Exercise 6 – Place Recognition and Line Extraction

Overview

In this exercise, we are going to cover place recognition as an important component for robotic systems and analyze different approaches for fitting lines to a set of measurement points.

Q1 Place Recognition

Place recognition is a crucial part for providing high autonomy in robotics. It essentially covers the study of efficiently recognizing known objects using salient image regions in large-scale databases [3]. It allows robots to recognize previously visited places, typically termed loop closures or the kidnapped robot problem, enabling them to improve their localization and mapping estimates. Consequently, it becomes a crucial component of a robotic system as the incorporation of wrong loop closures can significantly decrease the system's performance and even lead to divergence.

In this part, we will look at how we construct, populate and use the Vocabulary Tree [1] to identify candidate matches for a given query image.

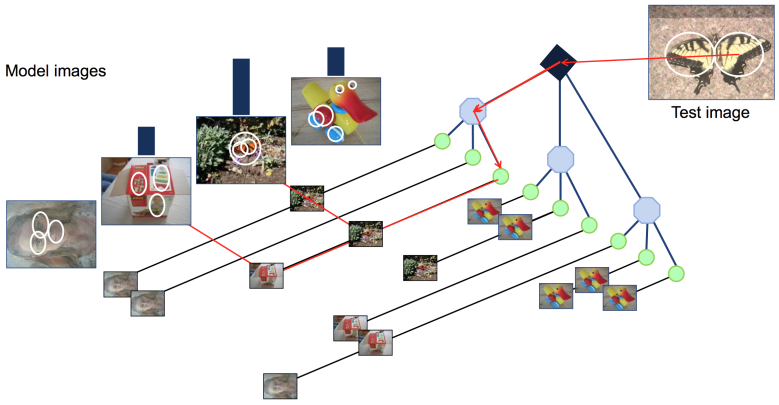

Figure 1: Place recognition by a hierarchical quantization.

Q1.1 Multiple Choice

1. Visual words are clusters of visual descriptors.
2. Keypoint detection and description using SIFT or SURF features is computationally expensive but yields good results.
3. The bag of words approach uses geometric information to retrieve candidate images.
4. We consider a successful lookup only if the two images have the same visual words with the exact same relative position in both images.

Answer:

1. True.
2. True. SIFT and SURF tend to be yield relatively robust feature detections and descriptors, that can match across a wider variety of viewpoints and scales, but this comes at the cost of speed versus keypoint detectors such as FAST and descriptors such as ORB and BRISK (binary descriptors).
3. False. We use visual words and feature histograms to retrieve candidate images.
4. False. The relative displacement is not considered.

Q1.2 Place Recognition

You are given the task of a place recognition by matching the image features to a database. Correctly order the following sub-tasks:

- Extract image features from the image collection.
- The Vocabulary Tree is ready
- Identify the visual word corresponding to an extracted feature
- Populate the descriptors' space with the descriptors of the extracted features
- Look-up the visual word in the inverted-file database
- Extract fetures from the test image
- Select the most voted image as the best candidate match of the test image
- Perform k-means clustering
- Increment the element of the voting array corresponding to the obtained visual word
- Extract features from the Model images (Model images, as seen in the Lecture segment)
- Link the visual word to the Model image it appears in
- Identify the visual word corresponding to an extracted feature

Answer:

1. Extract image features from the image collection.
2. Populate the descriptors' space with the descriptors of the extracted features
3. Perform k-means clustering
4. Extract features from the Model images (Model images, as seen in the Lecture segment)
5. Identify the visual word corresponding to an extracted feature
6. Link the visual word to the Model image it appears in
7. The Vocabulary Tree is ready
8. Extract fetures from the test image
9. Identify the visual word corresponding to an extracted feature
10. Look-up the visual word in the inverted-file database
11. Increment the element of the voting array corresponding to the obtained visual word
12. Select the most voted image as the best candidate match of the test image

Thus, the initial order of the tasks is given by: 1, 7, 5, 2, 10, 8, 12, 3, 11, 4, 6, 9.

Q2 Line Extraction

For many applications in robotics, knowledge of the position and orientation of the platform is absolutely necessary. This exercise could be motivated by an autonomous vehicle exploring an environment. On its way, it might come across walls and hallways, which would be perceived as measurements located along lines by a laser scanner mounted in a way that its scanning plane is parallel to the ground.

In this part of the exercise, we choose to express a line in polar parameters (r, α) as defined by the line equation (1) for the Cartesian coordinates (x, y) of the points lying on the line

$$x\cos(\alpha) + y\sin(\alpha) = r\,, \tag{1}$$

where $-\pi < \alpha \leq \pi$ is the angle between the x-axis and the shortest connection between the origin and the line (see Figure 2).

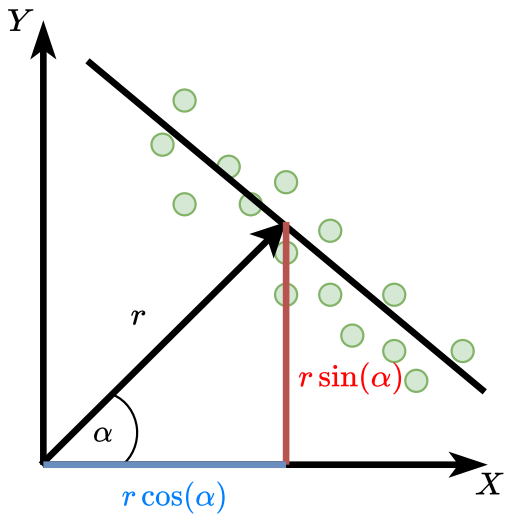

Figure 2: Expression of a line using the polar coordinates r and α.

The polar coordinate system is generally a simple and convenient approach for expressing certain models, e.g., lines and circles, with only a few parameters. The line equation (1) itself can be derived by the Pythagorean theorem: $x^2 + y^2 = r^2$ by substituting x and y with $r\cos(\alpha)$ and $r\sin(\alpha)$, respectively.

In this first part of the line extraction exercise, we will first look at fitting the line parameters using a least squares approach, often known as line regression. In other words, we will minimize a cost function, i.e. the sum of squared distances between each point and a line hypothesis.

Q2.1 Multiple Choice

1. Vertical lines can not be expressed with Cartesian coordinates ($y = mx + b$) but with polar coordinates.
2. The least squares approach randomly selects several sub-sets and evaluates them individually to find the best fit.
3. The least squares approach for model regression is sensitive to outliers (measurement noise).

Answer:

1. True.
2. False, RANSAC is following this approach. Least squares just minimizes the given error function.
3. True. Although least squares is not directly a robust regression method, it can be extended with robust error functions.

Q2.2 Line Regression using Least Squares

Consider the following expression for the squared error $S(\cdot)$ between the line parameterised by r and α and all the points $\mathbf{p} = \{(x_i, y_i) \,|\, i = 1, 2, \dots N\}$.

$$S(r, \alpha) = \sum_i^N \left(r - x_i \cos(\alpha) - y_i \sin(\alpha)\right)^2 \tag{2}$$

Derive an expression for the optimal parameters of the line, i.e. r and α. That means we need to compute

$$\frac{\partial S(r,\alpha)}{\partial r} = 0 \qquad \text{and} \qquad \frac{\partial S(r,\alpha)}{\partial \alpha} = 0$$

(*Hint:* Start with r and use the result to get α)

Answer: To find the optimal parameter for r, we perform a least square estimation.

$$\begin{aligned}\frac{\partial S(r,\alpha)}{\partial r} &= 2\sum_i^N \left(r - x_i \cos(\alpha) - y_i \sin(\alpha)\right) \\ &= 2\left(Nr - \sum_i^N \left(x_i \cos(\alpha) + y_i \sin(\alpha)\right)\right)\end{aligned}$$

Setting

$$\begin{aligned}&\frac{\partial S(r,\alpha)}{\partial r} \overset{!}{=} 0 \\ &\Leftrightarrow Nr - \sum_i^N \left(x_i \cos(\alpha) + y_i \sin(\alpha)\right) = 0 \\ &\Leftrightarrow r = \frac{1}{N}\sum_i^N \left(x_i \cos(\alpha) + y_i \sin(\alpha)\right) \\ &\Leftrightarrow r = x_c \cos(\alpha) + y_c \sin(\alpha)\end{aligned}$$

Here, x_c and y_c are the center coordinates of the point set. Next, we find the optimal parameter for α by using the optimal paramter for r.

$$\begin{aligned}\frac{\partial S(r,\alpha)}{\partial \alpha} &= \frac{\partial}{\partial \alpha}\sum_i^N \left(x_c \cos(\alpha) + y_c \sin(\alpha) - x_i \cos(\alpha) - y_i \sin(\alpha)\right)^2 \\ &= \frac{\partial}{\partial \alpha}\sum_i^N \left(\cos(\alpha)\left(x_c - x_i\right) + \sin(\alpha)\left(y_c - y_i\right)\right)^2\end{aligned}$$

We substitute $\tilde{x} = x_c - x_i$ and $\tilde{y} = y_c - y_i$, respectively and derive

$$\begin{aligned}\frac{\partial S(r,\alpha)}{\partial \alpha} &= 2\sum_i^N \left(\tilde{x}\cos(\alpha) + \tilde{y}\sin(\alpha)\right)\frac{\partial}{\partial \alpha}\sum_i^N \left(\tilde{x}\cos(\alpha) + \tilde{y}\sin(\alpha)\right) \\ &= 2\sum_i^N \left(\tilde{x}\cos(\alpha) + \tilde{y}\sin(\alpha)\right)\left(-\tilde{x}\sin(\alpha) + \tilde{y}\cos(\alpha)\right)\end{aligned}$$

Finally, setting again

$$
\begin{aligned}
&\frac{\partial S(r,\alpha)}{\partial r} \overset{!}{=} 0 \\
&\Leftrightarrow 2\sum_i^N \left(\tilde{x}\cos(\alpha) + \tilde{y}\sin(\alpha)\right)\left(-\tilde{x}\sin(\alpha) + \tilde{y}\cos(\alpha)\right) = 0 \\
&\Leftrightarrow -\cos(\alpha)\sin(\alpha)\sum_i^N \tilde{x}^2 + \cos^2(\alpha)\sum_i^N \tilde{x}\tilde{y} - \sin^2(\alpha)\sum_i^N \tilde{x}\tilde{y} + \sin(\alpha)\cos(\alpha)\sum_i^N \tilde{y}^2 = 0 \\
&\Leftrightarrow \underbrace{\sin(\alpha)\cos(\alpha)}_{\frac{1}{2}\sin 2\alpha}\sum_i^N(\tilde{y}^2 - \tilde{x}^2) + \underbrace{(\cos^2(\alpha) - \sin^2(\alpha))}_{\cos 2\alpha}\sum_i^N \tilde{x}\tilde{y} = 0 \\
&\Leftrightarrow \sin(2\alpha)\sum_i^N(\tilde{y}^2 - \tilde{x}^2) + 2\cos(2\alpha)\sum_i^N \tilde{x}\tilde{y} = 0 \\
&\Leftrightarrow \frac{\sin(2\alpha)}{\cos(2\alpha)} = -\frac{2\sum_i^N \tilde{x}\tilde{y}}{\sum_i^N(\tilde{y}^2 - \tilde{x}^2)} \\
&\Leftrightarrow \alpha = \frac{1}{2}\tan^{-1}\left(\frac{-2\sum_i^N \tilde{x}\tilde{y}}{\sum_i^N(\tilde{y}^2 - \tilde{x}^2)}\right)
\end{aligned}
$$

Q3 Line Extraction with RANSAC

RANSAC (Random Sample Consensus) is an iterative algorithm for fitting model paramters to a specific problem set. The approach is not limited to only fitting lines but can generally be applied to any model for which a mathematical parameterization is available. The algorithm is well-known to be able to efficiently handle problems with a large fraction of outliers, thus, enabling the development of more robust and reliable frameworks [2]. In robotics, it plays a significant role in various tasks, such as loop closure detection and point cloud registration.

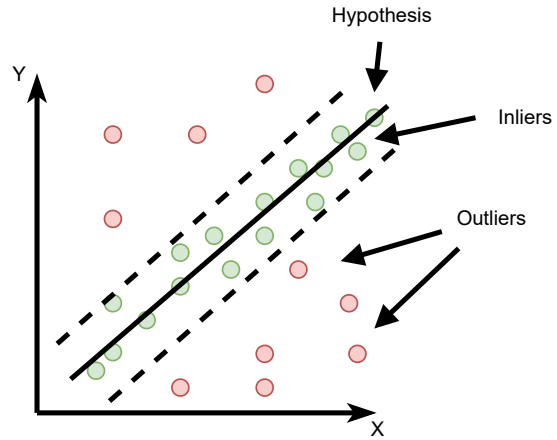

Figure 3: Illustration of a RANSAC problem. In this example, the current model hypothesis has 16 inliers and 10 outliers.

Q3.1 Multiple Choice

1. Applying RANSAC to a specific problem set will always end up with the same result. In other words, it is a deterministic algorithm.
2. RANSAC requires at least 3 points for estimating a line.

3. In the presence of many outliers, RANSAC requires more iterations.
4. RANSAC can be used to fit curves given a set of points.

Answer:

1. False. RANSAC, as it name implies, is generally an indeterministic algorithm.
2. False. The estimation of a line requires two points, while the estimation of a plane requires 3.
3. True.
4. True, RANSAC is not limited to any specific type of model. It only needs a mathematical expression of the desired type.

Q3.2 Statistics with RANSAC

Suppose that you are given a point cloud of 1000 points, 20% of which are inliers.

1. How many RANSAC iterations are required to find at least one set of all inliers with probability $p = 0.95$? You may assume that the set of points is large such that the portion of outliers does not change when removing a point.
2. If we performed RANSAC iterations indefinitely instead of selecting a pre-specified number of iterations, then all the possible line candidates would have been checked. How many possible line candidates are there in this dataset?
3. What is the general probability for getting an outlier model? What is the probability of getting an outlier model after 42 iterations?
4. What is the probability of a successful line extraction after 100 iterations?
5. Instead of fitting a line model to the point cloud, suppose that we are now interested in fitting a plane that fits most points in the point cloud. How many RANSAC iterations are needed to find a plane with at least one set of all inliers with probability $p = 0.95$?

Answer: Let $w = 0.2$ be the inlier count and $p = 0.95$ the probability of all inliers.

1. The number of iterations k is computed using
$$k = \frac{\log(1-p)}{\log(1-w^2)} \approx 74 \tag{3}$$
2. The number of all possible line hypotheses in a set of N points can be derived by considering all different pairs of points that can be selected in this set: For example, let $N = 1000$ then the number of all possible line hypotheses is given by
$$\frac{N(N-1)}{2} = 499500 \tag{4}$$
3. The probability of getting an outlier model is $1 - w^2$. Thus, the probability of having an outlier model after 42 iterations is given by
$$p_f = (1-w^2)^{42} \approx 0.18 \tag{5}$$
4. In this case, we simply need to negate the probability of getting an outlier model after 100 iterations
$$p_s = 1 - (1-w^2)^{100} \approx 0.98 \tag{6}$$
5. Now we are interested in planes, and therefore, we need a minimal set with at least 3 points. The number of iterations k is then given by
$$k = \frac{\log(1-p)}{\log(1-w^3)} \approx 373 \tag{7}$$

References

[1] David Nister and Henrik Stewenius. Scalable recognition with a vocabulary tree. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2161–2168. Ieee, 2006.

[2] Rahul Raguram, Jan-Michael Frahm, and Marc Pollefeys. A comparative analysis of RANSAC techniques leading to adaptive real-time random sample consensus. In *European conference on computer vision*, pages 500–513. Springer, 2008.

[3] Henrik Stewenius, Steinar H. Gunderson, and Julien Pilet. Size matters: Exhaustive geometric verification for image retrieval. In *12th European Conference on Computer Vision (ECCV)*, pages 674–687, 2012.