Practical Computer Vision: Theory & Applications

Carmen Alonso Montes
23rd-27th November 2015
Wrap up

Today, we are still here

Operational Level

LOW LEVEL

Computer Vision Stages

Acquisition

Preprocessing

Detection & Segmentation

High-level processing (recognition, registration, etc)
Learned concepts

- Image binarization
- Thresholding
- Edge detection
- Morphological operators
Contents

- Wrap up
- Image Transforms
  - Distance transform
- Watershed Algorithm
  - Landscape/Waterfall/
- Active contour model (Snakes)
- Clustering techniques: k-means
- Summary
- Practical exercises
Hough Transform
The classical Hough transform [Richard Duda et al.] was concerned with the identification of lines in the image, but later the Hough transform has been extended to identifying positions of arbitrary shapes, most commonly circles or ellipses.

**Goal:** to find imperfect instances of objects within a certain class of shapes by a **voting procedure**.

This voting procedure is carried out in a parameter space, from which object candidates are obtained as local maxima in a so-called accumulator space that is explicitly constructed by the algorithm for computing the Hough transform.
The simplest case of Hough transform is detecting **straight lines**. A straight line $y = mx + b$ can be represented as a point $(b, m)$ in the parameter space.

$$ r = x \cos \theta + y \sin \theta $$

It is therefore possible to associate with each line of the image a pair $(r, \theta)$. The $(r, \theta)$ plane is referred to as **Hough space** for the set of straight lines in two dimensions.

Given a single point in the plane, then the set of all straight lines going through that point corresponds to a sinusoidal curve in the $(r, \theta)$ plane, which is unique to that point.

A set of two or more points that form a straight line will produce sinusoids which cross at the $(r, \theta)$ for that line.

Thus, the problem of detecting collinear points can be converted to the problem of finding concurrent curves.
Hough Transform

- The linear Hough transform algorithm uses a two-dimensional array, called an **accumulator**, to detect the existence of a line described by \( r = x \cos \theta + y \sin \theta \)

  - The dimension of the accumulator equals the number of unknown parameters
  - For each pixel at \((x, y)\) and its neighborhood, the Hough transform algorithm determines if there is enough evidence of a straight line at that pixel.
    - If yes → vote + in the accumulator's bin
    - Bins with the highest values → local maxima in the accumulator space → the most likely lines can be extracted

- The final result of the linear Hough transform is a matrix similar to the accumulator
  - angle \( \theta \)
  - distance \( r \).

- Each element of the matrix has a value equal to the number of points or pixels that are positioned on the line represented by quantized parameters \((r, \theta)\). So the element with the highest value indicates the straight line that is most represented in the input image

- See:
Example

Find peaks in the Hough transform of the image.
Distance transform
The **distance transform (distance map or distance field)** is an operator normally only applied to binary images.

The result of the transform is a graylevel image where foreground point intensity shows the distance to the closest **boundary**.

Distance fields can also be signed, in the case where it is important to distinguish whether the point is inside or outside of the shape.

Important concepts to have them clear:
- Pixel neighbourhood and connectivity
- Pixel distance computation

**4-neighbors** $N_4(p)$
Distance transform (II)

- Usually the transform/map is qualified with the chosen metric.
  - Euclidean distance
  - City block distance or Manhattan distance.
  - Chessboard distance
- It can be made by morphological operations, like multiple successive erosions.
  - Erode till all foreground regions have been completely removed.
  - Each pixel is labeled with the number of performed erosions (distance transform)
- **Advantages**: easy to implement and intuitive
- **Disadvantages**: extremely inefficient.
- **Applications**: blurring effects, skeletonizing, motion planning in robotics, pathfinding, etc

![Distance transform example]
Examples (I)

Bwdist
Subimage
Imcontour
mat2gray

Euclidean
City block
Chessboard
Quasi-Euclidean
Examples (II)
Watershed Algorithm
A watershed is a basin-like **landform** defined by **highpoints** and **ridgelines** that descend into lower elevations and stream valleys.

A gray-scale image can be viewed as a landscape, where the heights are given by the grey levels in the image.

- **Idea**: A drop of water following the gradient of an image flows along a path to finally reach a local minimum.

It is used in image processing primarily for segmentation purposes.

http://accu.org/index.php/journals/1469
Fill with water → Objective: to have a lake

Water rising → pools of water (catchment basin) will gradually form at each of the minima

When catchment basins meet at a point → a dam or watershed is “built”

Fully flooded landscape → the created dams will segment the different regional minima from each other at the points where their catchment basins would have met

http://goo.gl/p9e18t
Watershed: An example

Segmentation result

http://goo.gl/p9e18t
Watershed usage

- In **medical images**, organs tend to be relatively homogeneous in the grey levels throughout the organ
  - The gradient at the edges of organs will be quite significant
  - Using the gradient image combined with watershed technique is quite interesting
    - Organ is located in the valleys of the image which are surrounded by hills.
- **Rainfall simulation**: alternative implementation and perspective instead of using a flooding method is only one way of approaching the problem.
  - **Idea**: to imagine rain falling on each point of the landscape from above.
  - Water always take the path with least resistance
  - Water runs downhill to a regional minimum via a path of steepest descent

![Diagram of rainfall simulation]

- Water here may flow either way.
Disadvantages

- Big issues with watershed approaches
  - There's no reason to suppose that the things we want to segment (e.g. organs in a medical image) will have low grey levels and thus be in a 'valley'.
    - If they are on top of a 'hill', we're apparently stuffed.
  - The idea is simpler, but the implementation is hard.
  - **Oversegmentation** due to the presence of noise and regional minima.
Active Contours (Snakes)
Active contour model (Snakes)

- **Active contour model**, also called **snakes**, is a framework in computer vision for delineating an object boundary from a possibly noisy 2D image. Contour models describe the boundaries of shapes in an image.

- **Applications**: object tracking, shape recognition, segmentation, edge detection and stereo matching.

http://personal.ee.surrey.ac.uk/Personal/R.Bowden/publications/bmvc97/image4.gif
Active Contours: Video examples

https://www.youtube.com/watch?v=CeU_yZjdVqY

http://personalpages.manchester.ac.uk/staff/p.dudek/scamp2/default.htm
An active contour (snake) can be conceptually defined as an **elastic curve** that evolves from its initial shape and position as a result of the combined action of external and internal forces [Kass et al. 1998]

- An energy functional $E$ is associated with the curve
- The problem of finding object boundary is an energy minimization problem.

- Imaging forces (behaviour of the snake):
  - **External forces**: push towards the object contours (target)
  - **Internal forces**: keep the shape to avoid deformation
- Contour initialization
  - Prior knowledge over the location of the objects is needed
  - Depending on their location, this will affect on how the energy equations are configured
Active contour schema

- Basic model is a **closed contour**
- Made by **nodes (control points)** and **lines** that connect them
- Nodes have the ability to **move** and deform towards our target
- The lines are intended to keep the topological relationship among the nodes
- The contour will move/evolve towards the direction of the target
  - The target needs to be properly defined
- Our **target** usually corresponds to the boundaries of an object we want to segment
How does the contour move?

- The contour moves towards our target object.
- If we define a distance map based on gradients computation, we have defined a path where the contour will flow.
- One of the constraints in movement is that the topological relationship shall be maintained.
- The classical snakes model involves an edge detector, which depends on the gradient of the image \( u_0 \), to stop the evolving curve at the boundary of the object.
The parametric representation of the active contour is described as a curve:

$$u(s) = (x(s), y(s)), \quad s \in [0, 1]$$

- $x$ and $y$ are the coordinate functions
- $s \in [0, 1]$ is the parametric domain.

The shape of the contour subject to an image $l(x, y)$ is associated with the function:

$$E_{\text{snake}} = E_{\text{internal}} + E_{\text{external}}$$

This function can be viewed as a representation of the energy of the contour in such a way that the final shape of the contour corresponds to the minimum of this energy.
The internal energy of the snake depends on the intrinsic properties of the curve.

It characterises the deformation of a stretchy, flexible contour, and it controls

- continuity of the contour
- the smoothness of the contour

The equation of the internal deformation energy is the sum of elastic energy and bending energy

\[
\int_0^1 \left( \alpha \left| \frac{\partial u}{\partial s} \right|^2 + \beta \left| \frac{\partial^2 u}{\partial s^2} \right| \right) ds
\]

The parameters \( \alpha \) and \( \beta \) control the tension and rigidity of the contour, respectively.

- A large weight \( \alpha \) for the continuity term penalizes changes in distances between points in the contour.
- A large weight \( \beta \) for the smoothness term penalizes oscillations in the contour and will cause the contour to act as a thin plate.
Elastic & Bending Energy

- **Elastic Energy**
  - The curve is treated as an elastic rubber band possessing elastic potential energy.
  - It discourages stretching by introducing tension.
  - Responsible for shrinking of the contour.

- **Bending Energy**
  - The snake is also considered to behave like a thin metal strip giving rise to bending energy.
  - Bending energy is minimum for a circle.
  - Total internal energy of the snake can be defined as

\[
E_{\text{int}} = E_{\text{elastic}} + E_{\text{bending}}
\]
External energy

- The external energy guides the contour evolution towards the boundaries of interest. It is derived from the image.

- Define a function that takes its smaller values at the features of interest, such as boundaries.

\[
\int_{0}^{1} P_{\text{ext}}(u(s)) \, ds
\]

- where \( P(x, y) \) denotes a scalar potential function defined on the image plane.

- External potentials are designed in a way that a local minimum coincides with edges or other features of interest, depending on the problem under study.
Internal vs External Energy (I)

Evolution using only internal forces

Initial Contour

Guiding Information

Evolution using only external forces

Initial Contour
Internal vs External Energy (II)
The problem at hand is to find a contour that minimize the energy functional

$$E_{\text{snake}} = E_{\text{internal}} + E_{\text{external}} + E_{\text{constraint}}$$

$$\mathbf{v}(s) = (x(s), y(s)) \quad E_{\text{snake}} = \int_s \frac{1}{2} (\alpha(s) | v_s |^2 + \beta(s) | v_{ss} |^2) + E_{\text{image}}(\mathbf{v}(s))\,ds$$

Using variational calculus and by applying Euler-Lagrange differential equation we get following equation

$$\alpha v_{ss} - \beta v_{ssss} - \nabla E_{\text{image}} = 0$$

Equation can be interpreted as a force balance equation.

Each term corresponds to a force produced by the respective energy terms. The contour deforms under the action of these forces.
Forces

- **Elastic force**
  \[ F_{\text{elastic}} = \alpha \nu_{ss} \]

- **Bending force**: tries to smooth out the curve

- **External force**: acts in the direction so as to minimize $E_{\text{ext}}$
  \[ F_{\text{ext}} = -\nabla E_{\text{image}} \]

- Force equations applied to **each control point** separately.
- Each control point allowed to move freely under the influence of the forces.
Weakness

- False positives with local minima
- Fails to detect concave boundaries. External force can't pull control points into boundary concavity.
- The snake is not attracted to distant edges.
- The snake will shrink inwards if no substantial images forces are acting upon it.
- A snake larger than the minima contour will eventually shrink into it, but a snake smaller than the minima contour will not find the minima and instead continue to shrink.
The balloon model [Cohen et al.] addresses those situations where the snake cannot go forward, due to the edge distance, thus the external energy is not strong enough against the internal forces.

The balloon model introduces an inflation term into the forces acting on the snake

\[ \vec{F} = k_1 \vec{n}(s) \]

where \( n(s) \) is the unit vector normal to the curve \( u(s) \) and \( k_1 \) is a constraint which controls the inflation or deflation tendency (depending on the sign of \( k_1 \)) respect to the external forces.

Three issues arise from this model:

- Instead of shrinking, the snake expands into the minima and will not find minima contours smaller than it.
- The outward force causes the contour to be slightly larger than the actual minima. This can be solved by decreasing the balloon force after a stable solution has been found.
- The inflation force can overpower forces from weak edges, amplifying the issue with snakes ignoring weaker features in an image.
Balloon model

Evolution using only inflation forces

Initial Contour
Advantages & Disadvantages

**Advantages:**
- Designed to solve problems where the approximate shape of the boundary is unknown
- Able to adapt to differences and noise and to ignore missing boundary information
- They autonomously and adaptively search for the minimum state.
- External image forces act upon the snake in an intuitive manner.
- Incorporating Gaussian smoothing in the image energy function introduces scale sensitivity.
- They can be used to track dynamic objects.

**Disadvantages:**
- Prior knowledge about the shape location is needed to initialize accordingly the contours
- They are sensitive to local minima states
- Their accuracy depends on the convergence policy
Practical applications

- Medical applications
Another alternatives for traditional snakes

- Geodesic Snakes
  - **Contours split and merge** depending on the detection of objects in the image.
  - These models are largely implemented using level sets, and have been extensively employed in medical image computing.

- Pixel Level Snakes (PLS)
  - The contour is not formed by control points
  - The contour is defined by all the single pixels that form it
Pixel Level Snakes

Objective: Object segmentation and tracking based on versatile and high performance active contours
Resources

- Matlab + Image Processing toolbox
  - I provide you with some code obtained from Mathworks to play with it, in order to understand the concepts
- In OpenCV, there is currently a function `cvSnakelImage`
  - Check this blog for a potential explanation on the implementation with a practical example
    - http://eric-yuan.me/active-contour-snakes/
Clustering techniques
Cluster analysis or clustering is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar, based on some common characteristics.

**Applications**: data mining, machine learning, pattern recognition, image analysis, information retrieval, and bioinformatics.

There are some applications on image segmentation using clustering techniques based on characteristics:

- Texture
- Colour

A famous and simple technique is k-means.
k-means clustering is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining.

- **Goal:** to partition \( n \) observations into \( k \) clusters
  - each observation belongs to the cluster with the **nearest** mean
  - **Distance to the cluster** is the squared or absolute difference between a pixel and a cluster center.
    - The difference is typically based on pixel color, intensity, texture, and location, or a weighted combination of these factors.
- \( K \) can be selected manually, randomly, or by a heuristic.
- This algorithm is guaranteed to converge, but it may not return the optimal solution.
- The quality of the solution depends on the initial set of clusters and the value of \( K \).
Algorithm

1) Pick K cluster centers, either randomly or based on some heuristic

2) Assign each pixel in the image to the cluster that minimizes the distance between the pixel and the cluster center

3) Re-compute the cluster centers by averaging all of the pixels in the cluster

4) Repeat steps 2 and 3 until convergence is attained (i.e. no pixels change clusters)
K-means: Example
<table>
<thead>
<tr>
<th>Slide no.</th>
<th>Matlab</th>
<th>OpenCv tips &amp; equivalences</th>
</tr>
</thead>
<tbody>
<tr>
<td>5-9</td>
<td>Bwdist</td>
<td>-</td>
</tr>
<tr>
<td>5-9</td>
<td>Subimage</td>
<td>-</td>
</tr>
<tr>
<td>5-9</td>
<td>Imcontour</td>
<td>-</td>
</tr>
<tr>
<td>5-9</td>
<td>mat2gray</td>
<td>-</td>
</tr>
<tr>
<td>10-15</td>
<td>watershed</td>
<td>-</td>
</tr>
<tr>
<td>16-27</td>
<td>activecontour</td>
<td>-</td>
</tr>
<tr>
<td>28-32</td>
<td>kmeans</td>
<td></td>
</tr>
</tbody>
</table>
Summary

- Image Transforms
  - Distance transform
- Watershed Algorithm
  - Landscape/Waterfall/
- Active contour model (Snakes)
- Clustering techniques: k-means
Practical Exercise

Exercise 1. Hough Transform

- Load the circuit figure
- Apply Hough transform to try to find the lines
- Load the coins image, and segment it
- Apply hough to find the circles

Exercise 2. Distance mapping

- Load the coins image
- Binarise the image and get the exterior of the coins (edges)
- Fill the interior, such as the complete coin is white
- Compute the distance mapping over it, using different types of kernels
  - Which is the differences you see?
- Get the contours and identify which objects are closer just based on the colours of the contours.
Practical Exercise

**Exercise 3.** Active contours (only Matlab)

- Using the same image as before, you will create all the images you need to run an active contours example

- Initial contours
  - From the binarised image, and using morphological operators, get a contour that is outside the coins

- External potential
  - Use the distance mapping technique to compute a distance mapping
  - Use morphological operators to compute another distance mapping (following the same philosophy)

  For matlab, you don't need the computed external potential, you need only your original image, but it will help to understand its behaviour.

  Play with the smooth value, what does it happen?

**Exercise 4.** k-means

- Select a color image with distinct colours
- Apply k-means with different k values
  - What do you notice in the output?
  - Is it good to define a k value by ourselves? Which are the main issues of this?
What did you learn today?

- You have learned different advanced techniques for image segmentation
- Conceptually are easy to understand but hard to implement
- Active contours are widely used in domains like medicine or surveillance due to its huge flexibility
- Clustering techniques is a big field
  - Some techniques are used in computer vision, particularly for object classification