Universidad Nacional del Altiplano
Escuela de Posgrado
Doctorado en Ciencias de la Computación

Computer Vision
Introduction

Prof. Dr. Ivar Vargas Belizario

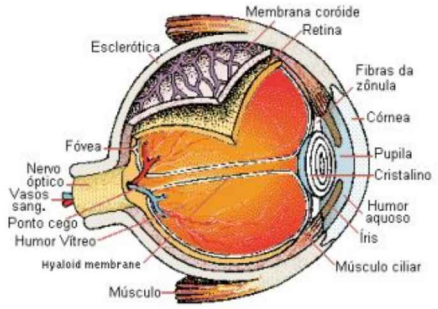
ivargasbelizario@gmail.com

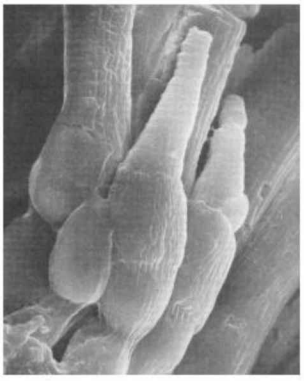
2024 - II

Visión computacional

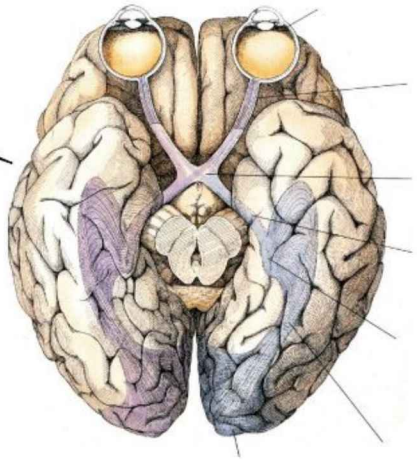
- Presentación:
  - **– Nombre?**
  - – Tema de tesis?
  - – Llevaron el curso?
  - **– Qué espera del curso de visión computacional?**

Visión Natural: sistema super paralelo

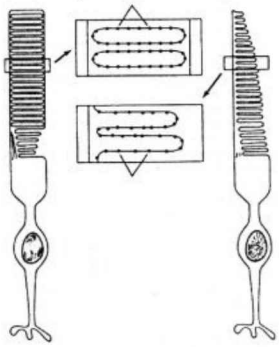




Macro

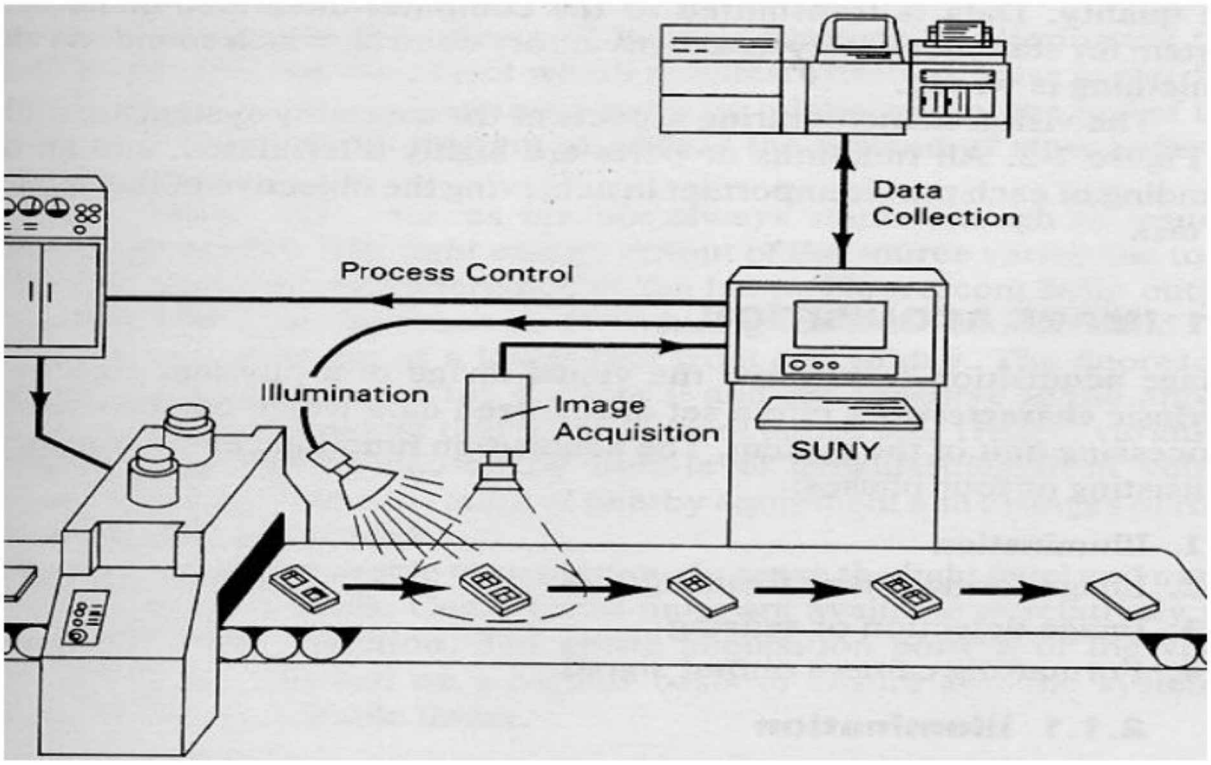


Micro



Intermediário

Un sistema de visión computacional para la industria



Visión computacional

- Visión
  - Es el proceso de descubrir, a partir de imágenes, lo que está presente en el mundo y donde está localizado

- Disciplinas relacionadas

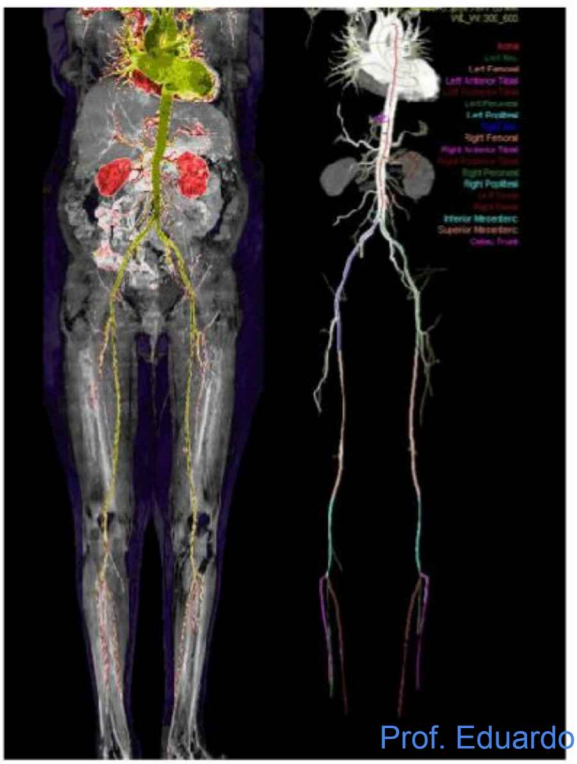
  - **Procesamiento de Imágenes**
  - Computacioń Gráfica
  - **Reconocimiento de Patrones**
  - Robótica
  - **Inteligencia artificial**

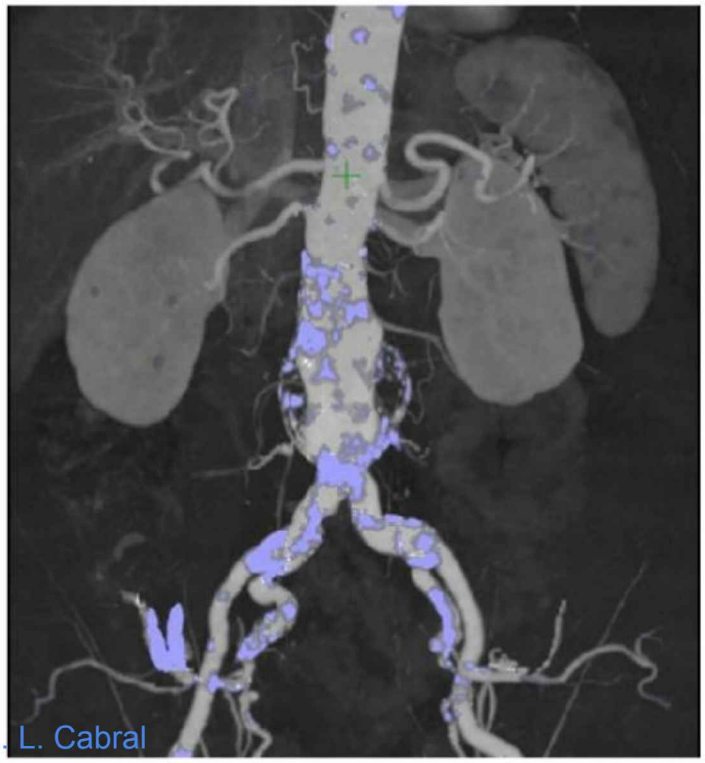
Aplicaciones

- Análisis de imagens médicas
- Seguridad: biometria, monitoramento, localización,
- reconhecimento de de objetivo
- Sensoriamento remoto
- Robótica
- Inspección e controle de calidad industrial
- Análisis de documentos
- Multimedia
- Asistencia a deficientes
- Interface humano-computador

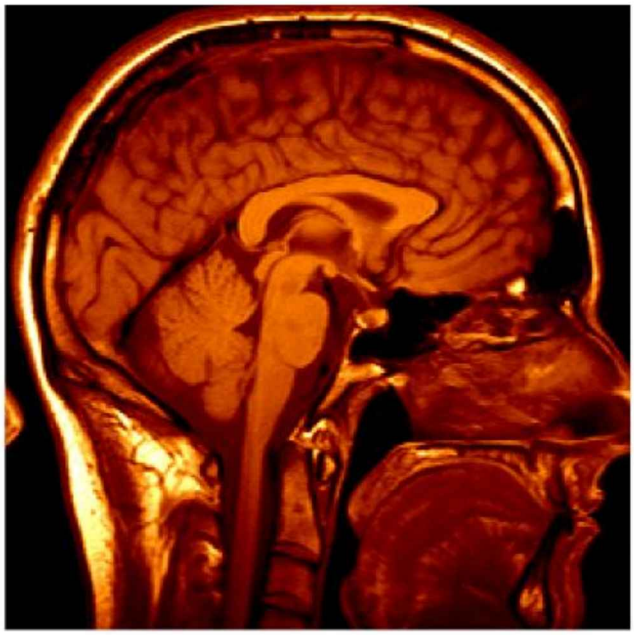
...

Análisis de imagens médicas





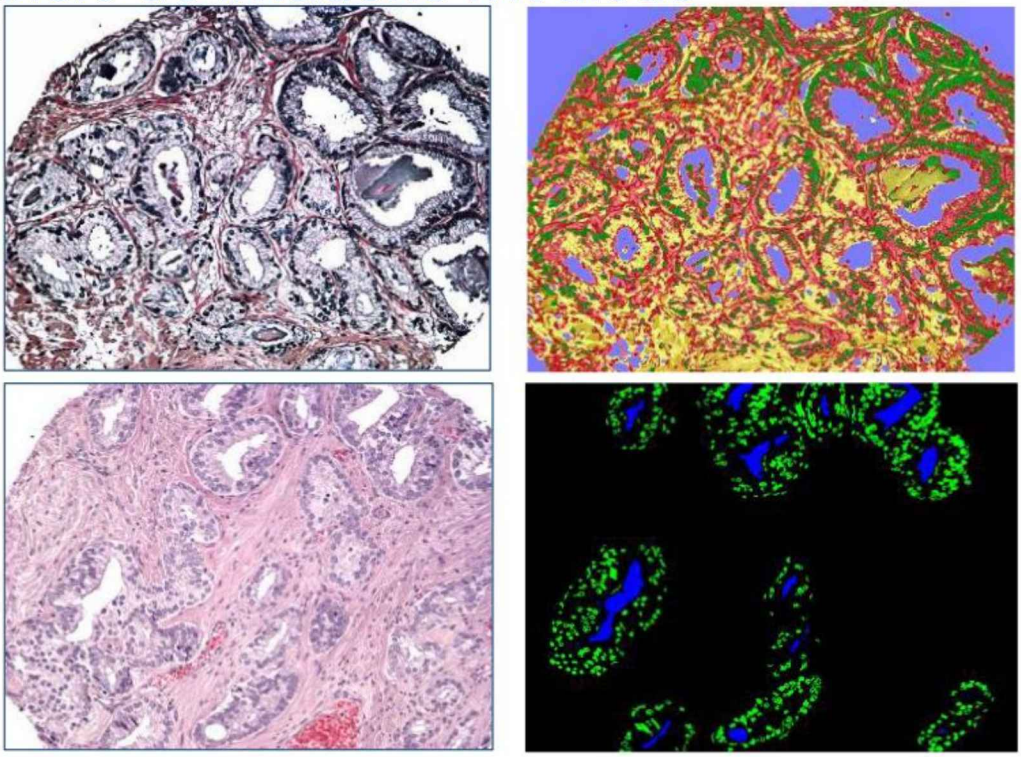
Análisis de imagens médicas



Imagens 3D: MRI, CT

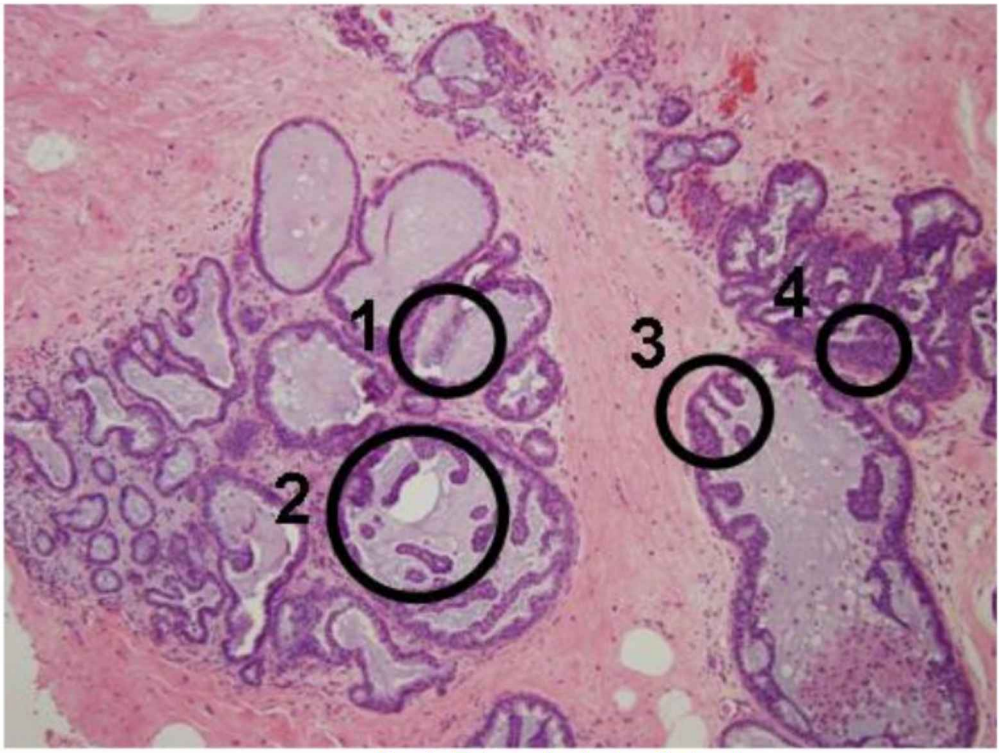
Cirúrgia guiada por imagem
(Grimson et al., MIT)

Análisis de imagens médicas



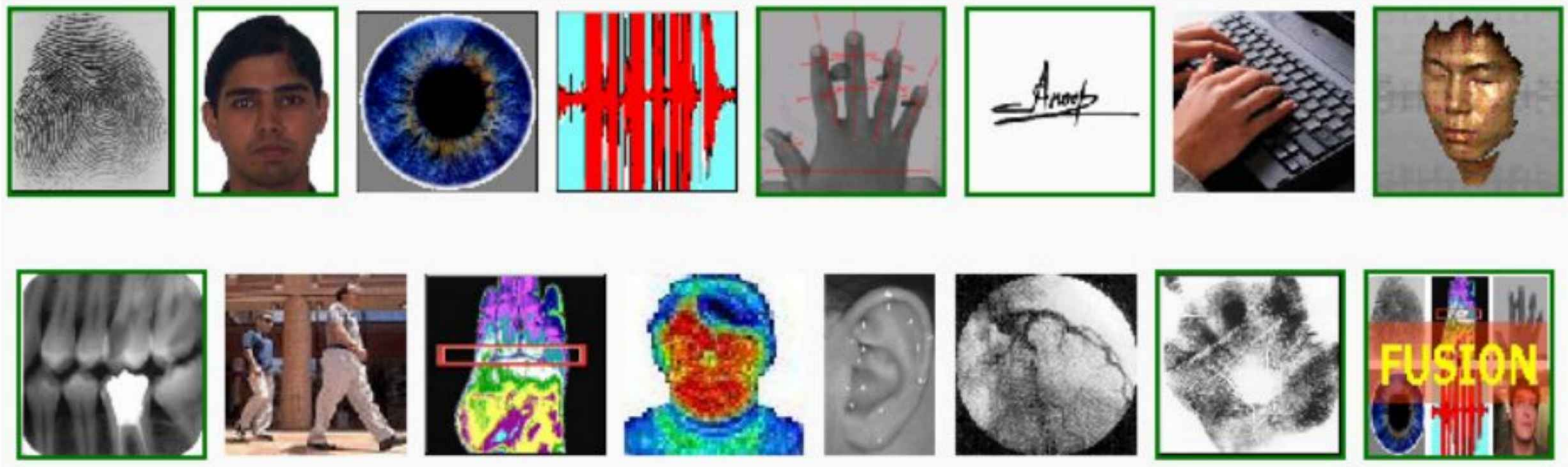
Detecção e classificação de cancer

Análisis de imagens médicas



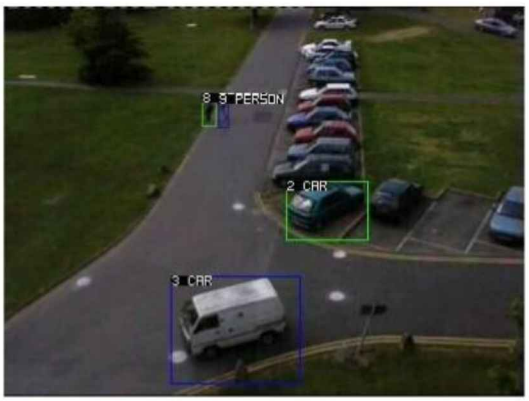
Imagens de biópsias: detecção de regiões de interesse

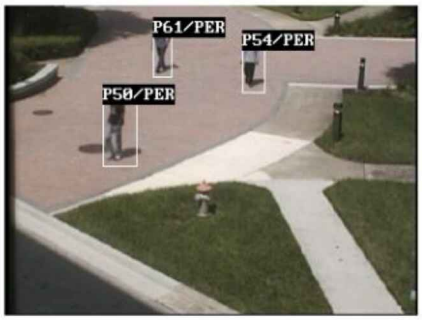
Biometría



De Anil Jain, Michigan State

Monitoreamiento





University of Central Florida, Computer Vision Lab

Monitoreamiento

De Martial Hebert, CMU

Seguridad de peatones y vehículos

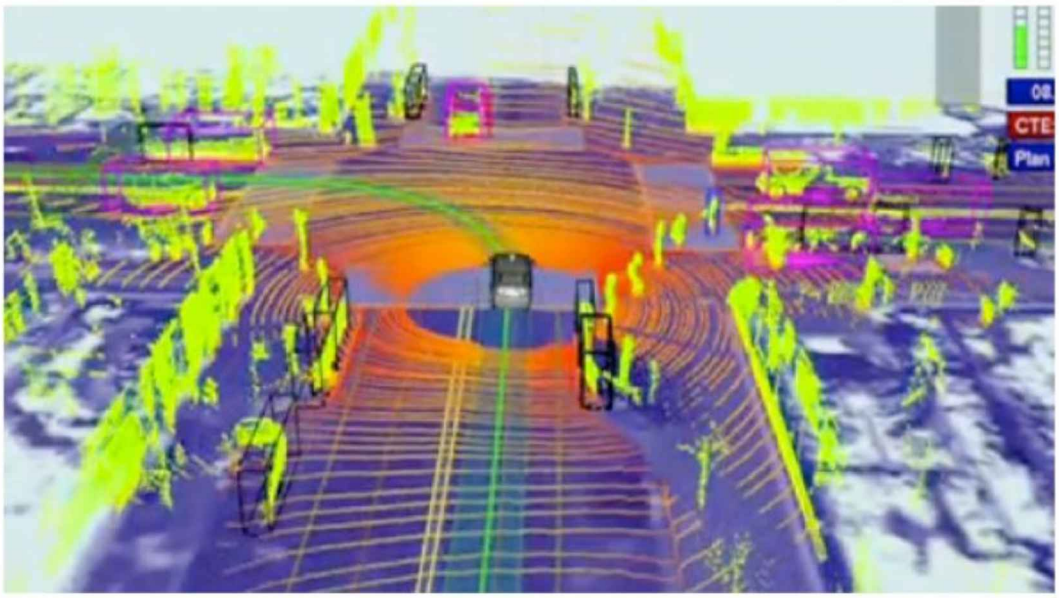
Aviso de saída da faixa, aviso de colisão, reconhecimento de sinal de trânsito, reconhecimento de pedestre, aviso de ponto cego

http://www.mobileye-vision.com

Automobiles inteligentes

http://www.darpa.mil/grandchallenge/index.asp
http://en.wikipedia.org/wiki/DARPA_Grand_Challenge

Automobiles inteligentes



"Our self-driving cars have now traveled nearly 200,000 miles on public highways in California and Nevada, 100 percent safely. They have driven from San Francisco to Los Angeles and around Lake Tahoe, and have even descended crooked Lombard Street in San Francisco. They drive anywhere a car can legally drive."

- Sebastian Thrun, Google

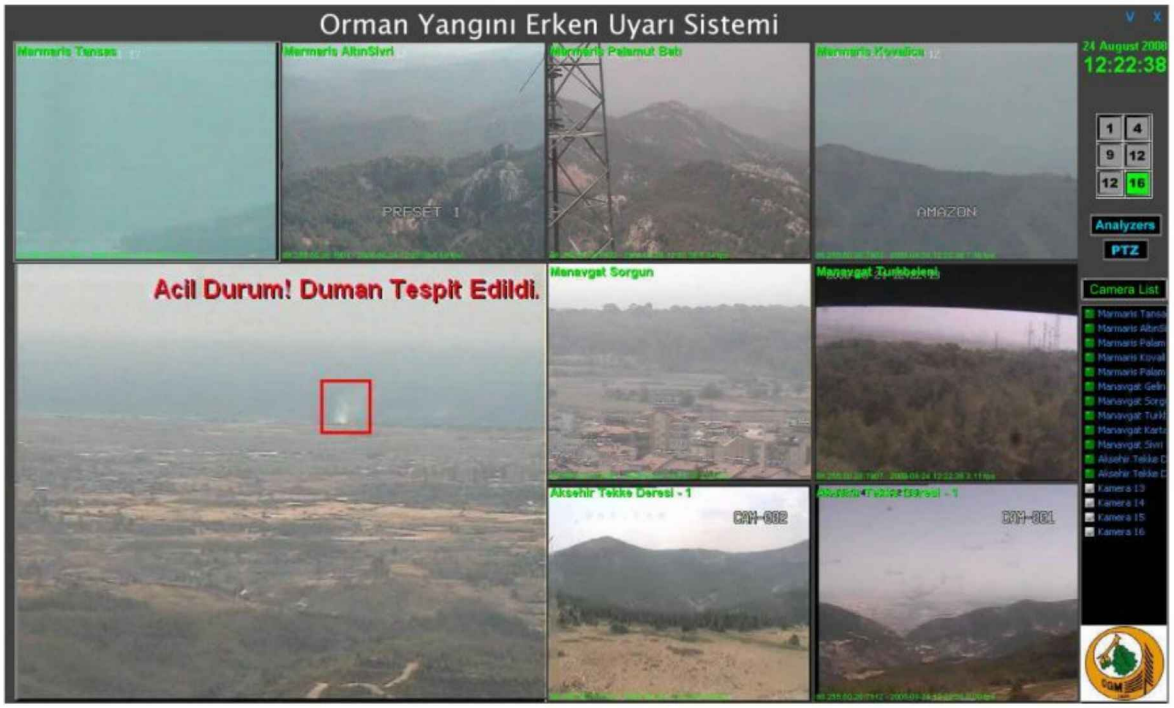
Navegación automática

Michigan State University

General Dynamics Robotics Systems
http://www.gdrs.com

Prof. Eduardo L. L. Cabral

Monitoreamiento de incendios forestales



Alerta precoce de incêndios

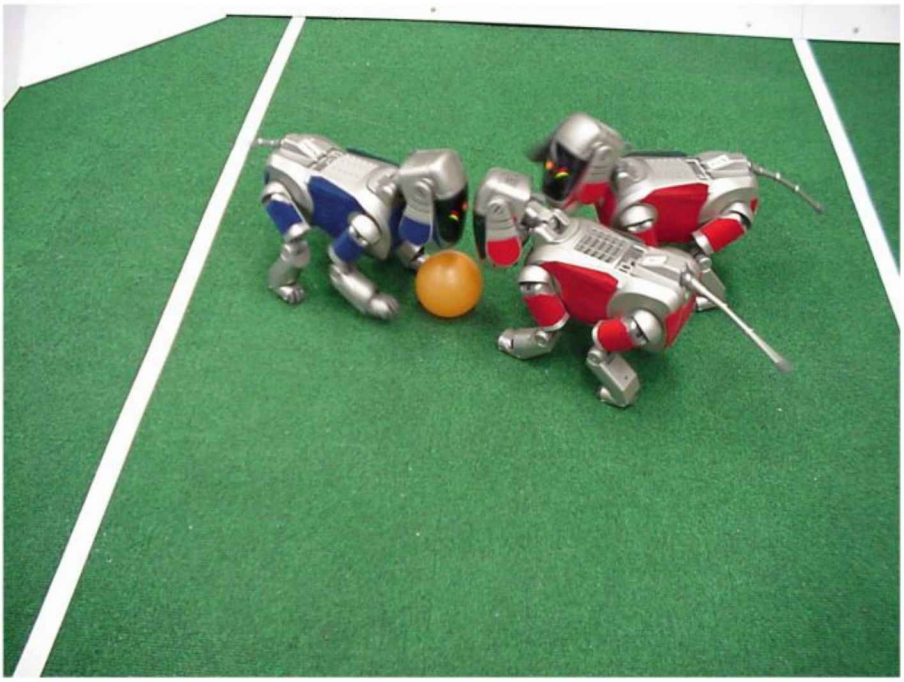
De Enis Cetin, Bilkent University

Prof. Eduardo L. L. Cabral

Robótica

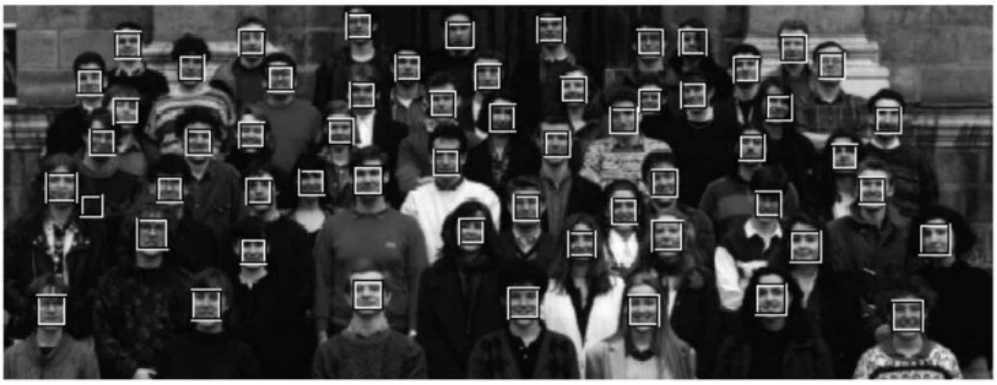
De CSE 455, U of Washington

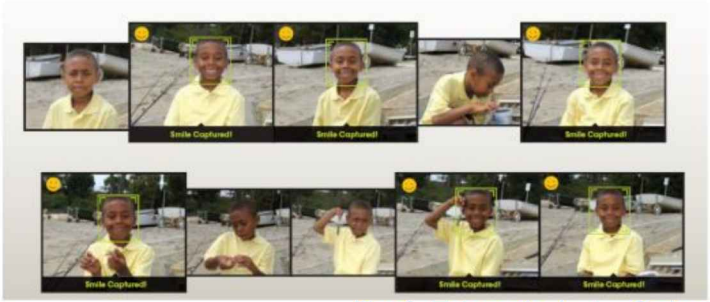
Robótica



De Steven Seitz, U of Washington

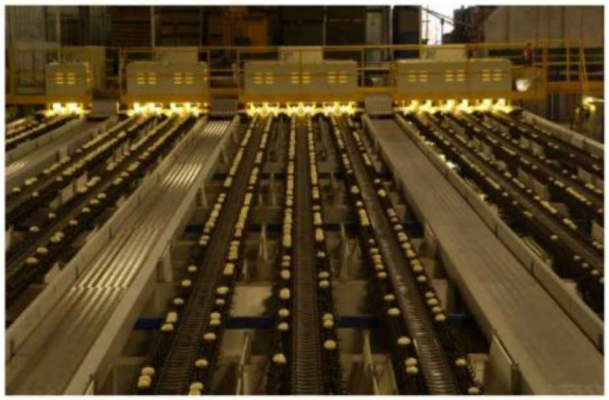
Detección de rostros





De CSE 455, U of Washington

Automatización industrial



Seleção automatica de frutas

Color Vision Systems
http://www.cvs.com.au

Automatización industrial

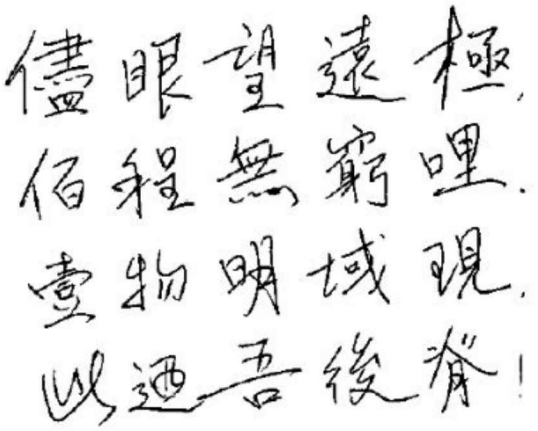
Robô industrial realizando tarefa de "pick and place" guiado por câmera

http://www.braintech.com

Realidad aumentada

De CSE 455, U of Washington

Análisis de documentos

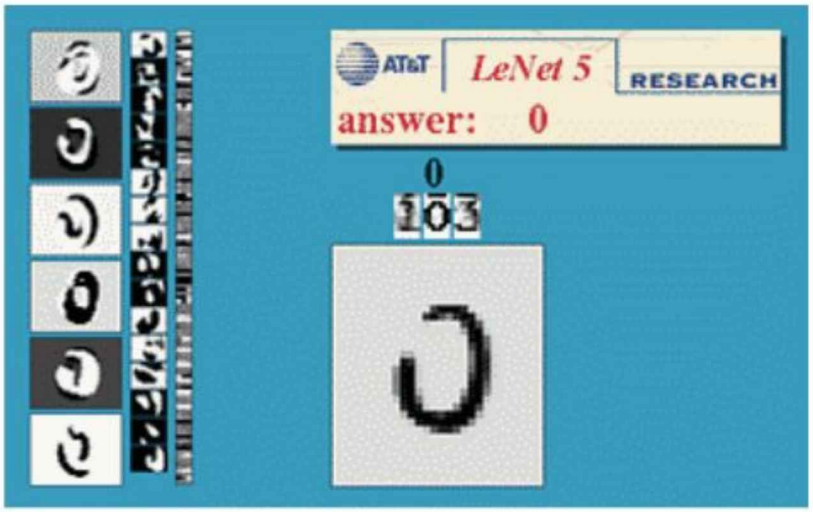


I looked as hard as I could see,
beyond 100 plus infinity
an object of bright intensity
– it was the back of me!

Figure 1.5: (Left) Chinese characters and (right) English equivalent. Is it possible that a machine could automatically translate one into the other? Chinese characters and poem courtesy of John Weng.

De Shapiro and Stockman

Reconocimiento de caracteres



Reconhecimento de letras, AT&T labs
http://www.research.att.com/~yann

De Steven Seitz, U of Washington

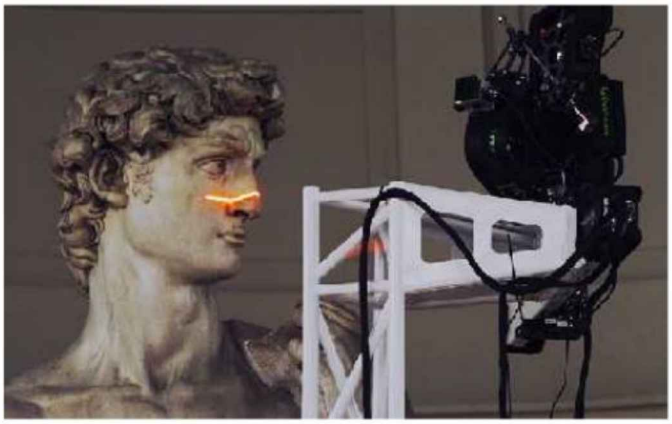
Reconhecimento de placas

Reconocimiento de objetos

Reconhecimento de prédios e grupo de prédios

Prof. Eduardo L. L. Cabral

3D scanning and reconstruction



De Linda Shapiro, U of Washington

Prof. Eduardo L. L. Cabral

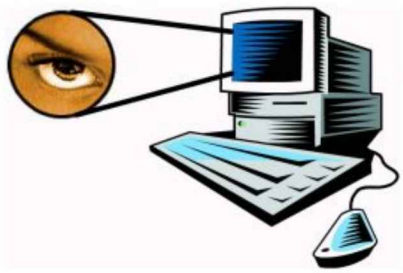
Efectos visuales

De CSE 455, U of Washington

Captura de movimiento

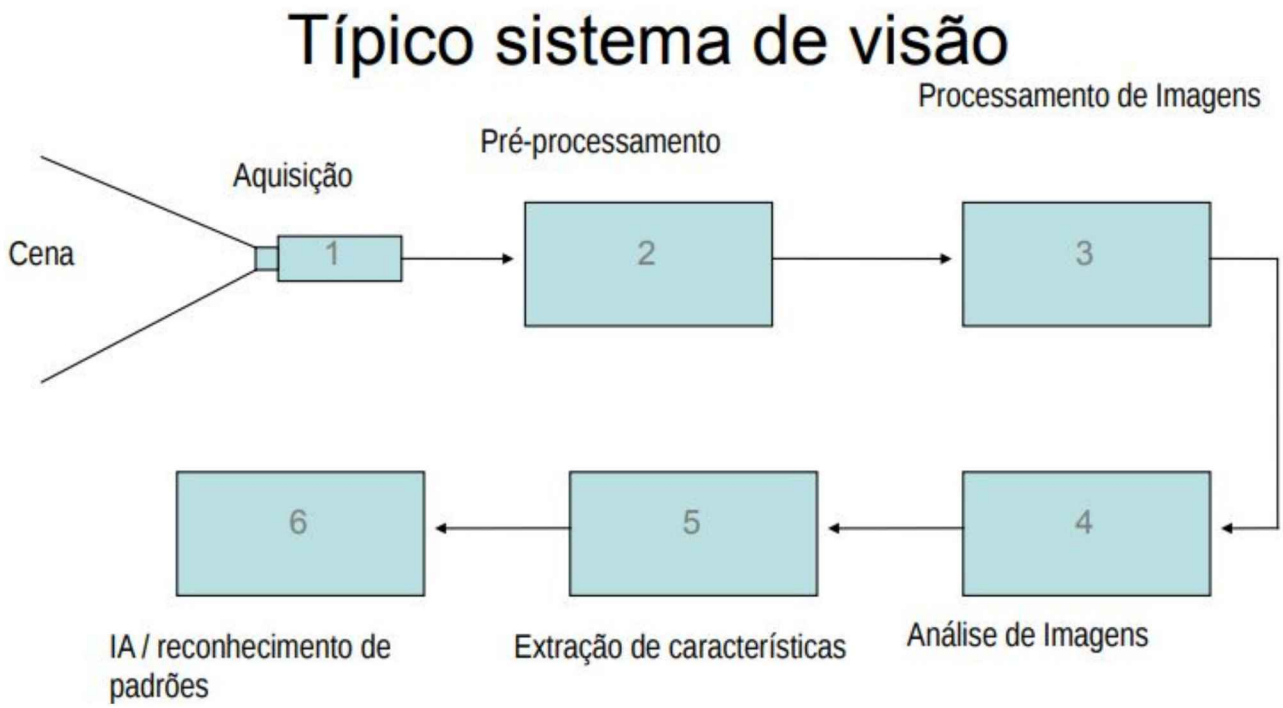
Kinect do Xbox da Microsoft

Ejemplo

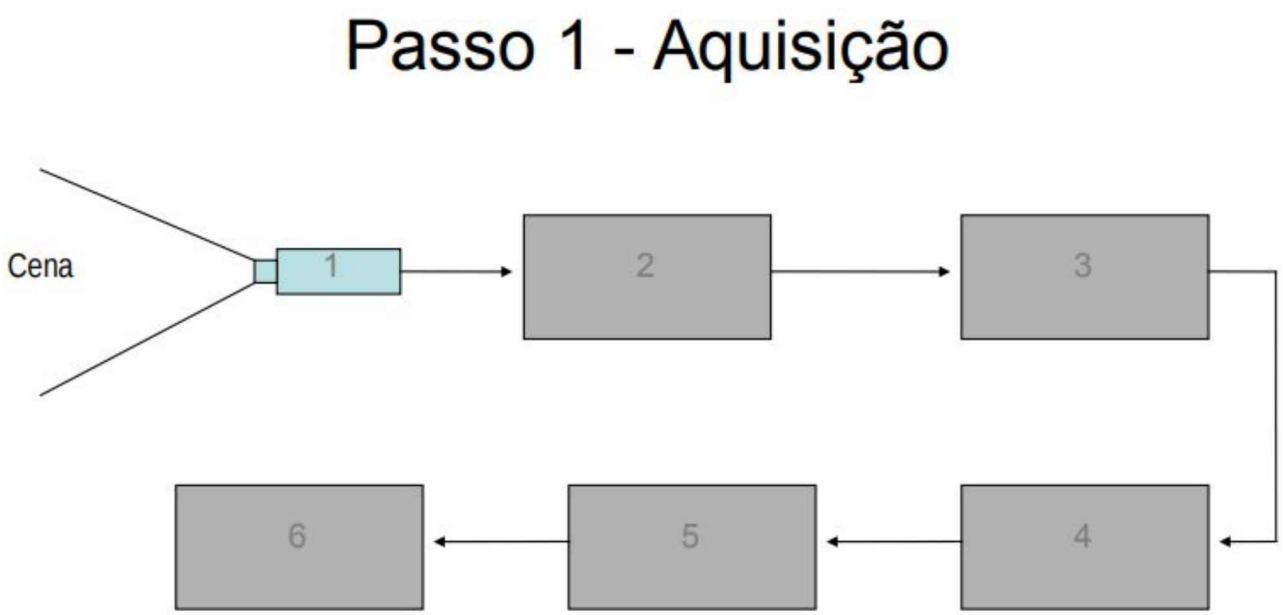


Exemplo: um sistema de visão para reconhecer digitais

Ejemplo

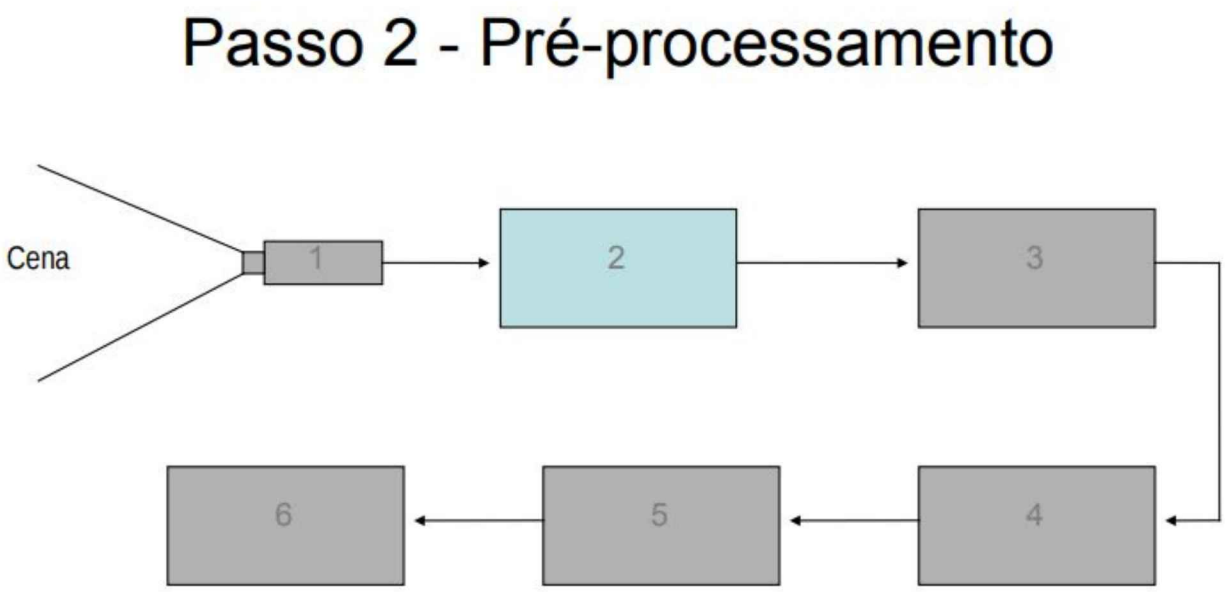


Ejemplo

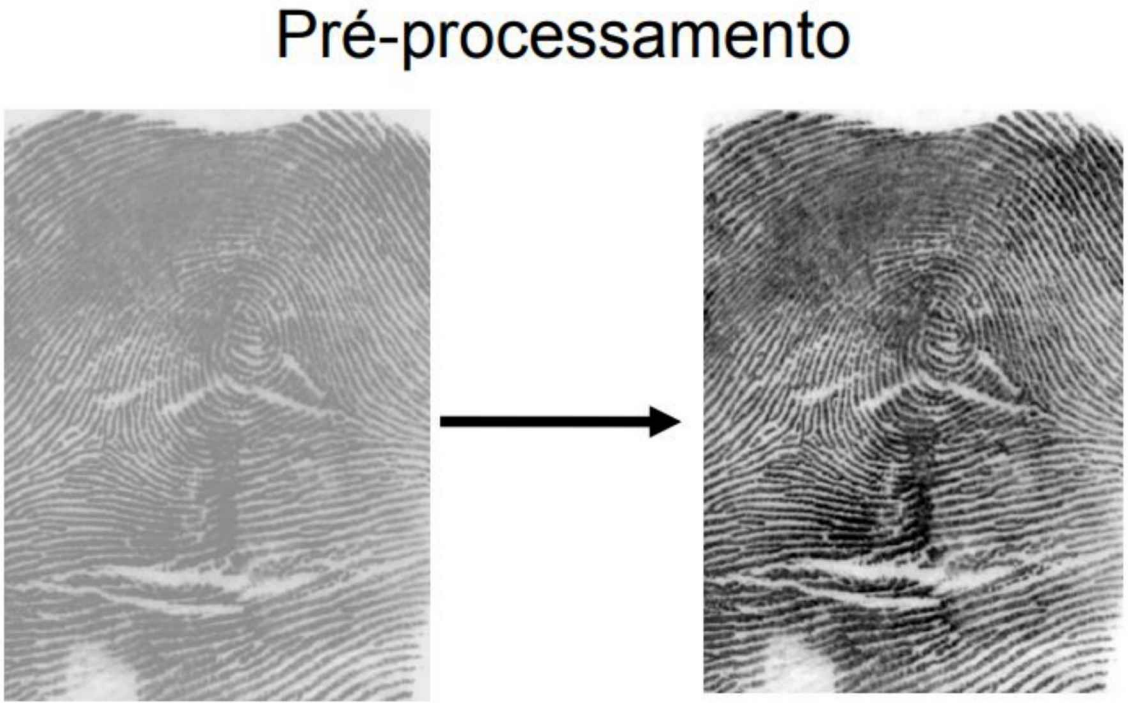


Ejemplo

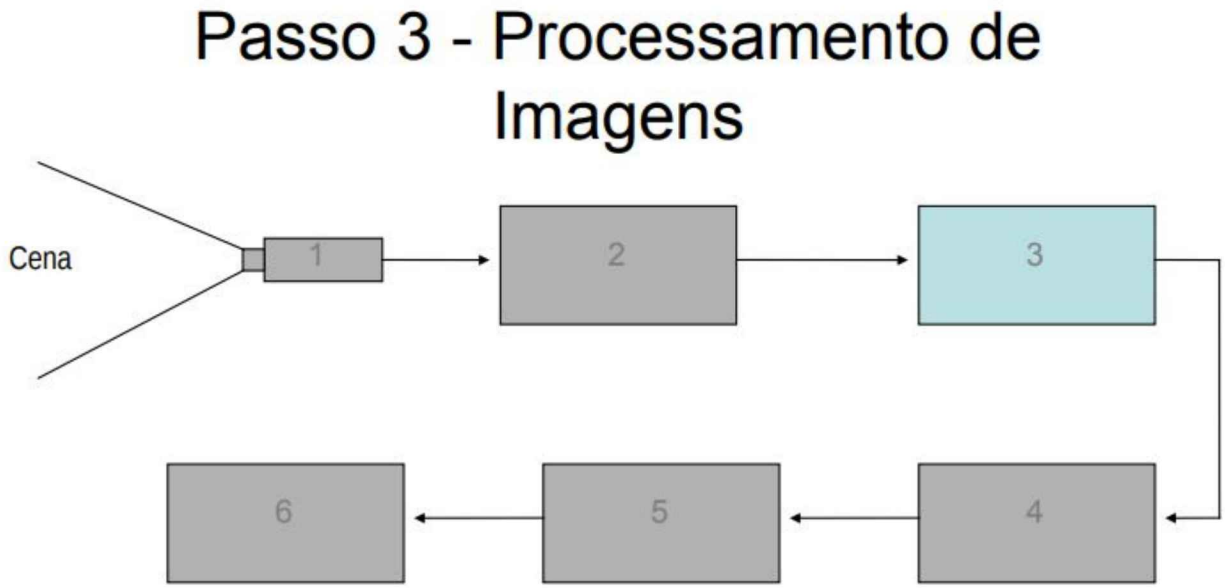
Ejemplo



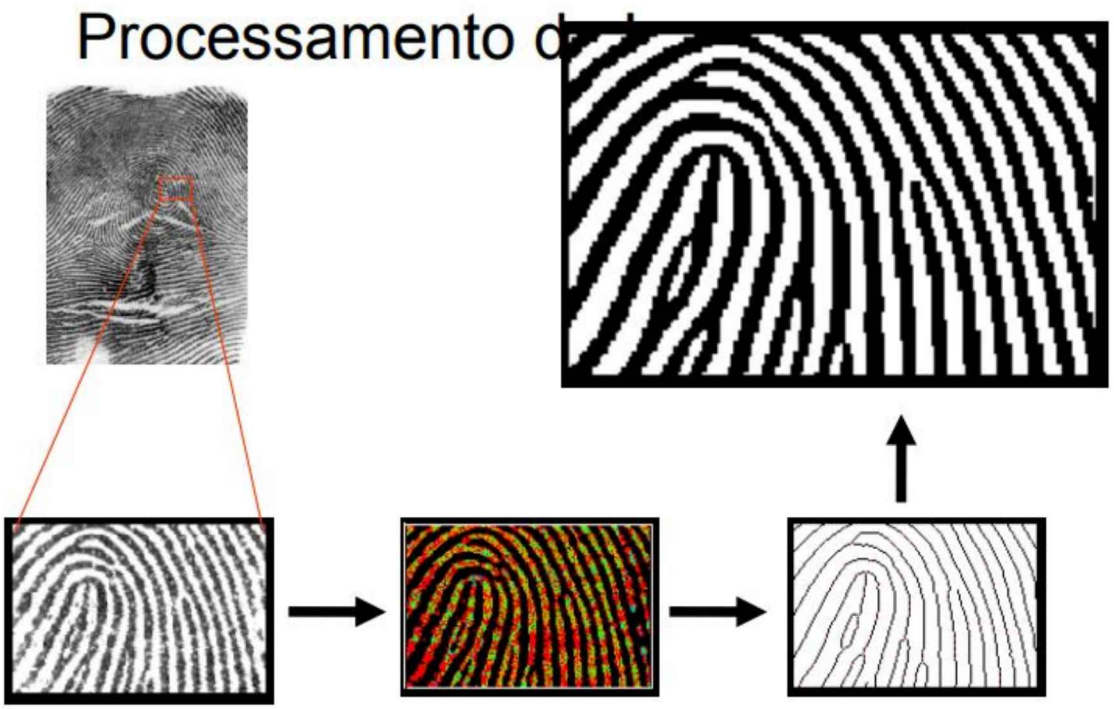
Ejemplo



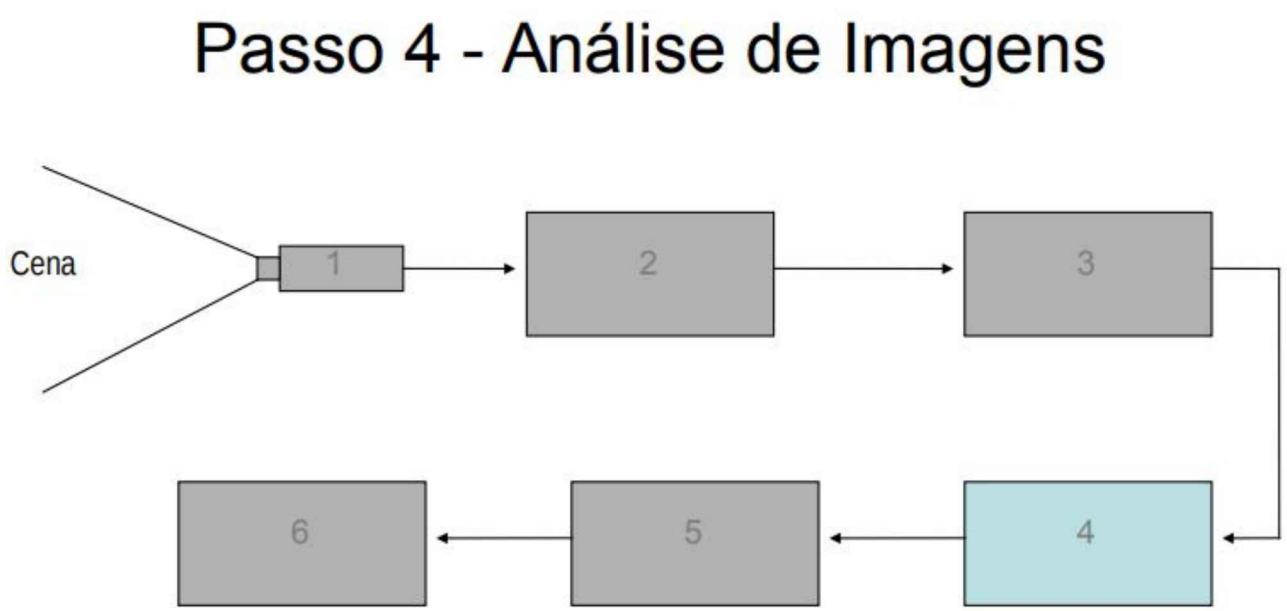
Ejemplo



Ejemplo

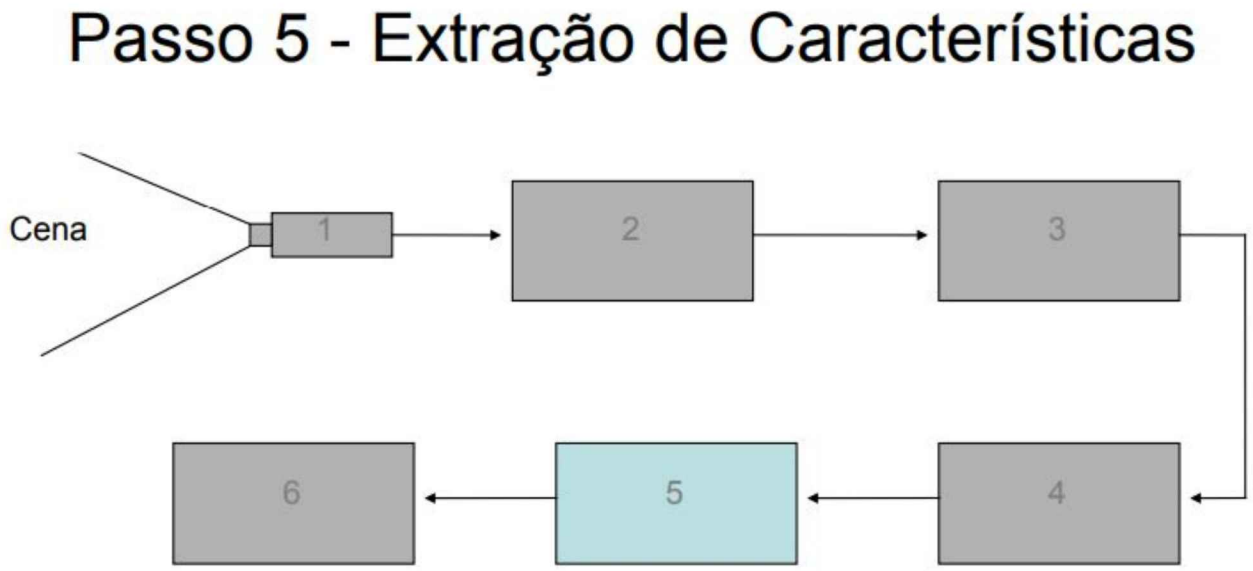


Ejemplo



Ejemplo

Ejemplo



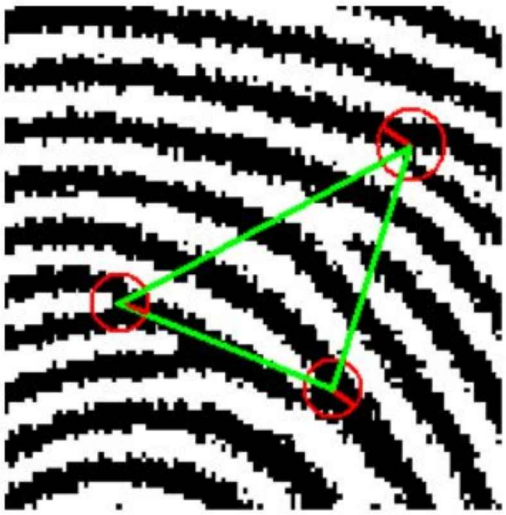
Ejemplo

Extração de Características: Modelo Matemático

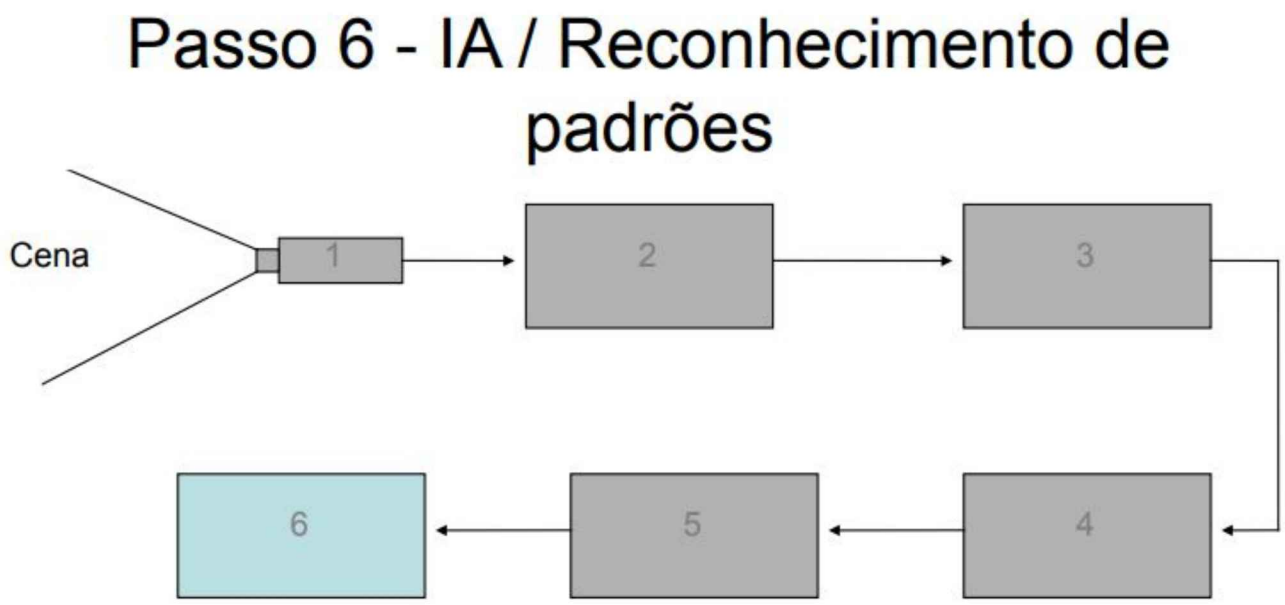
Modelo Matemático

- Semelhança de Triângulos

  Combinar as marcações 3 a 3

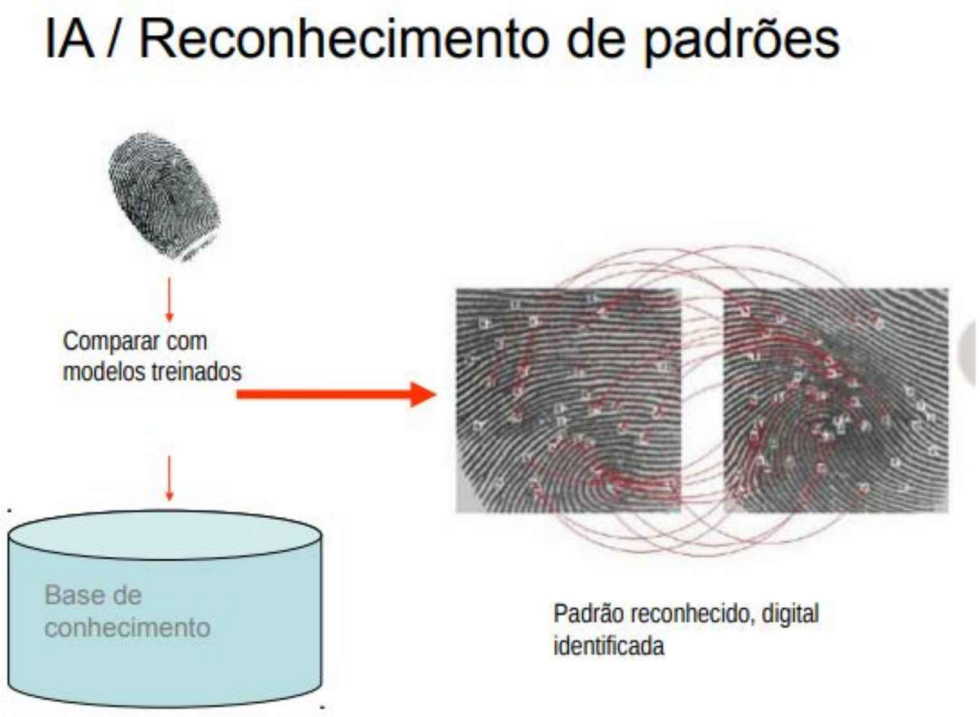


Ejemplo



Ejemplo

Ejemplo



Prof. Dr. João do E. S. Batista Neto

Universidad Nacional del Altiplano
Escuela de Posgrado
Doctorado en Ciencias de la Computación

Computer Vision

Unit 1. Image Processing:

- Image Segmentation

Prof. Dr. Ivar Vargas Belizario

ivargasbelizario@gmail.com

2024 - II

Content

- Introduction
- Region-Based
  - Thresholding
  - Watershed
  - Split and Merge
  - Clustering
  - Graph Cut
  - Normalized Cut
  - Complex Networks

Introduction (gray level images)

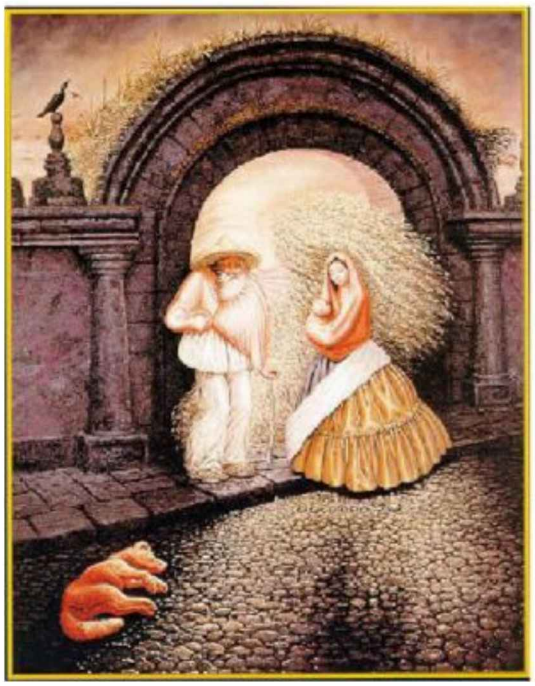
Introduction (color images)

Red (**R**)

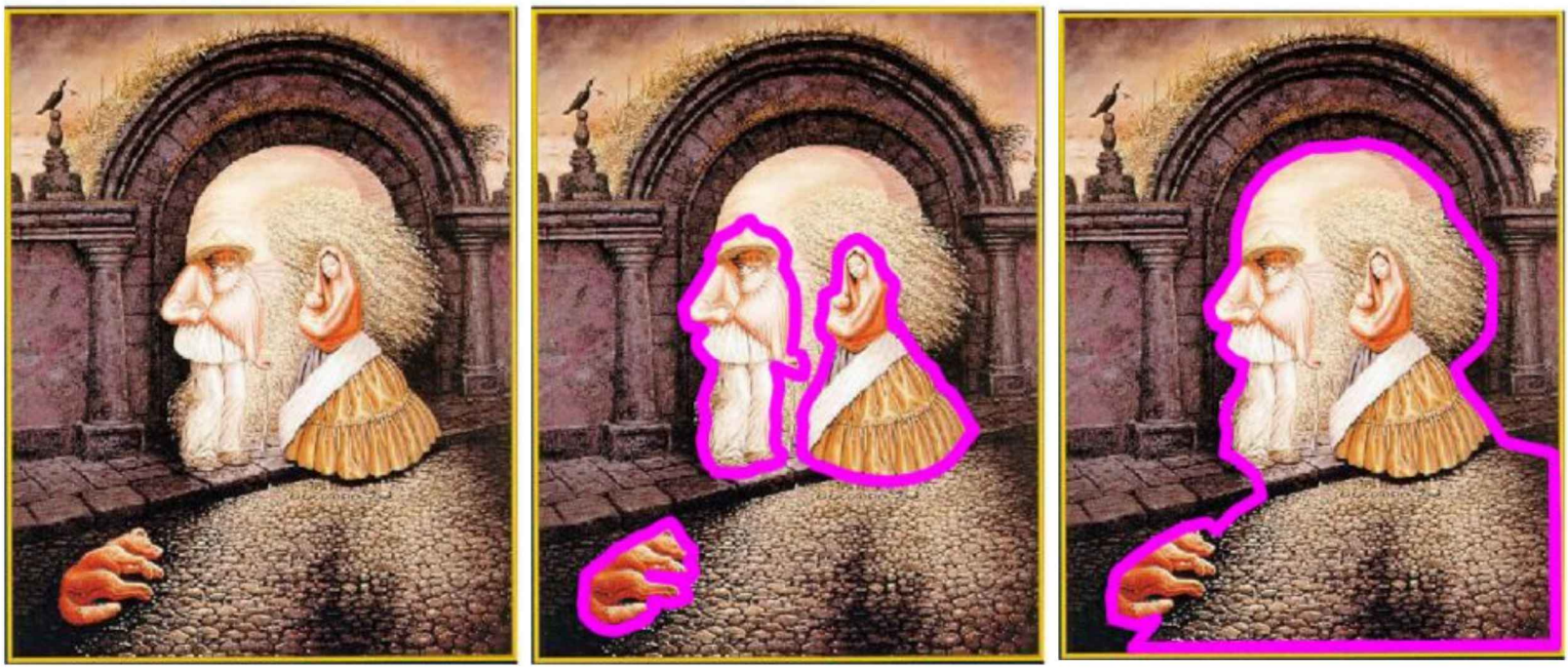
Green (**G**)

Blue (**B**)

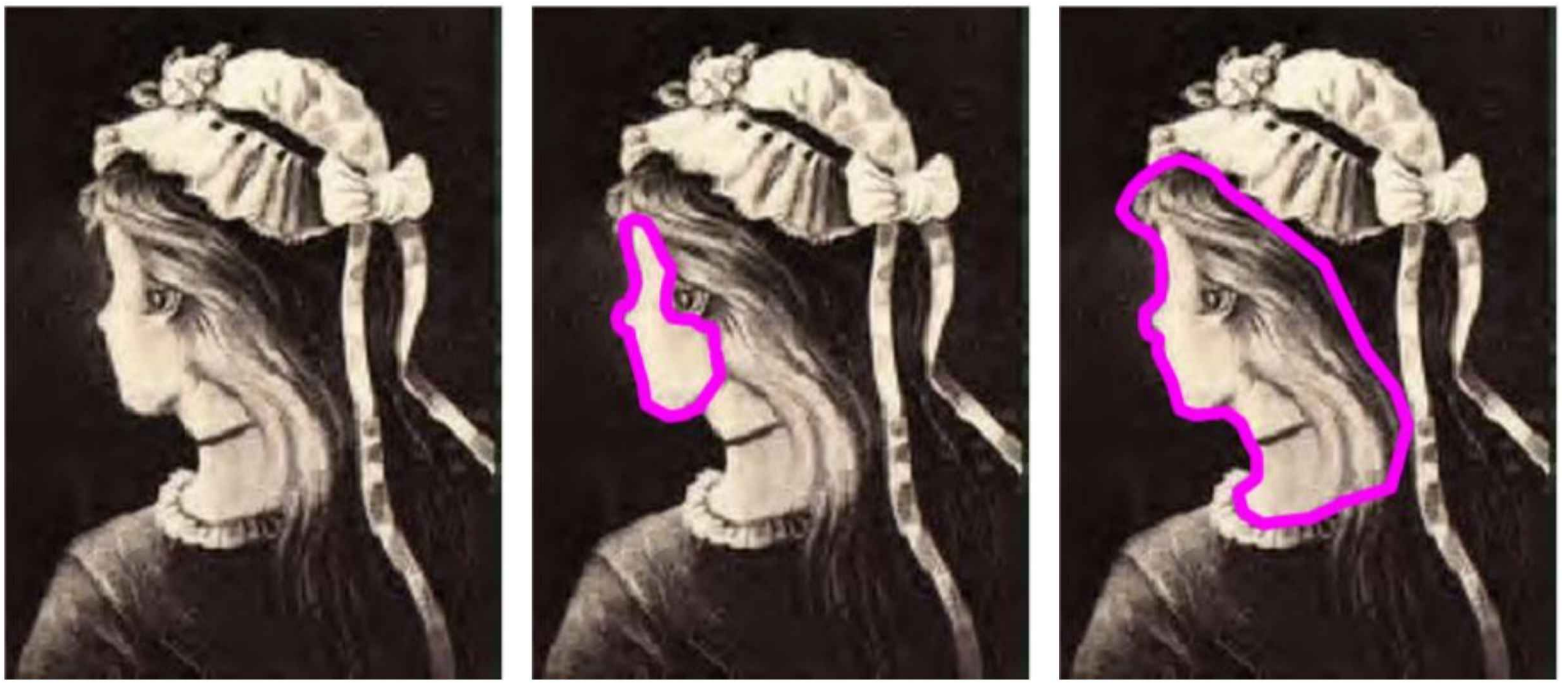
Introduction



Introduction



Introduction

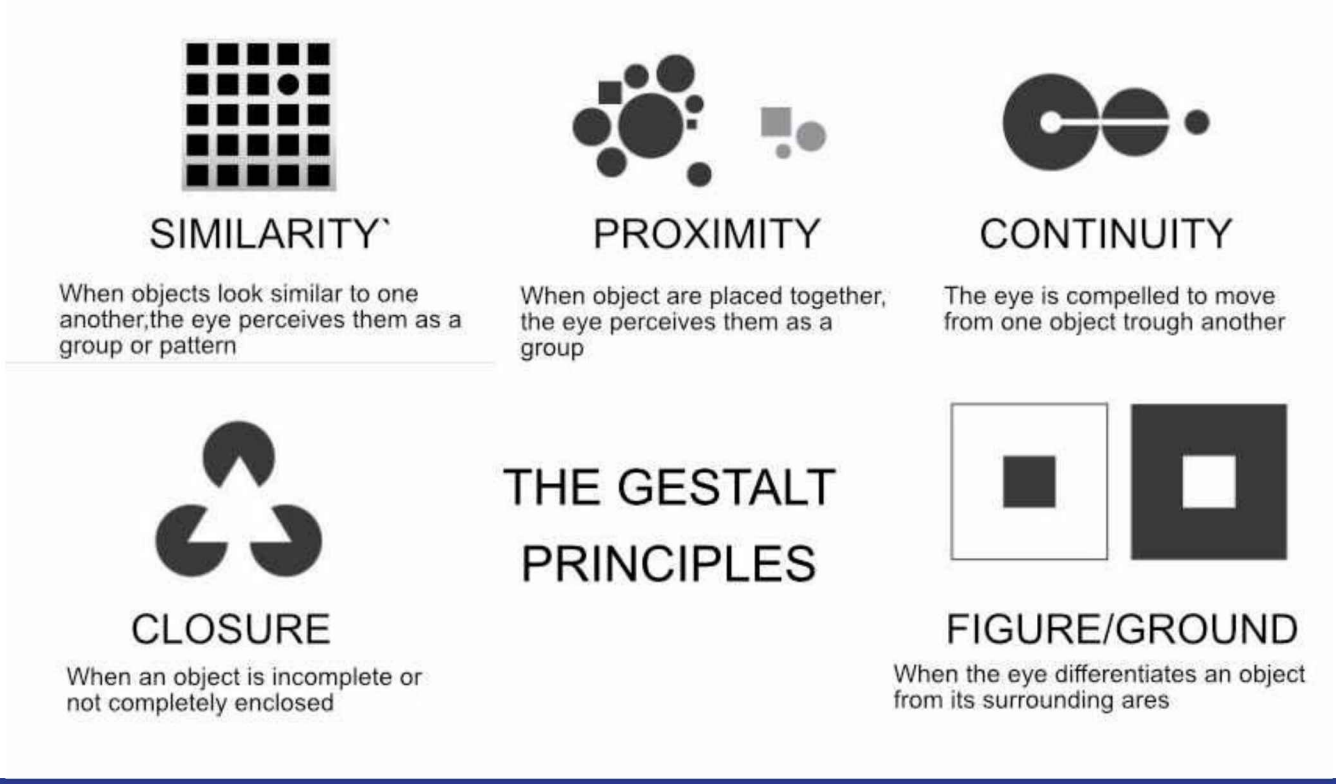


Content

- Introduction
  - Principles of Gestalt Theory
  - What is Segmentation?
  - Applications
- Region-Based

Introduction: Principles of Gestalt Theory

Introduction: Principles of Gestalt Theory



Introduction: Principles of Gestalt Theory

Content

- Introduction
  - Principles of Gestalt Theory
  - What is Segmentation?
  - Applications
- Region-Based

Introduction: What is Segmentation?

- **Segmentação de imagens**
  - O principal objetivo da segmentação de imagens é dividir uma imagem em partes que tenham uma forte correlação com objetos ou áreas do mundo real representados na imagem.

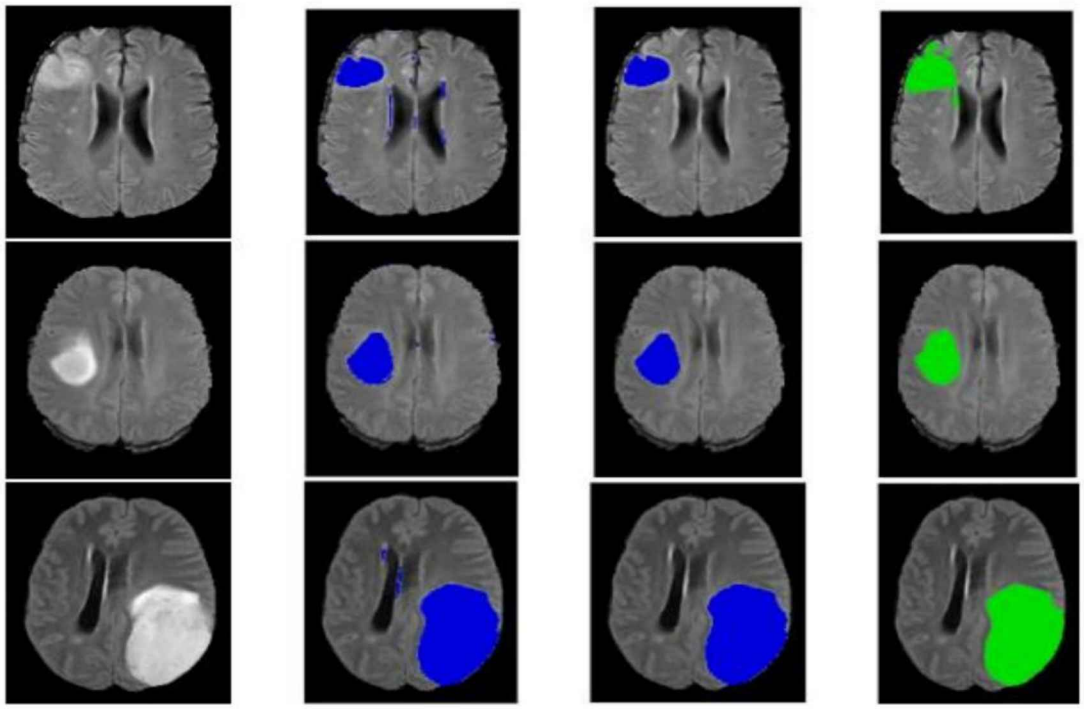
    Sonka, M.; Hlavac, V. & Boyle, R. (2014), Image Processing: Analysis and Machine Vision, CL-Engineering.

  - Tem importância na área de visão computacional por ser um dos primeiros processos para o reconhecimento de padrões.

Content

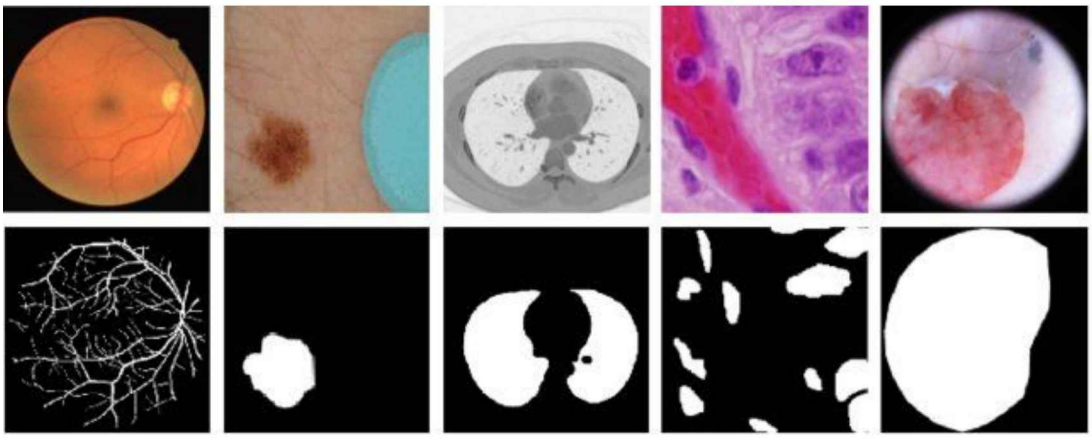
- Introduction
  - Principles of Gestalt Theory
  - What is Segmentation?
  - Applications
- Region-Based

Introduction: Applications

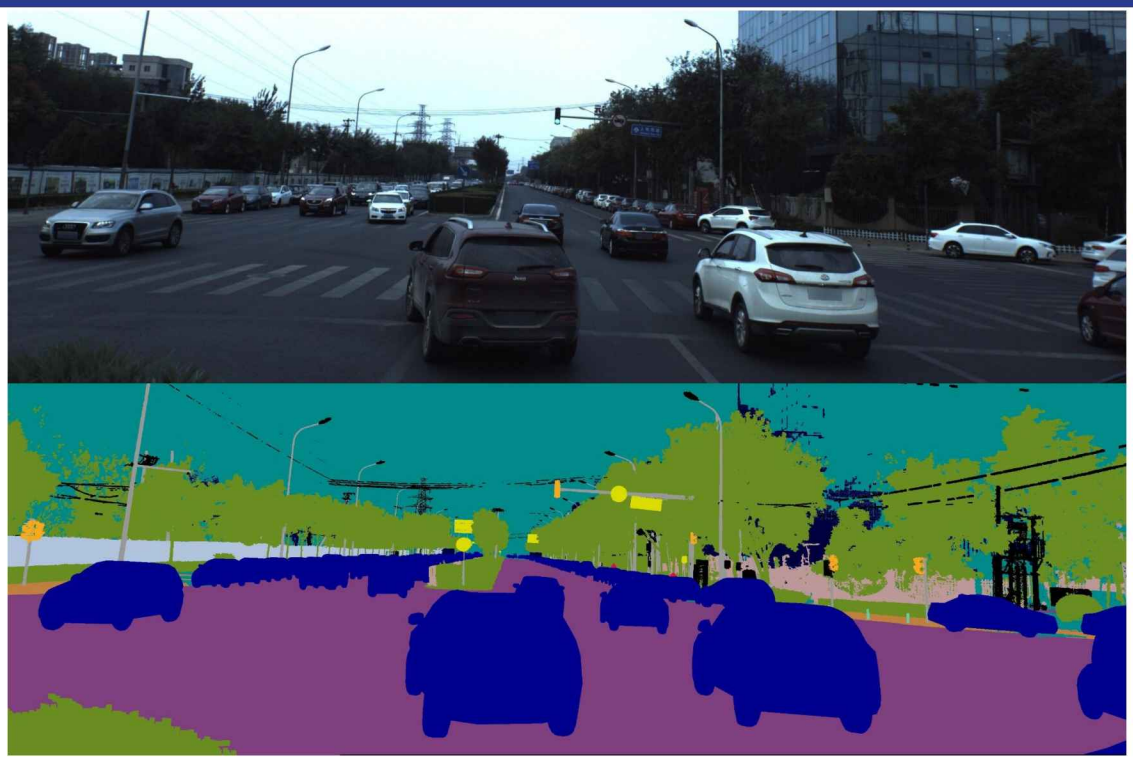


https://doi.org/10.3390/jimaging7120269

Introduction: Applications



Introduction: Applications



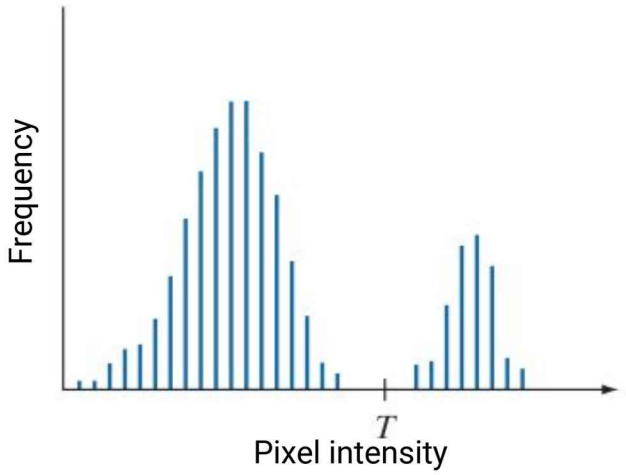
https://www.kaggle.com/c/cvpr-2018-autonomous-driving

Content

- Introduction
- Region-Based
  - Thresholding
  - Watershed
  - Splitting and Merging
  - Clustering
  - Graph Cut
  - Normalized Cut
  - Complex Networks

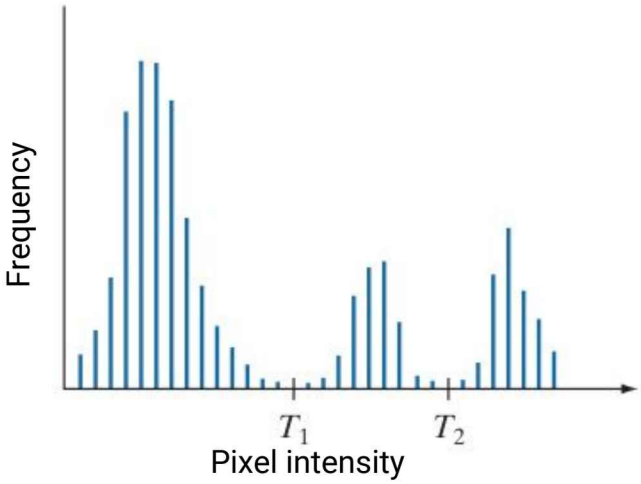
Thresholding

$$g(x,y) = \begin{cases} 1 & \text{if } f(x,y) > T \\ 0 & \text{if } f(x,y) \le T \end{cases}$$



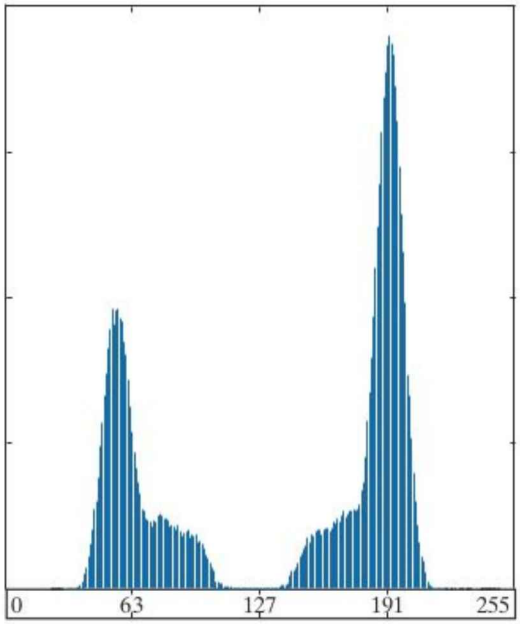
Thresholding

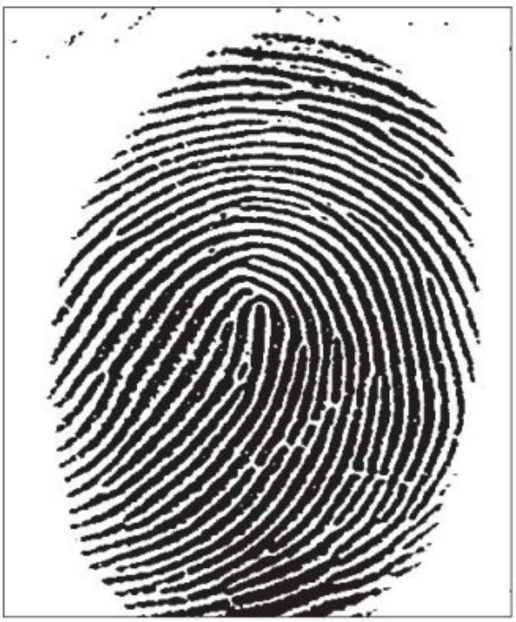
$$g(x,y) = \begin{cases} a & \text{if } f(x,y) > T_2 \\ b & \text{if } T_1 < f(x,y) \le T_2 \\ c & \text{if } f(x,y) \le T_1 \end{cases}$$



Thresholding

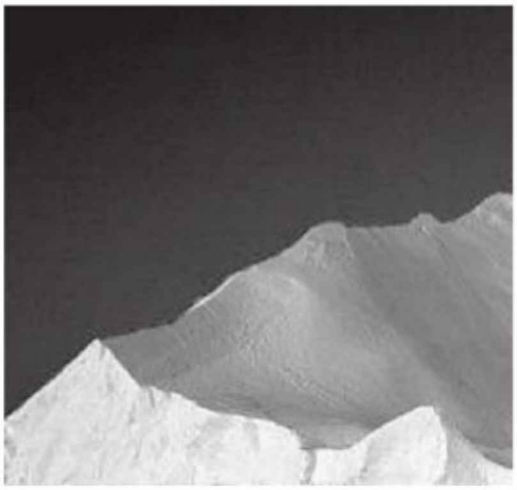
- **Simple thresholding**

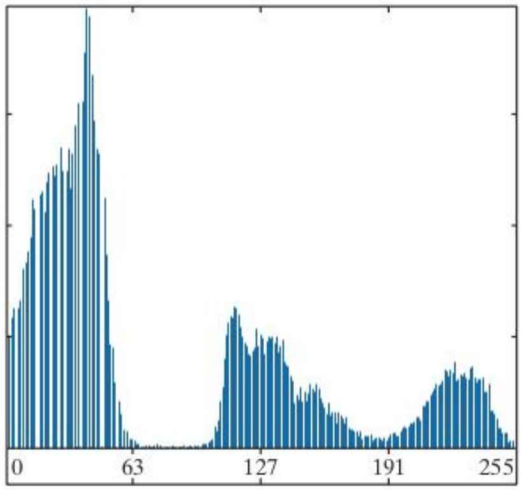


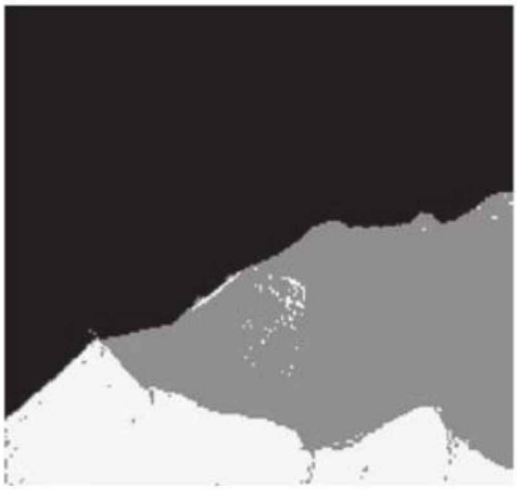


Thresholding

- **Multiple thresholding.**



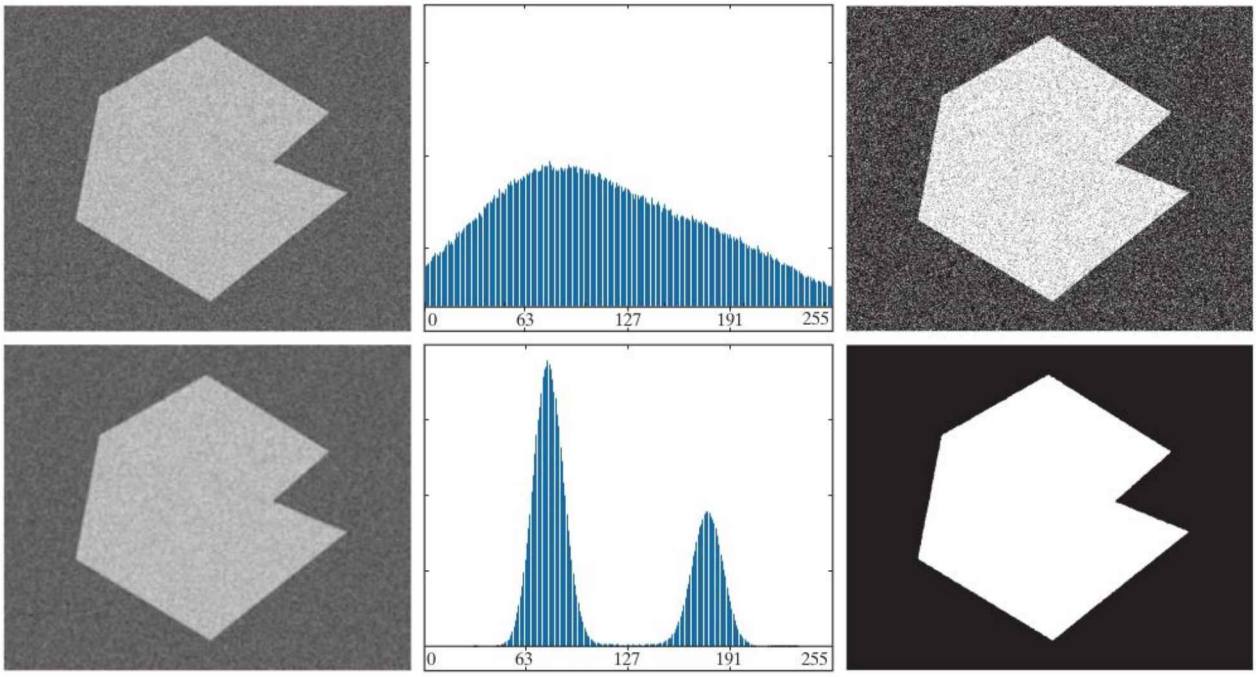




Thresholding

- **Add noise**

Noise



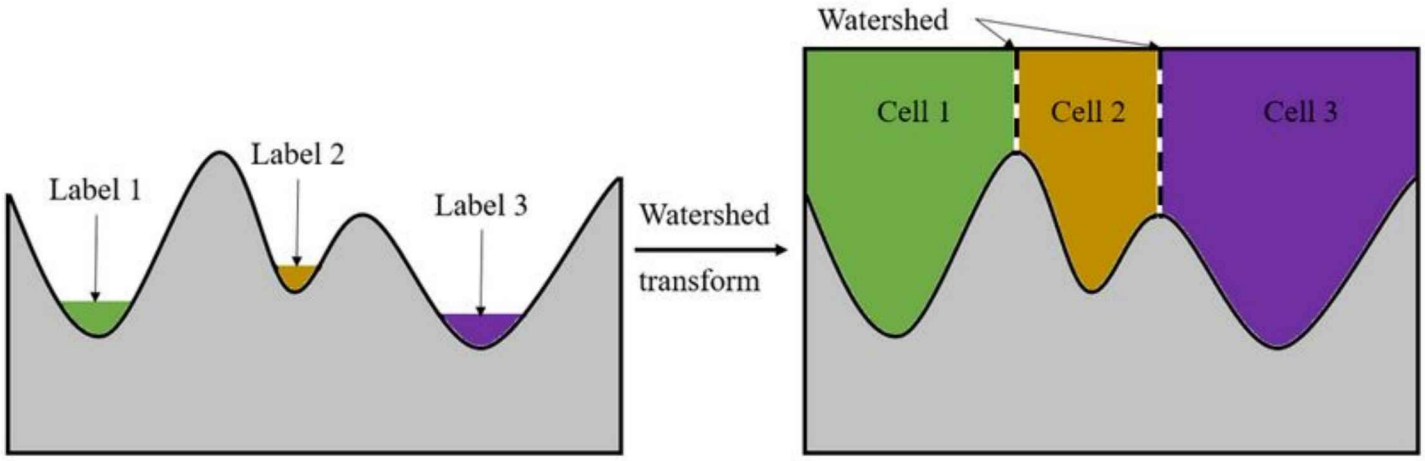
Smoothed using a 5 × 5 averaging kernel

Thresholding

Content

- Introduction
- Region-Based
  - Thresholding
  - Watershed
  - Splitting and Merging
  - Clustering
  - Graph Cut
  - Normalized Cut
  - Complex Networks

Watershed

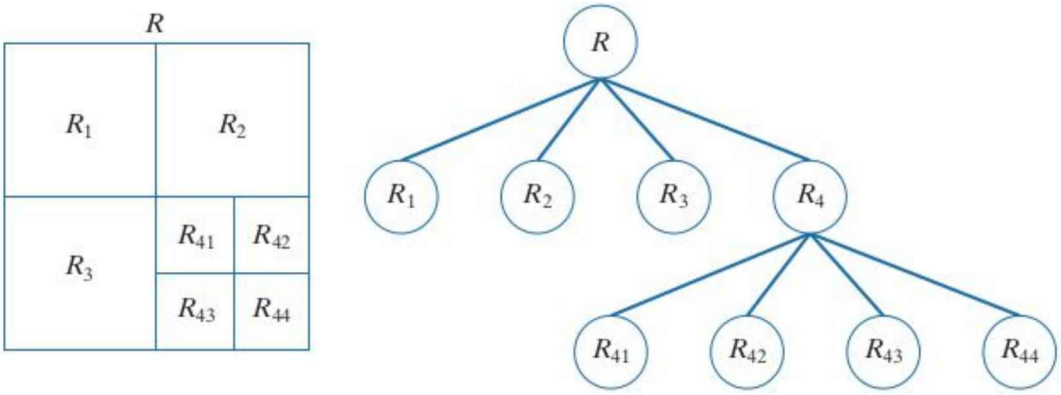


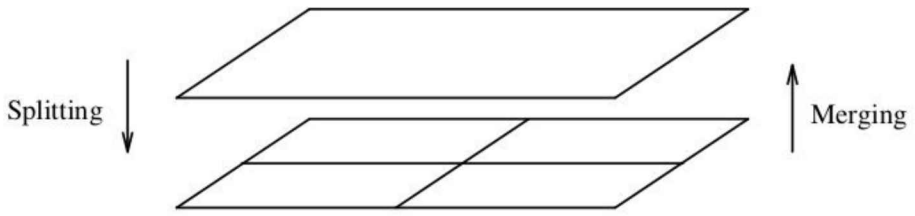
Watershed segmentation

Content

- Introduction
- Region-Based
  - Thresholding
  - Watershed
  - Splitting and Merging
  - Clustering
  - Graph Cut
  - Normalized Cut
  - Complex Networks

Splitting and Merging

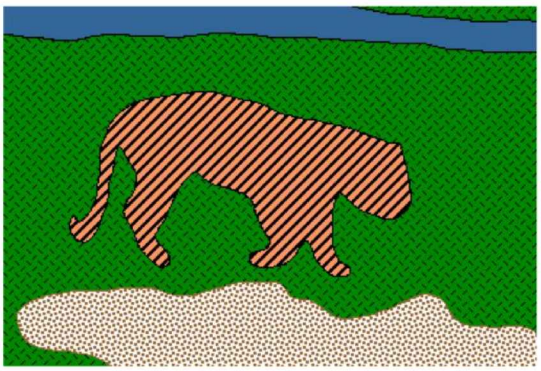




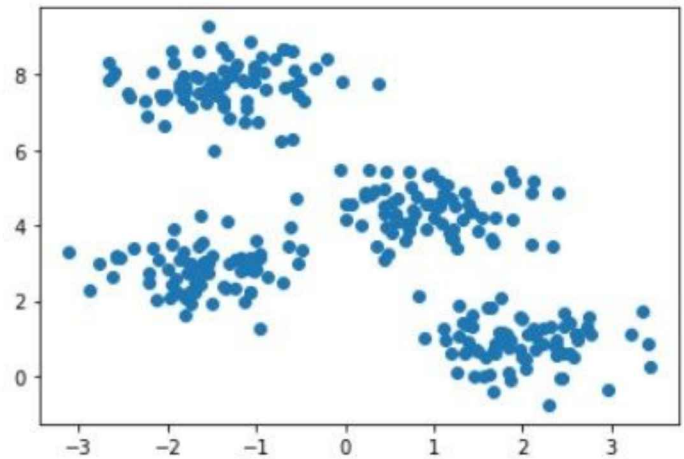
Content

- Introduction
- Region-Based
  - Thresholding
  - Region Growing
  - Splitting and Merging
  - Clustering
  - Graph Cut
  - Normalized Cut
  - Complex Networks

Clustering



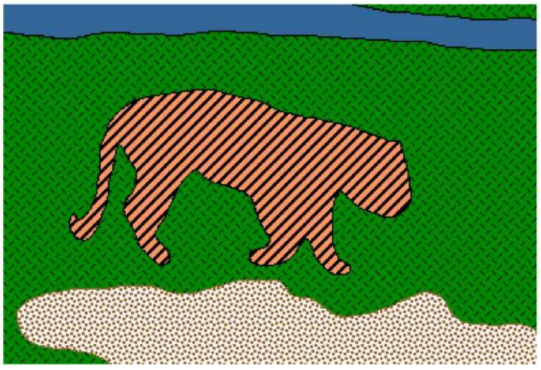
K-means algorithm



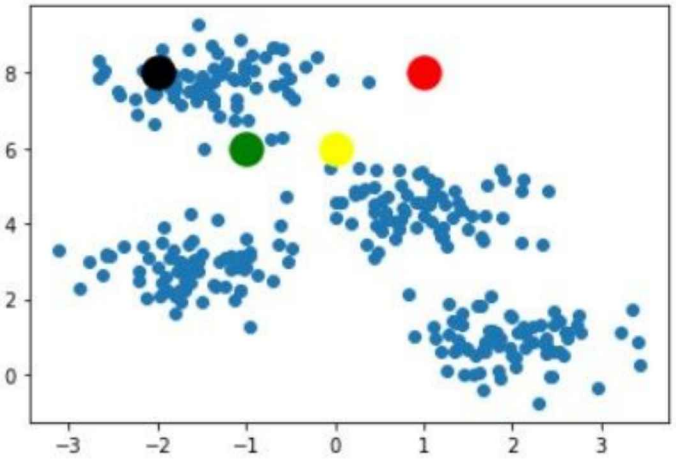
$$\text{objective function} \leftarrow J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2$$

number of clusters; number of cases; case i; centroid for cluster j; Distance function

Clustering



K-means algorithm

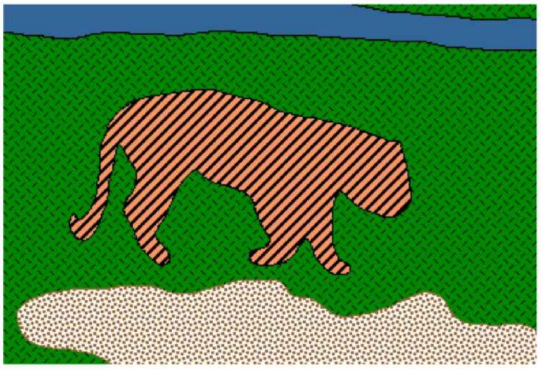


number of clusters — number of cases — case i — centroid for cluster j

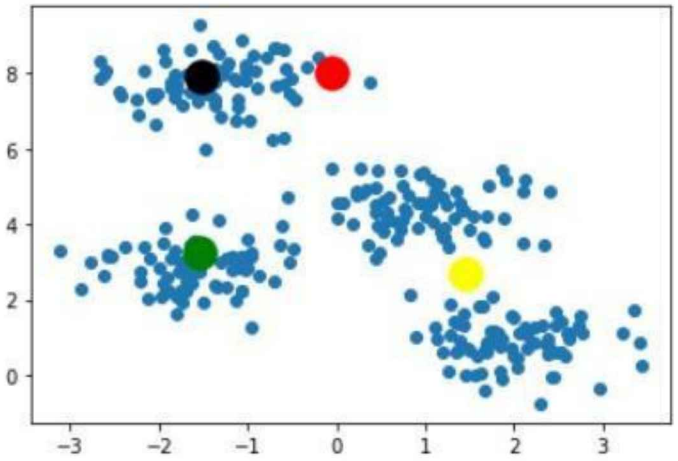
objective function $\leftarrow J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2$

Distance function

Clustering



K-means algorithm

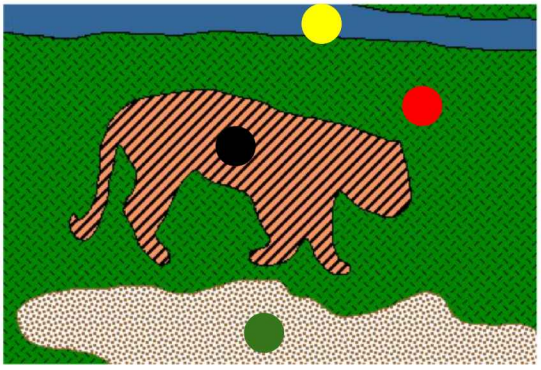


number of clusters — number of cases — case i — centroid for cluster j

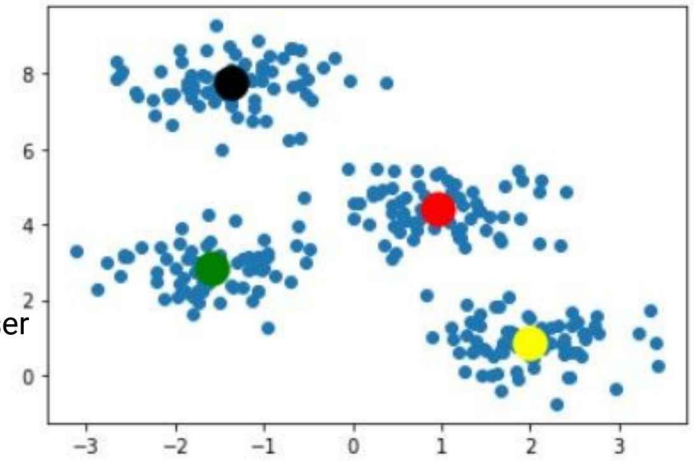
objective function $\leftarrow J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2$

Distance function

Clustering



K-means algorithm



k = is indicated by the user

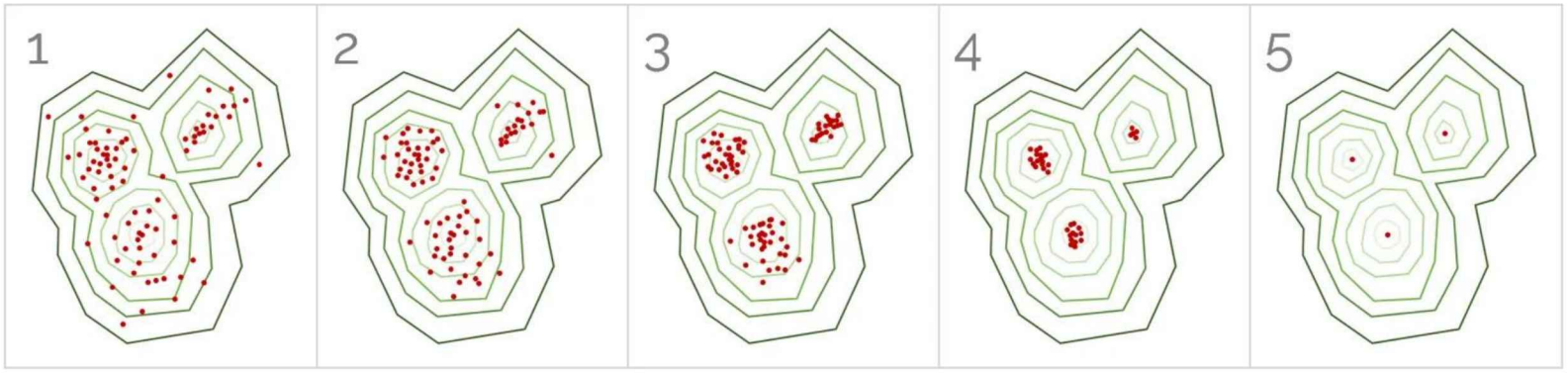
number of clusters
number of cases
centroid for cluster j
case i

objective function $\leftarrow J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2$

Distance function

Clustering

Mean shift algorithm



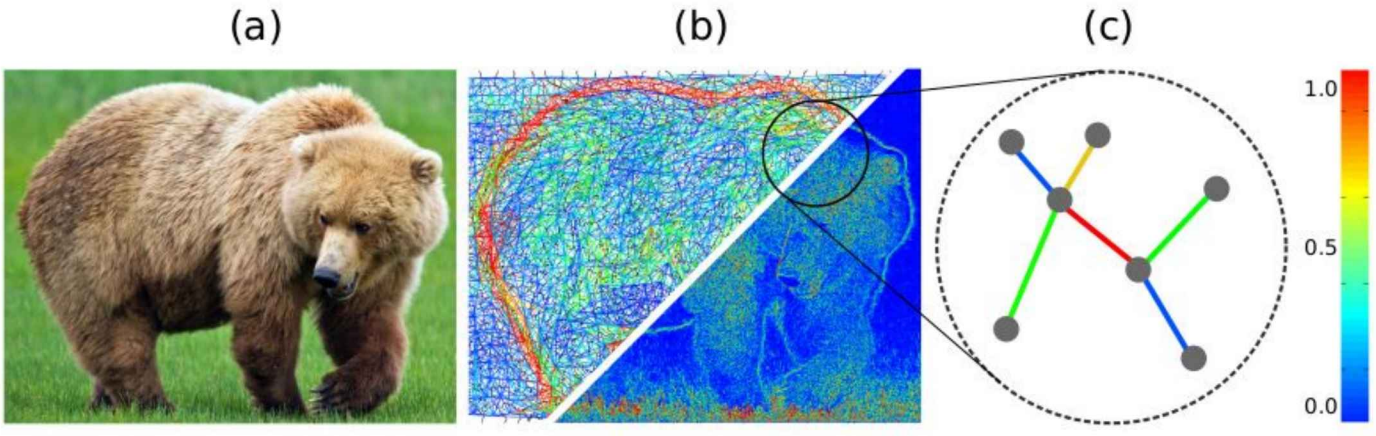
k = is calculated automatically

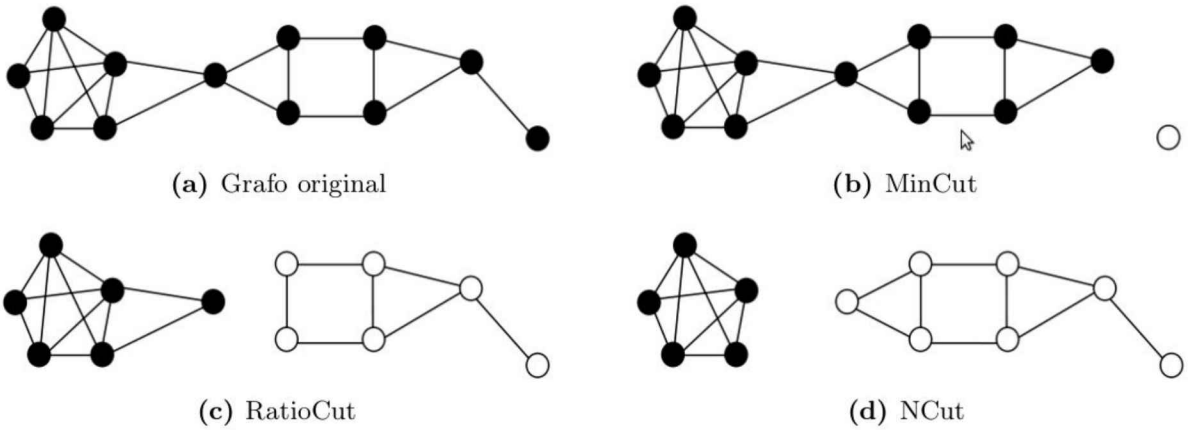
Content

- Introduction
- Region-Based
  - Thresholding
  - Watershed
  - Splitting and Merging
  - Clustering
  - Graph Cut
  - Normalized Cut
  - Complex Networks

Graph Cut: Image as a graph representation

Figura – Visualização de um grafo que representa a imagem: (a) imagem original, (b) visualização das arestas entre vértices que podem ser *superpixels* ou pixels, (c) visualização de uma subamostra de arestas: em termos de distância, as arestas com pesos altos indicam que os vértices adjacentes a arestas são diferentes, enquanto que as arestas com pesos baixos indicam que os vértices adjacentes a aresta são parecidos.





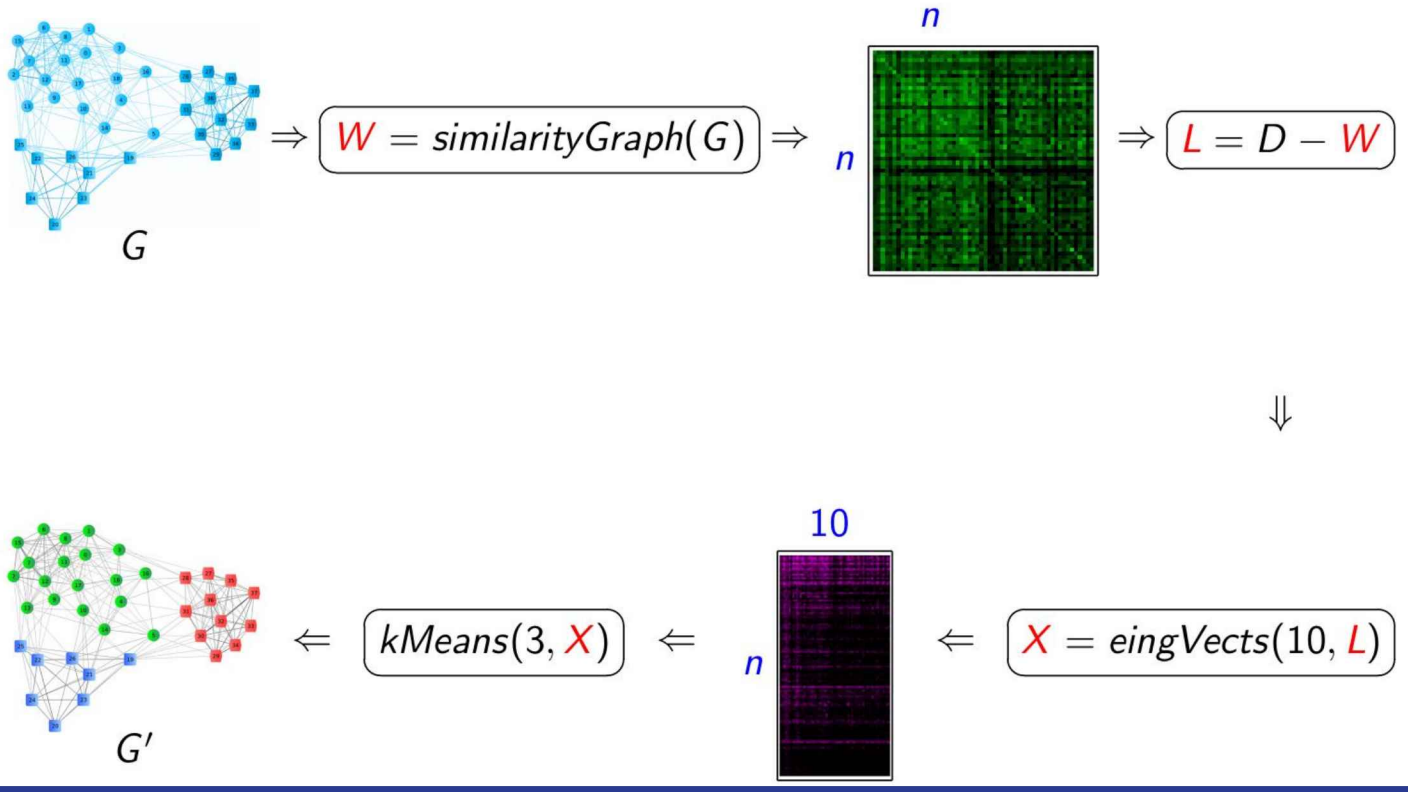
(a) Grafo original (b) MinCut

(c) RatioCut (d) NCut

Content

- Introduction
- Region-Based
  - Thresholding
  - Watershed
  - Splitting and Merging
  - Clustering
  - Graph Cut
  - Normalized Cut
  - Complex Networks

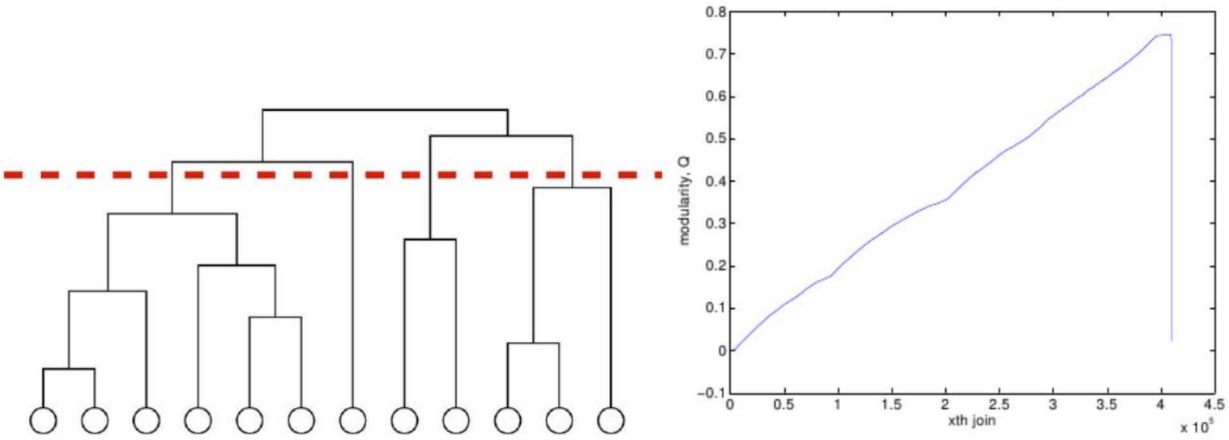
Normalized Cut



Content

- Introduction
- Region-Based
  - Thresholding
  - Watershed
  - Splitting and Merging
  - Clustering
  - Graph Cut
  - Normalized Cut
  - Complex Networks

Complex Networks: Fast Greedy



(a) Dendrograma (b) $\triangle Q$

Figura. (a) Representação do dendrograma, para o algoritmo *fast greedy*, a linha vermelha representa o corte do dendrograma empregando o máximo valor de $\triangle Q$. (b) No eixo x é mostrado o número de uniões realizadas entre duas comunidades ao longo do algoritmo, nela é mostrada o acrescentamento de $\triangle Q$, onde $\triangle Q$ apresenta um único valor máximo.

Complex Networks: Label Propagation

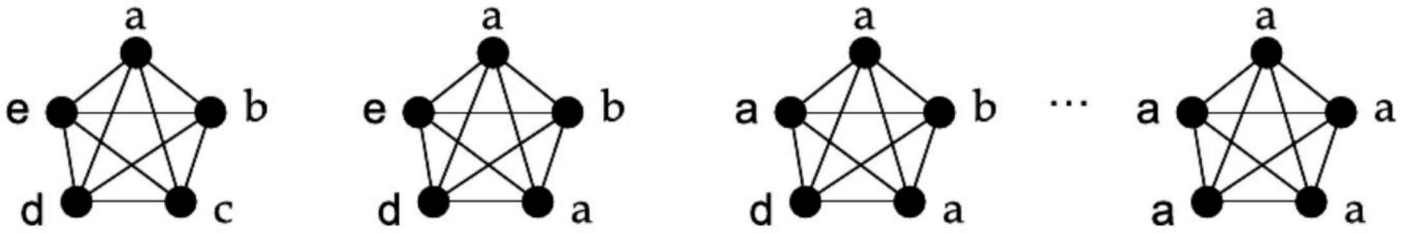
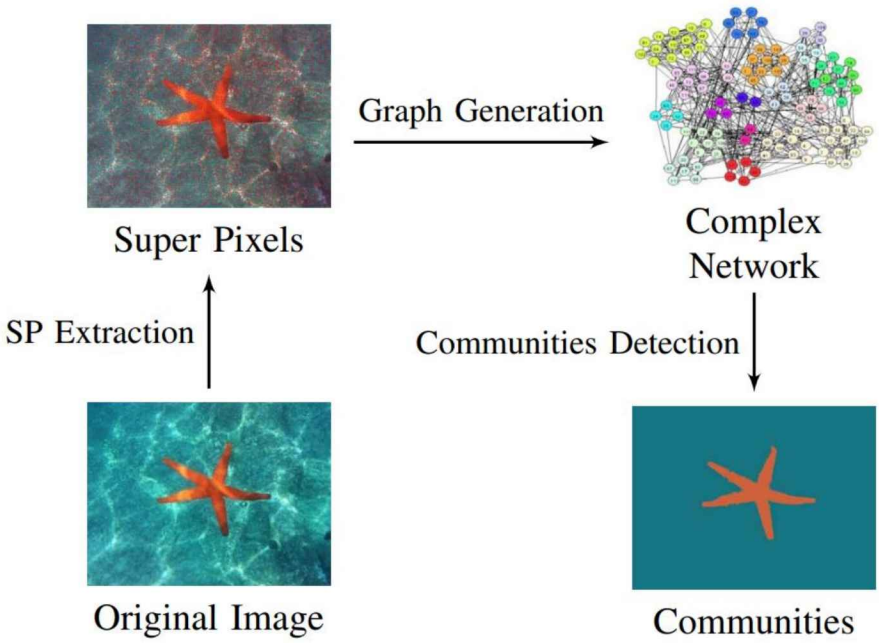


Figura. Atualização de etiquetas no *label propagation*: na figura de esquerda a direita, os nodos são atualizados um por um. Neste caso existe uma grande densidade de arestas, isto faz possível que todos os nodos adquiram a mesma etiqueta.

Content

- Introduction
- Region-Based
  - Thresholding
  - Watershed
  - Splitting and Merging
  - Clustering
  - Graph Cut
  - Normalized Cut
  - Complex Networks
    - Speed Up Turbo Pixel with Fast Greedy (SUTP-FG)
    - Simple Graph Label Propagation (SGLP)
    - Multi-level Graph Label Propagation (MGLP)

Complex Networks: Speed Up Turbo Pixel with Fast Greedy (SUTP-FG)



https://doi.org/10.1049/iet-ipr.2016.0072

Content

- Introduction
- Region-Based
  - Thresholding
  - Watershed
  - Splitting and Merging
  - Clustering
  - Graph Cut
  - Normalized Cut
  - Complex Networks
    - Speed Up Turbo Pixel with Fast Greedy (SUTP-FG)
    - Simple Graph Label Propagation (SGLP)
    - Multi-level Graph Label Propagation (MGLP)

Complex Networks: Simple Graph Label Propagation (SGLP)

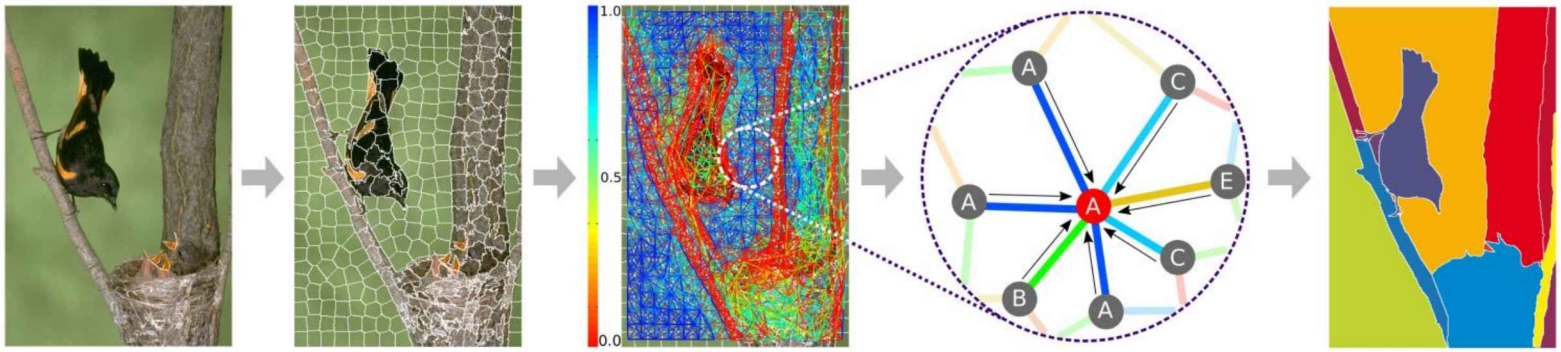
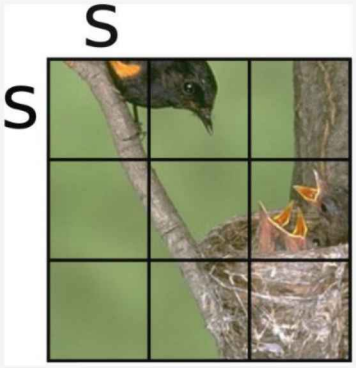


Figure: Método proposto: (a) Imagem de entrada; (b) pre-segmentação com superpixels; (c) criação do grafo; (d) propagação de etiquetas; (e) segmentação.

https://doi.org/10.1049/ipr2.12242

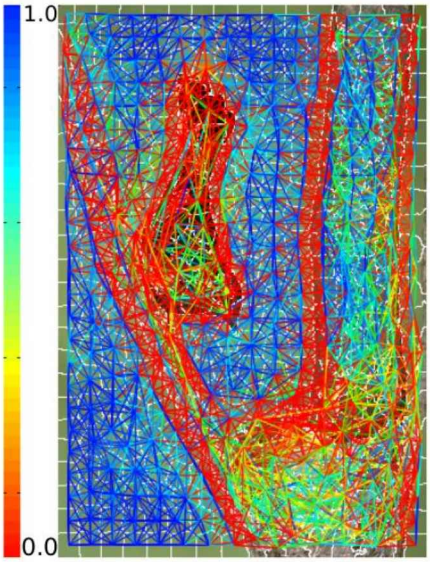
Complex Networks: Simple Graph Label Propagation (SGLP)

1. Simple Non-iterative Clustering (SNIC) [AS17]
2. Comprimento inicial do lado dos superpixels (S=?)



3. $R = N/S^2$

Complex Networks: Simple Graph Label Propagation (SGLP)



1. Vértices:
   - Descritores ME e CM
2. Arestas:
   - Vetores de caraterísticas (ME e CM)
   - Nível de vizinhança (H)
   - Limiar (T)
   - Funções peso F(i,j)

Complex Networks: Simple Graph Label Propagation (SGLP)

Peso das arestas:

$$W_{ij} = \begin{cases} F(i,j), & \text{If } j \in H \text{ and } F(i,j) \geq T \\ \nexists, & \text{else} \end{cases} \quad (1)$$

Funções peso ou de similaridade F(i,j):

$$GAU(i,j) = \exp\left(\frac{-d(i,j)^2}{2\sigma^2}\right) \quad (2)$$

$$COS(i,j) = \begin{cases} c = \frac{i \cdot j}{||i||||j||}, & \text{If } c > 0 \\ 0, & \text{else} \end{cases} \quad (3)$$

$$TAN(i,j) = \begin{cases} t = \frac{i \cdot j}{||i||^2 + ||j||^2 - i \cdot j}, & \text{If } t > 0 \\ 0, & \text{else.} \end{cases} \quad (4)$$

Limiar (T); Nível de vizinhança (H)

Complex Networks: Simple Graph Label Propagation (SGLP)

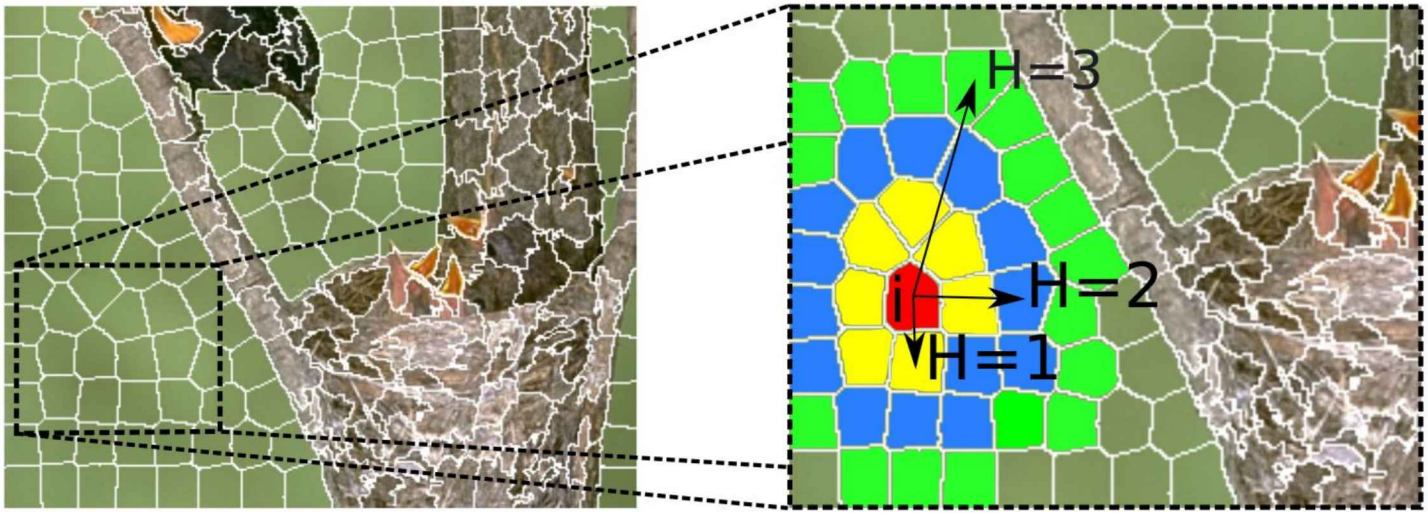
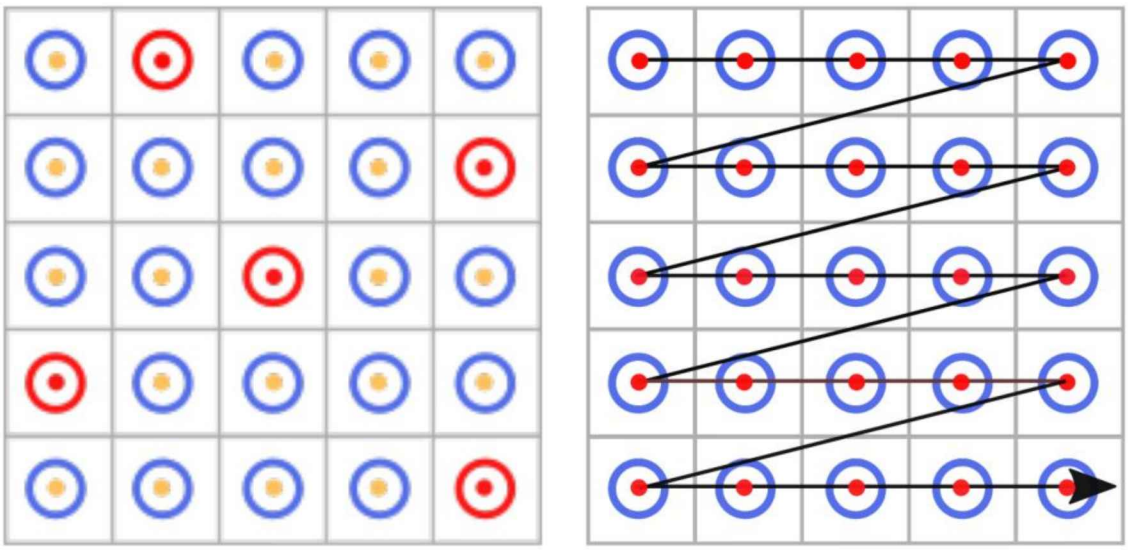


Figure: Exemplo de criação de arestas utilizando níveis de vizinhança. (a) Superpixels, (b) 3 Níveis de vizinhança (H=3)

Complex Networks: Simple Graph Label Propagation (SGLP)



(a) Random (b) **Convolução (CON)**

Complex Networks: Simple Graph Label Propagation (SGLP)

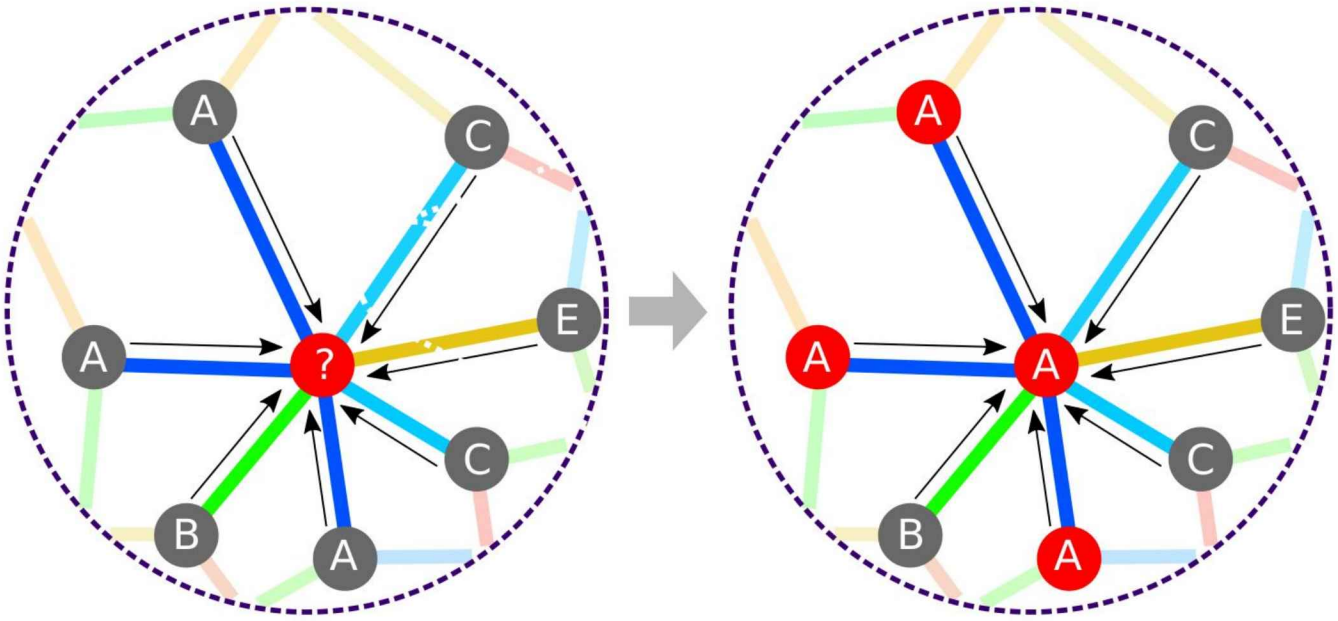


Figure: Exemplo de propagação de etiquetas. Com uma travessia determinística na forma de **convolução** lineal (CON)

Complex Networks: Simple Graph Label Propagation (SGLP)

- **Propagação Iterativa de Etiquetas** (ILP)

$$f^{(t)}(i, L) = \underset{j \in N(i)}{\text{argmax}} \left(\sum L_j^{(t)}, ..., \sum L_j^{(t-1)} \right) \tag{5}$$

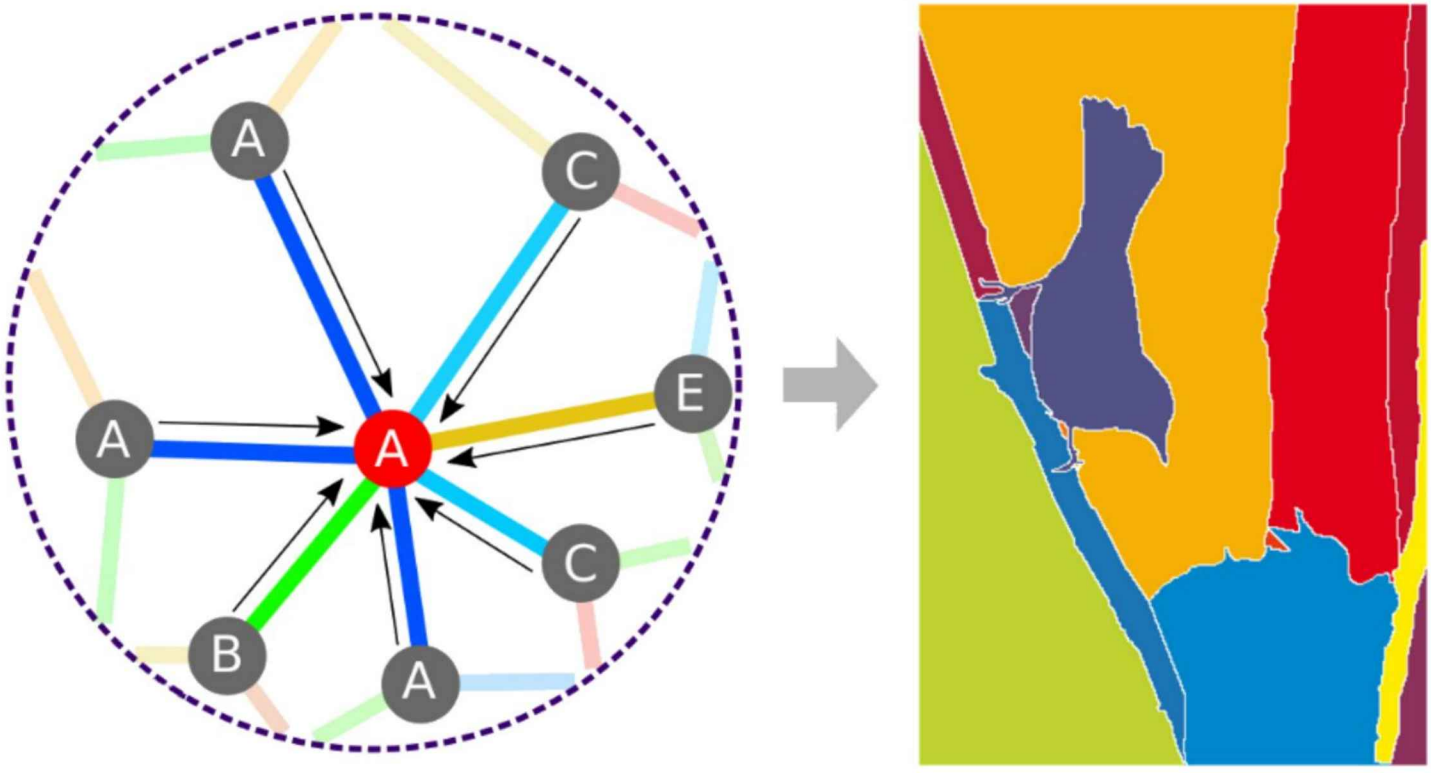
- **Propagação Recursiva de Etiquetas** (RLP)

$$f(i, L) = \begin{cases} \alpha = L_i \\ \beta = \text{argmax}_{j \in N(i)} \left(\sum L_j \right) \\ \text{If } \ \alpha \neq \beta \\ \quad L_i = \beta \\ \quad \delta_{j \in N(i)} (\ f(j, L) \ \text{ If } L_j \neq \beta \ ) \end{cases} \tag{6}$$

- **Propagação Recursiva e Ponderada de Etiquetas** (WRLP)

$$f(i, L, W) = \begin{cases} \alpha = L_i \\ \beta = \text{argmax}_{j \in N(i)} \left(\sum L_j (1 + W_{ij}) \right) \\ \text{If } \ \alpha \neq \beta \\ \quad L_i = \beta \\ \quad \delta_{j \in N(i)} (\ f(j, L, W) \ \text{ If } L_j \neq \beta \ ) \end{cases} \tag{7}$$

Complex Networks: Simple Graph Label Propagation (SGLP)



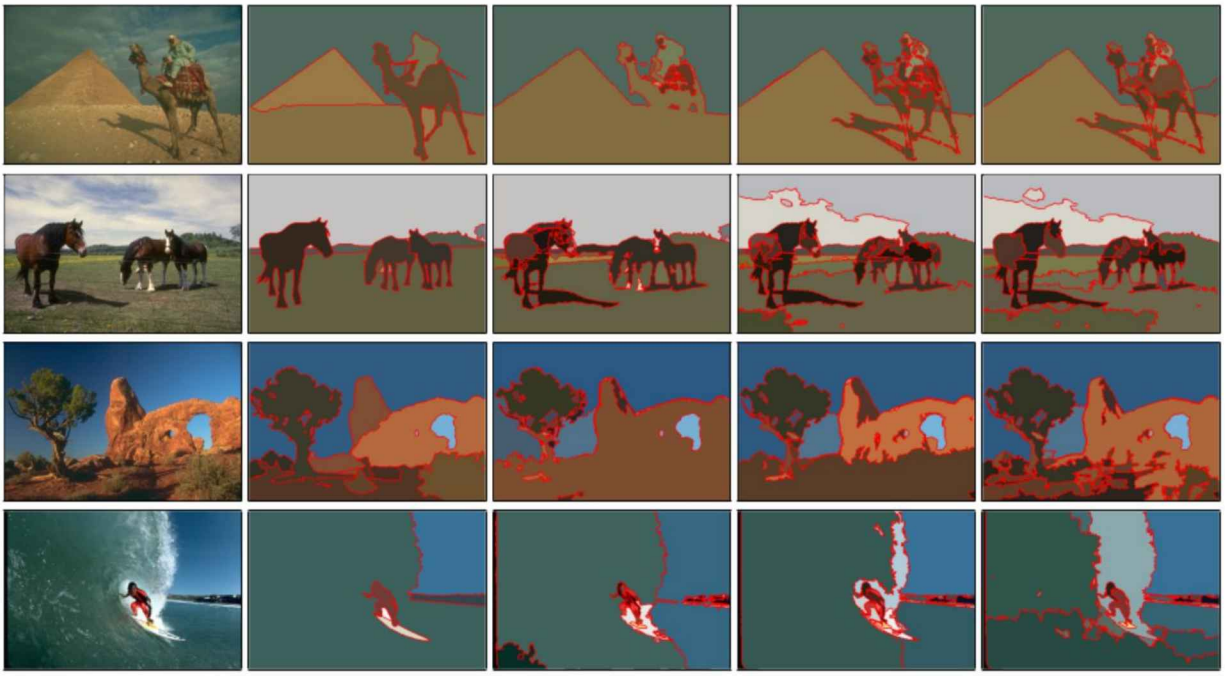


Figure: Resultados quantitativos para 4 imagens. Por coluna: imagem original, *Ground truth*, SUTP-FG, **ILP-GAU-CM** e **ILP-TAN-CM**.

Content

- Introduction
- Region-Based
  - Thresholding
  - Watershed
  - Splitting and Merging
  - Clustering
  - Graph Cut
  - Normalized Cut
  - Complex Networks
    - Speed Up Turbo Pixel with Fast Greedy (SUTP-FG)
    - Simple Graph Label Propagation (SGLP)
    - Multi-level Graph Label Propagation (MGLP)

Complex Networks: Multi-level Graph Label Propagation (MGLP)

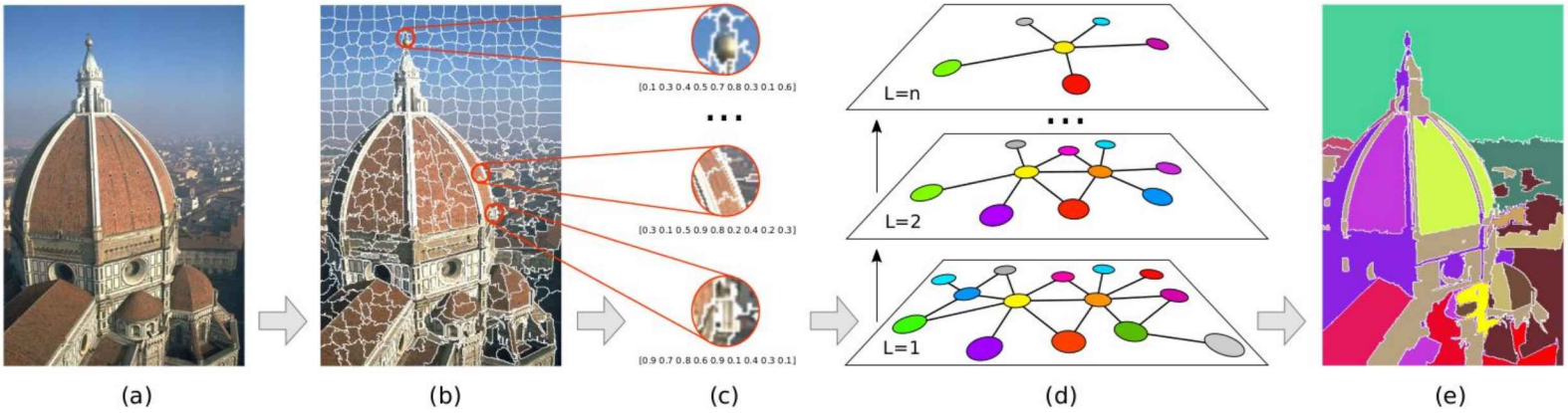
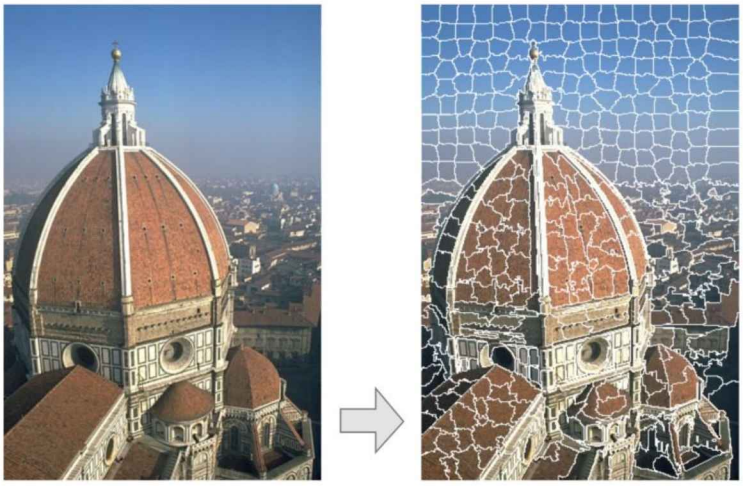


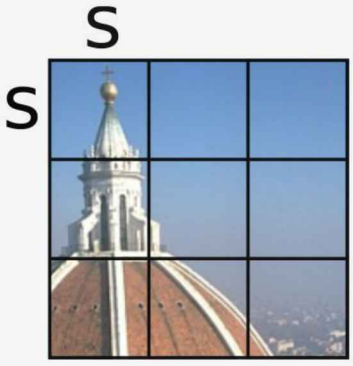
Figure: Método proposto: (a) Imagem de entrada; (b) pre-segmentação com superpixels; (c) extração de caraterísticas; (d) propagação multi-nível onde novos grafos são criados combinando superpixels similares desde níveis prévios; (e) segmentação final no último nível.

https://doi.org/10.1109/SIBGRAPI51738.2020.00034

Complex Networks: Multi-level Graph Label Propagation (MGLP)

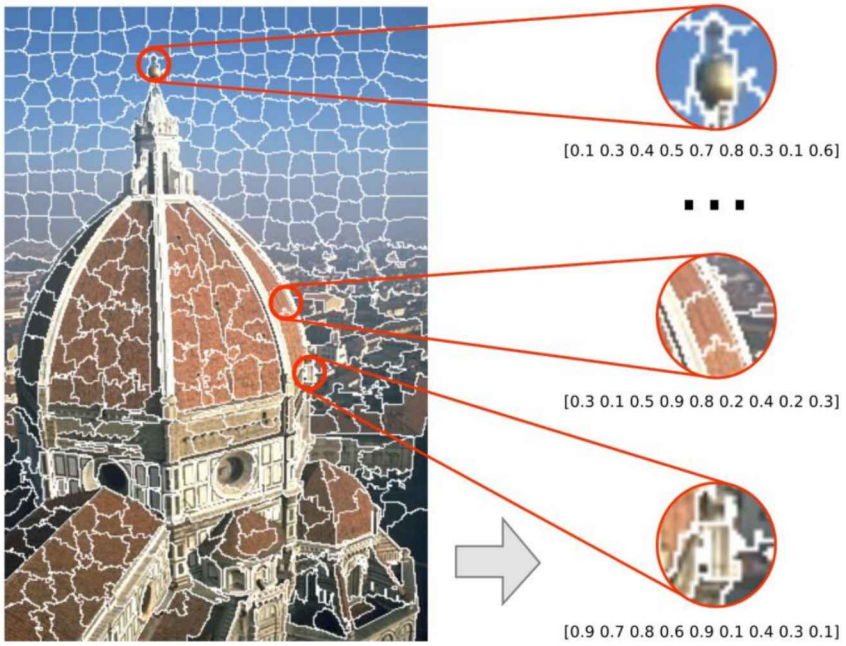


1. Simple Non-iterative Clustering (SNIC) [AS17]
2. Comprimento inicial do lado dos superpixels (S=?)



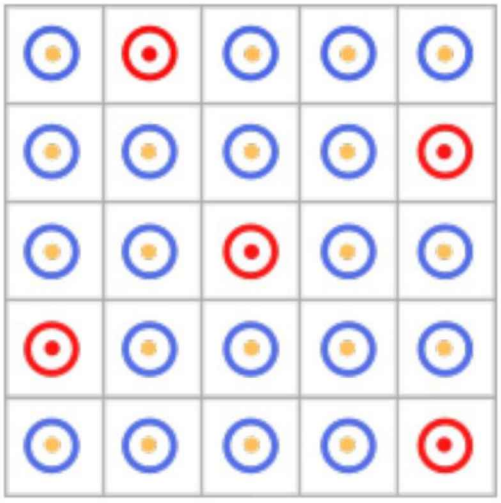
3. $R = N/S^2$

Complex Networks: Multi-level Graph Label Propagation (MGLP)

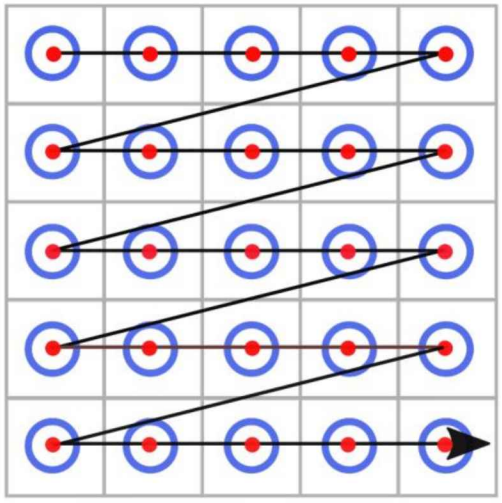


1. No espaço de cor CIELAB
2. 3 primeiros momentos estadísticos (**mean**, **variance** e **skewness**) para cada canal de cor
3. Total 9 caraterísticas por superpixel

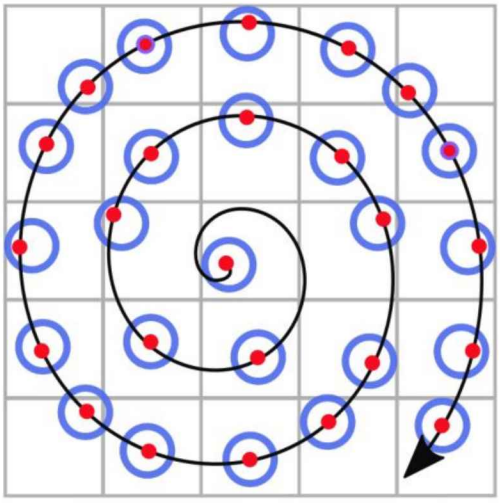
Complex Networks: Multi-level Graph Label Propagation (MGLP)



(a) Aleatório

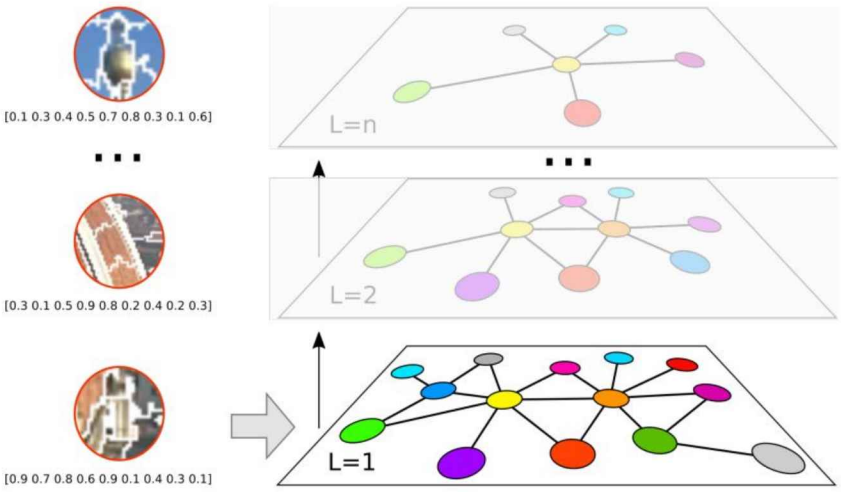


(b) **Convolução (CON)**



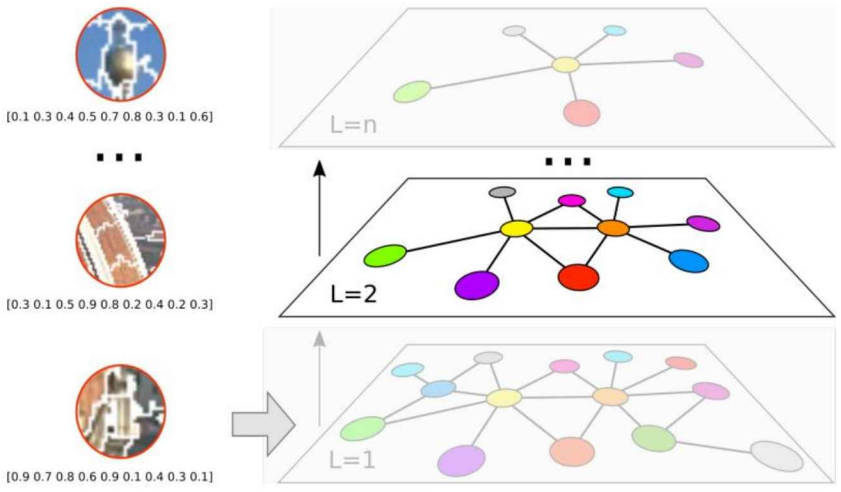
(c) **Espiral (ESP)**

Complex Networks: Multi-level Graph Label Propagation (MGLP)



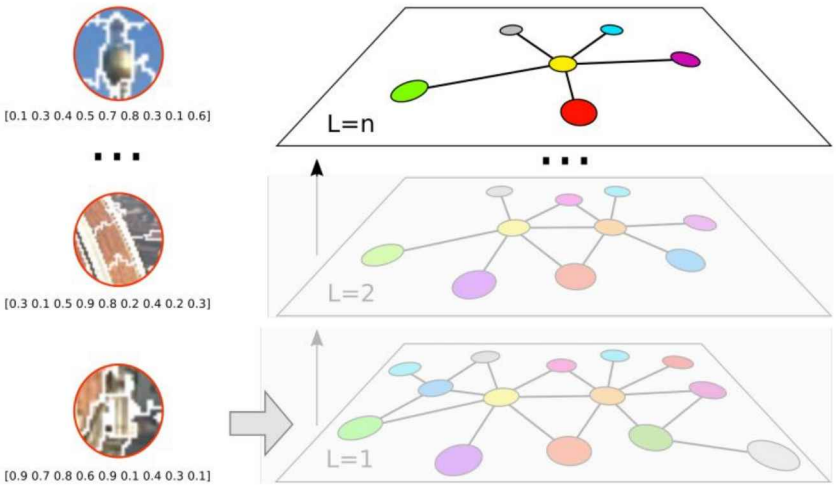
1. Criação do grafo
2. Propagação de etiquetas
3. Re-fazer os superpixels
4. Re-fazer as caraterísticas

Complex Networks: Multi-level Graph Label Propagation (MGLP)



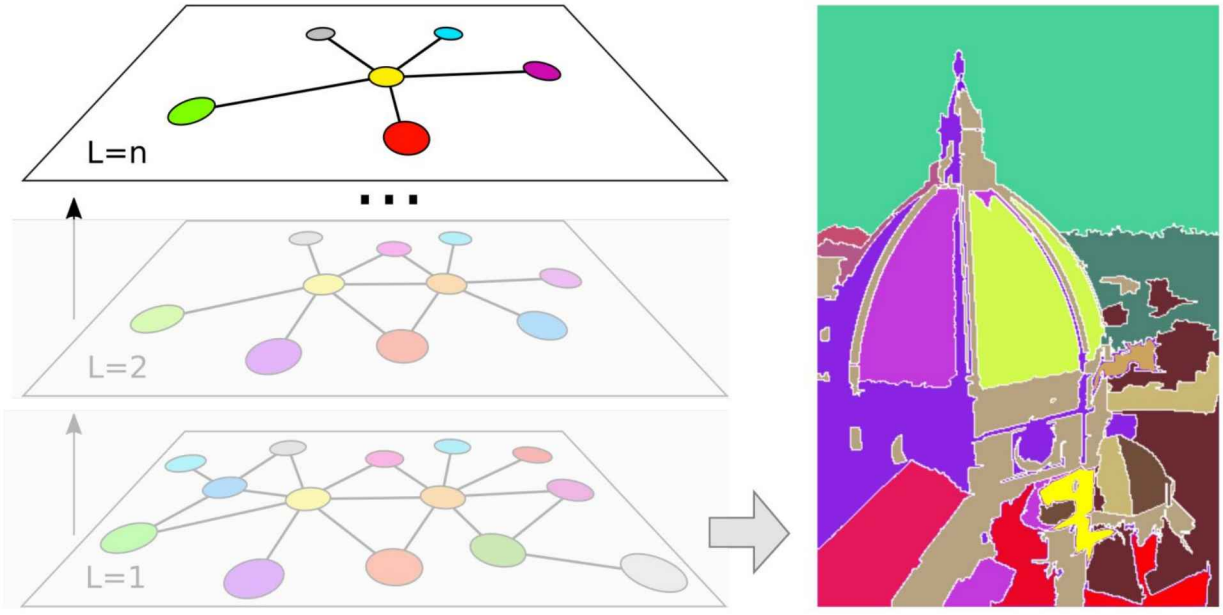
1. Criação do grafo
2. Propagação de etiquetas
3. Re-fazer superpixels
4. Re-fazer caraterísticas

Complex Networks: Multi-level Graph Label Propagation (MGLP)



1. Construção do grafo
2. Propagação de etiquetas
3. Re-fazer superpixels
4. Re-fazer caraterísticas
5. **Deter quando não exista uma redução no número de etiquetas**

Complex Networks: Multi-level Graph Label Propagation (MGLP)



A segmentação é dada no último nível, onde as regiões são criadas por pixels com as mesmas etiquetas

Complex Networks: Multi-level Graph Label Propagation (MGLP)

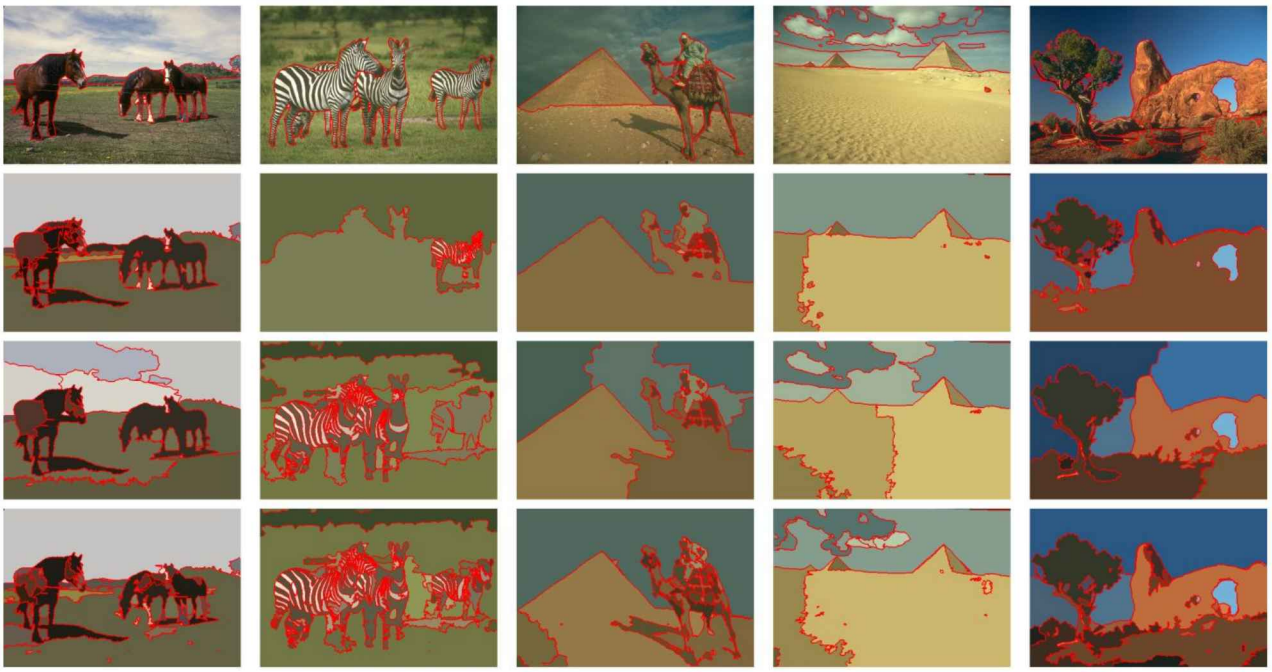


Figure: Resultados qualitativos. Por linha: Ground truth, SUTP-FG, LV-SP e **MGLP-SPI** .

Práctica 1:

https://www.kaggle.com/code/ivarvb/notebook-dccpc01?scriptVersionId=197809519

1. Filtro.
2. Histograma.
3. Binarización

Ref. Filtros
https://visioncompy.com/filtros-de-imagens-com-opencv/

Ref. Site do curso:
https://ivarvb.github.io/lectures/2024/unap-dcc104/

Universidad Nacional del Altiplano
Escuela de Posgrado
Doctorado en Ciencias de la Computación

Computer Vision

Unit 1. Image Processing:

- Morphological transformations
- Superpixels

Prof. Dr. Ivar Vargas Belizario

ivargasbelizario@gmail.com

2024 - II

Content

- Morphological transformations
- Superpixels
  - Introduction
  - Speed-Up Turbo Superpixel (SUTP)
  - Simple Linear Iterative Clustering (SLIC)
  - Simple Not Iterative Clustering (SNIC)
  - Exemplos

Morphological transformations

Erosion

https://docs.opencv.org/4.x/d9/d61/tutorial_py_morphological_ops.html
https://opencv24-python-tutorials.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_morphological_ops/py_morphological_ops.html

Morphological transformations

Dilation

https://docs.opencv.org/4.x/d9/d61/tutorial_py_morphological_ops.html
https://opencv24-python-tutorials.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_morphological_ops/py_morphological_ops.html

Content

- Morphological transformations
- Superpixels
  - Introduction
  - Speed-Up Turbo Superpixel (SUTP)
  - Simple Linear Iterative Clustering (SLIC)
  - Simple Not Iterative Clustering (SNIC)
  - Examples

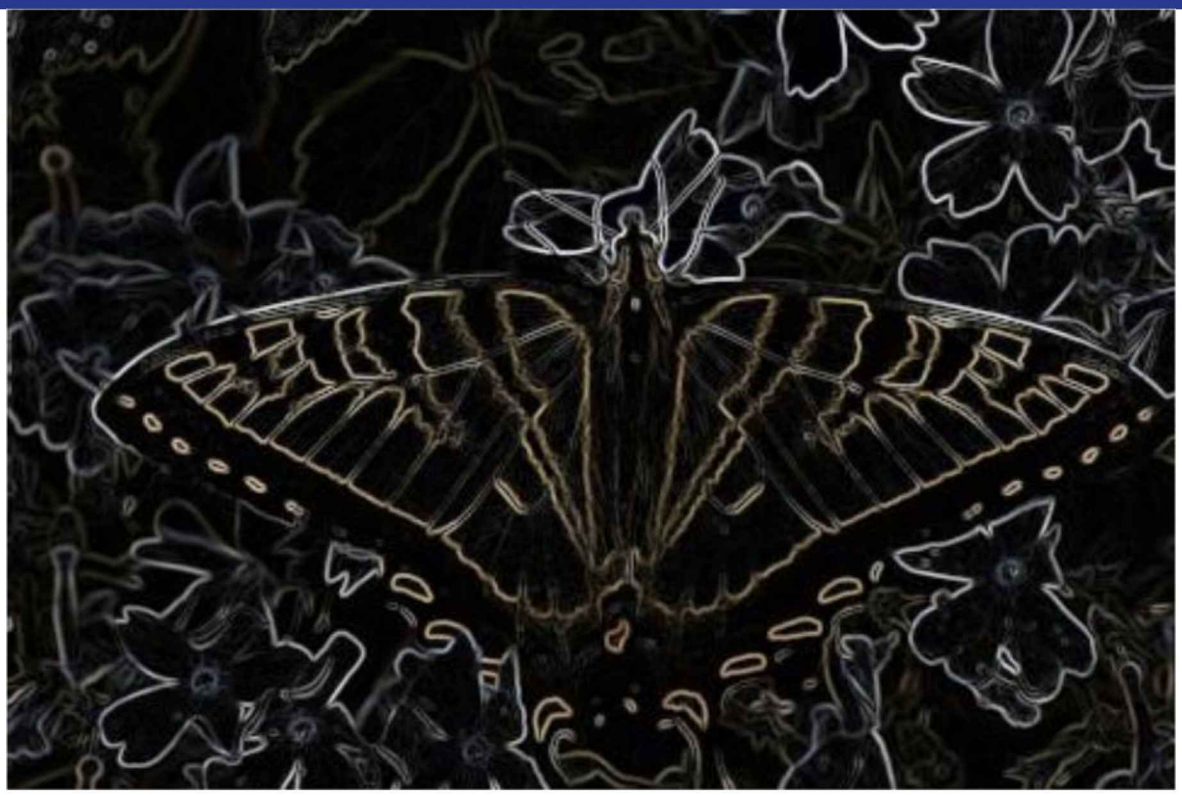
Introduction

- **Segmentação de imagens**
  - O principal objetivo da segmentação de imagens é dividir uma imagem em partes que tenham uma forte correlação com objetos ou áreas do mundo real representados na imagem.

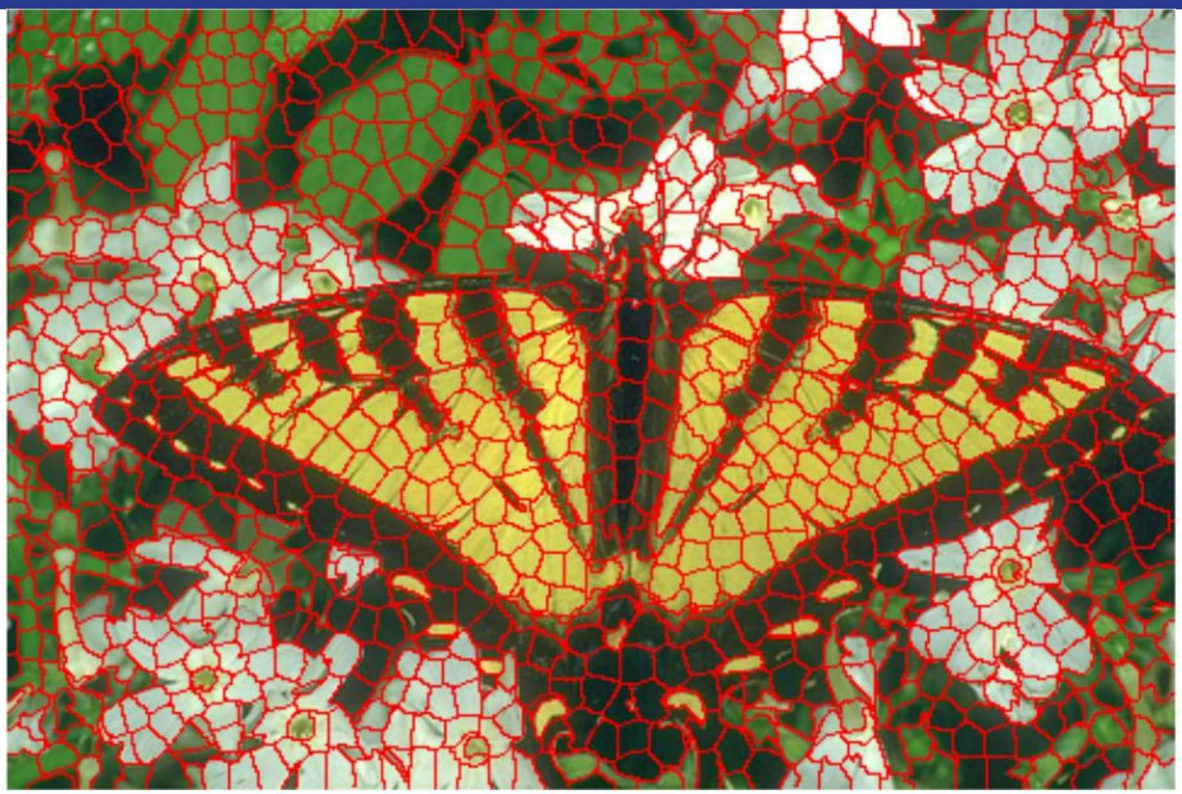
    Sonka, M.; Hlavac, V. & Boyle, R. (2014), Image Processing: Analysis and Machine Vision, CL-Engineering.

  - Tem importância na área de visão computacional por ser um dos primeiros processos para o reconhecimento de padrões.

Introduction



Introduction



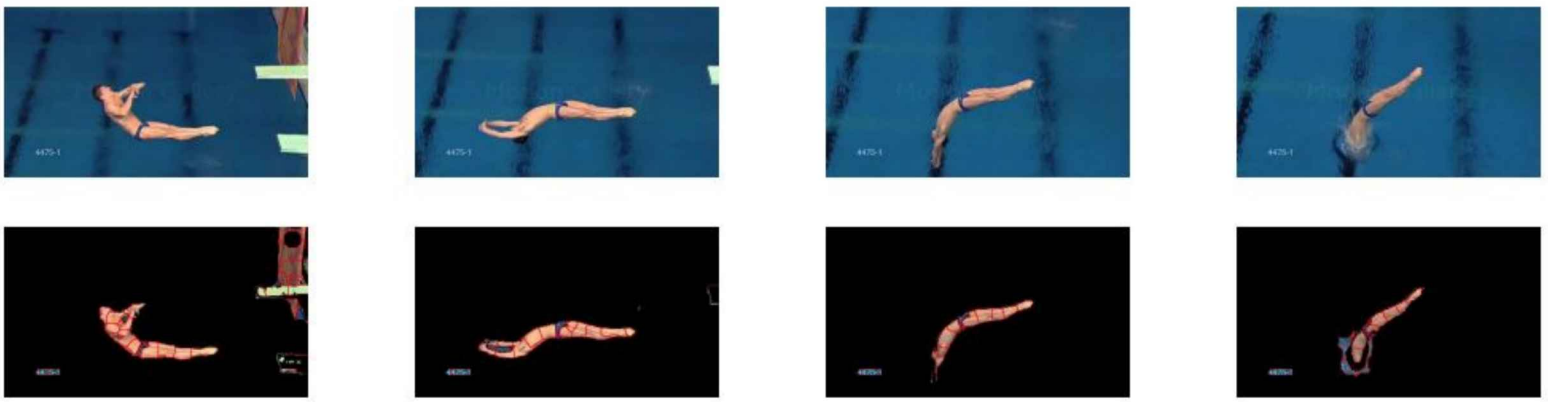
Introduction: quality of superpixels

- **Quality of superpixels**
  - ★ Costo computacional: O(n)
  - ★ Control de convexidad: compactos o convexos
  - ★ Adherencia a los bordes: com los contornos do superpixels

Introduction: Aplications

- ***Oversegmentation:*** pre-segmentação
- **Reducir la complejidad de la imagen:** para hacer tareas más complejas:
  - ★ Segmentation
  - ★ Description
  - ★ Classification
  - ★ Identification

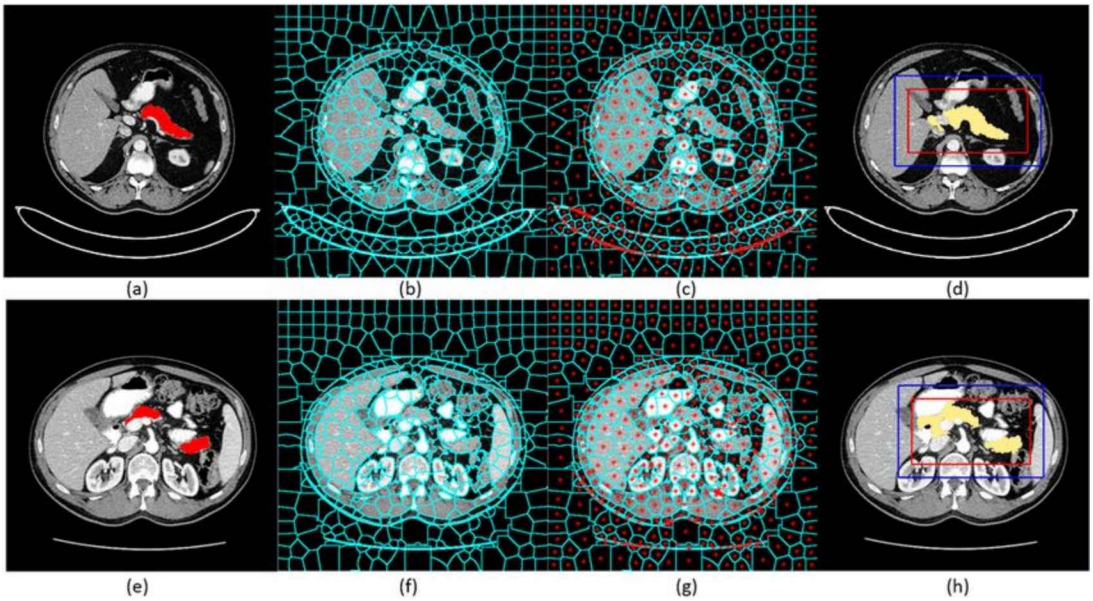
Introduction: Aplications



Temporal Superpixels based Human Action Localization

[7] https://doi.org/10.1109/ICET.2018.8603608

Introduction: Aplications



Automatic Pancreas Segmentation via Coarse Location and Ensemble Learning

[8] https://doi.org/10.1109/ACCESS.2019.2961125

Introduction: Aplications

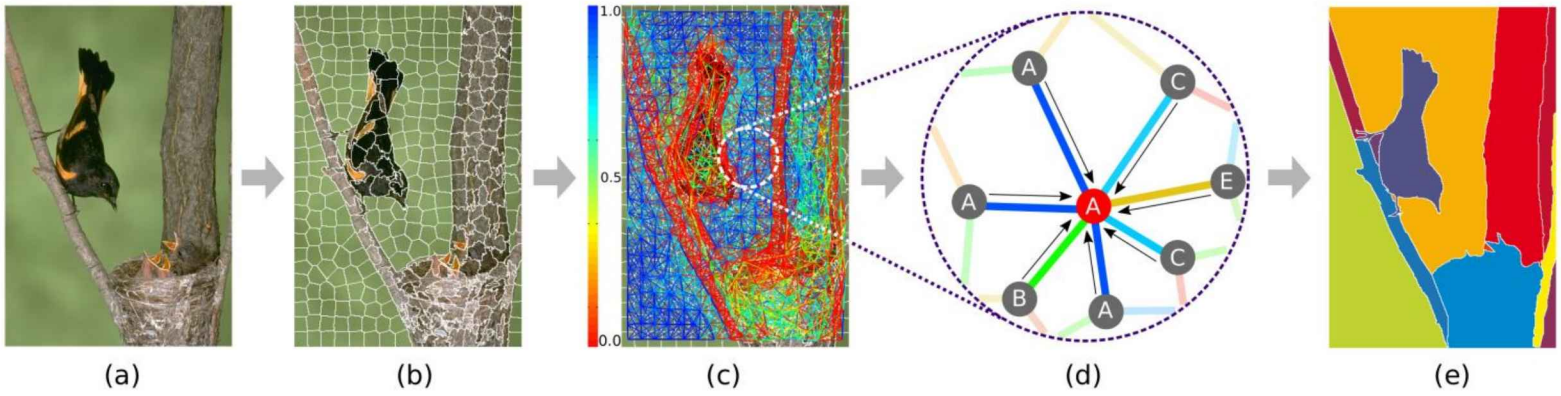


Figure: Método proposto: (a) Imagem de entrada; (b) pre-segmentação com superpixels; (c) criação do grafo; (d) propagação de etiquetas; (e) segmentação.

[9] https://doi.org/10.1049/ipr2.12242

Introduction: Aplications

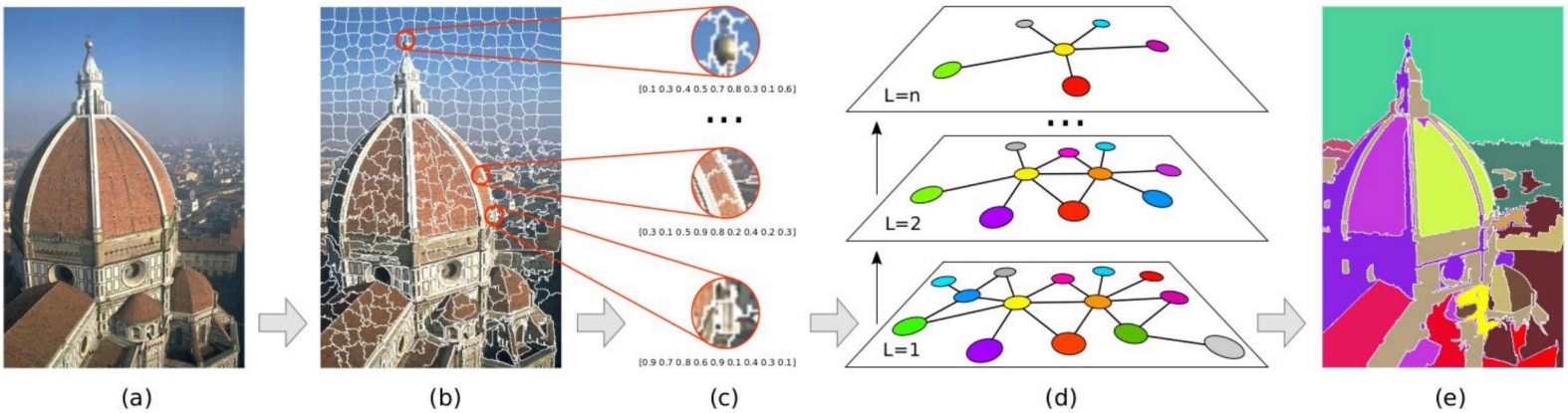


Figure: Método proposto: (a) Imagem de entrada; (b) pre-segmentação com superpixels; (c) extração de caraterísticas; (d) propagação multi-nível onde novos grafos são criados combinando superpixels similares desde níveis prévios; (e) segmentação final no último nível.

[10] https://doi.org/10.1109/SIBGRAPI51738.2020.00034

Content

- Morphological transformations
- Superpixels
  - Introduction
  - Speed-Up Turbo Superpixel (SUTP)
  - Simple Linear Iterative Clustering (SLIC)
  - Simple Not Iterative Clustering (SNIC)
  - Examples

Speed-Up Turbo Superpixel (SUTP)

EFFICIENT GRAPH-BASED IMAGE SEGMENTATION VIA SPEEDED-UP TURBO PIXELS

Cevahir Çığla and A.Aydın Alatan

Department of Electrical and Electronics Engineering M.E.T.U,Turkey
e-mail: cevahir@eee.metu.edu.tr, alatan@eee.metu.edu.tr

ABSTRACT

An efficient graph based image segmentation algorithm exploiting a novel and fast turbo pixel extraction method is introduced. The images are modeled as weighted graphs whose nodes correspond to super pixels; and normalized cuts are utilized to obtain final segmentation. Utilizing super pixels provides an efficient and compact representation; the graph complexity decreases by hundreds in terms of node number. Connected K-means with convexity constraint is the key tool for the proposed super pixel extraction. Once the pixels are grouped into super pixels, iterative bi-partitioning of the weighted graph, as introduced in normalized cuts, is performed to obtain segmentation map. Supported by various experiments, the proposed two stage segmentation scheme can be considered to be one of the most efficient graph based segmentation algorithms providing high quality results.

Index Terms— Super pixel, normalized cuts, color segmentation

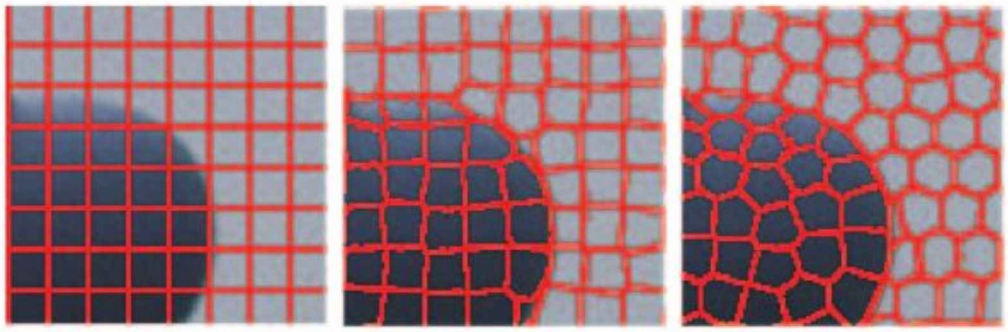
capture the details among pixels and prevent local variations to be lost in huge graphs. However, such an approach might also cause irregularities in the graph representation resulting in biases in the segmentation, as clearly explained in [5].

There are various studies on the extraction of super pixels, i.e. representing images with limited number of pixels groups [8][9]. In a recent work [8], a robust method is proposed yielding convex and quasi-uniform super pixels which are important for obtaining nearly regular graphs. In addition, quasi-uniform super pixels has an important role in preventing under-segmentation errors during graph bi-partitioning originating from tiny and irregular shaped super pixels [8]. The *TurboPixel* algorithm [8] can be accepted to be the superior over-segmentation algorithm suitable for image representation in terms of speed, compactness of segments and over-segmentation errors according to the results given in [8].

In this work, previous study [5] is improved by modifying the super pixel extraction step with a much faster method compared to [8] and [7], providing quasi-uniform over-segments with a similar performance of *TurboPixel*. The proposed method exploits a

[11 https://doi.org/10.1109/ICIP.2010.5653963

Speed-Up Turbo Superpixel (SUTP)



(a) (b) (c)

Figura. Processo para gerar superpixels com a abordagem STUP. Pixels no contorno do segmento (vermelho) são atualizadas a cada iteração. (a) Inicialização, (b) passo intermediário e (c) passo final.

Speed-Up Turbo Superpixel (SUTP)

Entrada:
- I : a imagem, N a quantidade de pixel da imagem.
- nI : número de iterações.
- k : número de *superpixels*.
- λ_1 : variável de ponderação para a similaridade entre as intensidades dos *pixels* e *superpixels*.
- λ_2 : variável de ponderação para a restrição da convexidade do *superpixel*.

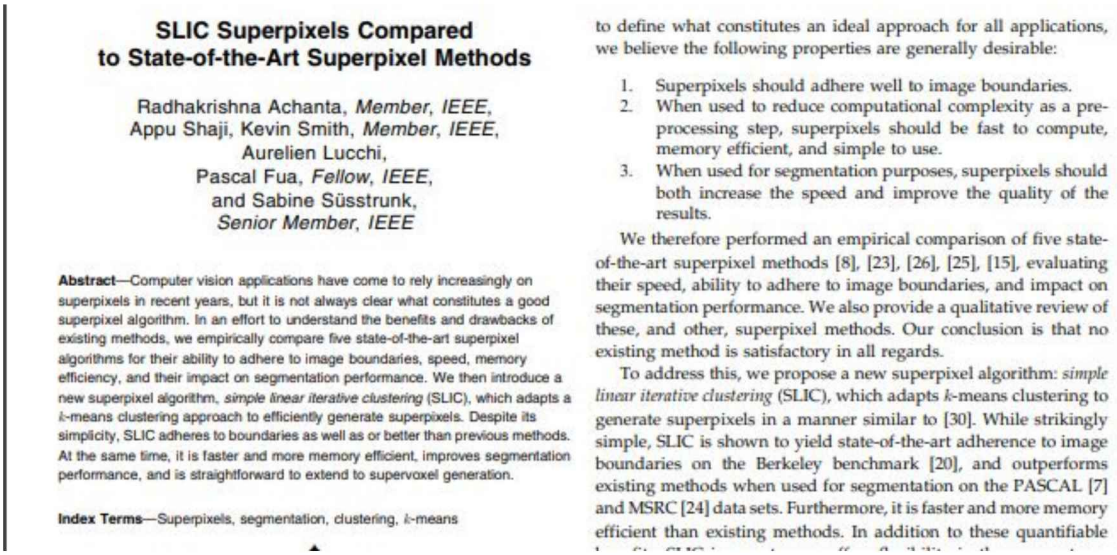
Saída:
- $l(i)$: i *pixels* agrupados em k *superpixels*

```
início
    S = √(N/k)                    /* calcular o comprimento aproximado do lado dos superpixels */
    grid(I,S)                     /* particionar a imagem em regiões retangulares */
    l(i) = k                      /* agrupar os i pixels com os respectivos k superpixels */
    C_k = [I_k, x_k, y_k]         /* inicializar os k centroides */
    enquanto o nI tenha sido completado faça
        d(i) = ∞                  /* inicializar as distâncias para cada pixel i */
        para cada superpixel k faça
            para cada pixel i que está sobre o contorno do superpixel k faça
                D = λ1|I(i) − I_k| + λ2|(x_i − x_k)^2 + (y_i − y_k)^2|
                se D < d(i) então
                    d(i) = D      /* atualizar a distância mínima entre C_k e i */
                    l(i) = k      /* agrupar o pixel i no superpixel k */
                fim
            fim
        fim
        atualizarCentroides()     /* atualizar os k centroides */
    fim
fim
```

Content

- Morphological transformations
- Superpixels
  - Introduction
  - Speed-Up Turbo Superpixel (SUTP)
  - Simple Linear Iterative Clustering (SLIC)
  - Simple Not Iterative Clustering (SNIC)
  - Examples

Simple Linear Iterative Clustering (SLIC)



SLIC Superpixels Compared to State-of-the-Art Superpixel Methods

Radhakrishna Achanta, *Member, IEEE*,
Appu Shaji, Kevin Smith, *Member, IEEE*,
Aurelien Lucchi,
Pascal Fua, *Fellow, IEEE*,
and Sabine Süsstrunk,
Senior Member, IEEE

Abstract—Computer vision applications have come to rely increasingly on superpixels in recent years, but it is not always clear what constitutes a good superpixel algorithm. In an effort to understand the benefits and drawbacks of existing methods, we empirically compare five state-of-the-art superpixel algorithms for their ability to adhere to image boundaries, speed, memory efficiency, and their impact on segmentation performance. We then introduce a new superpixel algorithm, *simple linear iterative clustering* (SLIC), which adapts a k-means clustering approach to efficiently generate superpixels. Despite its simplicity, SLIC adheres to boundaries as well as or better than previous methods. At the same time, it is faster and more memory efficient, improves segmentation performance, and is straightforward to extend to supervoxel generation.

Index Terms—Superpixels, segmentation, clustering, k-means

to define what constitutes an ideal approach for all applications, we believe the following properties are generally desirable:

1. Superpixels should adhere well to image boundaries.
2. When used to reduce computational complexity as a preprocessing step, superpixels should be fast to compute, memory efficient, and simple to use.
3. When used for segmentation purposes, superpixels should both increase the speed and improve the quality of the results.

We therefore performed an empirical comparison of five state-of-the-art superpixel methods [8], [23], [26], [25], [15], evaluating their speed, ability to adhere to image boundaries, and impact on segmentation performance. We also provide a qualitative review of these, and other, superpixel methods. Our conclusion is that no existing method is satisfactory in all regards.

To address this, we propose a new superpixel algorithm: *simple linear iterative clustering* (SLIC), which adapts k-means clustering to generate superpixels in a manner similar to [30]. While strikingly simple, SLIC is shown to yield state-of-the-art adherence to image boundaries on the Berkeley benchmark [20], and outperforms existing methods when used for segmentation on the PASCAL [7] and MSRC [24] data sets. Furthermore, it is faster and more memory efficient than existing methods. In addition to these quantifiable

[12] https://doi.org/10.1109/TPAMI.2012.120

Simple Linear Iterative Clustering (SLIC)

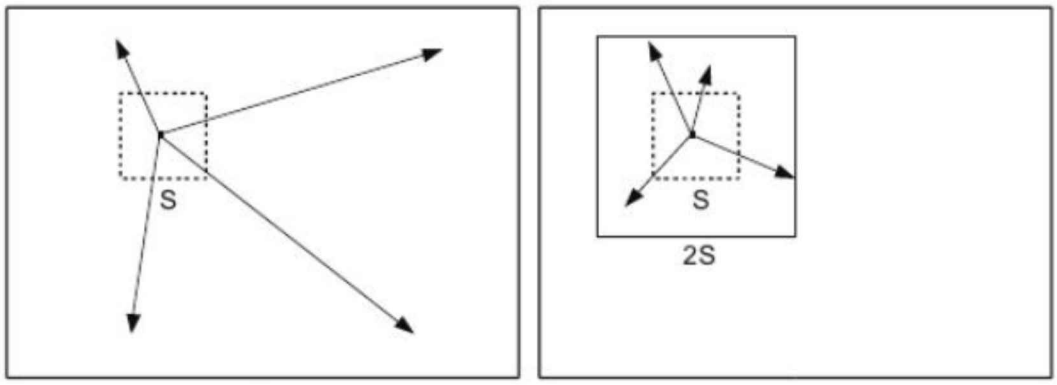


Figura. Redução do espaço da busca na imagem para o refinamento dos superpixels: (a) espaço de busca empregado por um algoritmo k-means convencional, (b) espaço de busca reduzido utilizado por o SLIC representado por a região 2Sx2S.

Simple Linear Iterative Clustering (SLIC)

Entrada:
I : a imagem, N a quantidade de pixel da imagem.
nI : número de iterações.
k : número de *superpixels*.
m : constante que garante a qualidade dos *superpixels*.

Saída:
$l(i)$: i *pixels* agrupados em k *superpixels*

início
  $S = \sqrt{\frac{N}{k}}$ /* calcular o comprimento aproximado do lado dos *superpixels* */
  $grid(I,S)$ /* particionar a imagem em regiões retangulares */
  $l(i) = -1$ /* para cada pixel i */
  $C_k = [l_k, a_k, b_k, x_k, y_k]$ /* inicializar os k centroides */
  enquanto *o nI tenha sido completado* **faça**
    $d(i) = \infty$ /* inicializar as distâncias para cada pixel i */
    para cada *centroide* C_k **faça**
      para cada *pixel i da região 2Sx2S em torno ao* C_k **faça**
        $d_c = \sqrt{(l_k - l_i)^2 + (a_k - a_i)^2 + (b_k - b_i)^2}$
        $d_s = \sqrt{(x_k - x_i)^2 + (y_k - y_i)^2}$
        $D = \sqrt{d_c^2 + (\frac{d_s}{S})^2 m^2}$ /* calcular distância D entre C_k e i */
        se $D < d(i)$ **então**
          $d(i) = D$ /* atualizar a distância mínima entre C_k e i */
          $l(i) = k$ /* agrupar o *pixel* i no *superpixel* k */
        fim
      fim
    fim
    atualizarCentroides() /* atualizar os k centroides */
  fim
  corrigirError()
fim

SUTP vs SLIC

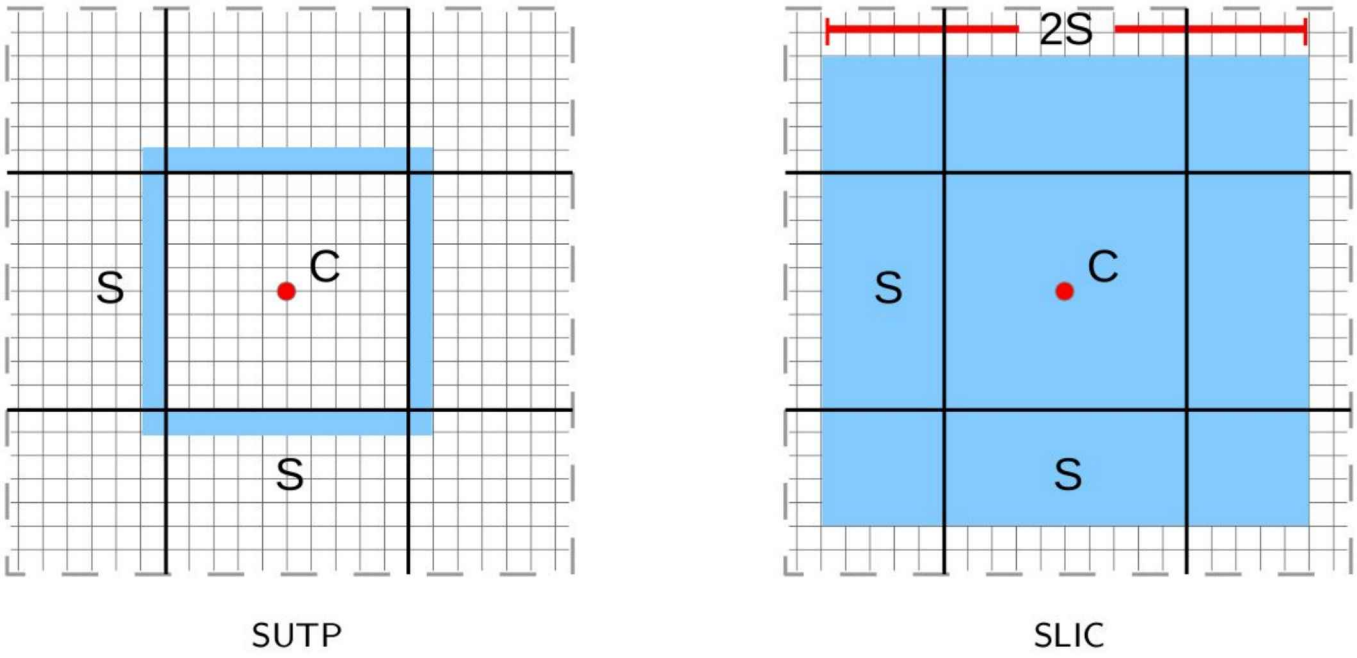
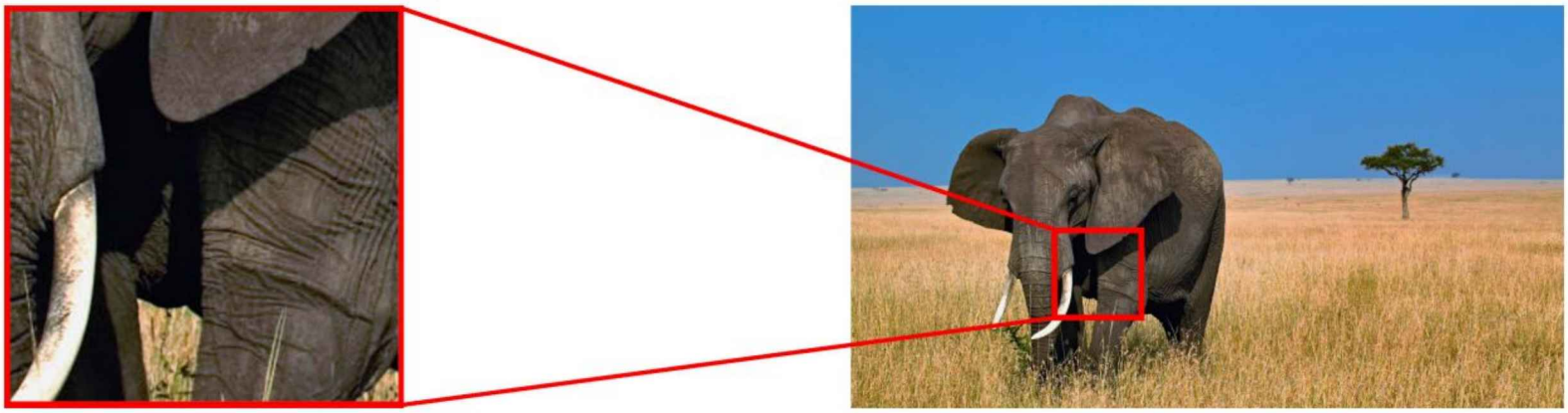


Figura. Intercâmbio de pixels para o SUTP e SLIC: centróide (C), lado do superpixel (S).

SUTP vs SLIC



SUTP vs SLIC

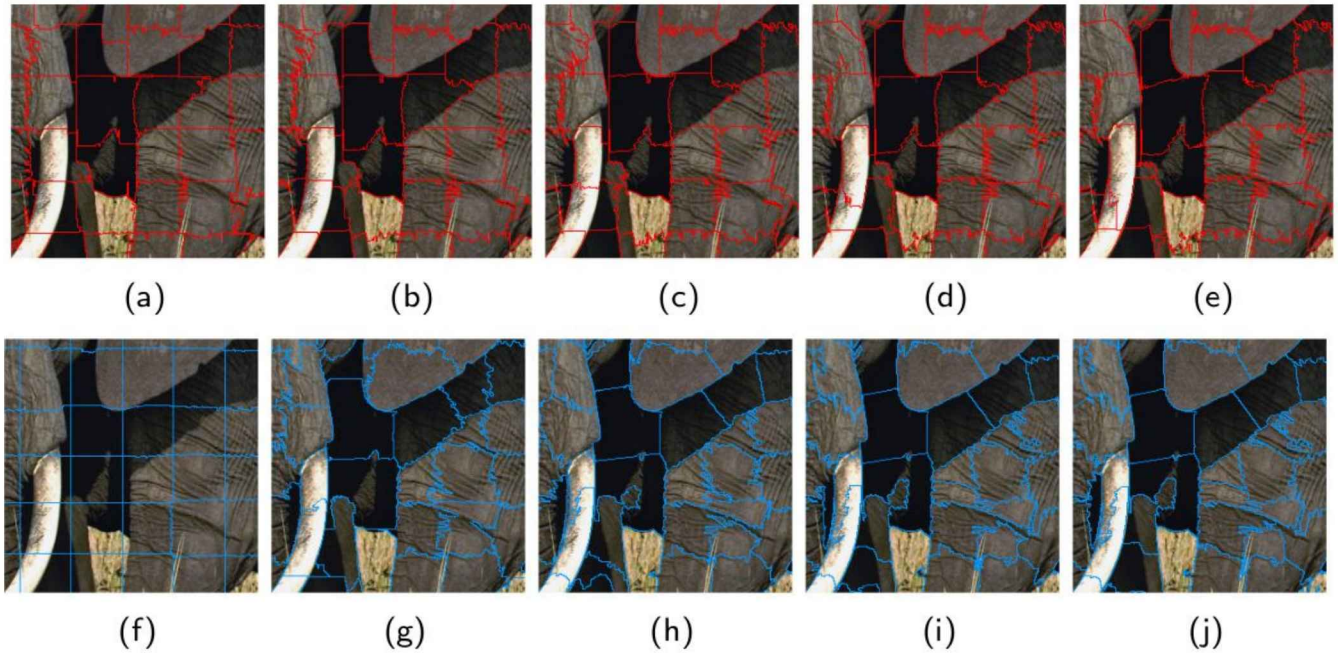


Figura. Comparação dos resultados do SUTP e do SLIC para as 5 primeiras iterações: (a-e) resultados SUTP, (f-j) resultados SLIC. s = 50.

Content

- Morphological transformations
- Superpixels
  - Introduction
  - Speed-Up Turbo Superpixel (SUTP)
  - Simple Linear Iterative Clustering (SLIC)
  - Simple Not Iterative Clustering (SNIC)
  - Examples

Simple Not Iterative Clustering (SNIC)

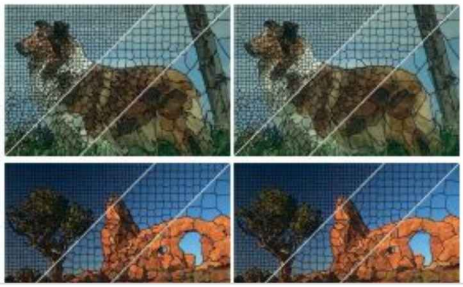
2017 IEEE Conference on Computer Vision and Pattern Recognition

Superpixels and Polygons using Simple Non-Iterative Clustering

Radhakrishna Achanta and Sabine Süsstrunk
School of Computer and Communication Sciences (IC)
École Polytechnique Fédérale de Lausanne (EPFL)
Switzerland
{radhakrishna.achanta,sabine.susstrunk}@epfl.ch

Abstract

We present an improved version of the Simple Linear Iterative Clustering (SLIC) superpixel segmentation. Unlike SLIC, our algorithm is non-iterative, enforces connectivity from the start, requires lesser memory, and is faster. Relying on the superpixel boundaries obtained using our algorithm, we also present a polygonal partitioning algorithm. We demonstrate that our superpixels as well as the polygonal partitioning are superior to the respective state-of-the-art algorithms on quantitative benchmarks.



[13] https://doi.org/10.1109/CVPR.2017.520

Simple Not Iterative Clustering (SNIC)

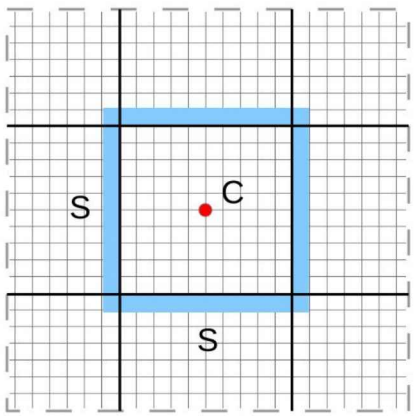
1. Definir uma **fila de prioridade** baseada em distância
   a. Q = PriorityQueue()
2. Dividir a imagem SXS:
   a. Inicializar os centroides: C[l, a, b, y, x, k, z]
   b. **Q.put**((**-1**, (cy, cx, k))) `//inserir os centroides`
3. L[,] = -1 `//iniciar etiquetas dos pixels`
4. Enquanto Q não estiver vazia:
   a. {iy, ix, k} <- **Q.get**() `//obter o pixel prioritário`
   b. Se L[iy, ix] == -1: `//se o pixel não tiver etiqueta`
      i. C[k][l, a, b, y, x, k, z] <- {lab(iy, ix), iy, ix, k,1}
      `//atualizar o centróide com a informação do pixel`
      ii. L[iy, ix] = k `//atualizar etiqueta do pixel`
      iii. Para os j vizinhos do pixel (iy, ix):
      - Se L[jy, jx] == -1: `//se o vizinho não tiver etiqueta`
        a. $\mathbf{d_{jk}}$ `//computar distância de j para o C[k]`
        b. **Q.put**((**d**, (jy, jx, k))) `//inserir o pixel j`

Simple Not Iterative Clustering (SNIC)

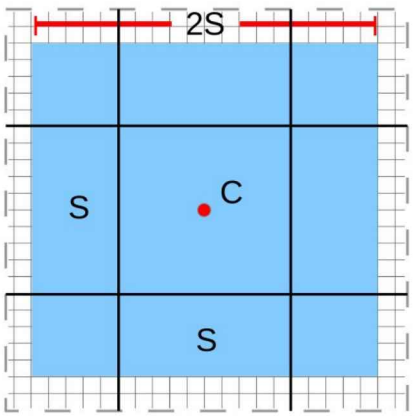
$$d_{j,k} = \sqrt{\frac{||x_j, x_k||_2^{\,2}}{s} + \frac{||c_j, c_k||_2^{\,2}}{m}}$$

onde, S é o comprimento inicial dos superpixels, X_j faz referência à localização do pixel j, X_k faz referência à média da localização do centróide k, c_j faz referência ao vetor de intensidades da cor do pixel j, c_k representa à média das intensidades de cor do centroide k, o m é um fator de compactação fornecido pelo usuário. Quanto maior o m, mais compactos (convexos) serão os superpixels mas com uma baixa aderência às bordas. Quanto menor o m, mais irregulares são os superpixels mas com uma alta aderência às bordas.

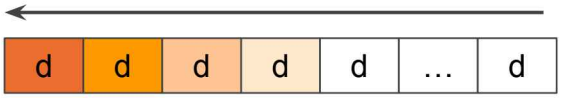
Simple Not Iterative Clustering (SNIC)



SUTP



SLIC



Q = PriorityQueue()
d = menor distância do pixel em relação a um determinado centróide

SNIC

Content

- Morphological transformations
- Superpixels
  - Introduction
  - Speed-Up Turbo Superpixel (SUTP)
  - Simple Linear Iterative Clustering (SLIC)
  - Simple Not Iterative Clustering (SNIC)
  - Examples

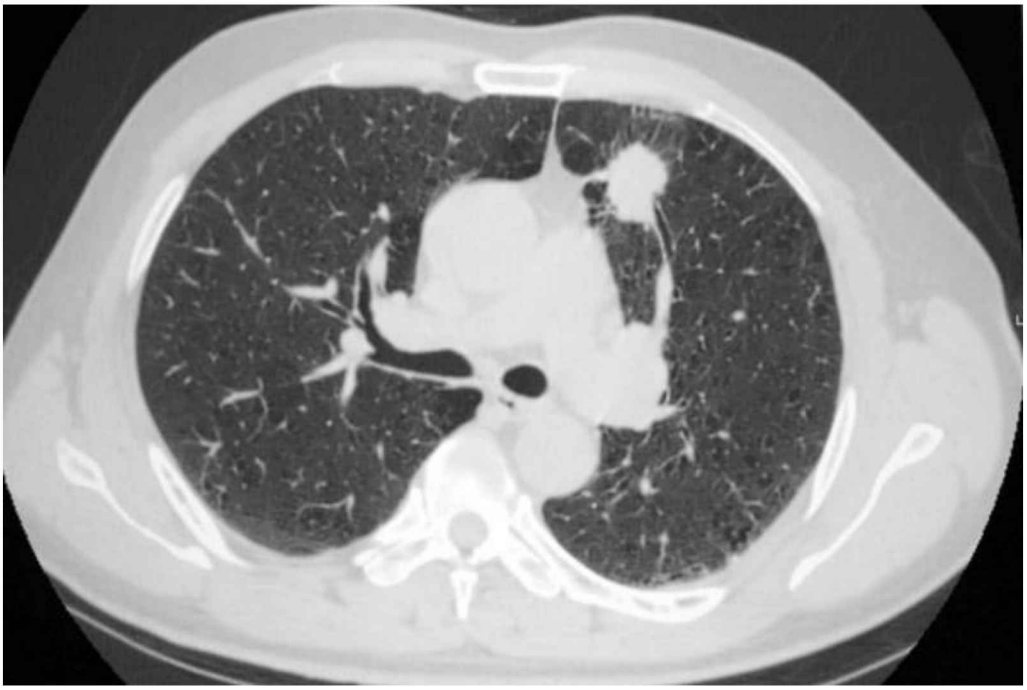
Examples

Examples

Examples

Examples

Examples



Examples

- **SLIC**
  - ★ https://www.kaggle.com/ivarvb/slic-sp
- **SNIC**
  - ★ https://www.kaggle.com/ivarvb/snic-sp

Ref.
https://pyimagesearch.com/2017/06/26/labeling-superpixel-colorfulness-opencv-python/