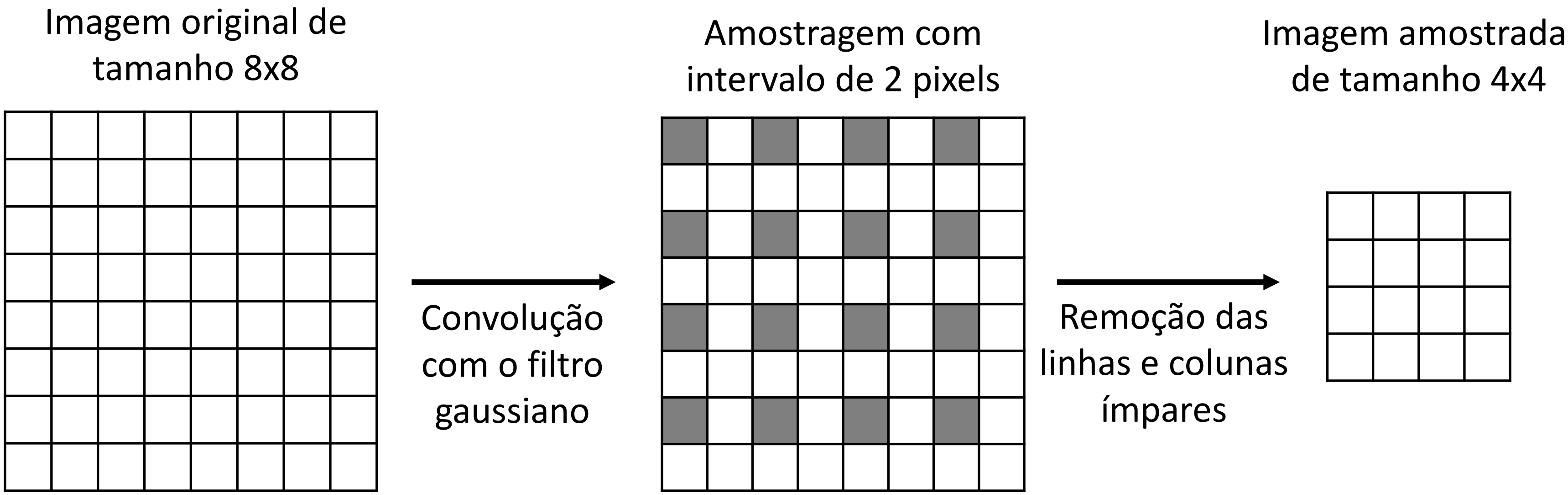
Pirâmide gaussiana

- No caso da pirâmide gaussiana, é muito comum reduzir o tamanho da imagem pela metade para cada nível da pirâmide.

- O seguinte filtro (kernel) é utilizado para suavizar a imagem antes da reamostragem:

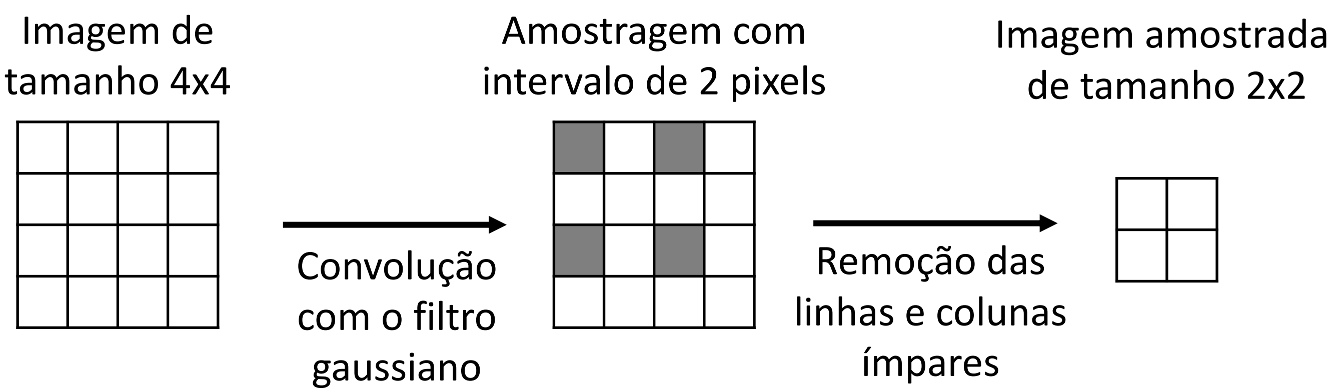
$$\frac{1}{256} \times \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

Pirâmide gaussiana – Construção do nível 2



Filtro gaussiano: $\frac{1}{256} \times \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$

Pirâmide gaussiana – Construção do nível 3



Filtro gaussiano: $\frac{1}{256}\times$

1	4	6	4	1
4	16	24	16	4
6	24	36	24	6
4	16	24	16	4
1	4	6	4	1

Pirâmide gaussiana

Notebook “**Pirâmide gaussiana**”

Pirâmide gaussiana

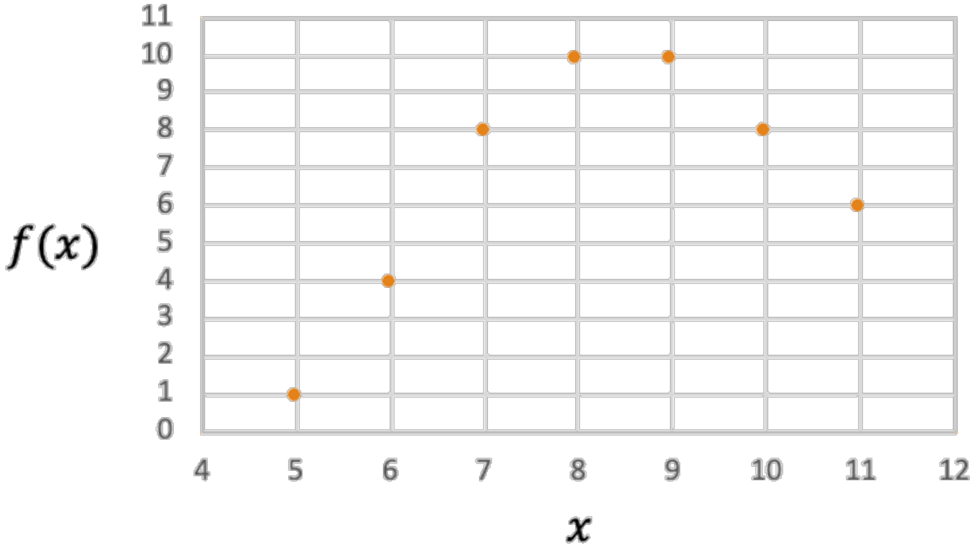
Vimos como reduzir o tamanho de uma imagem. Será que podemos também aumentar o seu tamanho?

Interpolação de imagens

Sim! O procedimento utilizado para aumentar a resolução de uma imagem (ou de um sinal qualquer) é chamado de **interpolação**.

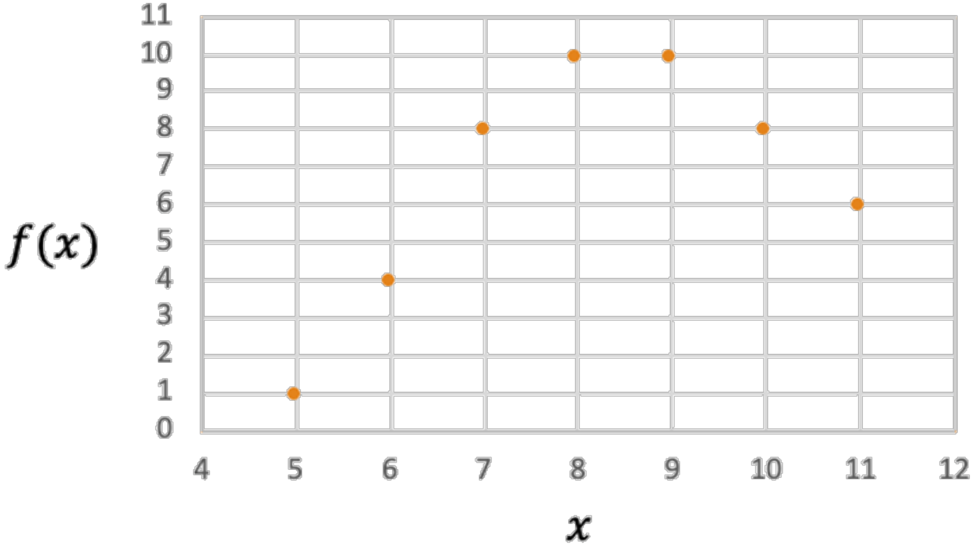
Interpolação de um sinal 1D

Vamos supor que temos o seguinte sinal



Interpolação de um sinal 1D

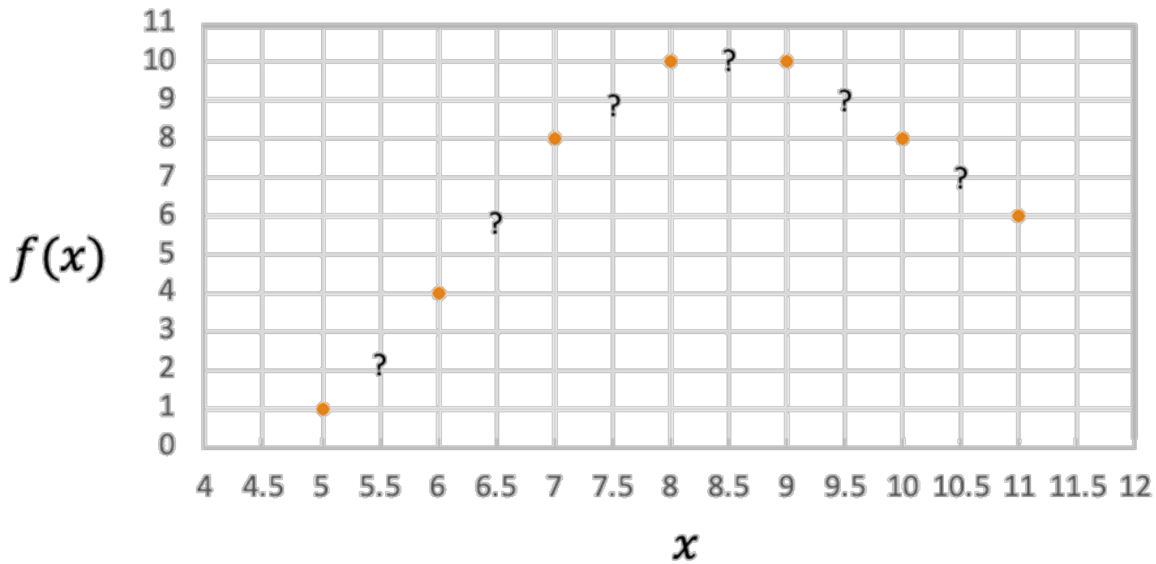
- Conhecemos o valor desse sinal nos pontos $x = 5$, $x = 6$, $x = 7$, $x = 8$, $x = 9$, $x = 10$ e $x = 11$. Ou seja, temos um intervalo de amostragem $\Delta x = 1$



$f(x)$	1	4	8	10	10	8	6
	0	1	2	3	4	5	6

Interpolação de um sinal 1D

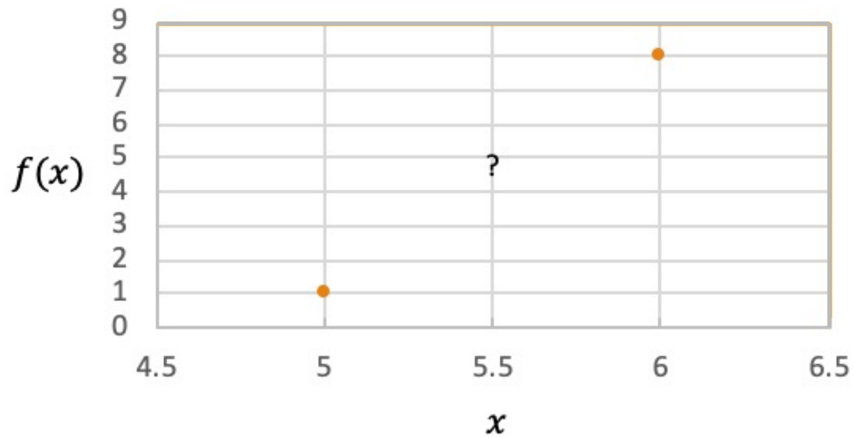
- Conhecemos o valor desse sinal nos pontos $x = 5$, $x = 6$, $x = 7$, $x = 8$, $x = 9$, $x = 10$ e $x = 11$. Ou seja, temos um intervalo de amostragem $\Delta x = 1$
- Nosso objetivo é diminuir o intervalo de amostragem para $\Delta x = 0.5$
- Para isso, precisamos estimar os valores do sinal nos pontos $x = 5.5$, $x = 6.5$, $x = 7.5$, $x = 8.5$, $x = 9.5$ e $x = 10.5$



$f(x)$	1	?	4	?	8	?	10	?	10	?	8	?	6
	0	1	2	3	4	5	6	7	8	9	10	11	12

Interpolação de vizinho mais próximo

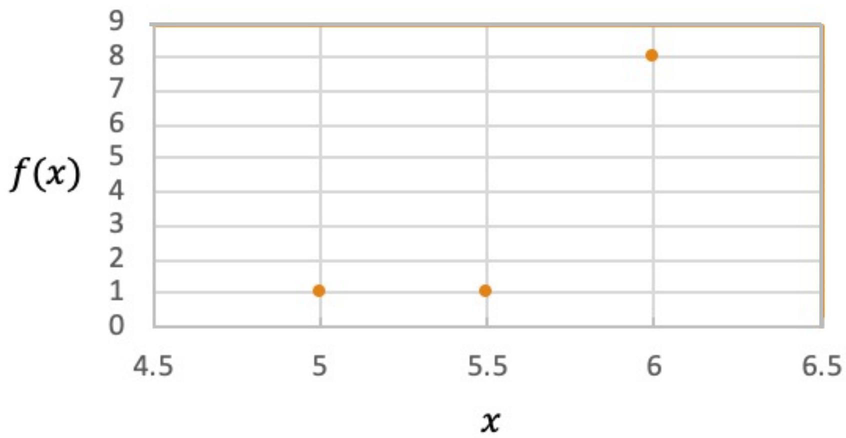
- A interpolação mais simples que podemos fazer é a de vizinho mais próximo
- Nesse caso, associamos ao ponto desconhecido o valor mais próximo que conhecemos do sinal
- Caso haja dois pontos mais próximos, escolhemos apenas um dos pontos



$f(x)$	1	?	8
	0	1	2

Interpolação de vizinho mais próximo

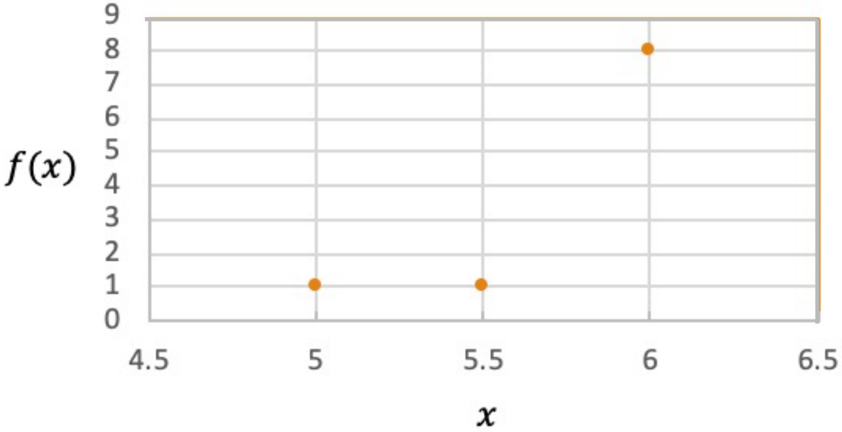
- A interpolação mais simples que podemos fazer é a de vizinho mais próximo
- Nesse caso, associamos ao ponto desconhecido o valor mais próximo que conhecemos do sinal
- Caso haja dois pontos mais próximos, escolhemos apenas um dos pontos



$f(x)$	1	1	8
	0	1	2

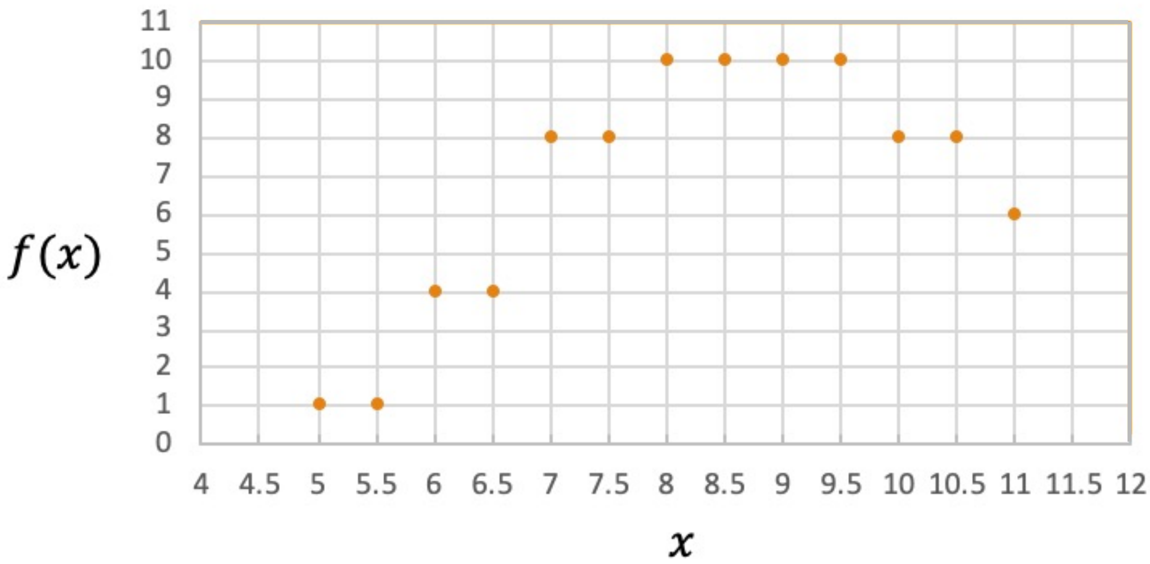
Interpolação de vizinho mais próximo

- A interpolação mais simples que podemos fazer é a de vizinho mais próximo
- Nesse caso, associamos ao ponto desconhecido o valor mais próximo que conhecemos do sinal
- Caso haja dois pontos mais próximos, escolhemos apenas um dos pontos
- Dizemos que essa interpolação possui ordem 0



$f(x)$	1	1	8
	0	1	2

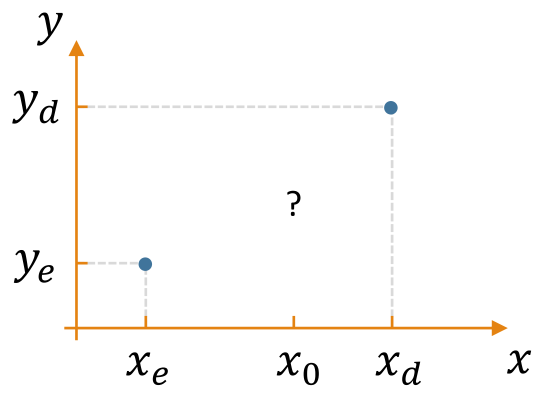
Interpolação de vizinho mais próximo



$f(x)$	1	1	4	4	8	8	10	10	10	10	8	8	6
	0	1	2	3	4	5	6	7	8	9	10	11	12

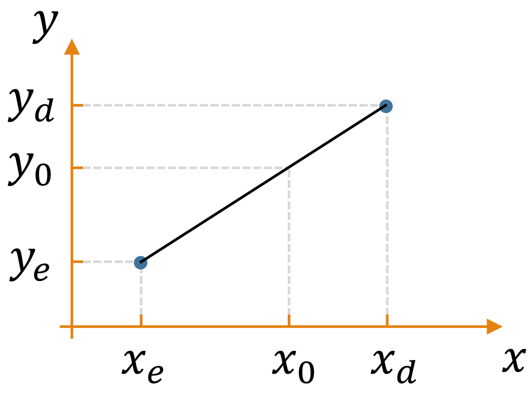
Interpolação linear

- Na interpolação linear, para cada ponto desconhecido x_0 utilizamos o ponto conhecido à esquerda de x_0, que chamaremos de x_e, e o ponto conhecido à direita de x, que chamaremos de x_d.



Interpolação linear

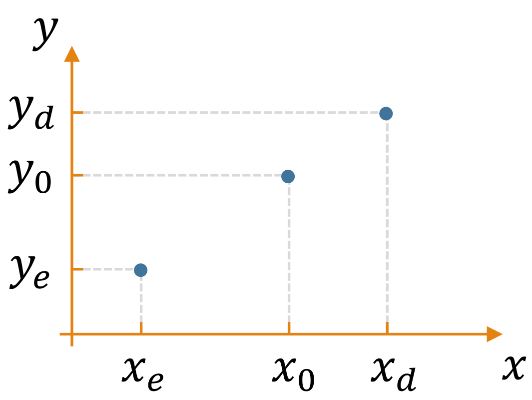
- Na interpolação linear, para cada ponto desconhecido x_0 utilizamos o ponto conhecido à esquerda de x_0, que chamaremos de x_e, e o ponto conhecido à direita de x, que chamaremos de x_d.
- Definimos uma reta passando pelos pontos x_e e x_d, e utilizamos a equação dessa reta para encontrar o valor do sinal no ponto x



$$y_0 = y_e + \frac{y_d - y_e}{x_d - x_e}(x_0 - x_d)$$

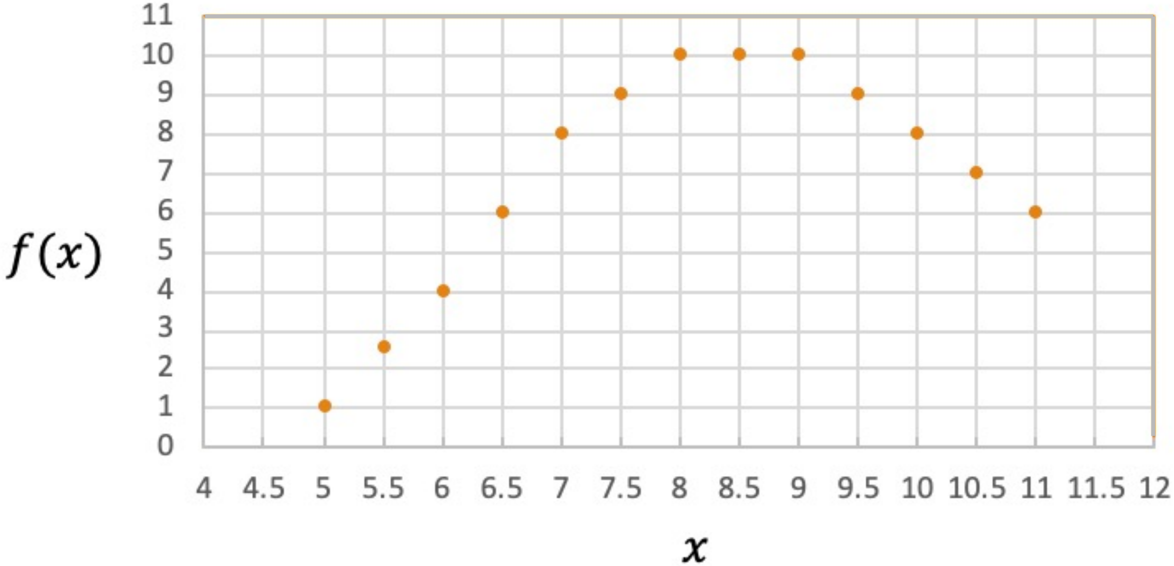
Interpolação linear

- Na interpolação linear, para cada ponto desconhecido x_0 utilizamos o ponto conhecido à esquerda de x_0, que chamaremos de x_e, e o ponto conhecido à direita de x, que chamaremos de x_d.
- Definimos uma reta passando pelos pontos x_e e x_d, e utilizamos a equação dessa reta para encontrar o valor do sinal no ponto x
- Essa interpolação possui ordem 1, pois ela é formada por retas e garante a continuidade do sinal (o que não ocorre na interpolação de ordem 0)



$$y_0 = y_e + \frac{y_d - y_e}{x_d - x_e}(x_0 - x_e)$$

Interpolação linear



$f(x)$	1	2.5	4	6	8	9	10	10	10	9	8	7	6
	0	1	2	3	4	5	6	7	8	9	10	11	12

Interpolação cúbica

- Provavelmente o tipo de interpolação mais utilizado.
- Possui diferentes definições
- Tipicamente, os dois vizinhos à esquerda e à direita do ponto desconhecido são utilizados (4 pontos no total)

Interpolação cúbica

- Provavelmente o tipo de interpolação mais utilizado.
- Possui diferentes definições
- Tipicamente, os dois vizinhos à esquerda e à direita do ponto desconhecido são utilizados (4 pontos no total)
- Essa interpolação possui ordem 3, pois é formada por polinômios de grau 3 e garante a continuidade do sinal e de sua primeira e segunda derivada
- O sinal resultante é mais suave do que no caso da interpolação linear
- Veremos mais adiante como calcular essa interpolação

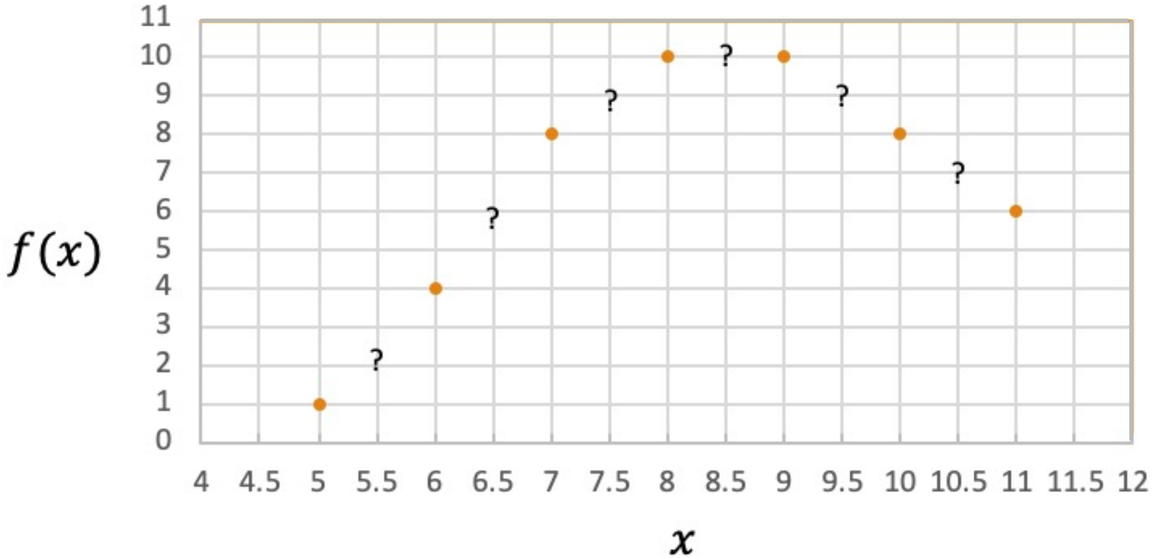
Interpolação por convolução

Interpolação por correlação-cruzada

Os procedimentos de interpolação que acabamos de ver podem ser definidos utilizando correlação-cruzada ou convolução.

Interpolação por correlação-cruzada

- Lembrando do nosso objetivo, encontrar os valores desconhecidos do sinal superamostrado (com maior resolução que o original)



$f(x)$	1	?	4	?	8	?	10	?	10	?	8	?	6
	0	1	2	3	4	5	6	7	8	9	10	11	12

Interpolação por correlação-cruzada

- Vamos definir o seguinte filtro caixa

  w = [0, 0, 1, 1, 0, 0 ,0]

Interpolação por correlação-cruzada

- Vamos definir o seguinte filtro caixa

  w = [0, 0, 1, 1, 0, 0 ,0]

- Vamos agora definir um novo array, possuindo os valores do sinal original intercalados pelo valor 0

	0	1	2	3	4	5	6	7	8	9	10	11	12
$f(x)$	1	0	4	0	8	0	10	0	10	0	8	0	6

- Se calcularmos a correlação-cruzada entre $f(x)$ e w, obtemos exatamente a interpolação por vizinho mais próximo!

Interpolação por correlação-cruzada

$f(x)$	1	0	4	0	8	0	10	0	10	0	8	0	6
	0	1	2	3	4	5	6	7	8	9	10	11	12

w = [0, 0, 1, 1, 0, 0 ,0]

g(x)	1												
	0	1	2	3	4	5	6	7	8	9	10	11	12

Interpolação por correlação-cruzada

$f(x)$	1	0	4	0	8	0	10	0	10	0	8	0	6
	0	1	2	3	4	5	6	7	8	9	10	11	12

w = [0, 0, 1, 1, 0, 0 ,0]

$g(x)$	1	1											
	0	1	2	3	4	5	6	7	8	9	10	11	12

Interpolação por correlação-cruzada

$f(x)$	1	0	4	0	8	0	10	0	10	0	8	0	6
	0	1	2	3	4	5	6	7	8	9	10	11	12

w = [0, 0, 1, 1, 0, 0 ,0]

$g(x)$	1	1	4										
	0	1	2	3	4	5	6	7	8	9	10	11	12

Interpolação por correlação-cruzada

$f(x)$	1	0	4	0	8	0	10	0	10	0	8	0	6
	0	1	2	3	4	5	6	7	8	9	10	11	12

w = [0, 0, 1, 1, 0, 0 ,0]

g(x)	1	1	4	4									
	0	1	2	3	4	5	6	7	8	9	10	11	12

Interpolação por correlação-cruzada

$f(x)$	1	0	4	0	8	0	10	0	10	0	8	0	6
	0	1	2	3	4	5	6	7	8	9	10	11	12

w = [0, 0, 1, 1, 0, 0 ,0]

g(x)	1	1	4	4	8	8	10	10	10	10	8	8	6
	0	1	2	3	4	5	6	7	8	9	10	11	12

Interpolação por correlação-cruzada

Vamos agora definir um novo filtro w_2, calculado pela correlação-cruzada de dois filtros caixa e normalizado por 2

$$w_2 = \frac{w \circ w}{2}$$

w_2 = [0, 0, 0.5, 1, 0.5, 0 ,0]

Esse filtro é uma função triângulo

A correlação-cruzada do sinal com o filtro w_2 é equivalente a fazermos uma interpolação linear.

Interpolação por correlação-cruzada

$f(x)$	1	0	4	0	8	0	10	0	10	0	8	0	6
	0	1	2	3	4	5	6	7	8	9	10	11	12

w = [0, 0, 0.5, 1, 0.5, 0 ,0]

$g(x)$	1												
	0	1	2	3	4	5	6	7	8	9	10	11	12

Interpolação por correlação-cruzada

$f(x)$	1	0	4	0	8	0	10	0	10	0	8	0	6
	0	1	2	3	4	5	6	7	8	9	10	11	12

w = [0, 0, 0.5, 1, 0.5, 0 ,0]

g(x)	1	2.5											
	0	1	2	3	4	5	6	7	8	9	10	11	12

Interpolação por correlação-cruzada

$f(x)$	1	0	4	0	8	0	10	0	10	0	8	0	6
	0	1	2	3	4	5	6	7	8	9	10	11	12

w = [0, 0, 0.5, 1, 0.5, 0 ,0]

g(x)	1	2.5	4	6									
	0	1	2	3	4	5	6	7	8	9	10	11	12

Interpolação por correlação-cruzada

$f(x)$													
	1	0	4	0	8	0	10	0	10	0	8	0	6
	0	1	2	3	4	5	6	7	8	9	10	11	12

w = [0, 0, 0.5, 1, 0.5, 0 ,0]

g(x)													
	1	2.5	4	6	8	9	10	10	10	9	8	7	6
	0	1	2	3	4	5	6	7	8	9	10	11	12

Interpolação por correlação-cruzada

Podemos também definir o filtro w_3, calculado pela correlação-cruzada de dois filtros w_3 e normalizado por 2

$$w_3 = \frac{w_2 \circ w_2}{2}$$

w_3 = [0, 0.125, 0.5, 0.75, 0.5, 0.125 ,0]

Esse filtro é **aproximadamente** uma interpolação cúbica. Não temos realmente uma interpolação cúbica porque o filtro altera os valores conhecidos do sinal.

Interpolação cúbica por correlação-cruzada

Para aplicarmos a interpolação cúbica (ordem 3) exata, utilizamos o seguinte filtro

w_c = [-0.0625, 0, 0.5625, 1, 0.5625, 0, -0.0625]

Interpolação por correlação-cruzada

Definição geral de interpolação por convolução:

- Dado um sinal f_n possuindo intervalo de amostragem Δx, a interpolação exata de ordem 0 ou 1, ou a interpolação aproximada de ordem 3 desse sinal pode ser feita utilizando correlação-cruzada.
- Se queremos interpolar o sinal com um intervalo de amostragem igual a $\Delta x/r$, onde r é um número inteiro maior que 1, utilizamos o seguinte procedimento:

Interpolação por correlação-cruzada

Definição geral de interpolação por convolução:

- Dado um sinal f_n possuindo intervalo de amostragem Δx, a interpolação exata de ordem 0 ou 1, ou a interpolação aproximada de ordem 3 desse sinal pode ser feita utilizando correlação-cruzada.
- Se queremos interpolar o sinal com um intervalo de amostragem igual a $\Delta x/r$, onde r é um número inteiro maior que 1, utilizamos o seguinte procedimento:

1. Gere um novo sinal $\tilde{f}_n$, definido da seguinte forma:

$$\tilde{f}_n = \begin{cases} f_n, & \text{se } n \text{ divisível por } r \\ 0, & \text{caso contrário} \end{cases}$$

2. Defina um filtro caixa w de altura 1 e largura r
3. Para interpolação de ordem 0, 1, ou 3, utilize o respectivo filtro $w_0 = w$, $\mathrm{w}_1 = w \circ w$ ou w_c
4. Calcule a correlação-cruzada entre $\tilde{f}_n$ e o filtro gerado

Caso r não seja inteiro, a interpolação é feita pelos procedimentos que vimos anteriormente (sem usar convolução)

Interpolação de sinais

Notebook “**Interpolação 1D**”

Pirâmides laplacianas

- Usualmente implementadas em conjunto com pirâmides gaussianas.

- Cada nível da pirâmide é dado pela diferença entre a interpolação do nível i da pirâmide gaussiana e o nível i+1 da mesma.

Pirâmides laplacianas

Imagem original

Suavização e subamostragem

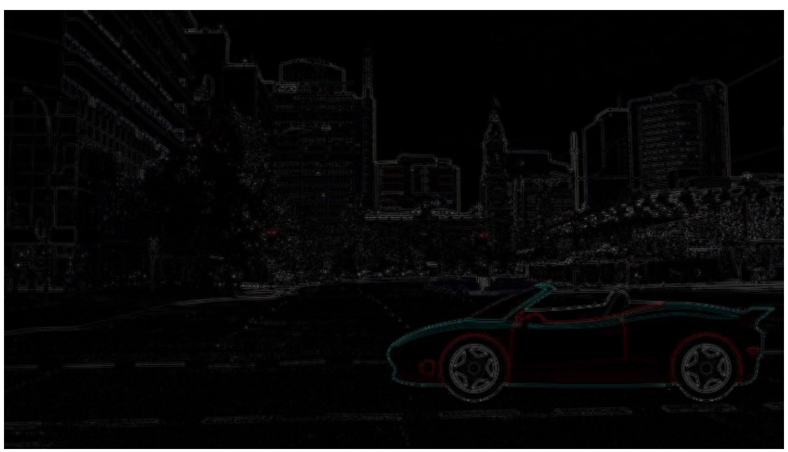
Imagem subamostrada

Superamostragem e interpolação

Imagem interpolada

Subtração

Imagem diferença



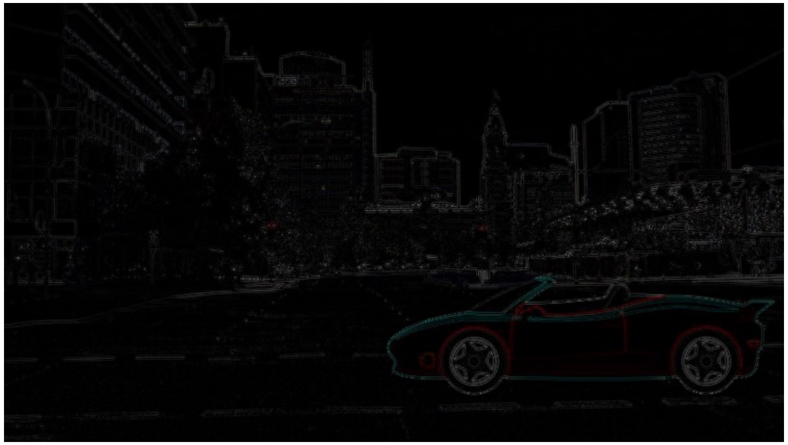
Pirâmides laplacianas

- Note que podemos recuperar exatamente a imagem original se tivermos a imagem subamostrada e a imagem diferença

Pirâmides laplacianas

Imagem subamostrada

Imagem diferença

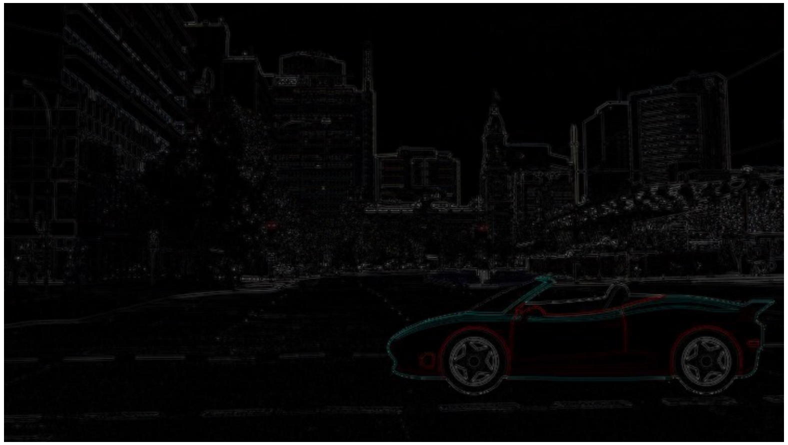


Pirâmides laplacianas

Imagem subamostrada

Superamostragem
e interpolação

Imagem diferença



Pirâmides laplacianas

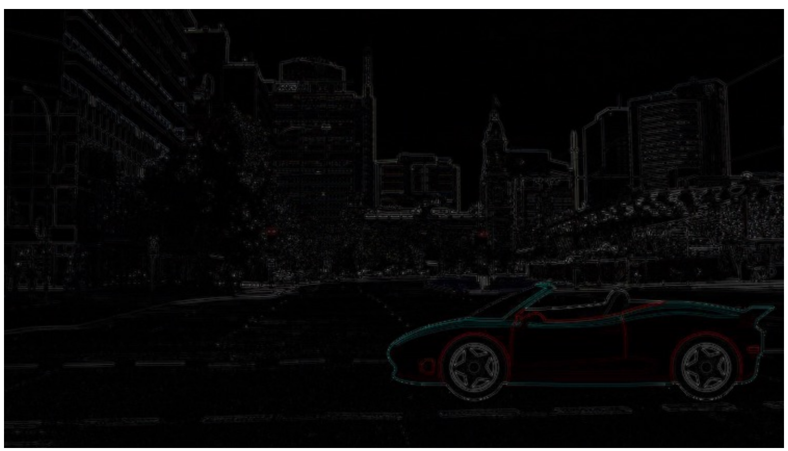
Imagem original

Imagem subamostrada

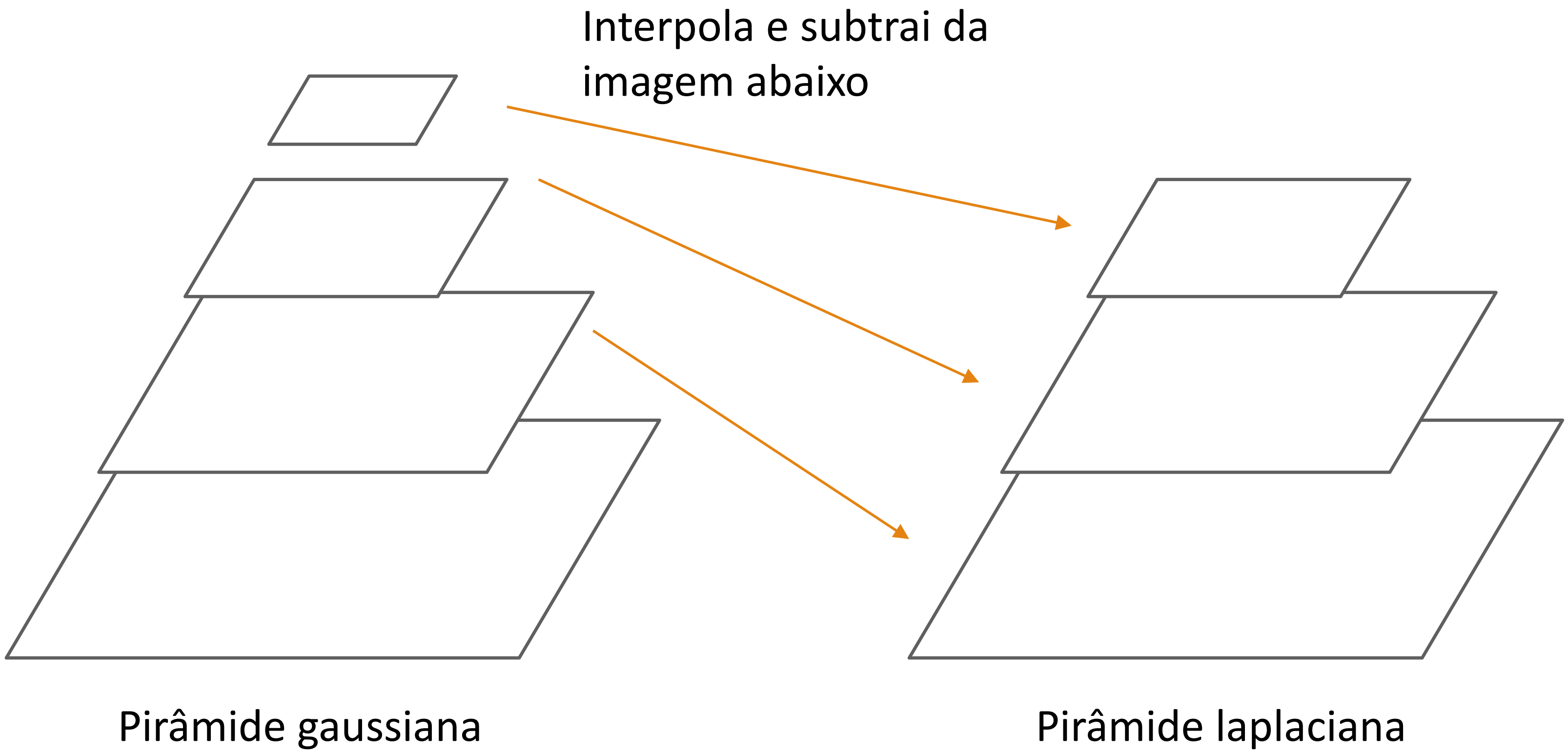
Superamostragem
e interpolação

Soma

Imagem diferença



Pirâmides laplacianas

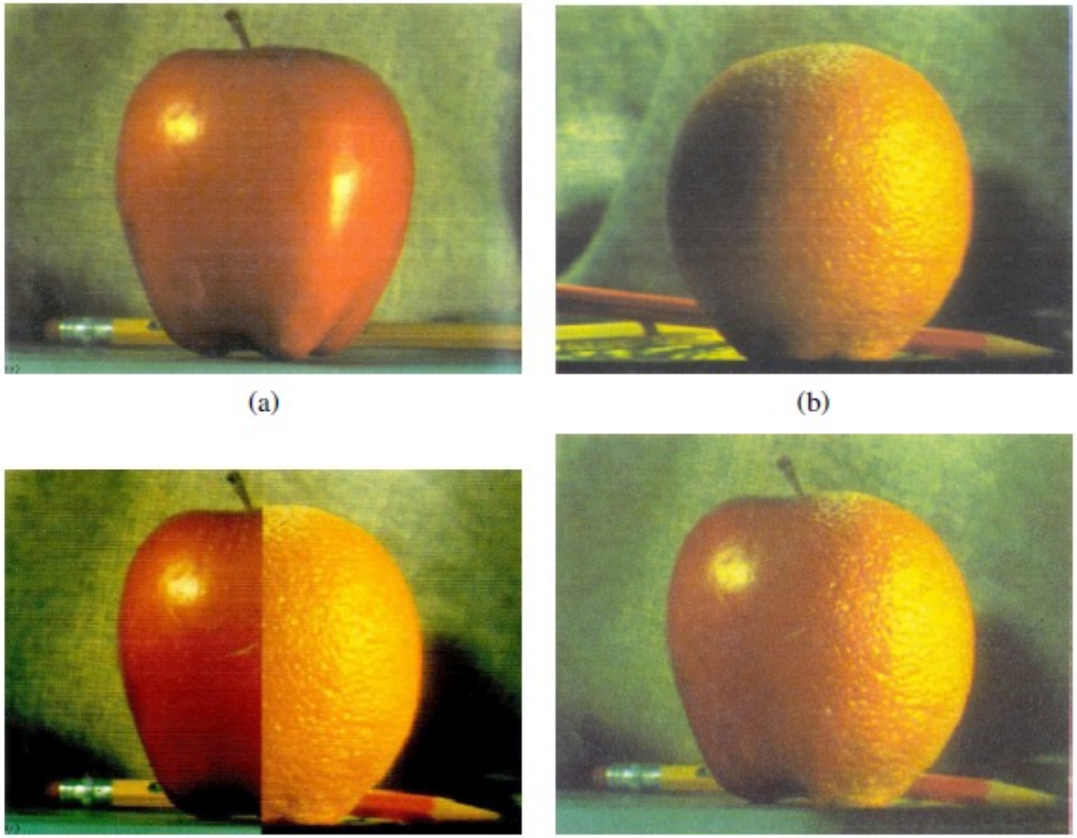


Pirâmides laplacianas

- Se armazenarmos apenas a imagem do topo da pirâmide gaussiana e as imagens da pirâmide laplaciana, podemos reconstruir exatamente a imagem original.
- Algoritmos de compressão conseguem reduzir bastante o espaço necessário para armazenar a pirâmide laplaciana, pois esta possui muitos valores iguais ou próximos a zero.

Pirâmides de imagens

Pirâmides laplacianas também podem ser utilizadas para misturar duas imagens



(a) (b) (c) (d)

Projeto 2

- Não utilizar funções prontas para implementar o principal conceito associado ao tema. Na dúvida, pergunte que funções/bibliotecas podem ser utilizadas no projeto.
- Entregáveis:
  - Código produzido (a organização do código também será avaliada!)
  - Um artigo escrito em Latex (∼6 páginas em coluna dupla ou ∼10 em coluna única) contendo:
    - Resumo
    - Introdução
      - Motivação do uso do método (porque usar? Em que situações ele é importante?)
    - Objetivos
      - O que será analisado sobre o método?
    - Metodologia
      - Explicação sobre a teoria do método
      - Explicação sobre a parte mais importante do código
    - Resultados
    - Conclusões
- Data de entrega: 07/02

Projeto 2 - Temas

1. Análise experimental de complexidade da convolução espacial e por FFT
2. Filtro passa baixa e passa-alta butterworth
3. Template matching por correlação de Pearson
4. Template matching com variação de luminosidade
5. Construção da pirâmide laplaciana
6. Template matching com variação de escala

Projeto 2 – Tema 1

- Análise experimental de complexidade da convolução espacial e por FFT
- Meça experimentalmente o tempo necessário para calcular a convolução entre um sinal 1D e um filtro 1D em dois casos:
    - Convolução espacial
    - Convolução utilizando a FFT
- Considere diferentes tamanhos para o sinal e para o filtro
- A partir de qual tamanho de sinal e de filtro a FFT passa a ser mais vantajosa?
- Faça o mesmo procedimento para imagens

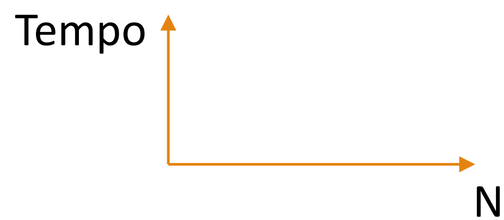
- Para calcular as convoluções, utilize as funções

Convolução espacial: convolve(signal, filter, method='direct’)
Convolução por FFT: convolve(signal, filter, method=‘fft’)

- Para medir o tempo, utilize o módulo time

```
import time
current_time = time.time()
```

Faça gráficos com as medidas de tempo



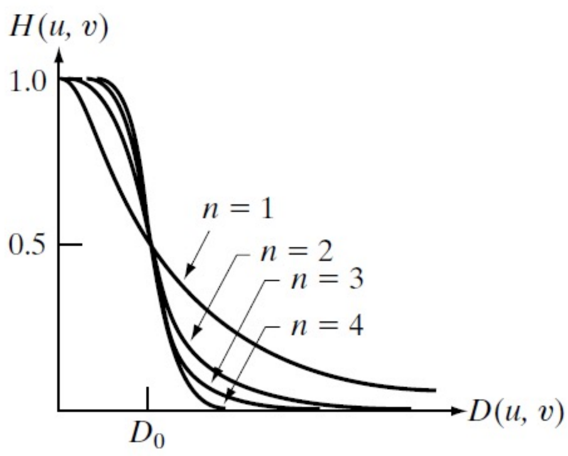
Projeto 2 – Tema 2

- Implemente a filtragem passa-baixa e passa-alta utilizando o filtro Butterworth
- Dois parâmetros: D_0 e n
- Faça uma análise exploratória (com figuras) da influência dos diferentes valores de n e D_0 no resultado
- Qual um possível critério para selecionar o melhor n?

Passa-baixa

$$H(\mu,\nu) = \frac{1}{1 + \left(\frac{D(\mu,\nu)}{D_0}\right)^{2n}}$$

$$D(\mu,\nu) = \sqrt{\mu^2 + \nu^2}$$



Passa-alta

$$H(\mu,\nu) = \frac{1}{1 + \left(\frac{D_0}{D(\mu,\nu)}\right)^{2n}}$$

Projeto 2 – Tema 3

- Template matching utilizando correlação de Pearson

$$d(r,c) = \frac{\sum_{s=0}^{R-1}\sum_{t=0}^{C-1}(I_g(r+s-R/2, c+t-C/2)-\mu_{rc})(I_o(s,t)-\mu_{I_o})}{\sqrt{\sigma_{rc}^2\sigma_{I_o}^2}}$$

Seja $\tilde{I}_g$ a região da imagem I_g que está sob o template I_o quando este está na posição (r, c). As quantias que aparecem na equação são dadas por:
μ_{rc}: Média dos valores de $\tilde{I}_g$
μ_{I_o}: Média dos valores da imagem template I_o
σ_{rc}: Desvio padrão dos valores de $\tilde{I}_g$
σ_{I_o}: Desvio padrão dos valores da imagem template I_o

A média e desvio padrão de uma imagem I podem ser calculadas, respectivamente, pelas funções numpy.mean(I) e numpy.std(I).

Projeto 2 – Tema 4

Template matching com variação de iluminação

- Na técnica de template matching, podemos ter uma imagem de template com iluminação diferente da cena global
- Implemente uma técnica de template matching com variação de luminosidade utilizando a transformação pontual de lei de potência.
- A ideia é aplicar o template matching entre a imagem global (imagem maior) e diferentes versões da imagem template I_o, cada uma gerada através da transformação de potência da imagem template com respectivo expoente γ:

$$I_o(\gamma) = (I_o)^\gamma$$

O procedimento é o seguinte:

1. Gere uma imagem de diferenças quadráticas entre a imagem global e o template transformado $I_o(\gamma)$
2. Calcule a menor diferença $D_{min}(\gamma)$
3. Repita 1 e 2 para diferentes valores de γ
4. Calcule o valor de γ que leva ao menor valor de $D_{min}(\gamma)$. Esse é o valor ideal de γ para transformar o template
5. Retorne a posição da menor diferença quadrática encontrada para o γ ideal

Projeto 2 – Tema 4

Imagem global

Template

Transformação lei de potência

Template transformado

Projeto 2 – Tema 5

Construção da pirâmide Laplaciana

- Faça um programa que constrói a pirâmide Laplaciana da forma especificada na aula 7 (alguns slides acima)
- Verifique o que acontece com a reconstrução se alguns níveis específicos da pirâmide forem apagados (associar valor 0 a todos os pixels).
- Verifique o que acontece se as pirâmides de duas imagens forem misturadas (a imagem no topo de uma pirâmide ser usada com as imagens diferenças de outra pirâmide)

Projeto 2 – Tema 6

- Implemente a técnica de template matching na pirâmide gaussiana
- O programa calcula as diferenças quadráticas entre cada nível da pirâmide e a imagem template. O resultado para cada nível é plotado.
- O programa então identifica o menor valor de diferença entre todos os níveis da pirâmide.
- Importante plotar a diferença em função da escala: