Computer vision: project assignment

1 Recognition based indoor positioning

Many techniques exist to determine the position of a user or a device relative to its environment. One example is satellite navigation, where a device compares time delays of microwave signals obtained from different satellites in known orbits to estimate its position. Satellite navigation is ubiquitous in outdoor applications, but performs poorly indoors because there is no direct path for signal transmission. Different techniques are used indoors, mostly relying on local radio beacons placed throughout the infrastructure. Examples include ultra-wideband (UWB) and Bluetooth localization. These approaches are acceptable in industrial settings, but the need to deploy active transmitters in the building makes them unattractive for many non-critical applications like navigating through a hospital, office building or convention center.

An alternative to these communication based techniques is visual positioning. In visual positioning, live camera images are compared to a database of previously recorded images with annotated positions. If the live image matches one of the database images with high certainty, the user is known to be at the location associated with this database image. This type of recognition based positioning is closely related to landmark detection, on which much research has been done due to a Google challenge launched in 2018.

The feasibility of identifying your location by recognizing the environment in a picture is strongly dependent on two things. Firstly, the environment must contain enough distinctive elements unique to this location. A repetitive office corridor is a poor example as the entire corridor has mostly uniform appearance. Secondly, the search space must be limited. If you already have a rough estimate, limiting the search to a specific floor or wing of a building, you can avoid erroneously matching to similar looking but distant locations.

Figure 1: Examples of famous landmarks in the Google landmark recognition challenge.

2 Indoor localization in the MSK

In this project, you will develop an indoor localization application for visitors of the Museum of Fine Arts (MSK) in Ghent. Such a localization app can provide added value for visitors, underpinning in a later stage interactive audio guides, adaptive routes to explore the museum to reduce crowding in some halls, ...

The MSK has an abundance of easily recognizable landmarks: the artworks themselves. This makes for an ideal proving ground for indoor localization. While the museum also offers a collection of sculptures, we will limit the project to the detection and recognition of paintings on the wall. The end goal of the project is to process a video captured while walking through the MSK, and match the location of the cameraman at each time to a specific exhibition hall.

Figure 2: Interior view of the Museum of Fine Arts.

3 General method overview

You can find the necessary source material for this project at `telin.ugent.be/~dvhamme/computervisie_2022/`. There you will find:

1. a set of photographs taken in all halls of the MSK, sorted by hall,
2. a database of paintings manually extracted from these photographs, together with their coordinates in the original photos,
3. a collection of videos taken while exploring the MSK,
4. the floor plan of the MSK.

Your task is to develop software that will determine the path of the visitor in each of the videos, by detecting and recognizing pictures visible throughout the video and looking up the corresponding hall number. To successfully complete this task, at least the following parts will be required:

1. a method to automatically extract paintings from a video frame,
2. a method to match an extracted painting against the database,

3. a mechanism to deal with false matches and enforce continuity of the trajectory,

4. a visualization of the result.

4 Practicalities

The project is executed in groups of maximum five students. Make a github repository at `github.ugent.be` and add all your group members, as well as both supervisors (handles `sroegier` and `dvhamme`) as contributors. Use this github repo to share code and documents with each other and with your supervisors. Do not put datasets, databases or videos on this repository, instead provide the tools to create this data.

We expect from you the following deliverables.

1. **Sparsely documented source code**: provide at least a readme file detailing which functionality is in which source file, and provide basic in-line comments explaining what each block of code aims to accomplish. Place all source code in your github repo. You may update the code until the date of the exam.

2. **Paper**: write a scientific paper about your method. Describe the algorithms (algorithm != source code), provide a performance evaluation using graphs, tables and examples, and draw appropriate conclusions about the strengths, weaknesses and possible future improvements of your work. Due May 19th.

3. **Final presentation**: provide the slide deck you will use for the final presentation in your github repo, excluding any videos. Due by the date of your exam.

4.1 Assigment 1: painting detection

Write a program that will attempt to extract any paintings from a picture completely **unsupervised**, i.e. without any human intervention. This will rely on finding straight lines in the picture, associating lines that may together enclose a painting, and testing whether the enclosed area is likely to contain a painting or a part of the surrounding wall or floor. Take into account that paintings may be viewed at an angle!

Quantitatively evaluate the performance of your detector on the photographs taken in every hall and compare to the coordinates in the manually annotated dataset provided by us. Evaluate in terms of false negatives (paintings missed entirely), false positives (non-paintings extracted), and bounding box accuracy (e.g. average intersection-over-union).

Secondly, visually assess the performance of your detector on the individual frames of the videos. Identify the most common causes of failure and make recommendations for future solutions for these. Note that not all video frames contain paintings, and some that do may be unusable due to motion blur, occlusion or too acute a viewing angle. You may want to automatically select good candidate frames e.g. using a sharpness metric.

4.2 Assignment 2: painting matching

Write a program that can compare an extracted painting to a database painting and compute a matching score indicative of their similarity. You should follow two approaches here: keypoints or feature vectors. Keypoint matching relies on finding individual distinctive elements in the painting. Feature vectors contain more general numbers that describe properties of the painting e.g. color histograms in various parts of the painting. A combination of the two approaches should give the best results. Note that the size and proportions of your automatically detected paintings is likely to be different from those in the database provided by us!

Make sure your matching methods produce a *matching score* that can quantify the similarity (or difference) between the automatically extracted painting and each picture from the database.

Evaluate your matching algorithms in two ways. Firstly, compare the matching score of each painting in the database with itself to the matching score of this painting with all the other paintings in the database. From this you can draw conclusions about your method's ability to capture the uniqueness of each painting.

Secondly, qualitatively compare the matching of the paintings extracted from the video to the paintings in the database provided by us. Identify common mismatches and their causes e.g. imperfect extraction, similar painting content, changed light conditions, ... It is normal that this matching performs worse, the video frames are of significantly lower quality!

4.3 Assignment 3: localization

Design a program that estimates the location of the user in video sequence as well as possible, based on the painting extraction and matching in Assignments 1 and 2. This program will need at least the following two types of logic.

- Soft matching: the current video frame's extracted paintings are matched against the database, and not only the best match is retained but a list of potential matches and their relative likelihoods.
- Spatio-temporal consistency: using the history of observations, some potential matches will be impossible or very unlikely: the user cannot instantly move to a completely different part of the museum. Even in the absence of extracted paintings, the likelihood of each location gradually diffuses from the last known position. You can only evaluate this part using the video.

The best results can be obtained by intelligently combining the two mechanisms: if the user cannot possibly be in some parts of the museum, paintings from those parts need not be matched against. On the other hand, if the location is ambiguous, many consecutive low-confidence matches can still strongly boost the likelihood of the correct location.

Tip: you will need to make a connectivity graph based on the floor plan. You can describe this with a NxN matrix where N is the number of halls, and each element (i, j) in the matrix is 0 or 1 depending on whether you can move from hall i to hall j or not.

Optional tip: a Hidden Markov Model takes this one step further and assigns probabilities to the transitions between halls. Additionally, it offers an elegant and theoretically sound way to take into account both the quality of your detector and the possible confusions between paintings in the database that look similar to each other.

4.4 Assignment 4: demo

Create a visualization of your combined methods on the test video, showing detected paintings, their most likely match from the database, and the estimated likelihoods of being in each of the exhibition halls (e.g. using a coloured map or bar chart).