Computer Vision: Foundations

Lecture Notes

Lecturer: Prof. Dr. Fred Hamprecht

Edited by: Nils Friess

Last updated: April 24, 2020

Contents

Todo list

1 Introduction

Computer Vision

- useful in consumer devices (e. g. fingerprint login on smartphone)
- is used as machine vision in quality control (e. g. check that all pixels of a screen work $\rightarrow$ rather easy task, that there are no scratches $\rightarrow$ rather hard task)
- makes life-and-death decisions (e. g. in autonomous emergency braking, cyclist and pedestrian detection $\rightarrow$ needs extremely high accuracy)
  - Example procedure: Take last n frames (e. g. $n = 11$) and decide based on them wether or not to brake
  - Need extremely low false positive rate (do not want to brake if it is not necessary)
  - Need low false negative rate
  - Need to process the data in real-time, i. e., need to process ~ 30 MB input/s

Only two hand ful of algorithms are sufficiently efficient, i. e., work at high scale and will make it into consumer devices. We will therefore study these two hand ful of algorithms in-depth. They can then be combined in complicated pipelines and we will study some of these pipelines (see table).

Input	Output	Task
Image	0/1	Image classification
Image	One class per pixel	Semantic/pixel segmentation
Image	Which pixel belong to which instance	Instance segmentation
Image	Pose of one or more humans	Pose estimation
Video	Tracks of all targets	tracking

Table 1.1: Example tasks in computer vision {tab:ex:tasks}

1.1 Linear Filters: Convolution

Convolution is useful for

- Smoothing (Not SOTA)
- Edge/Blob detection
- General: Feature extraction

Has been mainstay of image analysis since it's very cheap (still matters now) and is well-understood.

1D Convolution

Consider the mean square estimator for $\{y_i\} \in \mathbb{R}$

$$\hat{y} = \arg\min_y \sum_{i=1}^{n} (y_i - y)^2$$

Add the derviations for the different estimator from notes

Different filters can produce very different results ($\rightsquigarrow$ filter optimazation). This is one instance of *discrete convolution*

$$\sum_{i=0}^{n-1} f_{l-1} g_i =: (f * g)_l$$

Properties:

- Convolution is **commutative**: $f * g = gast f$
- Convolution is **associative**: $f * g * h = f * (g * h) = (f * g) * h$
  - Important in practice, especially for image analysis
- Convolution is distributive

Important convolution filters include

- $f_i \geq 0$ smoothing
- $_i = \delta_{i-2}$ shifting
- $f = 1/2(1\ 0\ -1)$ central finite difference (differentiation)

Note: 1D convolution can be written as matrix multiplication with a *Töplitz matrix*.

Write up example

2D Convolution

In early days only small filters were possible (e. g. f was 3x3).

Some filters are **seperable**: $f_{i,j} = a_i \cdot b_j$ and this allows for storage reduction (instead of storing the full matrix it suffices to store the vectors a and b).

Aside: Some filters are not seperable but low rank and using singular value decomposition we can find a suitable represantation.

In practice you would use libraries because with the right memory layout, clever use of co-processors and GPUs speed can be increased.

There are different options to extrapolate at the boundary of the image.