CAPÍTULO 1

# *Encontrando a Linha Divisória: Detecção de Bordas*

**Contribuíram:** *Daniela Marta Seara, Geovani Cássia da Silva Espezim Elizandro*

## Encontrar Bordas também é Segmentar

A visão computacional envolve a identificação e classificação de objetos em uma imagem, desta forma a detecção de bordas é uma ferramenta essencial no processo de análise de imagens. Este trabalho tem como objetivo dar uma breve introdução sobre os conceitos que envolvem o processo de detecção de bordas. A importância de tal processo é devido ao fato de que ao se detectar uma borda podemos localizar todos os objetos de uma imagem definindo desta forma, propriedades como área, perímetro e forma.

### Detecção de Imagens

O primeiro passo na análise de imagens é a separação ou a segmentação dos objetos dentro da imagem. Algoritmos de segmentação permitem achar diferenças entre dois ou mais objetos. A segmentação é baseada em dois conceitos: similaridade e descontinuidade. Na segmentação procura-se distinguir as partículas umas das outras e do fundo. Esta distinção permitirá ao programa interpretar pixels contíguos e agrupá-los em regiões. Não existe um modelo formal para a segmentação, o processo é essencialmente empírico e deverá se ajustar a diferentes tipos de imagem. Esta etapa é a mais difícil do processo e também a mais delicada porque todas as medidas serão realizadas sobre as regiões identificadas nesta etapa. A segmentação é tão complexa porque tenta

traduzir para o computador um processo cognitivo extremamente sofisticado realizado através da visão humana.

A descontinuidade em uma imagem pode ser:

- um ponto isolado;
- uma linha;
- a borda de uma objeto.

### Algoritmos de Segmentação

**a) Detecção de pontos:** é a mais simples técnica de detecção. Um ponto terá uma mudança drástica do valor de cinza em relação aos seus vizinhos.

**b) Detecção de linhas:** é o processo mais complicado, pois é necessário achar os pixels que são semelhantes e testá-los para verificar se são parte de uma linha comum.

**c) Detecção de bordas:** é uma das técnicas básicas utilizadas pela visão humana no reconhecimento de objetos. É o processo de localização e realce dos pixels de borda, aumentado o contraste entre a borda e o fundo. Este processo verifica a variação dos valores de luminosidade de uma imagem. Como foi abordado anteriormente, neste trabalho daremos ênfase apenas para os algoritmos de segmentação para detecção de bordas.

## O que é uma Borda?

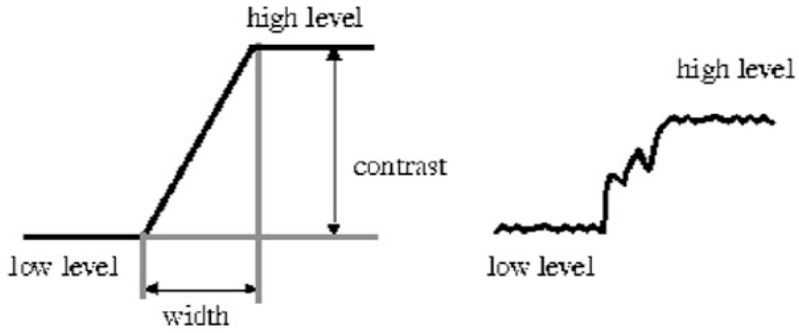
É o *contorno* entre um objeto e o fundo indicando o limite entre objetos sobrepostos Um *CONTORNO* é uma linha fechada formada pelas bordas de um objeto. Mas como conceituamos e detectamos uma borda computacionalmente ?

Variações de intensidade complexas que ocorrem em uma região são geralmente chamadas de textura. Bordas são definidas como picos da magnitude do gradiente, ou seja, são variações abruptas que ocorrem ao longo de curvas baseadas nos valores do gradiente da imagem. As bordas são regiões da imagem onde ocorre uma mudança de intensidade

em um certo intervalo do espaço, em uma certa direção. Isto corresponde a regiões de alta derivada espacial, que contém alta frequência espacial.

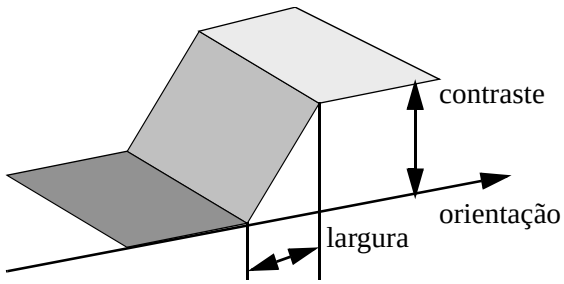
- **Borda unidimensional :** pode ser definida como uma mudança de uma intensidade baixa para alta. A intensidade do sinal pode ser interferida por ruídos.

**Figura 1.1.** **Gráfico da intensidade de uma imagem**




- **Borda bidimensional**: as descontinuidades ocorrem ao longo de certas linhas ou orientações A orientação é uma característica importante em bordas 2-D

**Figura 1.2.** **Gráfico da intensidade representando uma borda bidimensional**




## Ruídos

Toda aquisição da imagem está sujeita a algum tipo de ruído. A situação ideal, sem ruído, na prática não existe. Ruídos não podem ser previstos pois são de natureza randômica e não podem nem mesmo ser medidos precisamente. Porém, algumas vezes ele pode ser caracterizado pelo

efeito na imagem, e é geralmente expresso como uma distribuição de probabilidade com uma média específica e um desvio padrão. Existem dois tipos de ruídos específicos:

- ***Ruído independente do sinal:*** é um conjunto randômico de níveis de cinza, estatisticamente independente dos dados da imagem. Este tipo de ruído acontece quando a imagem é transmitida eletronicamente de um lugar para outro.
- **A** - imagem perfeita
- **N** - é o ruído
- **B** - imagem final
- **B**= A+N

- ***Ruído de sinal dependente****:* o nível do valor de ruído a cada ponto na imagem é uma função do nível de cinza.

## Operadores para detecção de bordas que utilizam derivadas

Desde que uma borda é definida por uma mudança no nível de cinza, quando ocorre uma denscontinuidade na intensidade, ou quando o grandiente da imagem tem uma variação abrupta, um operador que é sensível a estas mudanças operará como um detector de bordas.

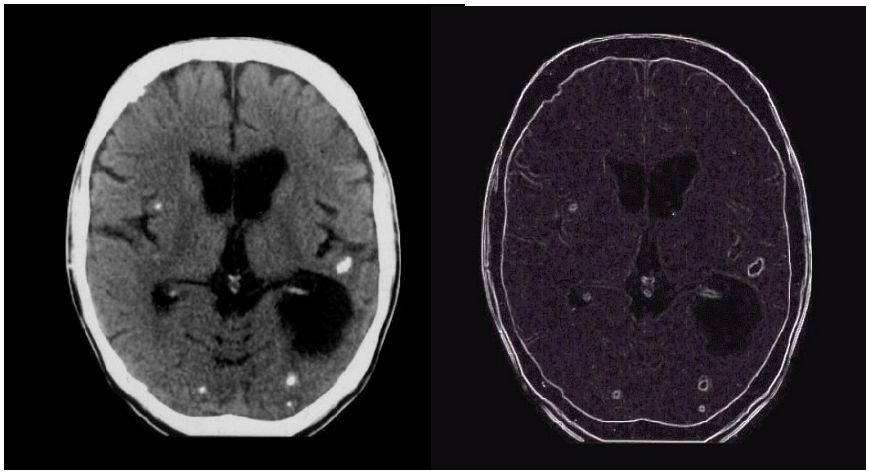
Um operador de derivada faz exatamente esta função. Uma interpretação de uma derivada seria a taxa de mudança de uma função, e a taxa de mudança dos níveis de cinza em uma imagem é maior perto das bordas e menor em áreas constantes. Se pegarmos os valores da intensidade da imagem e acharmos os pontos onde a derivada é um ponto de máximo, nós teremos marcado suas bordas. Dado que as imagens são em duas dimensões, é importante considerar mudanças nos níveis de cinza em muitas direções. Por esta razão, derivadas parciais das imagens são usadas, com as respectivas direções X e Y. Uma estimativa da direção atual da borda pode ser obtida usando as derivadas x e y

como os componentes da direção ao longo dos eixos, e computar o vetor soma. O operador envolvido é o gradiente, e se a imagem é vista como uma função de duas variáveis A(x,y) então o gradiente é definido como:

Fórumula 1:

$$\nabla A(x,y) = \frac{\partial A}{\partial x}, \frac{\partial A}{\partial y}$$

**Figura 1.3.** **Exemplo de uma imagem de tomografia e sua respectiva imagem de gradientes**



## Máscaras para Algoritmos de Segmentação

A maioria dos algoritmos de segmentação utilizam uma máscara sobre os pixels da imagem para detecção de uma descontinuidade. Cada pixel e seus pixels vizinhos tem um valor do nível de cinza multiplicados por uma constante. A soma destes valores representa a máscara de resposta daquele ponto.

Fórumula 2: **Exemplo de uma máscara de resposta**

$$\begin{bmatrix} m_1 & m_2 & m_3 \\ m_4 & m_5 & m_6 \\ m_7 & m_8 & m_9 \end{bmatrix} \cdot \begin{bmatrix} p_1 & p_2 & p_3 \\ p_4 & p_5 & p_6 \\ p_7 & p_8 & p_9 \end{bmatrix} = m_1p_1 + m_2p_2 + \ldots + m_9p_9 = \sum_{i=1}^{9} m_ip_i$$

Onde:

$\boldsymbol{m_i}$: é valor para um pixel

$\boldsymbol{p_i}$: é o valor do nível de cinza para um pixel

***R***: é a máscara de resposta para o pixel central ($p_5$).

Ampliando esta máscara atravessando a imagem linha por linha, um novo vetor é criado. É o mesmo que a imagem original mais contém os valores da máscara de resposta ao invés do valor do pixel. Estes valores da máscara de resposta podem então ser comparados com o valor mínimo de *threshold* para determinar quais pixels são mais prováveis de ser parte de uma borda. Este *threshold* pode ser ajustado para variar seletivamente de acordo com os pixels de borda, permitindo um usuário conduzir o algoritmo para um melhor desempenho para uma imagem específica.

Considerando o exemplo da máscara de entrada anterior e supondo um valor de threshold 20. A matriz 3x3 representa uma parte de uma determinada imagem, desde que os valores de saída que cancelem o valor central, não ultrapassem o valor de threshold, o que significa que a imagem não apresenta descontinuidade.

Analisando a máscara de resposta para o conjunto contínuo de pixels dados na Figura 1.4., percebemos que neste caso o valor central da máscara é 16, e a soma de todos os pixels subtraída do valor central fornece zero como resposta. Como zero é um valor menor do que o

**Figura 1.4.** **Exemplo de uma máscara**

| máscara de detecção de ponto | imagem | resposta da máscara |
|---|---|---|
| $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$ | $\begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix}$ | $\begin{bmatrix} -2 & -2 & -2 \\ -2 & 16 & -2 \\ -2 & -2 & -2 \end{bmatrix}$ |

threshold (20), o pixel central na imagem não é um ponto de descontinuidade.

**Figura 1.5.** **Outro exemplo de máscara**

| máscara de detecção de ponto | imagem | resposta da máscara |
|---|---|---|
| $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$ | $\begin{bmatrix} 2 & 2 & 2 \\ 2 & 8 & 2 \\ 2 & 2 & 2 \end{bmatrix}$ | $\begin{bmatrix} -2 & -2 & -2 \\ -2 & 64 & -2 \\ -2 & -2 & -2 \end{bmatrix}$ |

Neste caso (Figura 1.5.) o valor central da máscara é 64, a soma de todos os pixels subtraída do valor central fornece um valor de 48 que é maior do 20 (threshold), o pixel central na imagem é um ponto de descontinuidade, portanto seria um pixel de borda.

# Algoritmos de Detecção de Bordas

Os algoritmos para detecção de bordas que serão estudados no presente trabalho são:

- **Roberts**
- **Sobel**

- **Robinson**
- **Canny**
- **Marr-Hildreth**

## Operador de Roberts

É o mais antigo e simples algoritmos de detecção de bordas. Utiliza uma matriz 2x2 para encontrar as mudanças nas direções x e y.

**Figura 1.6.** **Máscara de Roberts**

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

$G_x$ $G_y$

Para determinar onde o pixel avaliado é ou não um pixel de borda, o gradiente é calculado da seguinte forma:

Fórumula 3:

$$|G| = \sqrt{G_x^2 + G_y^2}$$

Se a magnitude calculada é maior do que o menor valor de entrada (definido de acordo com a natureza e qualidade da imagem que esta sendo processada), o pixel é considerado ser parte de um borda. A direção do gradiente da borda, perpendicular a direção da borda, é encontrada com a seguinte fórmula:

Fórumula 4:

$$\alpha = \operatorname{atan}\left(\frac{G_y}{G_x}\right)$$

O pequeno tamanho da máscara para o operador de Roberts torna o mesmo fácil de se implementar e rápido para calcular a máscara de resposta. As respostas são muito sensíveis ao ruído na imagem.

## Operador de Sobel

Utiliza duas máscaras para encontrar os gradientes vertical e horizontal das bordas.

**Figura 1.7.** **Máscaras de Sobel**

$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

$G_y$ $G_x$

### Considerações sobre o operador de Sobel

- A fórmula para encontrar o gradiente e o ângulo são as mesmas do operador de Roberts.
- O ângulo do gradiente corresponde direção de máxima variação da intensidade
- Devido as máscaras serem 3X3 ao invés de 2X2, Sobel é muito menos sensível ao ruído do que Roberts.
- Os resultados são mais precisos.
- A computação de |G| se torna mais complexa. Na prática |G| é aproximada da seguinte forma: |G|= |Gx| + |Gy|.
- O módulo do gradiente é proporcional a derivada local da intensidade.

## Operador de Robinson

É similar em operação ao de Sobel, porém usa um conjunto de oito máscaras, onde quatro delas são as seguintes:

**Figura 1.8.** **Máscaras de Robinson**

$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad \begin{bmatrix} 2 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -2 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \quad \begin{bmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{bmatrix}$$

As outras quatro são simplesmente negações destas quatro. A magnitude do gradiente é o valor máximo obtido ao aplicar todas as oito máscaras ao pixel vizinho, e o ângulo do gradiente pode ser aproximado como o ângulo na linha de zeros na máscara dando a resposta máxima. Este algoritmo aumenta a precisão de |G|, mas requer mais computação do que Roberts e Sobel, devido ao tamanho das máscaras.

## Detector de bordas de Canny

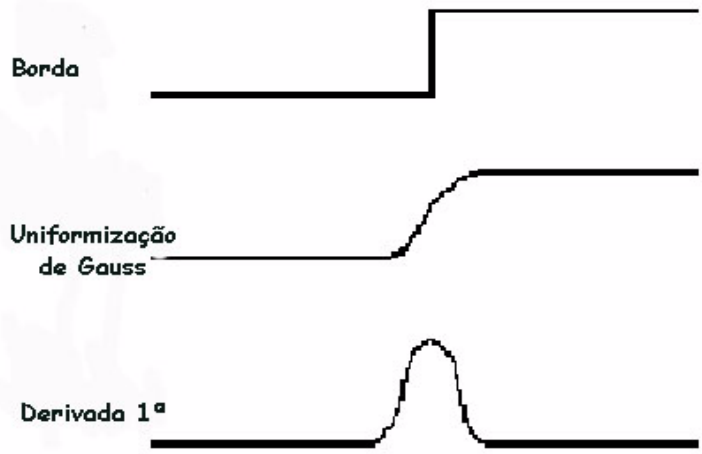
Na criação do algoritmo canny, definiu-se um conjunto de requisitos que um detector de bordas deveria satisfazer. São eles:

1. **Taxa de erro:** o detector de bordas deveria detectar e achar somente bordas, nenhuma borda deveria faltar;
2. **Localização:** a distância entre os pixels de borda encontradas pelo detector de bordas e a borda atual deveriam ser o menor possível;
3. **Resposta:** o detector de bordas não deveria identificar múltiplos pixels de borda onde somente exista um único pixel.

O detector de bordas de Canny é um filtro de convolução *f* que uniformizaria o ruído e localizaria as bordas. O problema é identificar um

filtro que otimize os três critérios do detector de bordas. Se considerarmos uma borda de uma dimensão variando no contraste e então convolucionando a borda com a função de uniformização de Gauss, o resultado será uma variação contínua do valor inicial ao final, com uma inclinação máxima no ponto onde existe um "degrau". Se esta continuidade é diferenciada em relação a x, esta inclinação máxima será o máximo da nova função em relação a original.

**Figura 1.9.** **Detecção de bordas por Canny**




Os máximos da convolução da máscara e da imagem indicarão bordas na imagem. Este processo pode ser realizado através do uso de uma função de Gauss de 2-Dimensões na direção de x e y. Os valores das máscaras de Gauss dependem da escolha do sigma na equação de Gauss:

Fórumula 5:

$$G(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{-x}{2\sigma}}$$

$$G'(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{-x}{2\sigma}}$$

A aproximação do filtro de Canny para detecção de bordas é G'. Convolucionando a imagem com G' obtemos uma imagem *I* que mostrará as bordas, mesmo na presença de ruído. A convolução é relativamente simples de ser implementada, mas é cara computacionalmente, especialmente se for em 2-dimensões. Entretanto,uma convolução de Gauss de 2-dimensões pode ser separada em duas convoluções de Gauss de 1-dimensão.

A intensidade computacional do detector de bordas de Canny é relativamente alta, e os resultados são geralmente pós-processados para maior clareza. Entretanto, o algoritmo é mais eficiente no processamento de imagens com ruídos ou com bordas difusas.

### Algoritmo de Canny

1. Ler a imagem I a ser processada;
2. Criar uma máscara de Gauss de 1-D G para convolucionar I. O desvio S de Gauss é um parâmetro para o detector de bordas;
3. Criar uma máscara de 1-D para a primeira derivada de Gauss nas direções x e y; nomear como Gx e Gy. O mesmo valor S é usado;
4. Convolucionar a imagem I com G percorrendo as linhas na direção x (Ix) e percorrer as colunas na direção y (Iy);
5. Convolucionar Ix, com Gx, para dar I´x, o componente x e de I convolucionado com a derivada de Gauss, e convolucionar Iy, com Gy para dar I´y;
6. Se você deseja ver o resultado neste ponto os componentes x e y devem ser " combinados". A magnitude do resultado é computada para cada pixel (x,y).

## Operador de Marr-Hildreth

Um máximo da derivada primeira ocorrerá a cada zero crossing da derivada Segunda. Para localizar bordas horizontais e verticais procuramos a derivada Segunda nas direções de x e y. Isto é Laplacian de I:

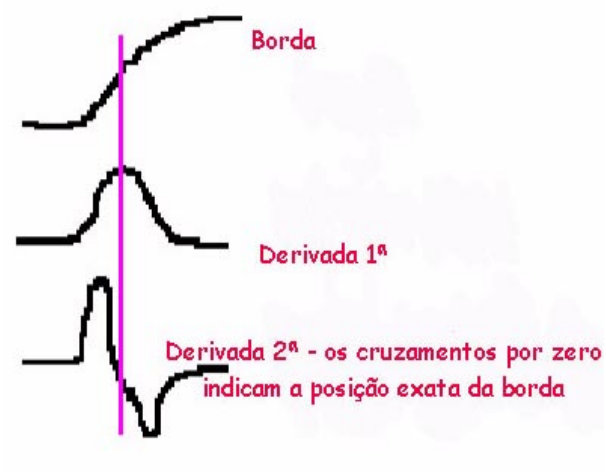
Fórumula 6:

$$\nabla^2 I = \frac{\partial^2 I}{\partial^2 x}, \frac{\partial^2 I}{\partial^2 y}$$

- Laplaciano

O Laplaciano é linear e simétrico rotacionalmente. Podemos procuar pelos zeros crossings da imagem primeiro uniformizando com uma máscara de Gauss e então calculando a derivada Segunda. Isto define o operador de Marr-Hildreth.

**Figura 1.10.** **Zero Crossing**




O operador de Marr-Hildreth tornou-se largamente utilizado devido:

- a reputação de Marr;
- pesquisadores encontraram campos nos olhos de animais que se comportavam de forma semelhante com este operador (*campos receptivos*);
- o operador é simétrico. Bordas são encontradas em todas as direções, diferente dos operadores que usam a primeira derivada (são unidirecionais).

# Considerações sobre os Operadores de Derivadas de Segunda Ordem

- Zero crossing da derivada segunda são mais simples de determinar do que o ponto de máximo em uma derivada de primeira ordem;
- Zero crossing sempre formam contornos fechados. Isto é bom quando estamos tentando separar objetos em uma imagem.
- No operador da Segunda derivada o ruído é considerado. Este método é muitas vezes usado em conjunção com limiarização normal para reduzir sua sensibilidade a ruído;
- Gerar sempre contornos fechados não realista;
- Marca bordas em algumas localizações que não são bordas.

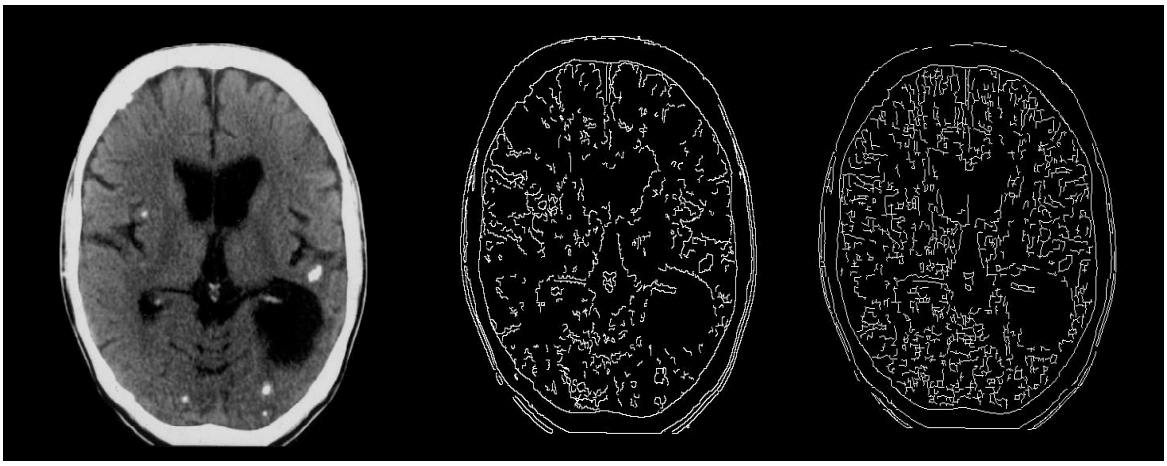
### Método de Marr-Hildreth

1. Borra-se a imagem com um kernel gaussiano;
2. Obtém-se o Laplaciano;
3. Localiza-se os cruzamentos por zero;
4. Os pontos localizados formam contornos fechados.

# Ferramentas de Software para Detecção de Bordas

O sistema Khoros [1.13][1.14] apresenta uma considerável coleção bons detetores de bordas. Entre eles estão operações de convolução que implementam os filtros: Roberts, Sobel, Robinson, Marr-Hildreth. Existe também um Toolbox especial que implementa o Canny. Além desses, o Khoros disponibiliza filtros de primeira (vdrf) e segunda (vsderf) derivada que realizam um pós-processamento dos resultados de detecção de bordas, apresentando uma imagem binária. Estes filtros possibilitam ao usuário estabelecer inclusive tamanhos mínimos para segmentos a serem detectados, etc.

Figura 1.11. **Imagem de tomografia computadorizada e resultados da aplicação do filtro de primeira derivada (vdrf) e do filtro de segunda derivada (vsderf) provenientes do Khoros.**



Além destes, existe uma excelente implementação do Canny no pacote VISTA [1.15].

# Referências


[1.1] http://www.dcmm.puc-rio.br/Cursos/IPDI/index.htm

[1.2] http://www.eps.ufsc.br/~martins/fuzzy/fuz_ap/image/image_gi.htm

[1.3] http://www.coder.com/creations/banner/examples/edge.html

[1.4] http://vinny.bridgeport.edu/MailArchives/cs411x-list/0029.html

[1.5] http://ccl1-dee.poliba.it/~cafforio/edge.html

[1.6] http://www.cclabs.missouri.edu/~c621269/blackkite/blackkite.html

[1.7] http://WWW.ISIP.MsState.Edu/resources/courses/ece_4773/projects/1997/group_image

[1.8] http://rvl1.ecn.purdue.edu/v1/student-software/ee661.97/teffeau/teffeau.html

[1.9] http://www-mrips.od.nih.gov/uguide/ug13.htm#4520

[1.10] http://www.bath.ac.uk/BUCS/Software/image_analysis/visilog/html/refguide/chap8.html

[1.11] http://lummi.stanford.edu/class/cs223b/WWW/oldnotes/notesEdges/node2.html

[1.12] http://rome.cis.plym.ac.uk/cis/cesar/hilbert/ov/over.html

[1.13] **The Khoros Group**; *Khoros User's Manual, Release 1.0.5*, Vols. I - III, Department of Electrical Engineering & Computer Engineering, University of New Mexico, Albuquerque, USA, 1993.

[1.14] www.khoral.com/

[1.15] **Pope, Arthur R., Lowe, David G.**; *VISTA: A Software Environment for Computer Vision Research*. Internal Report, Department of Computer Science, University of British Columbia, Canada, 1994