Perception for Autonomous Systems

Lecture 2 - Image Processing (08/02/2021)

Outline/Content:

- Edge Detection
  - What is an Edge?
  - Image Derivative
  - Gradient
  - Sobel
  - Laplacian
  - Canny
- Image Feature Detection and Description
- Feature Description
- Feature Matching
- Fitting Data to a Model (Handling Outliers)
  - Least Squares
  - Hough Transform
  - RANdom Sample Consensus (RANSAC)

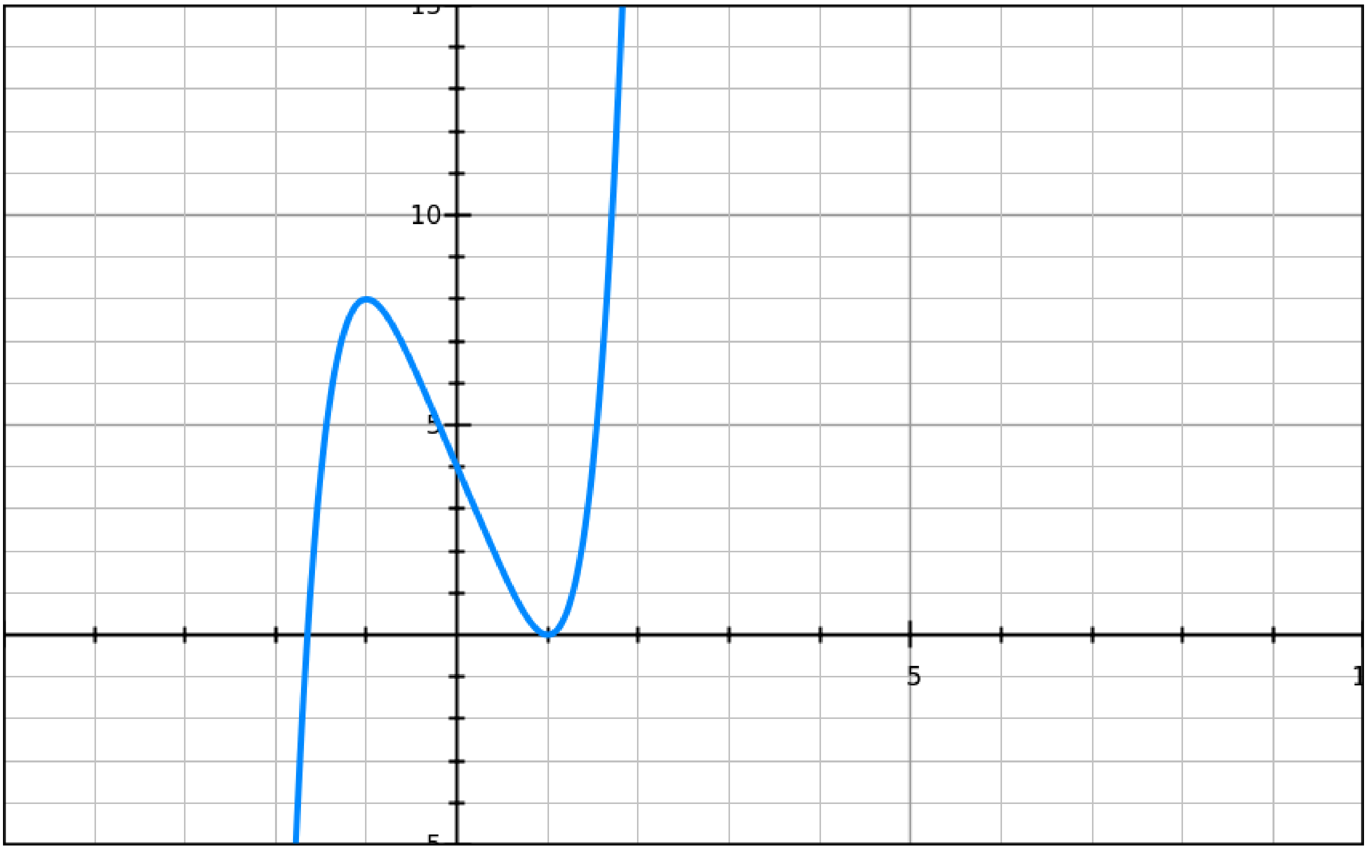
Topics and reading material

> - Feature detection
>   - Book A, Chapter 4.1.1
> - Feature description and Matching
>   - Book A, Chapters 4.1.2 - 4.1.3
> - Feature Tracking
>   - Book A, Chapter 4.1.4
> - Hough transform, RANSAC
>   - Book A, Chapter 4.3.2

Edge Detection

What is an edge?

Edge detection embodies mathematical methods to find points in an image where the brightness of pixel intensities changes distinctly. Simply put, edges occur at the boundaries between areas of different color, intensity, or texture.

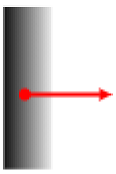


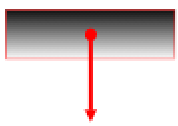
> Visually speaking, do edges occur at locations of steep slopes, or equivalently, in regions of closely packed contour lines (extremas). The edge detection is calculated based on the derivative of an image.

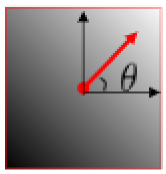
Gradient

The gradient is a vector which points in the direction of most rapid change in intensity:

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$







$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$ $\nabla f = \left[0, \frac{\partial f}{\partial y}\right]$ $\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$

Sobel

Sobel is a edge extractor operator, which is a seperable combination of a horizontal central difference (so called because the horizontal derivative is centered on the pixel) and a vertical filter (to smooth the results).

- Practically:

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

- Magnitude:

$$g = \sqrt{g_x^2 + g_y^2}$$

- Orientation:

$$\Theta = \tan^{-1}\left(\frac{g_y}{g_x}\right)$$

Derivative Filters

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

Scharr $\begin{bmatrix} 3 & 0 & -3 \\ 10 & 0 & -10 \\ 3 & 0 & -3 \end{bmatrix}$ $\begin{bmatrix} 3 & 10 & 3 \\ 0 & 0 & 0 \\ -3 & -10 & -3 \end{bmatrix}$

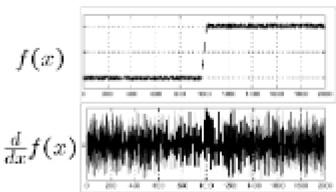
Prewitt $\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$ $\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$

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

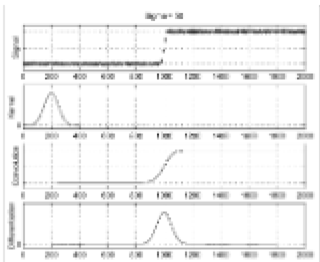
Preprocessing to Edge Detection

Taking image derivatives (required for edge detection) accentuates high frequencies and hence amplifies noise, since the proportion of noise to signal is larger at high frequencies. It is therefore prudent to smooth the image with a low-pass filter prior to computing the gradient.

- In reality derivatives are very prone to noise
  Consider the following example:



- To overcome this issue we can smooth the signal beforehand



> Because it's desirable for the response to be indepenedent of orientation it's necessary to use a circularly symmetric smoothing filter.

Laplacian of Gaussian

The Laplacian of Gaussian (LoG) is the convolution kernel of the Laplacian (gradient operator dot product with the gradient).

Canny Edge Detection

The Canny edge detector was developed way back in 1986 by John F. Canny. And it's still widely used today as one of the default edge detectors in image processing.

The Canny edge detection algorithm can be broken down into 5 steps:

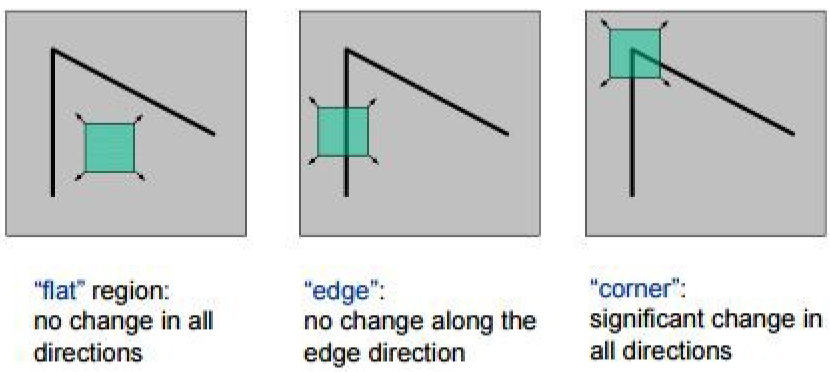
- Step 1: Smooth the image using a Gaussian filter to remove high frequency noise.
- Step 2: Compute the gradient intensity representations of the image.
- Step 3: Apply non-maximum suppression to remove "false" responses to to edge detection.
- Step 4: Apply thresholding using a lower and upper boundary on the gradient values.
- Step 5: Track edges using hysteresis by suppressing weak edges that are not connected to strong edges.

Image Features

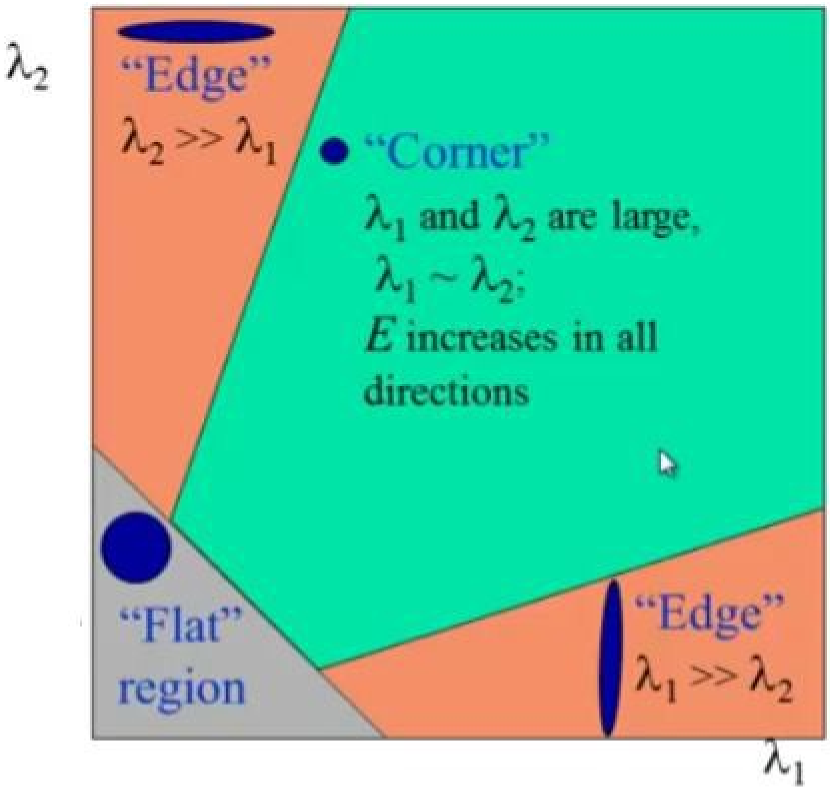
- Feature Detection:
  - Find the most "prominent" Points (areas) in an images
- Feature Description:
  - Create a "unique" descriptor fingerprint for each Feature point
- Feature matching:
  - Find correspondences among different images

Feature Detection: Harris corner detector Medium

A corner is a point whose local neighborhood stands in two dominant and different edge directions. In other words, a corner can be interpreted as the junction of two edges, where an edge is a sudden change in image brightness. Corners are the important features in the image, and they are generally termed as interest points which are invariant to translation, rotation, and illumination.



So let's understand why corners are considered better features or good for patch mapping. In the above figure, if we take the flat region then no gradient change is observed in any direction. Similarly, in the edge region, no gradient change is observed along the edge direction. So both flat region and edge region are bad for patch matching since they not very distinctive (there are many similar patches in along edge in edge region). While in corner region we observe a significant gradient change in all direction. Due this corners are considered good for patch matching (shifting the window in any direction yield a large change in appearance) and generally more stable over the change of viewpoint.



> The values of these eigenvalues (lambdas) decide whether a region is a corner, edge or flat. **More information is to be found in the linked medium article.**

Feature Detection: Scale Space Theory **Expert: Tony Lindeberg** | **Scale-space theory: A basic tool for analysing structures at different scales, Tony Lindeberg**

A theory of multi-scale representation of sensory data developed by the image processing and computer vision communities. The purpose is to represent signals at multiple scales in such a way that fine scale structures are successively suppressed, and a scale parameter t is associated with each level in the multi-scale representation.

> Scale space is a collection of images having different scales, generated from a single image. Fore more information check the linked source.

The mother of Features - SIFT Article

SIFT, or Scale Invariant Feature Transform, is a feature detection algorithm in Computer Vision.

It helps to locate the local features in an image, commonly known as the 'keypoints' of the image. These keypoints are scale & rotation invariant that can be used for various computer vision applications, like image matching, object detection, scene detection, etc.

> This implies that simple corner detectors are not able to match features for images with different scales and rotations. That's when SIFT comes into play.

Characteristics of the SIFT algorithm: Contains:

- Detection
- Description
- Matching

It's robust in:

- Change of Translation
- Change in Scale
- Change in Rotation
- Change in 3D View Point
- Change in Illumination

Scale Invariant Feature Transform - SIFT

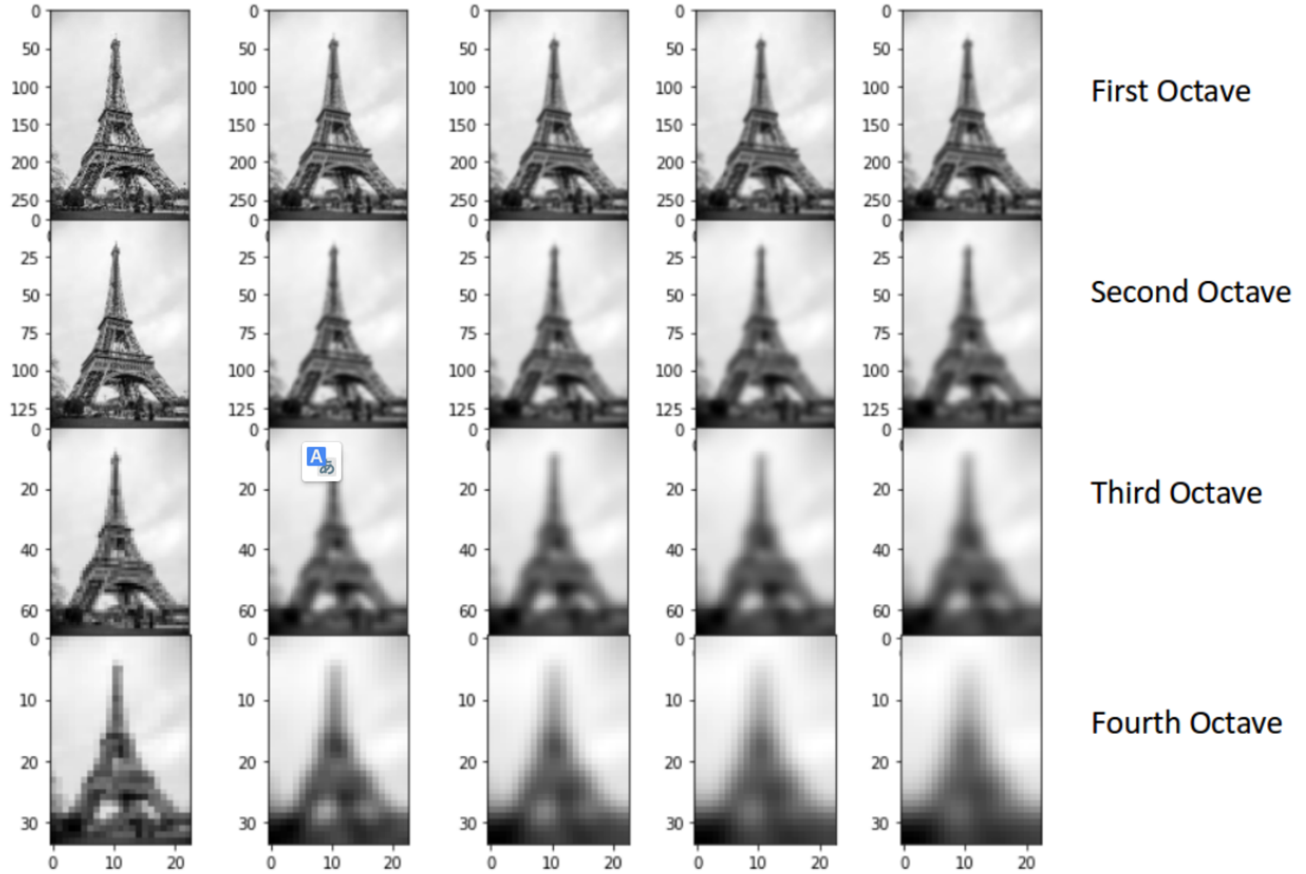
The entire process of that algorithm can be divided into 4 parts:

- Constructing a Scale Space: To make sure that features are scale-independent
- Keypoint Localisation: Identifying the suitable features or keypoints
- Orientation Assignment: Ensure the keypoints are rotation invariant
- Keypoint Descriptor: Assign a unique fingerprint to each keypoint

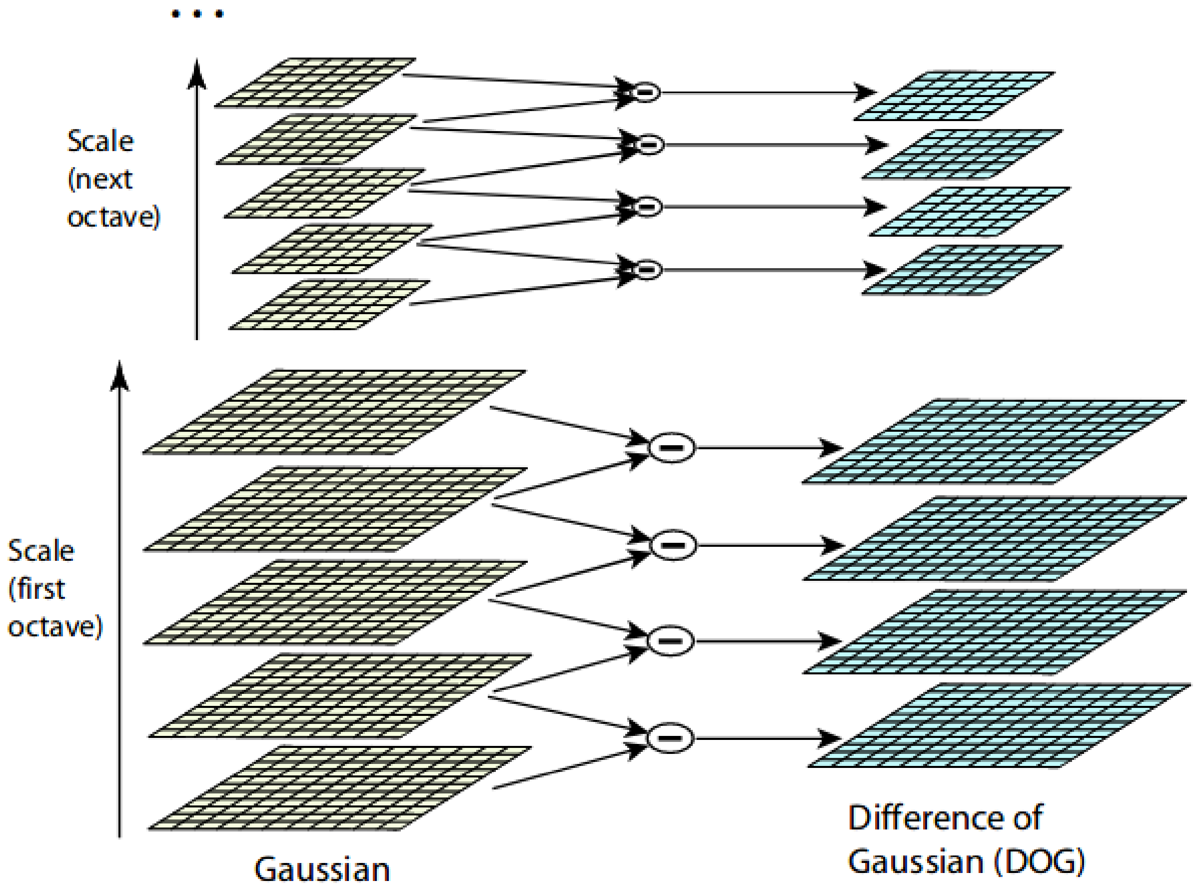
Applied Scale space

The first step of the sift algorithm is to create a number of octaves (images of different scales) and blur out all of those images repeatedly.

The ideal number of octaves should be four, and for each octave, the number of blur images should be five (recommended by the authors).



The second step is to enhance the images. DoG creates another set of images, for each octave, by subtracting every image from the previous image in the same scale. Here is a visual explanation of how DoG is implemented:



> Difference of Gaussian is a feature enhancement algorithm that involves the subtraction of one blurred version of an original image from another, less blurred version of the original.

The next step is to find the important keypoints from the image that can be used for feature matching. The idea is to find the local maxima and minima for the images. This part is divided into two steps:

- Find the local maxima and minima
- Remove low contrast keypoints (keypoint selection)

Fourth step is ensuring the orientation invariance. At this stage, we have a set of stable keypoints for the images. We will now assign an orientation to each of these keypoints so that they are invariant to rotation. We can again divide this step into two smaller steps:

- Calculate the magnitude and orientation
- Create a histogram for magnitude and orientation

> For more information check the linked article.

Feature points and areas

Feature matching algorithms:

- SIFT
- SURF
- BRISK
- FREAK
- MSER

- ORB

What applications do the Features have

More or less everything

- Motion estimation
- Localization
- Mapping
- Photogrammetry
- Image Retrieval
- Machine Learning:
  - Object Detection
  - Recognition
- Autonomous Driving
- etc.

Fitting Data to a Model (Handling Outliers)

Least Squares

Least Squares describes the process of minimizing the loss function by finding the partial derivative of that loss function.

Hough Transform Article

Hough transform is a feature extraction method for detecting simple shapes such as circles, lines etc in an image.

> The main advantage of using the Hough transform is that it is insensitive to occlusion.

It's important to use polar coordinates when applying the hough transform method, to have bounded parameters. Otherwise we could have unbounded parameters ranging from -infinity - infinity.

> Initialization of the hough space

RANSAC Article

The Random Sample Consensus (RANSAC) algorithm proposed by Fischler and Bolles is a general parameter estimation approach designed to cope with a large proportion of outliers in the input data. Its basic operations are:

- Select sample set
- Compute model
- Compute and count inliers
- Repeat until sufficiently confident

The RANSAC steps in more details are:

- Select randomly the minimum number of points required to determine the model parameters.
- Solve for the parameters of the model.

- Determine how many points from the set of all points fit with a predefined tolerance.
- If the fraction of the number of inliers over the total number of points in the set exceeds a predefined threshold, re-estimate the model parameters using all the identified inliers and terminate.
- Otherwise, repeat steps 1 through 4 (maximum of N times).

Briefly, RANSAC uniformly at random selects a subset of data samples and uses it to estimate model parameters. Then it determines the samples that are within an error tolerance of the generated model.

These samples are considered as agreed with the generated model and called as consensus set of the chosen data samples. Here, the data samples in the consensus as behaved as inliers and the rest as outliers by RANSAC. If the count of the samples in the consensus is high enough, it trains the final model of the consensus by using them.

It repeats this process for a number of iterations and returns the model that has the smallest average error among the generated models. As a randomized algorithm, RANSAC does not guarantee to find the optimal parametric model with respect to the inliers. However, the probability of reaching the optimal solution can be kept over a lower bound by assigning suitable values to algorithm parameters.