P4 of SE **Master 1 Informatique**

Detection of Animated Objects in Movies

1 Introduction

The film industry is making nowadays a larger and larger use of computer science in the different steps of film production. A new and interesting emerging problem in this context is the automatic analysis of movie scenes, where the main aim is to identify cinematographic *styles* in a completely automatic way [1, 2, 3].

Performing such an analysis is currently very challenging, because actors, backgrounds, and moving objects in general, are quite difficult to detect from a given scene. Moreover, other important features to detect include where the actor is looking at, or even its facial expression. A general methodology is supposed to deal with the different kind of movies, including cartoons, old black-and-white movies, and modern digital movies.

This project focuses on the following sub-problem: given a movie scene, our problem consists in generating a video clip that reproduces the same scene, but where the *main* moving objects are represented by their silhouettes. Every silhouette can have a different color, and the background color can be fixed to a neutral color, such as black. This would be an initial step in finding a more general methodology for movie scene analysis, as indicated above. The difficulty of the present project consists in the way the main moving objects in the scenes are identified.

2 The project

The project can be summarized in the following main steps.

2.1 Data conversion

Select an initial (small) set of scenes from movies that you like (please select scenes with no audience restrictions, of any kind). Pay attention to select scenes that consist of single camera shots. One possible source of movie scenes is `YouTube`; videos may be downloaded for local use. By using available freeware such as `ffmpeg`, convert your movie scenes in a list of images, in `jpg` format for example. A framerate of 10 images per second could be adequate for our purposes. Then, by using other freeware (an example is `Scilab`), find the silhouettes of the objects represented in every image. Represent therefore every image with a matrix of booleans, where `true` means that the corresponding pixel belongs to one of the identified silhouettes. Finally convert your scene in a new video format, where the scene is represented by a sequence of boolean matrices.

2.2 Data extraction

We want now to analyze the videos encoded in our new format, by executing the steps below.

- Turn to `false` all pixels that are `true` in all frames (we are interested in the moving objects).
- In every frame, assign a common label to all `true` pixels that can be connected by a path consisting of only `true` pixels; this operation is performed without considering the neighboring frames.
- For every pair of consecutive frames F1 and F2, find a *mapping* from every `true` pixel p_1 of F1 to a `true` pixel p_2 of F2 so that:

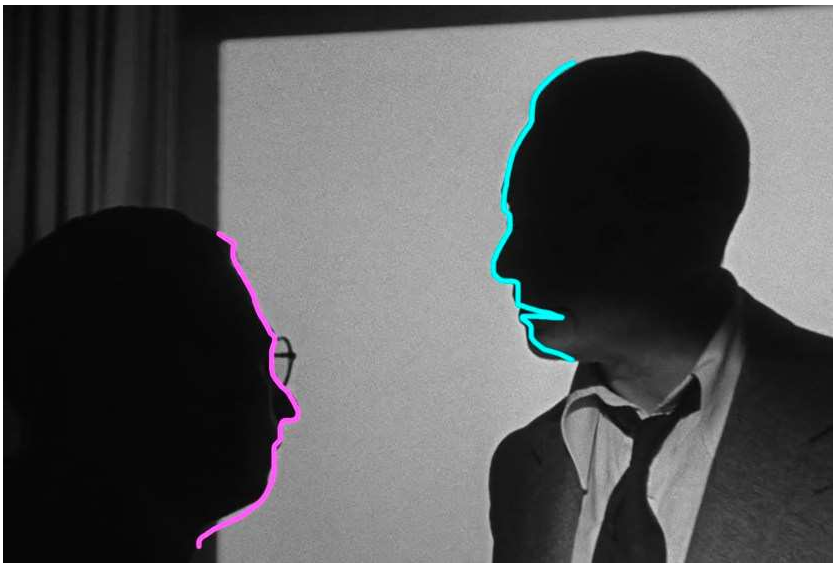

Figure 1: *Two actors with the silhouettes of their faces.*

1. the Manhattan distance between p_1 and p_2 is as small as possible;
2. all `true` pixels of F1 are associated to at least one `true` pixel of F2.

This mapping is subsequently exploited for associating the group labels assigned to the different frames composing the scene. The same color can therefore be assigned to a common group of labels.

Write a Java code implementing the steps above. The program should accept in input a movie scene encoded as a sequence of boolean matrices, and should provide in output the same scene encoded as a sequence of integer matrices, where every integer represents a different group color.

2.3 Multi-thread programming

When the Java code is running properly (i.e. it was tested on a certain number of movie scenes, you may want to increase the number of scenes to test at this stage), then we may attempt to develop a multi-threading version of the code. The most expensive part of our program consists in identifying the mappings: this will be the focus of our multi-threading version. Since we consider pairs of consecutive frames to create the mappings, this operation can be performed independently by several threads, in different frame subsequences of our scenes. A synchronization step is however necessary when two frames, initially assigned to two different threads, need to be considered.

References

[1] J.E. Cutting, *The Framing of Characters in Popular Movies*, Art & Perception **3**, 191–212, 2015.

[2] H-Y. Wu, M. Christie, *Analysing Cinematography with Embedded Constrained Patterns* Eurographics Workshop on Intelligent Cinematography and Editing (WICED16), Lisbon, Portugal, 8 pages, 2016.

[3] H-Y. Wu, M. Young, M. Christie, *A Cognitive-Based Model of Flashbacks for Computational Narratives*, Proceedings of Artificial Intelligence and Interactive Digital Entertainment, 12^{th} Annual AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE16), San Francisco, USA, 239–245, 2016.