Exercise 3 : Object Tracking

1 Tracking simple objects in videos

1.1 Particle Filters

In this exercise, you will implement a simple object tracker based on the Particle Filter. Some test video sequences can be downloaded from the link below :

`https://drive.google.com/file/d/1k4RoERodDDEmLUzy_EhYPUaMlfzkRKVP/view?usp=drive_link`

A motivating example is given in sequence seq23.avi that shows the tracking result of a (standard) Particle Filter on a real sequence.

The tracked area (object) is initialized by the user and its shape is assumed to be fixed. We will consider here a rectangle, parameterized by the position of its center. The hidden state is thus represented by a 2-dimensional vector.

Transition function. We are interested in the scenario where no a priori information on the movement of the tracked object is available. The transition model is therefore given by :

$$X_k = X_{k-1} + V_k, \tag{1}$$

where $V_k \sim \mathcal{N}(0, \mathbf{\Sigma})$ is a Gaussian white noise, $\mathbf{\Sigma}$ is a diagonal matrix.

Likelihood function. We calculate a normalized color histogram (N_b bins) associated with the tracked area and those associated with the particles in order to approximate the likelihood function. More precisely, at a time instant k, each particle $x_k^i, i = 1, \ldots, N$ will define an area on the image. A histogram is then calculated from this area. The likelihood function is constructed by comparing the obtained histogram with the reference histogram (the one associated with the tracked area). Here we assume that the histogram associated with the tracked area is time-invariant. The distance between two color histograms h and h' is defined as :

$$\text{dist}(h, h') = \Big(1 - \sum_{i=1}^{N} \sqrt{h(i)h'(i)}\Big)^{\frac{1}{2}}, \tag{2}$$

where $h(i)$ and $h'(i)$ are the i-th bins of histograms h and h', respectively.

The likelihood function is then approximated by :

$$g(y_k^i | x_k^i) \propto \exp(-\lambda \text{dist}^2(h_{ref}, h(x_k^i))), \tag{3}$$

where h_{ref} is the reference histogram, $h(x_k^i)$ is the histogram associated with particle x_k^i, and λ is a constant. Details of the tracking algorithm are given below :

- **Step 1 :** Initialize the tracked area by determining the position of center of rectangle and its size.
- **Step 2 :** Calculate the reference histogram h_{ref} associated with the initial tracked area.
- **Step 3 :** Initialize a set of N particles by generating random positions around the center of tracked area. Initial particle weights are set to $w_0^i = \frac{1}{N}, i = 1, \ldots, N$.
- **Step 4 :** It includes 3 main steps in the Particle Filter. At every time instant k, a set of N particles with associated weights $\{x_k^i, w_k^i\}, i = 1, \ldots, N$ is maintained to track the object. Note that particle weights are normalized so that $\sum_{i=1}^{N} w_k^i = 1$.
  - **Prediction :** The goal of this step is to predict the position of the tracked object using the transition model in Eq. 1.
  - **Correction :** This step is used to correct the predicted position obtained from the previous step using the observation from the current image. The likelihood function is constructed from Eqs. 2 and 3. The particle weights are then updated as :

$$w_{k+1}^i \propto g(y_k^i | x_k^i) w_k^i. \tag{4}$$

  - **Resampling :** This step alleviates a common problem of Particle Filter referred to as sample degeneracy. It replicates particles with high weights and remove those with low weights. The most commonly used resampling algorithms are :
    Systematic Resampling Sample $u_1 \sim \mathcal{U}[0, \frac{1}{N}]$ and define $u_i = u_1 + \frac{i-1}{N}, i = 2, \ldots, N$. Set $N_k^i = |\{u_j : \sum_{l=1}^{i-1} w_k^l \leq u_j \leq \sum_{l=1}^{i} w_k^l\}\ |$ with the convention $\sum_{l=1}^{0} = 0$.
    Residual Resampling Set $\tilde{N}_k^i = \lfloor N w_k^i \rfloor$, sample $\{\bar{N}_k^i\}, i = 1, \ldots, N$ from a multinomial of parameters $(N, \bar{w}_k^1, \ldots, \bar{w}_k^N)$ where $\bar{w}_k^i \propto w_k^i - \frac{1}{N}\tilde{N}_k^i$ then set $N_k^i = \tilde{N}_k^i + \bar{N}_k^i$.
    Multinomial Resampling Sample $\{N_k^i, i = 1, \ldots, N\}$ from a multinomial of parameters $(N, w_k^i, i = 1, \ldots, N)$.
    In the above resampling algorithms, N_k^i is the number of particles x_k^i after resampling. An estimate $\bar{x}_k$ of the position of tracked object at the current time instant can be obtained after the correction step or resampling step. More precisely, given a set of particles $\{x_k^i, w_k^i\}, i = 1, \ldots, N$, the position of tracked object can be estimated as

$$\bar{x}_k = \sum_{i=1}^{N} w_k^i x_k^i. \tag{5}$$

1.2 Practical Work

- Implement the Particle Filter algorithm described in the previous section to track a moving object in a video sequence.

- Test your tracker with different settings of parameters : $\mathbf{\Sigma}$ (transition model), λ (likelihood model), N_b (color histogram), N (number of particles). Implementation of the systematic resampling algorithm is mandatory. The other two resampling algorithms are optional (you'll get bonus points if you implement them).
- Test your tracker with a reduced number of video frames (e.g., remove one frame for every two consecutive frames). Compare the accuracy of your tracker in the two cases (tracking the object in the original video and in the video with a reduced number of frames). Which components (parameters, algorithms) you should change in order for the tracker to work ?
- The size of object is now assumed to be varied during tracking. Adapt your tracker to this scenario. What do you observe in terms of tracking accuracy ? Why ?
- Propose some other methods for constructing the likelihood model. Compare the performance (accuracy, computation time) of your tracker when using different likelihood models.
- In order to made a thorough study on the performance of your tracker, you are asked to test it on your own videos. For simplicity, you might create a square/rectangle/circle/triangle that moves in a video. Different levels of difficulties can be added to investigate the robustness of your tracker :
  - Vary the moving spead of objects.
  - Objects filled with more colors.
  - Clustered background videos instead of clean background ones.
  - Videos with multiple moving objects. What do you observe when some objects are occluded by some others ? Propose a solution to deal with the problem.

  By creating your own videos, you'll be able to calculate the tracking errors by comparing the positions of tracked object estimated by your tracker with the groundtruth positions. Plot the tracker accuracy over time for different cases considered above.

2 Tracking complex objects in videos

2.1 Particle Filters for high-dimensional problems

In this exercise, we will implement a Particle Filter algorithm for tracking an articulated object (e.g., human body, human hand) in a video. Some demo videos resulted from tracking human poses with such an algorithm are provided with this handout.

For simplicity, we will work with simulation videos like the one given in the previous exercise. The target object consists of a center part (square) connected with 4 hands. Each object's hand

consists of some rectangles linked together. Different objects have different numbers of rectangles in their hands. We note that once the algorithm for tracking such objects has been implemented, only a few modifications are needed to make it work as shown in the demo videos.
Since the target object has multiple parts to be tracked, we can no longer represent the hidden state by a 2-dimensional vector as in the previous exercise. Instead, we must use a high-dimensional vector to represent the hidden state. In theory, the number of particles required to track the object grows exponentially with the dimension of hidden state. We are thus facing the problem referred to as the curse of dimensionality in Particle Filter methods. In order to alleviate this problem, we will use an advanced version of Particle Filter algorithm whose steps are given below :

1. Initialize a set of particles to track the object. You might use the same method in the previous exercise.
2. Track the object's center part using a Particle Filter.
3. Track the object's hands sequentially using a Particle Filter. Note that when a specific part of a hand is tracked, the transition model is only applied to the part in question. Also, particle weights should be evaluated based only on the observation related to the target part. In practice, the order for tracking the object's hands might have an important impact on the tracking accuracy. For simplicity, you can assume a fixed order when tracking the object's hand.

You might use the same transition and likelihood models as well as the resampling algorithm that work well for you in the previous exercise.

2.2 Questions

1. Propose a method for representing the hidden state of the object.
2. Why do you think the algorithm can alleviate (to some extent) the problem mentioned above ? Can you improve it ?
3. What happens when we increase or decrease the number of rectangles in each object's hand ? Analyze the behavior of the algorithm in each case.

2.3 Practical Work

— Implement the Particle Filter algorithm described in the previous section to track a moving object in a video sequence.

- Test your tracker with different settings of parameters : $\mathbf{\Sigma}$ (transition model), λ (likelihood model), N_b (color histogram), N (number of particles). Implementation of the systematic resampling algorithm is mandatory. The other two resampling algorithms are optional (you'll get bonus points if you implement them).
- Test your tracker with a reduced number of video frames (e.g., remove one frame for every two consecutive frames). Compare the accuracy of your tracker in the two cases (tracking the object in the original video and in the video with a reduced number of frames). Which components (parameters, algorithms) you should change in order for the tracker to work ?
- The size of object is now assumed to be varied during tracking. Adapt your tracker to this scenario. What do you observe in terms of tracking accuracy ? Why ?
- Propose some other methods for constructing the likelihood model. Compare the performance (accuracy, computation time) of your tracker when using different likelihood models.
- In order to made a thorough study on the performance of your tracker, you are asked to test it on your own videos. For simplicity, you might create a square/rectangle/circle/triangle that moves in a video. Different levels of difficulties can be added to investigate the robustness of your tracker :
  - Vary the moving spead of objects.
  - Objects filled with more colors.
  - Clustered background videos instead of clean background ones.
  - Videos with multiple moving objects. What do you observe when some objects are occluded by some others ? Propose a solution to deal with the problem.

  By creating your own videos, you'll be able to calculate the tracking errors by comparing the positions of tracked object estimated by your tracker with the groundtruth positions. Plot the tracker accuracy over time for different cases considered above.