

# **Estudo de comparação do descritor de imagens coloridas BIC empregando diferentes abordagens de classificação de detecção de bordas: Canny e Operador Laplaciano**

Diego Martin Mancini

Orientador: Prof. Paulo A. V. de  
Miranda

Instituto de Matemática e Estatística  
Universidade de São Paulo

# Descritores

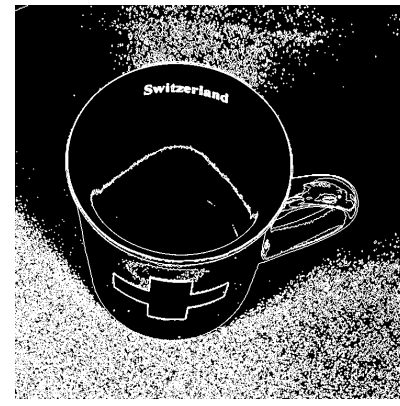
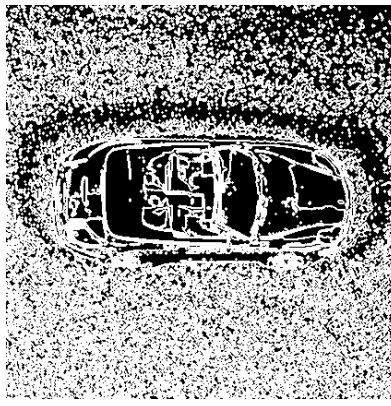
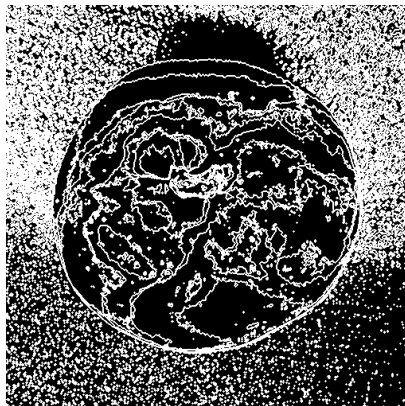
- Um descritor de imagem é definido como uma metodologia para comparar as imagens, usando um método computacional para extrair características das imagem.
- A metodologia de comparar as imagens pode ser, por exemplo, uma função de distância entre os histogramas das imagens.
- A extração de característica pode ser uma extração de cor.

# BIC

Proposto por Stehling, esse descritor de cores consiste em três passos:

- 1) Quantiza uniformemente o espaço de cores RGB em cores de  $4 \times 4 \times 4 = 64$  níveis (bins).
- 2) Classifica cada pixel da imagem em interior ou borda.
- 3) São construídos dois histogramas (um para interior e outro para borda) que são concatenados e dão origem a um vetor de extração de características de cor.

# Imagens geradas pelo BIC



Imagens geradas na 2º etapa do BIC.

# BIC

---

## Algorithm 1 BIC

---

### Entrada:

*img* é a imagem de tamanho  $n \times m$ .

### Saída:

o histograma de características de cor (vetor).

**função** BIC(*img*)

*/\*prop* é o vetor que conterà as cores em 64 níveis e a classificação de interior e borda.*\*/*

*Properties prop;*

*prop*  $\leftarrow$  *quantiza\_pixels(img)* */\*Quantifica as cores em 64 níveis (4x4x4).\*/*

*prop*  $\leftarrow$  *borda\_interior(prop)* */\*Classifica os pixels em interior e borda.\*/*

*vetor*  $\leftarrow$  *cria\_histograma(prop)* */\*Cria o histograma de interior/borda.\*/*

**devolve** *vetor*

**fim função**

---

# Proposta do Trabalho

- Algumas soluções adotadas nas diferentes etapas do BIC não consideram algoritmos clássicos de processamento de imagens.
- Este trabalho tem como proposta realizar essa comparação com os algoritmos *mainstream* e testar, se de fato, o algoritmo BIC é tão eficaz quanto sua versão modificada.
- Foram utilizados os algoritmos de Detector de Bordas de Canny e Operador Laplaciano.

# Variações do BIC

- Consiste em substituir o algoritmo de detecção de interior/borda do BIC pelos algoritmos de detecção de bordas de Canny e o Operador Laplaciano.
- Foram utilizados uma janela (kernel) de 3x3 e o mesmo limiar (threshold) nos algoritmos de Canny e o Laplaciano.

# Variações do BIC

---

## Algorithm 1 BIC

---

### Entrada:

*img* é a imagem de tamanho  $n \times m$ .

### Saída:

o histograma de características de cor (vetor).

**função** BIC(*img*)

*/\*prop* é o vetor que conterà as cores em 64 níveis e a classificação de interior e borda.*\*/*

*Properties prop;*

*prop*  $\leftarrow$  *quantiza\_pixels(img)*

*prop*  $\leftarrow$  *borda\_interior(prop)* */\* Serão substituídos AQUI \*/*

*vetor*  $\leftarrow$  *cria\_histograma(prop)*

**devolve** *vetor*

**fim função**

---



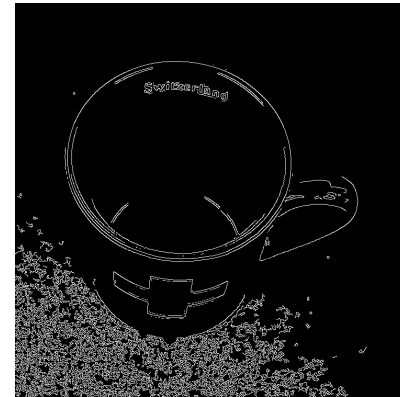
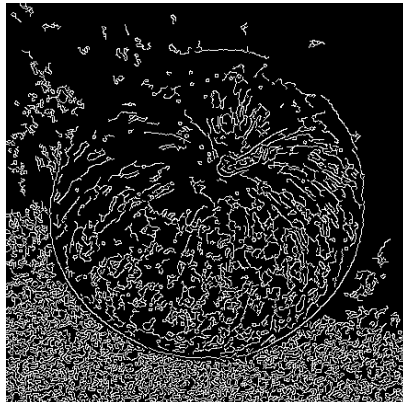
# Canny

- O detector de bordas de Canny segue três critérios: Boa detecção, boa localização e todas as bordas devem ser detectadas uma única vez.
- Aplica o filtro gaussiano para eliminar os ruídos.
- Utiliza o gradiente análogo ao de Sobel, usando dois limiares (threshold) superior e inferior no gradiente para verificar se o pixel é interior ou borda.
- Para ser considerado interior ou borda, deverá seguir os seguintes três critérios:

# Canny

- O pixel é borda se valor do gradiente for maior que o limiar superior.
- Se o valor do gradiente do pixel é abaixo do limiar inferior, então o pixel é rejeitado (interior).
- Se o gradiente do pixel está entre os dois thresholds, então este pixel é aceito com borda se estiver conectado a um pixel que estiver acima do limiar superior.

# Imagens geradas pelo Canny

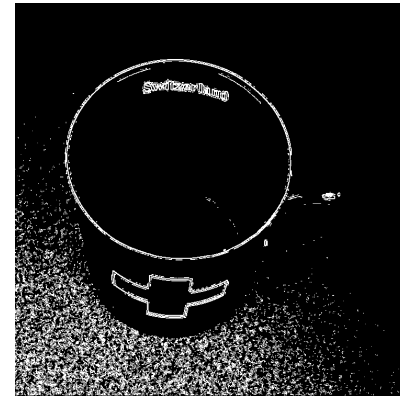
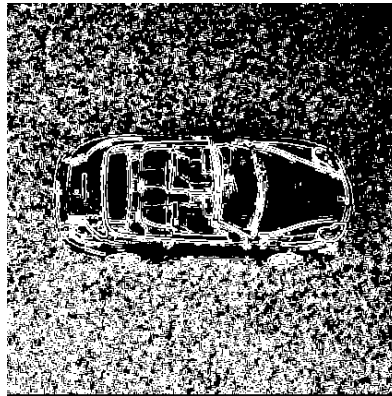
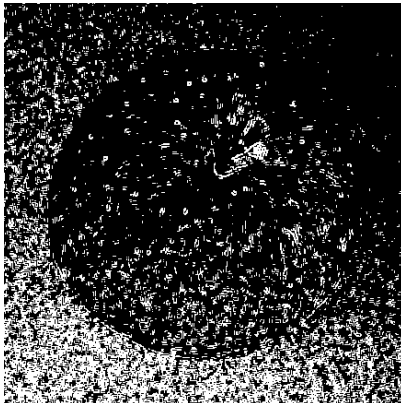


Imagens geradas na 2º etapa do BIC.

# Operador Laplaciano

- Operador Laplaciano utiliza o operador de Sobel para classificar a intensidade do pixel.
- Por definição, o Sobel utiliza duas derivadas e somente a segunda derivada será utilizada para detectar de maneira única se o pixel é interior ou borda.
- O zero crossing será usado para detectar as bordas, utilizando uma janela (kernel) 3x3 e um limiar para detectar valores mínimos e máximos (polaridades).

# Imagens geradas pelo Laplaciano



Imagens geradas na 2º etapa do BIC.

# Experimento

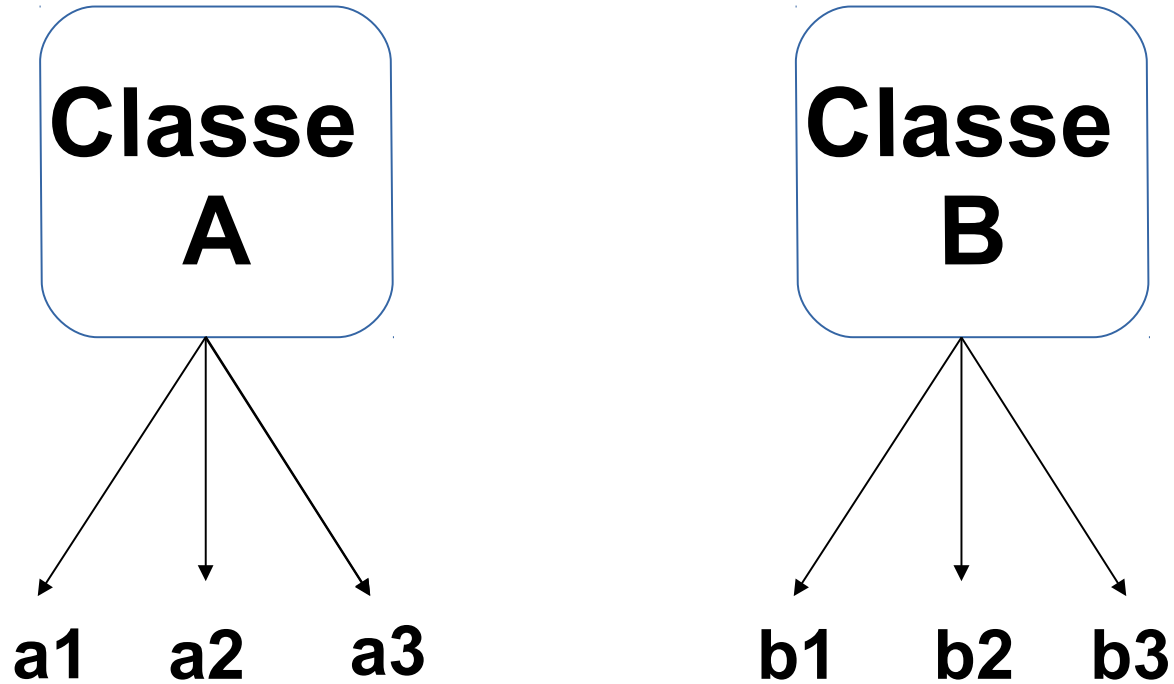
- Foram usado mais de 2200 imagens do site público ETH, com 54 classes, cada uma com 41 imagens.
- Utilizando os descritores, cada imagem é comparada com **todas** as imagens de **todas** as classes (inclusive a própria classe), usando a **distância dLog**.
- Calcula-se a curva da média dos gráficos de **precision x recall**, utilizando cada imagem da base como imagem de consulta.

# Esperimento

**Classe  
A**

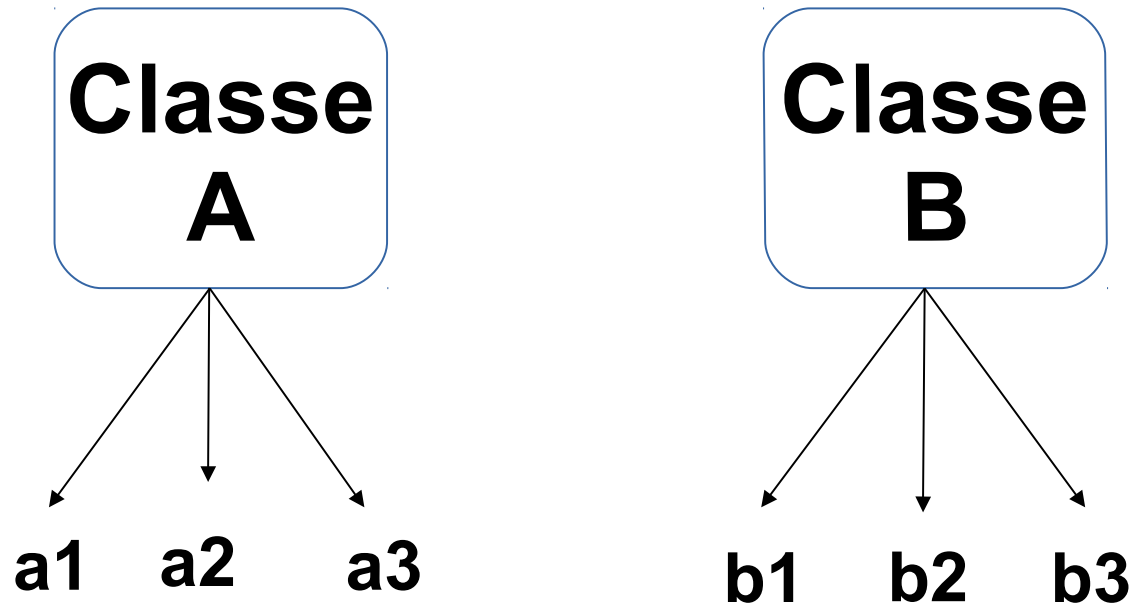
**Classe  
B**

# Esperimento





# Experimento



$d\text{Log}(a1, a1) = 0$   
 $d\text{Log}(a1, a2) = 0,6$   
 $d\text{Log}(a1, a3) = 0,3$   
 $d\text{Log}(a1, b1) = 0,2$   
 $d\text{Log}(a1, b2) = 0,8$   
 $d\text{Log}(a1, b3) = 0,4$

**a1: a1, b1, a3, b3, a2, b2**

# Experimento

$n_a$  = # de acertos da classe

$n_i$  = # de imagens até a  $i$ -ésima comparação

$N$  = # de imagens por classe

**Precision** =  $n_a / n_i$

**Recall** =  $n_a / N$

# Experimento

$n_a$  = # de acertos da classe

$n_i$  = # de imagens até a  $i$ -ésima comparação

$N$  = # de imagens por classe

**Precision** =  $n_a / n_i$

**Recall** =  $n_a / N$

a1: a1, b1, a3, a2, b3, b2

a1:  $n_a = 1, n_i = 1 \Rightarrow$  Precision =  $1/1$ , Recall =  $1/3$

b1:  $n_a = 1, n_i = 2 \Rightarrow$  Precision =  $1/2$ , Recall =  $1/3$

a3:  $n_a = 2, n_i = 3 \Rightarrow$  Precision =  $2/3$ , Recall =  $2/3$

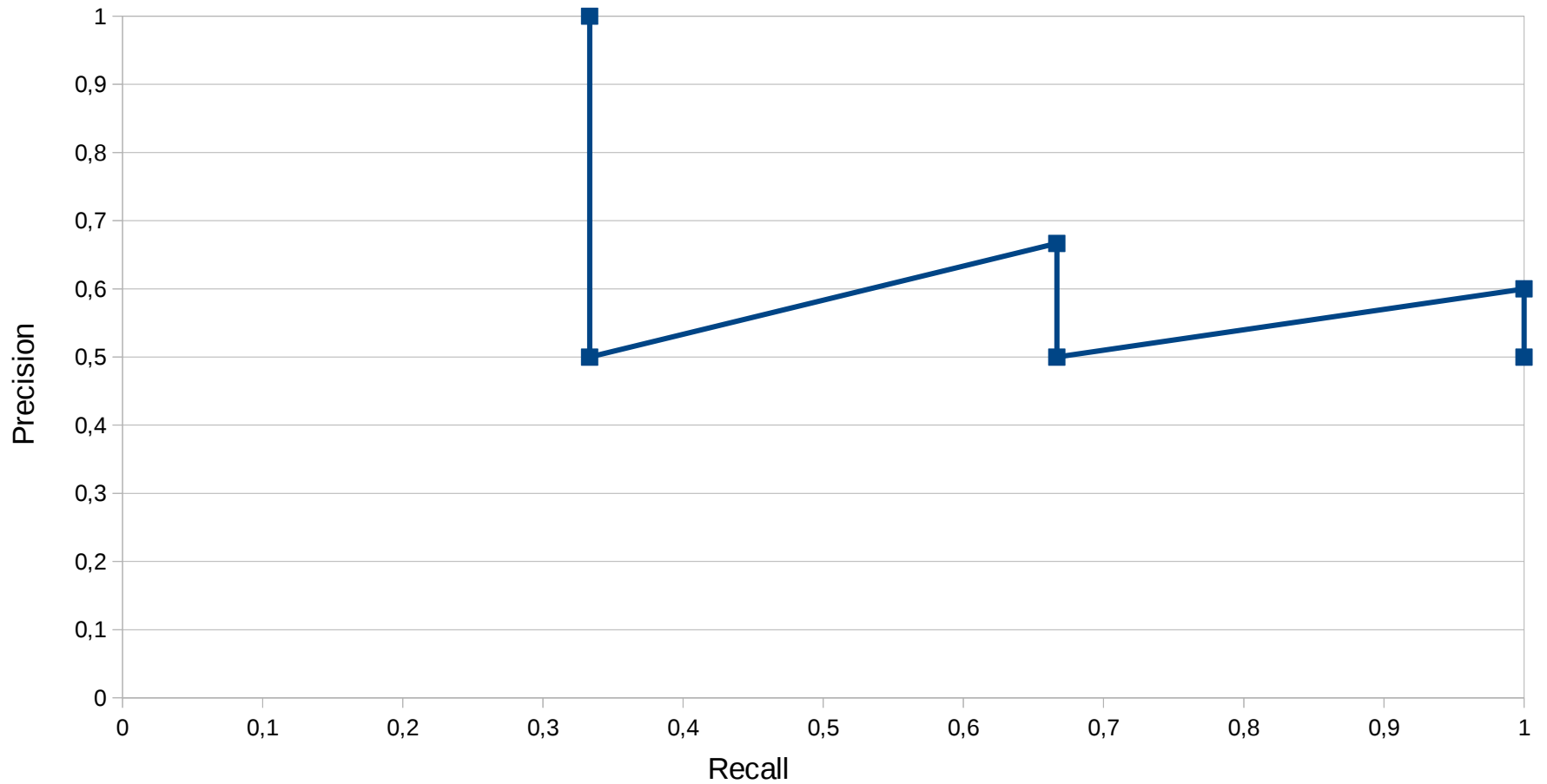
a2:  $n_a = 3, n_i = 4 \Rightarrow$  Precision =  $3/4$ , Recall =  $3/3$

b3:  $n_a = 3, n_i = 5 \Rightarrow$  Precision =  $3/5$ , Recall =  $3/3$

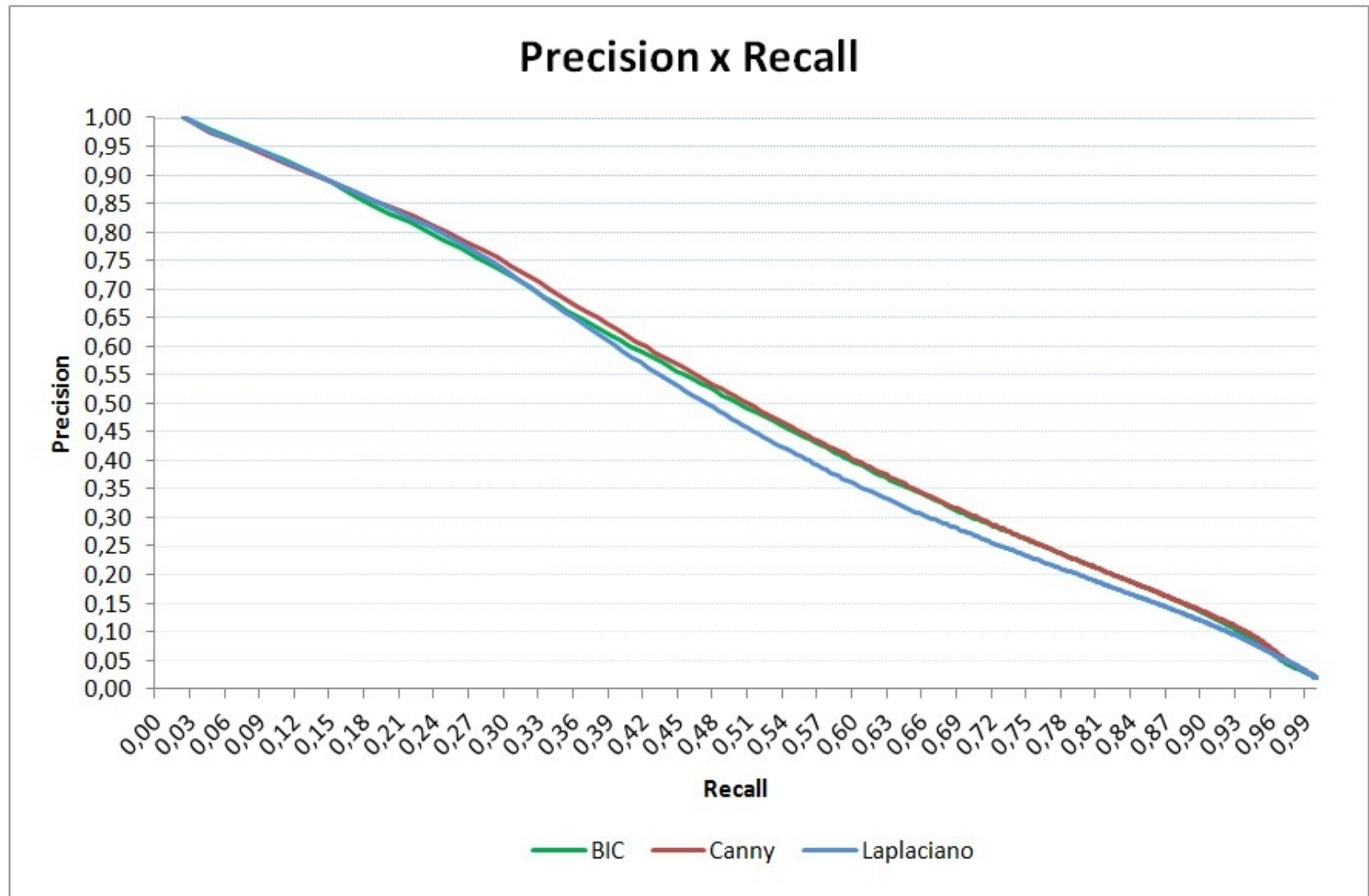
b2:  $n_a = 3, n_i = 6 \Rightarrow$  Precision =  $3/6$ , Recall =  $3/3$

# Experimento

Precision x Recall



# Resultado



# Conclusão

- Os resultados das variações do BIC/Canny e BIC/Laplaciano foram semelhantes ao descritor BIC original.
- O BIC/Laplaciano demonstrou uma menor acurácia do que o BIC/Canny e BIC.

# Trabalhos Futuros

- Usar o conceito de detecção de bordas Fuzzy.
- No experimento, foram utilizados somente um limiar para o Canny e o Laplaciano, podendo ser testados mais variações de limiares para testar acurácia.
- Estudar a eficiência do BIC em termos de velocidade de processamento.
- Aumentar o número de imagens para o experimento.

# Referências

- R. O. Stehling, M. A. Nascimento, and A. X. Falcão. A compact and efficient image retrieval approach based on border/interior pixel classification. CIKM '02 Proceedings of the eleventh international conference on Information and knowledge management, 2002.
- Wikipedia, [https://en.wikipedia.org/wiki/Canny\\_edge\\_detector](https://en.wikipedia.org/wiki/Canny_edge_detector)> Acessado em 08-2015.
- R. F. Gonzalez and R. E. Woods. Digital image processing. Pearson PrenticeHall, 2008.
- Zero crossing laplacian operator, <http://fourier.eng.hmc.edu/e161/lectures/gradient/node7.html>. Acessado em 10-2015



**Obrigado!**