

Cimarron

Stabilisation of videos in modern C++

Marius Herget

in partnership with

*Institute for Computer Science
Ludwig-Maximilians-Universität München*



8th October 2018

Contents

Chapter 1

Planing

1.1 Idea

1.2 System diagram

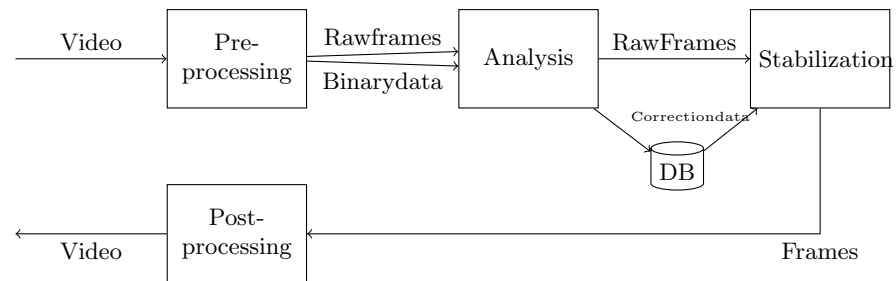
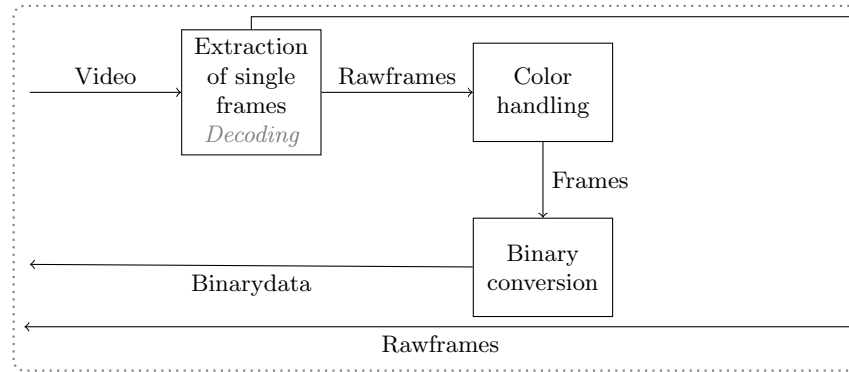
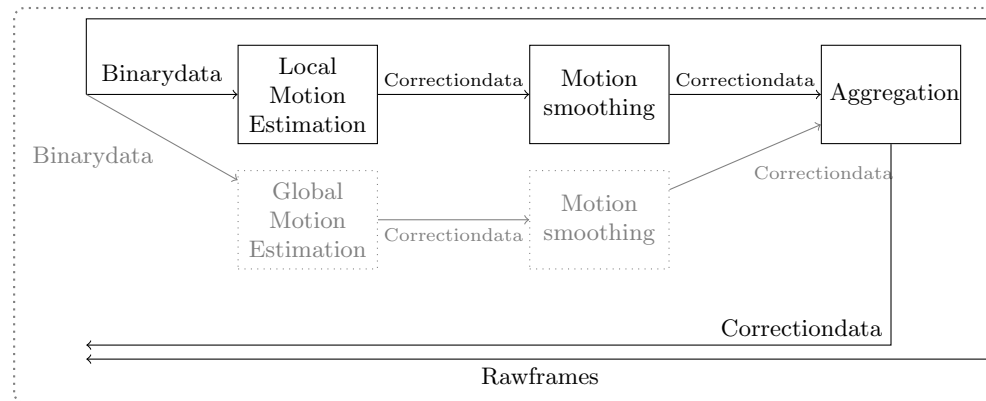
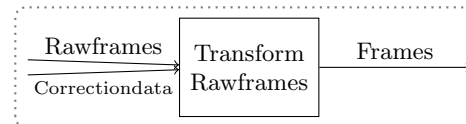


Figure 1.1: High-level system diagram

Figure 1.2: Detailed system diagram of the *Preprocessing*Figure 1.3: Detailed system diagram of the *Analysis*Figure 1.4: Detailed system diagram of the *Stabilization*

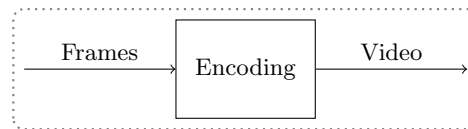


Figure 1.5: Detailed system diagram of the *Postprocessing*

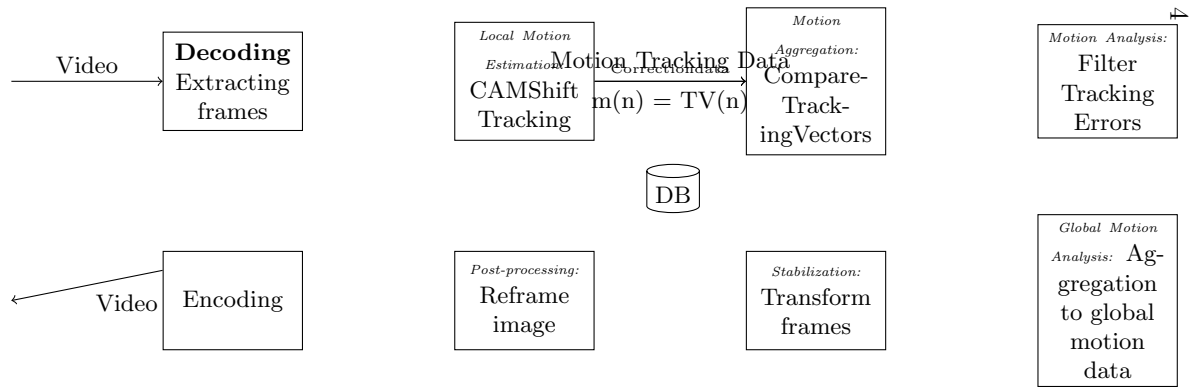


Figure 1.6: High-level system diagram

1.3 Practical evaluation

This section starts with an description of the scoring criterias which will be used in ?? to evaluate the in ?? described frameworks. The final decision with a detailed explanation is closing this section.

1.3.1 Evaluation criteria

The overall goal of this work is to develop a C++ driven software video stabilisation. Therefore, the following criteria are the basis to evaluate the perfect C++ framework with all needed features to extract, analyse and manipulate videos.

- * **Videocodex** The framework needs to be able to handle basic codecs. In detail these are *TBD*. This is only a gateway criterium.
- Extracting frames** *Decoding* One basic feature of a suitable framework is the extraction of every frame to process it further. The easier and efficient it is, the higher is the score.
- * **Edge detection** One approach to the problem is to recognize edges and to track their movement. Therefore, a suitable framework needs to be able to extract edges and to present them in an efficient way. This is only a gateway criterium. *[TBD: Score]*
- Manipulation of frames** A suitable framework needs to be able to transform, rotate and warp frames. The more methods to manipulate the frames are available, the better is the score.
- Tracking** An non-necessary but useful feature is the object tracking. These objects could be used to stabilize videos around an important object. This is just an decision helper, if two or more gateways have similiar scores.
- Merge frames** *Encoding* Another basic feature of a suitable framework is to merge the manipulated frames back into an video. This is only a gateway criterium.
- License** For publication and eventuall professional usage the *license* is major fact as well. A free, open framework is favored.
- Technical environment** This criterium considers the current state of the framework. If the technical dependencies in hard- or software are low the score is better. This includes hardware, operating system or other framework dependencies. Moreover, the ongoing amount of known bugs is also covered. Furthermore, it is scored whether the codes main goals are consistent.
- Performance** This criterium considers a good framework to have a fast response time. This includes an efficient implementation and in the best case the use of modern C++14 standard. Performance boosters like GPU support or special hardware lead to a higher score, as long as this does not interfere with a normal use or sets any limitations.

Confidence This criterium considers a good structure to be under the patronage of a admired company, an internationally respected university or based on an active open-source community with more than 1000 participants. In all cases, the project should be financed for the next few years. The internal calculations of each framework are rated on the present documentation, their developers, and publications. Another observable factor in this criterium is the overall usage of the code. If it is adopted by big companies and has a large community with help forums, tutorials or regular mailing lists, it is scored higher. This is important to since only a long-term, scientifically proved framework should be used to receive a steady and well usable applicaion.

1.3.2 Frameworks

In the following, there is a quick overview of various frameworks with short descriptions (in alphabetical order).

CImg

FFmpeg

GStreamer

OpenCV

Video++

1.3.3 Evaluation

		Caffe	Caffe2	Keras	Scikit-Learn	Tensor-Flow	Theano
Language	10%	7.5 5 9.5 7.5	7.5 9 9.5 9.1	<u>0</u> 9 8 7.6	6 5 9 7.1	9 9.5 8.5 8.95	6 7 7 6.9
Releases	40%						
Documentation	50%						
Environment							
License		9	9	9.5	9	9	9
Dependencies	50%	8 7.5 8 7.85	9 6 8 7.9	5 1 7 4.2	7.5 8 9 7.95	9.5 8.5 8 8.9	8 8 8 8
Known bugs	30%						
Code goals	20%						
Technique							
Activations ftk.	35%	8 7 10 7.5	7.5 8.5 5 7.975	7 5 9 5.9	9.5 9.5 8 9.425	8 8 7.5 7.975	7 7 6.5 6.975
Neural networks	60%						
Rest	5%						
Features							
Creation	15%	9 9 10 9.25	7 8 8 7.85	7 7 7 7	8 9 2 7.1	9 8 9.25 8.4625	8 7.5 5 6.95
Real-time	60%						
Booster	25%						
Performance							
Final Score		80.76	81.6	Disq.	83.775	86.32	69.13

Table 1.2: Scoring of the evaluated frameworks

1.3.4 Decision