Notes on Visual SLAM 14 lectures – Chapter 11

1 Loop closure and detection

Frontend errors will accumulate, use loop closure to determine if the vehicle reaches the same place when we pass the same road.
Pose graph can be regarded as a point-spring system ,loop detection is equivalent to adding additional springs to the graph, which improves the system's stability. (can also be used for relocation)

1.1 Close the loops

simplest way is to perform feature matching on any image pairs and determine which of them are related according to the number of correct matches.

A. blindly assume that any two images may have loops

B. for N possible loops, we have to detect C_N^2 times, not practical on real-time systems

C. relies on the assumption that the accumulated error is small so the loop can be detected

Another way is to select frames randomly among n frames: when the number of frames N increases, the probability of drawing a loop is significantly reduced.
(Mainstream)based on appearance: determine loop detection relationship based on two images' similarity. calculate similarity score between image A and image B:

$$s(A, B) = ||A - B|| \tag{1}$$

which is not feasible since:

A. the pixel grayscale is an unstable measurement value

B. when the camera's viewing angle changes a little, even if each other's luminosity does not change, their pixels will be transformed in the image.

- False-positive: perceptual bias
- False-negative: perceptual variation
- precision = TP/(TP + FP), recall = TP /(TP + FN)

1.2 Bag of words

A. determine the concepts of people, cars and dogs corresponding to the word

B. detect which predefined words in the dictionary appear in an image-we use appearance of words (histogram) to describe the entire image (convert image into a vector description)

C. compute the similarity by the histogram in the second step

For example, we can write the words as w_1, w_2, w_3, and then for any image A, according to the words they contain, it can be written as:

$$A = 1 \cdot w_1 + 1 \cdot w_2 + 0 \cdot w_3 \tag{2}$$

since the dictionary is fixed, we may use the vector $[1, 1, 0]^T$, we can then calculate the difference between two vectors $a, b \in \mathbb{R}^W$:

$$s(a, b) = 1 - \frac{1}{W}||a - b||_1 \tag{3}$$

where we take the $\mathcal{L}_1$ norm
Use K-means to construct the dictionary:

A. randomly select k centers: $c_1, ..., c_k$

B. compute the distance between each data sample to the cluster center, assign the closest cluster to this sample

C. recompute the centers of the clusters

D. if the centers converge, exit. Otherwise, return to step 2

A k-d tree can be used to build the dictionary:

A. At the root node, use K-means to cluster all samples into k classes (K-means++ is used to ensure clustering uniformity). This gives the first layer

B. For each node in the first layer, gather the samples belonging to that node into the k class to get the next layer

C. Repeat the second step until the leaf node. The leaf layer is the so-called words

TF-IDF(Term Frequency-Inverse Document Frequency) is a weighting method commonly used in text retrieval and is also used in BoW models:

If a word often appears in the image, its distinguishing degree is high. We can calculate IDF score when building a dictionary–count the ratio of the number of features in a leaf node w_i to the number of all features as the IDF part. Assume that the number of all features is n and the number of w_i is n_i, then the IDF of the word is:

$$\text{IDF}_i = \log \frac{n}{n_i} \tag{4}$$

On the other hand, TF part refers to a certain feature's frequency in a single image. Assuming that the word w_i appears n_i times in the image A, and the total number of words appears n, then:

$$\text{TF}_i = \frac{n_i}{n} \tag{5}$$

then we have:

$$\eta_i = \text{TF}_i \times \text{IDF}_i \tag{6}$$

For a certain image A, its feature points can correspond to many words, forming its bag-of-words:

$$A = \{(w_1, \eta_1), (w_2, \eta_2), ..., (w_N, \eta_N)\} \triangleq v_A \tag{7}$$

We can also calculate the difference between v_A and v_B:

$$s(v_A - v_B) = 2 \sum_{i=1}^{N} |v_{A_i}||v_{B_i}| - |v_{A_i} - v_{B_i}| \tag{8}$$

In practice, we can take a prior similarity $s(v_t, v_{t-\Delta t})$, which means the similarity between the keyframe at a certain moment and the keyframe at the previous moment. Other scores are normalized with reference to this value:

$$s(v_t, v_{t_j})' = s(v_t, v_{t_j})/s(v_t, v_{t-\Delta t}) \tag{9}$$

if the current frame and a previous keyframe exceeds 3 times the similarity between the current frame and the previous frame, it is considered that there may be a loop. Tricks:

A. Elimination of similar loops

B. Use a buffer to only use persistent loops

C. Use spatial consistency (ICP) to estimate the camera's movement, then use it to see if the frames differ a lot