Detecting missing frames in videos

3344959 Marc Vaarties
Supervisor: Remco Veltkamp

July 16, 2014

Abstract

Although there are many video completion algorithms, ranging from object removal to reconstructing missing frames, little work has been done in the field of automatically detecting missing frames to begin with. This means manual work, thus we seek to propose a method that detects the locations of missing frames automatically. Our proposed method is based on the *Edge Change Ratio* algorithm, which was originally developed to detect scene breaks. We look at a number of factors and how they influence the algorithm, and adapt it to our needs. Results indicate that our method works fairly well, and with some more work, can automatically and reliably detect missing frames regardless of resolution, framerate, or the actual content of the video.

Contents

Abstract **i**

1 Introduction **1**

2 Related work **2**

3 Approach **4**
3.1 Edge Change Ratio 4
3.2 Approach 5
3.2.1 Step one: segmenting the video 5
3.2.2 Step two: computing the Edge Change Ratio 7
3.2.3 Step three: detecting the missing frames 7

4 Experimentation **9**
4.1 Benchmark 9
4.2 Hypothesis 10
4.3 Results 12
4.3.1 Subhypothesis I: framerate and resolution 12
4.3.2 Subhypothesis II: motion 14
4.3.3 Subhypothesis III: variable r 18
4.3.4 False Positives and False Negatives 31
4.3.5 Conclusion 31

5 Concluding remarks **33**

A Benchmark **35**

Bibliography **44**

Chapter 1

Introduction

In the field of video restoration, many methods exist that try to inpaint missing frames in video sequences in a visually appealing way. These could be damaged frames, or perhaps the audio goes out of sync with the video and we wish to generate some extra frames. However, little work exists on detecting the optimal locations where to add an extra frame automatically, or mark locations in the video where a frame is most likely missing. Manual annotation is a lot of work, so automating this process to at least some amount is useful.

Our goal is to provide a method that detects locations of missing frames, regardless of a video's resolution, framerate, or actual content. After the method has been ran on a video, its output can then be used in combination with a video inpainting algorithm to complete the video. We limit ourselves to video only, thus we ignore the audio completely. We also limit ourselves to detecting frames that are actually missing, not suggesting the most optimal locations if we want to add some frames (though the latter is most likely merely a matter of changing some thresholds accordingly).

We will start with a discussion on related work in Chapter 2, followed by a more in-depth explanation of necessary background theory in Chapter 3. Our extension is also handled in this chapter, namely Section 3.2. The largest chapter will be on Experimentation, introducing first our benchmark set in Section 4.1. A more technical definition of our hypothesis, and the experiments that will be involved, is given in Section 4.2, of which the results and analysis are presented in Section 4.3. Finally, we summarize our work and discuss some possible future developments in Chapter 5.

Chapter 2

Related work

Existing relevant work mostly focuses on shot transition detection, for which numerous – and sometimes, simple – techniques exist. A survey [Lie01] conducted in 2001 presents a good overview of these methods.

By far the most common transition is a *hard cut*. A video changes from one scene to another abruptly, and many methods to automatically detect these have been suggested in the past. Applications range from creating a gallery of representative screenshots of a movie to simply have a video cut up into many pieces, since other video algorithms, for example video inpainting and video segmentation, often assume their input consists of only a single shot.

A hard cut causes discontuity, but note that a missing frame does this too. The most commonly used methods for detecting visual discontinuity are based on insensity or color histograms. They are fast and effective, and nowadays the FFmpeg tools have an implementation for this as well, based on the shotdetect algorithm [Mat, ffma, ffmb]. The survey [Lie01] also mentions that for these methods the right choice of the discontinuity classifier is far more important than in which colorspace calculations are performed. In reality, this often comes down to using an adaptive threshold instead of a global one, as demonstrated by [DNR05].

In [CSS12], the assumption is made that each shot lasts at least one second, thus making the number of frames per second an interesting variable to look at when designing an adaptive threshold. However, they also mention that this is not always true, especially not in movie trailers, where a lot of very short shots are quite common. Exponential decay [DNR05] is also commonly used for adaptive thresholds.

Aside from visual discontinuity, a hard cut also introduces structural and motion discontinuity. For the former, one method is the *Edge Change Ratio (ECR)* [ZMM95]. At the cost of being vastly more computationally expensive, it can also detect other types of cuts, like fades, dissolves, and wipes. For hard cuts, ECR-based methods generally do not outperform the aforementioned color histogram-based methods [Lie01].

While motion discontinuity sounds like a promising approach, it is mentioned [Lie01] that correctly detecting motion is much harder than discreet image operations, and as such results are often not on par with histogram or ECR methods. Although the MPEG2 codec uses motion vectors, these do not correlate to actual motion [HM13]. Another method [FFR05] aims to segment the input into different layers, and then warping those to reconstruct missing frames. It requires user input, and focuses more on video completion than the detection of

the missing frames.

A few recent papers [TSG12, Wol09], while focusing on the detection of missing frames, are more geared towards network transmission, missing IP packets, and the frames missing as a result of that. Many of them are 'simply' detecting duplicate frames, since most streams will simply keep the last frame shown until a new one arrives. Although a few No Reference (NR) metrics are presented, these are often accompanied by a comparison with a Reduced Reference (RR) metric, which compares a video with a known source video. Naturally, the RR variant performs better, however we don't have the source video in the case of film restoration. On top of that, scenes with very little motion are often challenging to the presented NR methods. Real-time computations are also not really an issue in film restoration.

Chapter 3

Approach

Our approach relies heavily on the Edge Change Ratio, which will be explained in Section 3.1. What features we intend to exploit, and how, is described in Section 3.2.

3.1 Edge Change Ratio

The Edge Change Ratio (ECR) algorithm [ZMM95] was developed to detect scene breaks. It provides a metric for this by computing and looking at the entering and exiting pixels between two frames. An overview of the ECR computation between two frames is given in Figure 3.1.

The ECR algorithm takes as input two consecutive images I and I', which may or may not be aligned to compensate camera motion. Before the edges are calculated, the image is smoothed by convolving it with a Gaussian of width σ. Then, binary edge images E and E' are created by thesholding the gradient magnitude τ of the smoothed images. Assume black pixels indicate the gradient magnitude was bigger than the threshold (thus are an edge); all other pixels are white.

Dilated images $\bar{E}$ and $\bar{E}'$ are created by dilating E and E', respectively, with a diamond of radius r. The slight dilation is to prevent false hits from when an object is moving at a moderate speed, small camera movements, or otherwise normal motions. We can now determine whether a pixel is an exiting pixel if:

- It is present (or black) in E, and
- It is not present (or white) in $\bar{E}'$

or in other words, if the pixel is an edge in one frame, but in the subsequent frame, it (or a small region around it) is not; the pixel has *exited*. We can compute the fraction of exiting edge pixels ρ_{out} with the following equation:

$$\rho_{out} = \frac{\text{number of exiting edge pixels}}{\text{number of black (edge) pixels in } E} \tag{3.1}$$

Similarly, we can define a measure of entering edge pixels ρ_{in} with the following:

$$\rho_{in} = \frac{\text{number of entering edge pixels}}{\text{number of black (edge) pixels in } E'} \tag{3.2}$$

The final ECR ρ between frames I and I' is then defined by

$$\rho = \max(\rho_{in}, \rho_{out}) \tag{3.3}$$

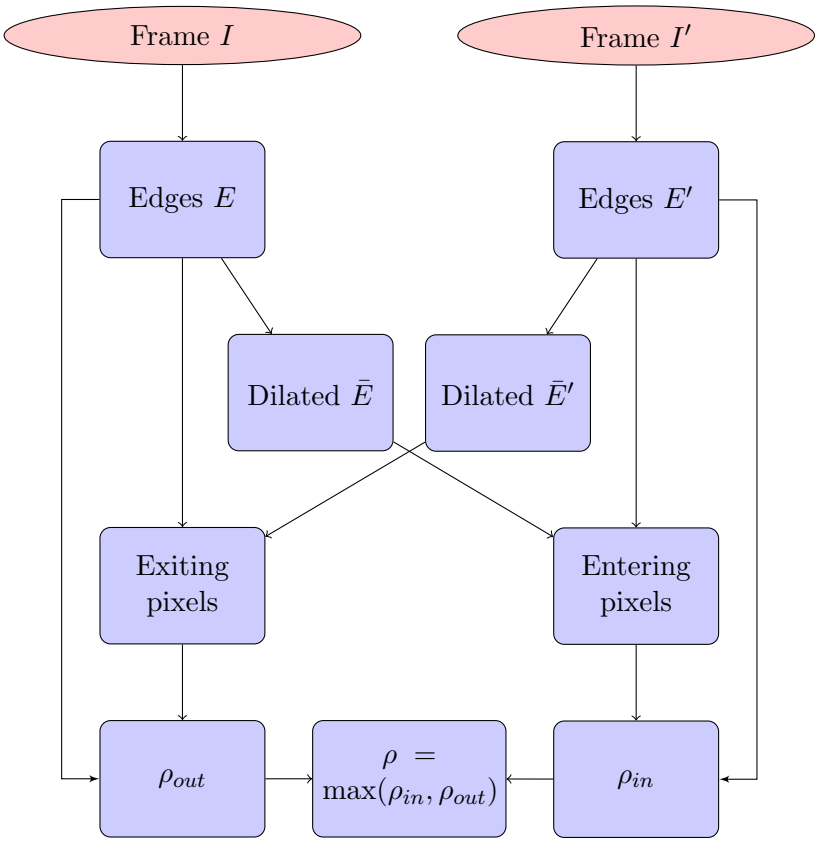


Figure 3.1: Schematic overview of the Edge Change Ratio algorithm

which gives high values when a scene change occurs, and low values otherwise. As can be seen in Figure 3.2, a hard cut leads to a single value peaking, while transitions with a temporal component (ie. a transition that spans multiple frames, for example a crossfade) create more spread-out peaks.

3.2 Approach

We want to exploit the ECR algorithm to detect missing frames. To this end, a pipeline has been designed (see Figure 3.3), whose steps will be explained in more detail in this section.

3.2.1 Step one: segmenting the video

In order to reliably use the exploit proposed in the next subsections, we need to make sure the input video contains no hard cuts. This is the only step that technically requires user input, but it can be automated to a great extent nowadays.

It should be noted that when using an automated method for detecting hard cuts, it is vital that it does not pick up locations of missing frames as well. Suggested methods are histogram-based methods, which are not only very fast, they also seldom pick up a location of a missing frame.

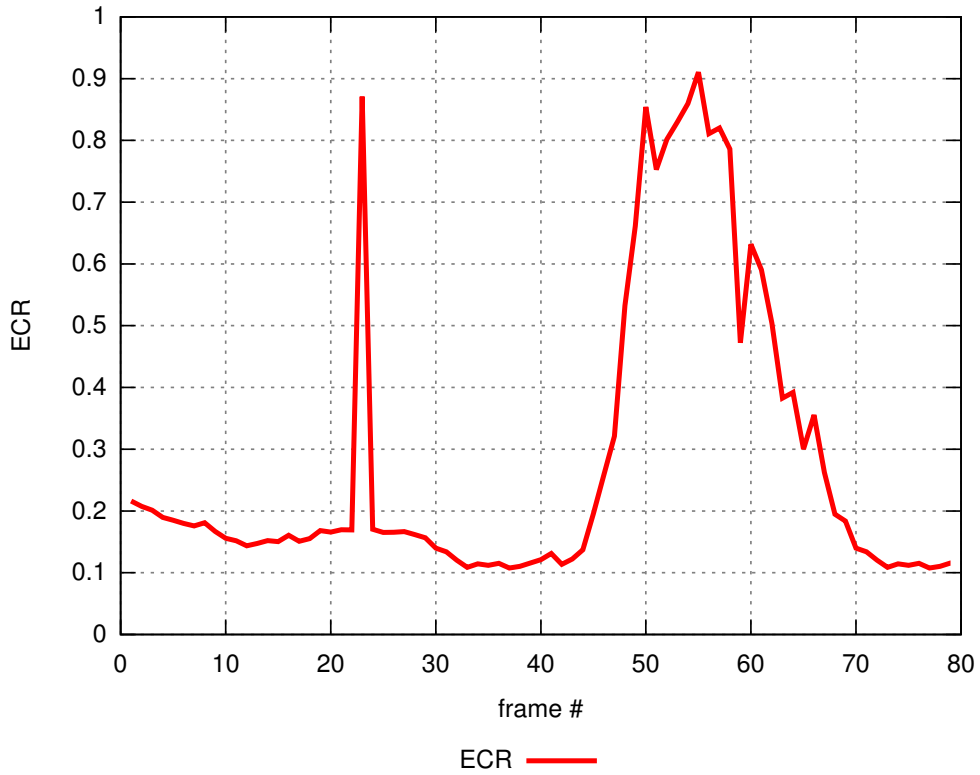


Figure 3.2: Example ECR output. The video has a hard cut at frame 23 and a dissolve at frames 46 to 66.

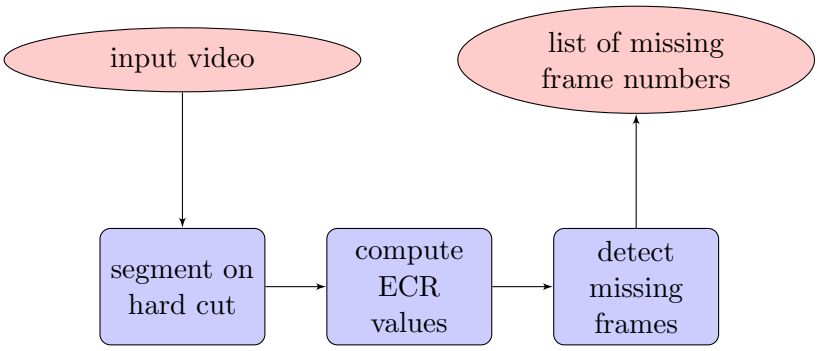


Figure 3.3: Schematic overview of the pipeline

3.2.2 Step two: computing the Edge Change Ratio

On each segment, we now run the regular ECR algorithm, which produces values as depicted in Figure 3.2. It is worth noting that the algorithm has three parameters:

- σ: edge detector's smoothing width,
- τ: edge detector's threshold,
- r: expansion distance.

In the original work, these parameters were set at $\sigma = 1.2$, $\tau = 24$ (out of 256) and $r = 3$. There is not much reason to vary the first two, though we expect the third to have some relation to at least the resolution of the video, possibly framerate as well. At least one of our experiments will involve a non-fixed r.

3.2.3 Step three: detecting the missing frames

At this point we have a sequence of ECR values. Every frame (except the first one) has an ECR value, and every frame except the first resp. the last has an ECR_{in} resp. ECR_{out} value as well. Similar to the original ECR algorithm, we use a local threshold to decide whether a frame is missing or not.

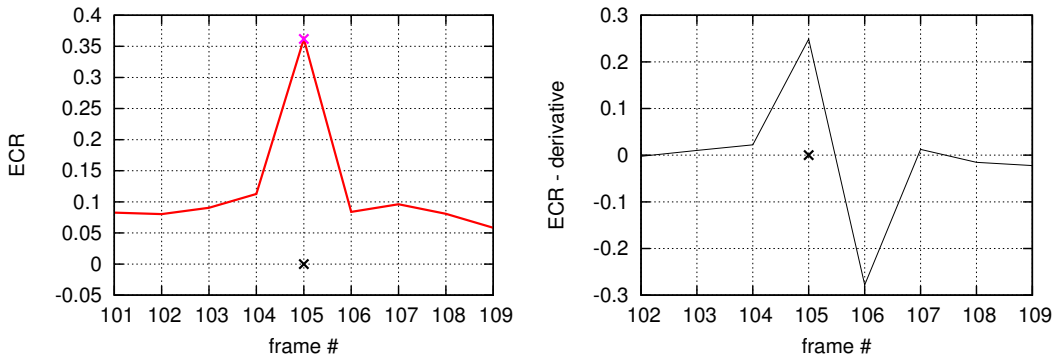
There are a few observations we can make in order to build a suitable detection method. Note that it is not our goal to create a *perfect* classifier; we only need a simple but resonably robust one that has explainable behaviour. The parameters of the ECR algorithm are our variables; as long as the ECR algorithm produces significant peaks at missing frames, any classifier will work – maybe not optimal, but results are still significant.

The first and most important observation is that no matter what algorithm we use, it will be impossible to detect *all* the locations of missing frames – aside from the trivial case where we assign everything to be missing a frame. Examples are equally trivial: just take any video and drop every second frame, thus halving the framerate. Or take a video of a talking person, where the only significant movement is that of the lips. In terms of resulting ECR values, we doubt that this will create a peak at all. Then again, would a human skipping through the video frame by frame notice it, and assuming so, would it be of much interest? We don't think so, which means it is not necessarily bad if not all missing frames get detected. It does mean that we should value precision more than recall.

The next observation is that we expect a missing frame to (normally) result in a peak in the ECR values, although not necessarily one as tall as the peaks that would occur at a hard cut. This is incidentally why we need to get the hard cuts out using a different method first. Transitions spread out over multiple frames (fades, dissolves, wipes, etc.) are not a problem: although they produce high ECR values, unlike a missing frame it will be for a number of consecutive frames.

Since we expect a missing frame to produce a *single-frame peak* in ECR values, it makes sense to construct our method around that. We also propose to take the derivative into account, since that tells us how much more or less motion there is compared to the previous frame. We expect that a complete video (no missing frames) will have its derivative near zero the entire time. An explosion might cause a series of high ECR values, but even then the derivative will only be high at the start of the explosion, and a dip at the end.

This is a very important observation, because an actual missing frame will have the dip immediately after the peak, as shown in Figure 3.4. So, for our simple classifier:



(a) (b)

Figure 3.4: Characteristics of a missing frame. We expect missing frames to occur at local maxima in ECR (left) and the derivative at that place to have a distinct shape (right).

- A (detectable) missing frame will always occur at a local maximum of the ECR values, and
- The derivative at the location of a missing frame has a distinct shape.

The first parameter we need is a window size w, then we define a set of candidate missing frame locations by the following equation:

$$F_i \in candidates\ iff\ ECR(F_i) = \max(ECR(F_{i-w/2}), ..., ECR(F_{i+w/2})) \tag{3.4}$$

or in normal words, if the ECR value of frame i is the highest ECR value in a window of size w centered around frame i. Note that this puts a constraint on how close to the start or end of a video missing frames can be detected.

Now that we have the candidates, we simply need to check if the derivative matches the certain characteristic. $ECR'(F_i)$ should be positive, $ECR'(F_{i+1})$ should be negative, and ECR' of the other frames in the window (the first frame is not taken into account) should be near zero. Let S be the set of absolute ECR values that don't belong to frame i or $i+1$. We can then define a local threshold s, which is just $\max(S)$. We propose the following classifier to determine whether something is a missing frame or not:

$$missing?(i) = \begin{cases} \text{yes} & \text{if } ECR'(F_i) > \lambda \cdot s \text{ and } ECR'(F_{i+1}) < -\lambda \cdot s \\ \text{no} & \text{otherwise} \end{cases} \tag{3.5}$$

Here, λ is a measure of how many times bigger the positive and negative peaks should be with respect to the derivatives of the rest of the window. It is not necessary for it to be a static number; a function that makes the threshold dependent on s is also entirely possible. Another extension could be a fixed minimum value for the ECR or ECR' values to be considered candidate.

Chapter 4

Experimentation

In order to verify our approach, a benchmark has been made, whose contents are listed in Section 4.1. We will then formulate our hypothesis (Section 4.2), followed by an in-depth analysis of the experiments (Section 4.3).

4.1 Benchmark

To test the algorithm's performance, a benchmark has been made. A complete overview, including the different resolutions and framerates for each video, is given in Appendix A.

It consists of a variety of videos, and is split into three distinctive subsets. The first set consists of videos often used in research. The second set consists of various additional real-world videos, including common defects like blocking, camera shake, and changes in illumination. The third set consists almost exclusively of scene transitions: fades, crossfades, dissolves, and so on. The usefulness of the last one debatable, because (depending on the transition) while it might be obvious to the human eye that a frame is missing in such a transition, even if the algorithm correctly detects it, restoring it will be hard.

Almost all videos are available in different resolutions or framerates, and all but one video have unedited versions as well as versions where frames have been removed. All videos were manually and subjectively annotated to indicate how difficult or significant it is should a missing frame occur at that location. It should give us an indication if it is failing consistenly in some cases, or picks up false positives with for example lighting. The five categories we used are listed in Table 4.1.

The different resolutions and framerates are to see what impact this has on the algorithm, as well as to see whether using 1080p video is really worth it. The unedited versions are there to have more information on false positives from the edited ones. If a false positive is detected in both the unedited and the edited version, it is a different kind of problem then if it were only present in one of the results.

Note that the class sizes are imbalanced. This is easily proved by assuming they are equal, because in that case it either means the video is just shorter, or the framerate has halved. In both cases, it is impossible to determine whether frames are missing, since they are perfectly 'valid' videos on their own as well. 'Valid' here means that without additional knowledge, a video is indistinguishable from a video we know doesn't have any missing frames. For example, if we drop every second frame of a 30fps video, the resulting video is exactly the same as when the video had been 15fps to begin with.

Category	Description
Normal	Adequate motion, reasonably stable camera, and so on. Subjectively, we expect the algorithm to work here.
Illumination	Sudden flashes, lamps being turned on or off, the camera auto-adjusting to new lighting conditions.
Talk	Scenes focused on people talking, and little motion of eg. limbs besides that.
Hard	Otherwise hard to detect, for example very little motion, excessive motion, excessive camera shake.
Undetectable	Completely still frames, *extremely* little motion (eg. a slow pan of a still frame), and generally anything that even if it did get detected correctly wouldn't be of much use, for example very strange temporal transitions.

Table 4.1: In the testset, we annotated which frames (subjectively) belong to what category. It will show us some insight if some category is consistently failing.

Equal class sizes are unfeasible; a much more likely scenario is where the number of missing frames are only a small fraction of the total number of frames. In our experiments, we randomly delete about 3% of the input frames. We do not delete frames near the beginning or end, and also do not delete frames in close succession.

A short overview of the videos, and what types of content they have, is given in Table 4.2. All in all, there is a good variety of framerates, resolutions, pans, zooms, and actual content present.

Note that none of the videos have audio, or if they do, we don't care about it. The few videos that contain hard cuts are annotated at the appropriate locations. Thus, we are only interested in steps two and three of the approach given in Section 3.2.

4.2 Hypothesis

In this section, we state our main hypothesis, and the subhypotheses to get there.

Our main hypothesis is that by making r variable, the Edge Change Ratio algorithm can be extended to detect missing frames, regardless of the video's resolution, framerate, or actual content. In order to do this, we must first understand the impact of resolution, framerate, and actual content on the algorithm.

We think that, using the standard ECR algorithm, the same video will perform differently depending on resolution and framerate, which translates to there being particular combinations of framerate and resolution that perform significantly better. This is our first subhypothesis.

The second builds upon the first, namely that aside from resolution and framerate, the content also affects the results, and more specifically, by how much. For example, it would be intuitive that a video with lots of motion performs better at high framerates, while a video of the same resolution but with little motion will perform better at a lower framerate. The key question here is how much this influences results.

Finally, we will make r variable, and depending on the results of the previous two points, propose a method to choose a well-performing r for any video. The method should work for

Name	Cat	N	I	T	H	U	#res	#fps
blackmamba1	RW	•	•				3	1
blackmamba2	RW	•			•		3	1
blackmamba3	RW	•	•				3	1
bus	RS	•					2	3
car360	RW	•			•		1	1
city	RS	•					3	3
crew	RS	•	•				3	3
flower	RS	•					2	2
football	RS	•			•		2	3
foreman	RS	•		•		•	2	3
greatwhite1	RW	•					3	1
greatwhite2	RW	•					3	1
greatwhite3	RW	•				•	3	1
greatwhite4	RW	•					3	1
greatwhite5	RW	•			•		3	1
gtaiv1	RW	•				•	3	1
gtaiv2	RW	•			•	•	3	1
hallmonitor	RS	•			•	•	2	2
harbour	RS	•					3	3
mix	S	•	•		•	•	3	3
mix-notext	S	•	•		•	•	2	1
mobile	RS				•		2	3
motherdaughter	RS			•			2	2
soccer	RS	•			•		3	3
stefan	RS	•			•		2	2
tempete	RS				•		2	2

Table 4.2: Short overview of the videos. *Cat* refers to Research (RS), Real-World (RW) or Synthetic (S). The columns *N*, *I*, *T*, *H* and *U* refer to the contents listed in Table 4.1. *#res* and *#fps* indicate how many different resolutions resp. framerats that video has. If a video has 2 different resolutions and 3 different framerates, it has 6 versions in total.

any video, regardless of its resolution, framerate, or content. We would like the method to prefer a high presicion over recall (within reasonable limits), since by default it's most likely not possible to detect all (synthetic) missing frames anyway.

4.3 Results

In this section, we will describe our findings regarding the hypothesis. Because the class sizes are imbalanced, our primary metric of performance is the Matthews correlation coefficient (MCC). Unlike plain recall and precision, this metric takes into account the class sizes.

In all experiments, a window size $w = 7$ was used. A smaller window size ($w = 5$) led to a *very* high recall, but at the cost of many false positives. Likewise, enlarging the window ($w = 9$) amounted to better precision, but wasn't able to detect as many missing frames compared to $w = 7$. Additionally, the increase in precision was not as significant compared to the increase it got between $w = 5$ and $w = 7$. Hence, we use $w = 7$ in our experiments.

The other parameters for our classifier are $\lambda = 2$. There are also some extra thresholds, which all reduce to that microscopic peaks (especially if the ECR or derivative ECR is very low) are ignored.

We will now continue with the evaluation of the results regarding the hypothesis. After that, we will also look at a few cases where the algorithm produced wrong results consistently, and propose a solution for some of them.

4.3.1 Subhypothesis I: framerate and resolution

Our first hypothesis was that for every video, there is a combination of framerate and resolution with which the algorithm performs best for that particular video. We think that there is also a relation between the two, thus that given a fixed r and a comnbination of resolution and framerate that performs good, there will also be some other, derived combination that produces similar results. However, it is not our goal to exploit this relation, if there even is one. Instead, we will first fix not only r but also the framerate, and see if they perform differently. The results of this are listed in Table 4.3.

video	resolution	rec.	prec.	MCC
blackmamba1	480x270	0.90	0.79	0.84
	960x540	0.96	0.82	0.88
	1920x1080	0.95	0.80	0.87
blackmamba2	480x270	0.75	0.85	0.79
	960x540	0.67	0.72	0.69
	1920x1080	0.60	0.88	0.72
blackmamba3	480x270	0.31	0.77	0.47
	960x540	0.21	0.82	0.40
	1920x1080	0.19	0.59	0.32
bus	176x144	1	0.65	0.80
	352x288	0.79	1	0.88
city	176x144	0.20	–	–
	352x288	0.87	0.75	0.80
	704x576	0.54	0.85	0.66
crew	176x144	0.42	1	0.64
	352x288	0.66	1	0.81
	704x576	0.56	1	0.73
flower	176x144	0.25	–	–
	352x288	0.87	1	0.93
football	176x144	0.51	1	0.70
	352x288	0.54	0.82	0.65
foreman	176x144	0.23	0.52	0.33
	352x288	0.27	0.37	0.29
greatwhite1	480x270	0.83	0.83	0.82
	960x540	0.58	0.61	0.58
	1920x1080	0.35	0.54	0.42
greatwhite2	480x270	0.52	0.70	0.59
	960x540	0.36	0.53	0.42
	1920x1080	0.21	0.80	0.39
greatwhite3	480x270	0.05	N/A	N/A
	960x540	0.06	0.28	0.12
	1920x1080	0.05	N/A	N/A
greatwhite4	480x270	0.21	0.69	0.36
	960x540	0.22	0.66	0.37
	1920x1080	0.09	N/A	N/A
greatwhite5	480x270	0.80	0.87	0.83
	960x540	0.75	0.94	0.83
	1920x1080	0.61	1	0.77
harbour	176x144	0.11	0.29	0.16
	352x288	0.61	1	0.77
	704x576	0.50	0.67	0.56
mix	176x144	0.31	0.56	0.40
	352x288	0.45	0.56	0.48
	704x576	0.42	0.45	0.41
mix-notext	352x288	0.47	0.66	0.54
	704x576	0.40	0.61	0.48
mobile	176x144	0.02	–	–
	352x288	0.17	–	–
motherdaughter	176x144	0.16	0.31	0.21
	352x288	0.27	0.56	0.38
soccer	176x144	0.55	0.85	0.66
	352x288	0.45	0.82	0.59
	704x576	0.36	0.76	0.51
stefan	176x144	0.60	–	–
	352x288	0.63	–	–
tempete	176x144	0.01	–	–
	352x288	0.06	–	–

Table 4.3: Effects on precision, recall and MCC when the resolution is increased, but framerate is kept the same. For the values for recall, precision, and MCC the average of 20 runs are taken, each run dropping 3% of the frames. Values are rounded to two digits after the comma at most. In all cases, framerate is 30 or 29.970 fps.

We can easily see that most of the time, a resolution between 300 and 500 pixels wide produces relatively good results. However, not every video is conveniently in that resolution, and although downscaling high-resolution videos is not much of a problem, upscaling might be. Additionally, not every video is conveniently around 30 frames per second either, so we then compared the performance if we allowed the framerate to be variable, which yielded similar observations: at the aforementioned resolution, it was usually the ±30fps variant that had the best performance, although for lower and higher resolutions, it was more often than not a lower resp. higher framerate. For example, if a 352x288 30fps video performed well, chances are high that a 704x576 60fps variant will also perform well. The relation did not appear to be precisely linear, though certainly monotonic.

As we expected, framerate and resolution influence the outcome.

4.3.2 Subhypothesis II: motion

In the previous subhypothesis, we mentioned that most of the videos perform relatively good at a medium resolution and framerate, and also that there were a few exceptions. These videos, however, all fall into either of three classes, in the sense that they are:

- videos with very little motion, or
- videos with a lot motion, or
- complex videos, ie. some parts have little motion, others a lot.

The results with different framerates and resolutions are listed in Table 4.4 If the average video performs well on a medium resolution and framerate, then videos with very little motion perform better at high resolution/medium framerate or medium resolution/low framerate. Similarly, the videos with lots of motion generally perform better at combinations of low resolution/medium framerate or medium resolution/high framerate. At a first glance, this looks about the same relation as before, except the optimal combination is different. For example, the 'soccer' video performs much better at 60fps, even at lower resolutions, whereas 'tempete' works better at 15fps on a medium resolution.

Table 4.4: Effects on precision, recall and MCC with variable resolution and framerate. Recall, precision and MCC are averaged over a number of runs. A * in the precision column indicates there were no false positives in any of the runs, but it did not always detect all the true positives. Thus, the positives it did detect were always true positives.

video	resolution	fps	rec.	prec.	MCC
blackmamba1	480x270	29.97	0.90	0.79	0.84
	960x540		0.96	0.82	0.88
	1920x1080		0.95	0.80	0.86
blackmamba2	480x270	29.97	0.75	0.85	0.79
	960x540		0.67	0.72	0.69
	1920x1080		0.60	0.88	0.72
blackmamba3	480x270	29.97	0.31	0.77	0.47
	960x540		0.21	0.82	0.40
	1920x1080		0.19	0.55	0.32

video	resolution	fps	rec.	prec.	MCC
bus	176x144	7.5	0.35	*	–
		15	0.63	*	–
		30	1	0.65	0.80
	352x288	7.5	0.50	*	–
		15	0.78	*	–
		30	0.79	1	0.88
city	176x144	15	0.76	0.64	0.68
		30	0.20	–	–
		60	0.05	0.13	0.06
	352x288	15	0.39	0.44	0.40
		30	0.87	0.75	0.80
		60	0.25	0.60	0.37
	704x576	15	0.20	*	–
		30	0.54	0.85	0.66
		60	0.88	1	0.94
crew	176x144	15	0.40	*	–
		30	0.42	1	0.64
		60	0.24	0.56	0.35
	352x288	15	0.41	*	–
		30	0.66	1	0.81
		60	0.48	0.73	0.58
	704x576	15	0.29	*	–
		30	0.56	1	0.73
		60	0.60	0.86	0.71
flower	176x144	15	0.77	1	0.86
		30	0.25	*	–
	352x288	15	0.95	1	0.97
		30	0.87	1	0.93
football	176x144	7.5	0.05	–	–
		15	0.13	*	–
		30	0.51	1	0.70
	352x288	7.5	0.10	–	–
		15	0.12	*	–
		30	0.54	0.82	0.65
foreman	176x144	7.5	0.08	–	–
		15	0.16	*	–
		30	0.23	0.52	0.33
	352x288	7.5	0.03	–	–
		15	0.38	0.46	0.40
		30	0.27	0.37	0.29
greatwhite1	480x270	29.97	0.83	0.83	0.82
	960x540		0.58	0.61	0.58
	1920x1080		0.35	0.54	0.42
greatwhite2	480x270	29.97	0.52	0.70	0.59
	960x540		0.36	0.53	0.42

video	resolution	fps	rec.	prec.	MCC
	1920x1080		0.21	0.80	0.39
greatwhite3	480x270	29.97	0.05	–	–
	960x540		0.06	0.28	0.12
	1920x1080		0.05	–	–
greatwhite4	480x270	29.97	0.21	0.69	0.36
	960x540		0.22	0.66	0.37
	1920x1080		0.09	–	–
greatwhite5	480x270	29.97	0.80	0.87	0.83
	960x540		0.75	0.94	0.83
	1920x1080		0.61	1	0.77
gtaiv1	320x180	29.97	0.22	0.46	0.30
	640x360		0.12	0.29	0.17
	1280x720		0.11	0.22	0.14
gtaiv2	320x180	29.97	0.10	–	–
	640x360		0.18	–	–
	1280x720		0.13	–	–
hallmonitor	176x144	15	0.13	*	–
		30	0.01	*	–
	352x288	15	0.29	*	–
		30	0.25	1	0.49
harbour	176x144	15	0.50	0.68	0.56
		30	0.11	0.29	0.16
		60	0.02	*	–
	352x288	15	0.58	0.39	0.45
		30	0.61	1	0.77
		60	0.05	0.25	0.10
	704x576	15	0.50	0.44	0.45
		30	0.50	0.67	0.56
		60	0.60	0.87	0.71
mix	176x144	7.5	0.08	0.10	0.07
		15	0.25	0.39	0.29
		30	0.31	0.56	0.40
	352x288	7.5	0.08	0.12	0.08
		15	0.15	0.28	0.19
		30	0.45	0.56	0.48
	704x576	7.5	0.10	0.14	0.09
		15	0.19	0.18	0.15
		30	0.41	0.45	0.41
mix-notext	352x288	30	0.47	0.66	0.54
	704x576		0.40	0.61	0.48
mobile	176x144	7.5	0.88	1	0.93
		15	0.16	*	–
		30	0.02	*	–
	352x288	7.5	0.93	1	0.96
		15	0.85	1	0.91

video	resolution	fps	rec.	prec.	MCC
		30	0.17	–	–
motherdaughter	176x144	15	0.16	–	–
		30	0.16	0.31	0.21
	352x288	15	0.15	0.22	0.16
		30	0.27	0.56	0.38
soccer	176x144	15	0.21	*	–
		30	0.55	0.85	0.66
		60	0.79	0.77	0.77
	352x288	15	0.20	–	–
		30	0.45	0.82	0.59
		60	0.95	0.89	0.92
	704x576	15	0.13	*	–
		30	0.40	0.76	0.51
		60	0.91	0.88	0.90
stefan	176x144	15	0.35	*	–
		30	0.60	*	–
	352x288	15	0.30	*	–
		30	0.63	*	–
tempete	176x144	15	0	–	–
		30	0.01	*	–
	352x288	15	0.48	0.62	0.53
		30	0.06	*	–

When looking at the ECR graphs in more detail, we make an interesting observation: regardless of whether there are missing frames or not, the optimal combinations all have very definite peaks, and 'normal' motion is reduced to ECR values around or below 0.1. We suspect this is related to the amount of motion and r. Additionally, we note that changing the resolution or framerate, in terms of effects on the algorithm, effectively also is a way of saying how much motion there is. For example, doubling the resolution is very similar to a video of the same resolution, but twice as much motion. Or even halving the framerate.

Note that in this context, one might think that 'lots of motion' translates to 'high ECR values', which we discovered is not always the case. In fact, even a static background with just one moving object can lead to very different ECR's.

Note that 'motion' generally refers to 'high ECR values', in other words, the fraction of edges that are (dis)appearing. A slow pan can have many edge pixels, but generally only a few are over a distance r from their previous location. The same holds true for a few fast-moving objects.

It should also be noted that even a static background can play an important role in succesful detection.

For example, if framerate is kept the same but the resolution gets doubled, in terms of computations, edge pixels and entering/exiting pixels, it's roughly equivalent to a video of the same resolution and framerate, but twice as much motion. In this context, 'motion' refers to how fast something moves, not how much edge pixels there are. For example, a slow pan can have many edge pixels, but little motion, whereas cars driving by can have less edge pixels, but more motion.

We will use this information in our next step, which is to see if a variable r really is the key to good performance, regardless of resolution, framerate, or how much motion there is. As for our subhypothesis, the optimal combination of framerate and resolution depends on the content of the video.

4.3.3 Subhypothesis III: variable r

As mentioned before, the parameter r controls what the algorithm determines is 'normal motion' for a particular video. In all the results, the better performing combinations have one thing in common, namely that r is big enough to diminish the normal motion, yet at the same time small enough such that a missing frame will create a lot of extra pixels. This results in low values for normal frames, but relatively high values when there is a sudden jump.

Not caring yet about how to determine the optimal r, we reran the experiment using different values of r to see its influence. The results of this are listed in Table 4.5.

Table 4.5: Effects of different r

video	resolution	fps	MCC						
			$r=0$	$r=1$	$r=2$	$r=3$	$r=4$	$r=5$	$r=6$
blackmamba1	480x270	29.97	0.90	0.89	0.86	0.84	0.84	0.80	0.79
	960x540		0.90	0.90	0.90	0.88	0.90	0.89	0.86
	1920x1080		0.83	0.85	0.85	0.86	0.85	0.86	0.86
blackmamba2	480x270	29.97	0.66	0.76	0.75	0.79	0.79	0.81	0.74
	960x540		0.68	0.64	0.68	0.69	0.75	0.75	0.74
	1920x1080		0.55	0.62	0.66	0.72	0.70	0.67	0.69
blackmamba3	480x270	29.97	0.21	0.36	0.46	0.47	0.51	0.47	0.48
	960x540		–	–	0.37	0.40	0.44	0.45	0.48
	1920x1080		–	0.27	0.29	0.32	0.34	0.41	0.43
bus	176x144	7.5	–	–	–	–	–	–	–
		15	–	–	0.84	–	–	0.66	0.70
		30	0.89	0.82	0.87	0.80	0.66	0.41	0.19
	352x288	7.5	–	–	0.18	–	–	–	–
		15	–	–	0.84	–	–	–	–
		30	0.81	0.93	0.90	0.88	0.90	0.95	0.87
city	176x144	15	–	–	0.65	0.68	0.58	–	0.20
		30	0.73	0.77	0.67	–	–	0.11	0.10
		60	0.79	0.52	0.13	0.06	0.06	0.04	–
	352x288	15	–	–	0.18	0.40	0.55	0.60	0.70
		30	–	0.69	0.82	0.80	0.74	0.65	–
		60	0.90	0.95	0.75	0.37	0.18	0.08	0.06
	704x576	15	–	–	–	–	0.18	0.31	0.39
		30	–	–	0.50	0.66	0.79	0.78	0.85
		60	0.75	0.96	0.97	0.94	0.74	0.49	0.36
crew	176x144	15	–	–	–	–	–	–	–
		30	0.59	0.74	0.66	0.64	0.50	0.40	0.28
		60	0.52	0.58	0.44	0.35	0.19	0.14	0.06
	352x288	15	–	–	–	–	0.71	0.75	0.79

video	resolution	fps	MCC						
			$r=0$	$r=1$	$r=2$	$r=3$	$r=4$	$r=5$	$r=6$
		30	0.66	0.74	0.79	0.81	0.76	0.70	0.64
		60	0.71	0.70	0.70	0.58	0.48	0.40	0.35
		15	–	–	–	–	–	–	–
	704x576	30	–	0.58	0.65	0.73	0.76	0.67	0.61
		60	0.70	0.72	0.69	0.71	0.66	0.63	0.56
	176x144	15	0.79	0.98	0.86	0.86	–	0.33	–
		30	1	0.95	0.81	–	–	–	–
flower	352x288	15	–	–	0.77	0.97	0.87	–	–
		30	0.98	0.94	1	0.93	–	–	–
		7.5	–	–	–	–	–	–	–
	176x144	15	–	–	–	–	–	–	–
		30	0.40	0.63	0.76	0.70	0.67	0.75	0.74
football		7.5	–	–	–	–	–	–	–
	352x288	15	–	–	–	–	–	–	–
		30	–	0.63	0.68	0.65	0.70	0.67	0.65
		7.5	–	–	–	–	–	–	–
	176x144	15	–	–	0.21	–	–	–	0.12
		30	0.51	0.47	0.40	0.33	0.28	0.26	0.28
foreman		7.5	–	–	–	–	–	–	–
	352x288	15	–	–	–	0.40	0.39	0.24	–
		30	–	0.61	0.48	0.29	0.30	0.29	0.29
	480x270		0.30	0.54	0.67	0.82	0.79	0.77	0.76
greatwhite1	960x540	29.97	0.14	0.23	0.42	0.58	0.69	0.75	0.78
	1920x1080		0.06	0.12	0.25	0.42	0.50	0.61	0.66
	480x270		0.26	0.34	0.49	0.59	0.65	0.69	0.68
greatwhite2	960x540	29.97	–	0.27	0.33	0.42	0.44	0.53	0.60
	1920x1080		0.14	0.26	0.30	0.39	0.40	0.43	0.48
	480x270		–	0.04	0.08	–	–	–	0.24
greatwhite3	960x540	29.97	–	0.05	0.11	0.12	0.12	0.15	0.14
	1920x1080		0.05	0.09	–	–	–	–	0.14
	480x270		0.16	0.31	0.36	0.36	0.44	0.39	0.40
greatwhite4	960x540	29.97	–	–	0.30	0.37	0.39	0.33	0.32
	1920x1080		–	–	–	–	0.27	0.36	0.39
	480x270		0.73	0.82	0.87	0.83	0.82	0.74	0.66
greatwhite5	960x540	29.97	0.67	0.75	0.83	0.83	0.86	0.80	0.88
	1920x1080		0.48	0.66	0.70	0.77	0.84	0.84	0.83
	320x180		–	–	–	0.30	0.32	0.50	0.34
gtaiv1	640x360	29.97	0.19	0.15	0.14	0.17	0.22	–	–
	1280x720		0.11	0.12	0.13	0.14	0.17	0.19	0.14
	320x180		–	–	0.20	–	–	–	–
gtaiv2	640x360	29.97	–	–	–	–	–	–	–
	1280x720		–	–	–	–	–	0.15	0.16
	176x144	15	–	–	–	–	–	–	–
hallmonitor		30	0.53	0.48	–	–	–	–	–

video	resolution	fps	MCC						
			$r=0$	$r=1$	$r=2$	$r=3$	$r=4$	$r=5$	$r=6$
	352x288	15	–	0.68	–	–	–	–	–
		30	0.36	0.68	0.57	0.49	–	–	–
harbour	176x144	15	0.41	–	–	0.56	–	–	–
		30	0.38	0.49	0.31	0.16	–	–	–
		60	0.47	0.43	0.12	–	–	–	–
	352x288	15	–	0.38	0.46	0.45	0.53	0.59	–
		30	0.39	0.65	0.81	0.77	–	–	–
		60	0.40	0.67	0.50	0.10	–	–	–
	704x576	15	0.11	0.28	0.37	0.45	0.59	0.52	0.43
		30	0.30	0.47	0.47	0.56	0.63	0.68	0.63
		60	0.38	0.68	0.79	0.71	0.32	0.11	–
mix	176x144	7.5	0.08	0.06	0.04	0.07	0.12	0.12	0.13
		15	0.17	0.23	0.31	0.29	0.20	0.19	0.16
		30	0.40	0.46	0.45	0.40	0.40	0.31	0.36
	352x288	7.5	–	0.09	0.08	0.08	0.06	0.03	0.04
		15	0.10	0.11	0.12	0.19	0.23	0.25	0.25
		30	0.35	0.45	0.49	0.48	0.44	0.44	0.39
	704x576	7.5	–	–	0.12	0.09	0.08	0.08	0.08
		15	–	0.13	0.12	0.15	0.19	0.21	0.22
		30	0.23	0.36	0.37	0.41	0.44	0.42	0.43
mix-notext	352x288	30	0.35	0.49	0.57	0.54	0.42	0.40	0.40
	704x576		0.32	0.40	0.42	0.48	0.51	0.48	0.52
mobile	176x144	7.5	0.96	0.96	1	0.93	–	–	–
		15	0.85	1	0.80	–	–	–	–
		30	0.95	0.86	–	–	–	–	–
	352x288	7.5	–	–	0.75	0.96	1	1	–
		15	0.89	1	1	0.91	0.78	–	–
		30	1	1	0.82	–	–	–	–
motherdaughter	176x144	15	–	–	–	–	–	–	–
		30	0.42	0.34	0.28	0.21	0.22	0.08	0.07
	352x288	15	–	–	0.22	0.16	0.15	–	–
		30	0.46	0.32	0.37	0.38	0.40	0.35	–
soccer	176x144	15	–	–	–	–	0.31	–	–
		30	–	0.48	0.60	0.66	0.63	0.64	0.60
		60	0.92	0.89	0.78	0.77	0.62	0.55	0.48
	352x288	15	–	0.20	–	–	–	–	0.19
		30	–	0.47	0.49	0.59	0.61	0.67	0.68
		60	0.90	0.95	0.97	0.92	0.86	0.84	0.77
	704x576	15	–	–	–	–	–	–	–
		30	–	0.39	0.45	0.51	0.53	0.68	0.76
		60	0.78	0.85	0.94	0.90	0.92	0.87	0.84
stefan	176x144	15	–	–	–	–	–	–	–
		30	–	–	–	–	–	–	–
	352x288	15	–	–	–	–	–	–	–

video	resolution	fps	MCC						
			$r = 0$	$r = 1$	$r = 2$	$r = 3$	$r = 4$	$r = 5$	$r = 6$
		30	–	–	–	–	–	–	–
tempete	176x144	15	1	0.88	–	–	–	–	–
		30	0.77	–	–	–	–	–	–
	352x288	15	1	1	0.85	0.53	–	0.03	–
		30	0.89	0.88	–	–	–	–	–

In addition to the already observed relations, we can clearly see that there are more, particularly:

- Under a constant framerate, if a video performs well at a certain resolution and r, (roughly) doubling the resolution and r also generally leads to good results.
- The relation between framerate and r under a constant resolution is less defined: it is clear that increasing the framerate decreases the optimal r, but it differs greatly per video by how much. On some it is not even halved, on others it drops to less than a quarter of the original value.

However, as already mentioned in the previous subhypothesis, nearly all the high-scoring ones have one thing in common: the bulk of the ECR values are around or below 0.1. An example graph is given in Figure 4.1.

Essentially, a well-chosen r will reduce the regular motion to low values, while at the same time still giving significantly high values for missing frames. That is, assuming the video is somewhat constant in terms of motion, like the city video is. Most videos however, aren't, and the effects of a different r start to become obvious. One particularly bad performing video is the 'mix' video, which under no r achieves an MCC above 0.5, and that is still quite unsatisfactory. Figure 4.2 shows the ECR graphs for this video.

On a difficult video like this, the importance of r becomes clear: for example, the peak at frame 97 is very significant for $r = 3$, but almost disappears for $r = 0$. It also shows that using a simple metric like the average is not going to give the best possible results on generic videos, since different parts of the video have a different optimal r. An intuitive method is to take another window (v) around the frame we're computing, and use the average or median of that. This window would most likely need to be larger than w, but not too large. Because $r+1$ means equal or lower ECR values than r, we could simply look for the average or median value of this window v closest to some ECR value that gives good results.

Another problem we noticed was the tendency of the algorithm to fluctuate quite badly, as well as having generally poor results on videos with little motion. Both have different causes, but lead to the same result: false positives.

The first has a rather surprising source, namely the static parts of the video, and is purely related to the background, not the actual motion. Figure 4.3 show two pictures of a car. In both pictures, the car is the only moving entity, so one would expect more or less the same ECR values. However, this is not true. In the left picture, there are almost no edges aside from the car itself, meaning that almost all edge pixels are also exiting or entering. This in turn leads to not only high ECR values, but also potentially fluctuating ones.

The right picture shows the same car, moving at the same speed, but the background has changed. The amount of exiting/entering pixels is still (roughly) the same, but there are significantly more edge pixels overall. The result is very low ECR values.

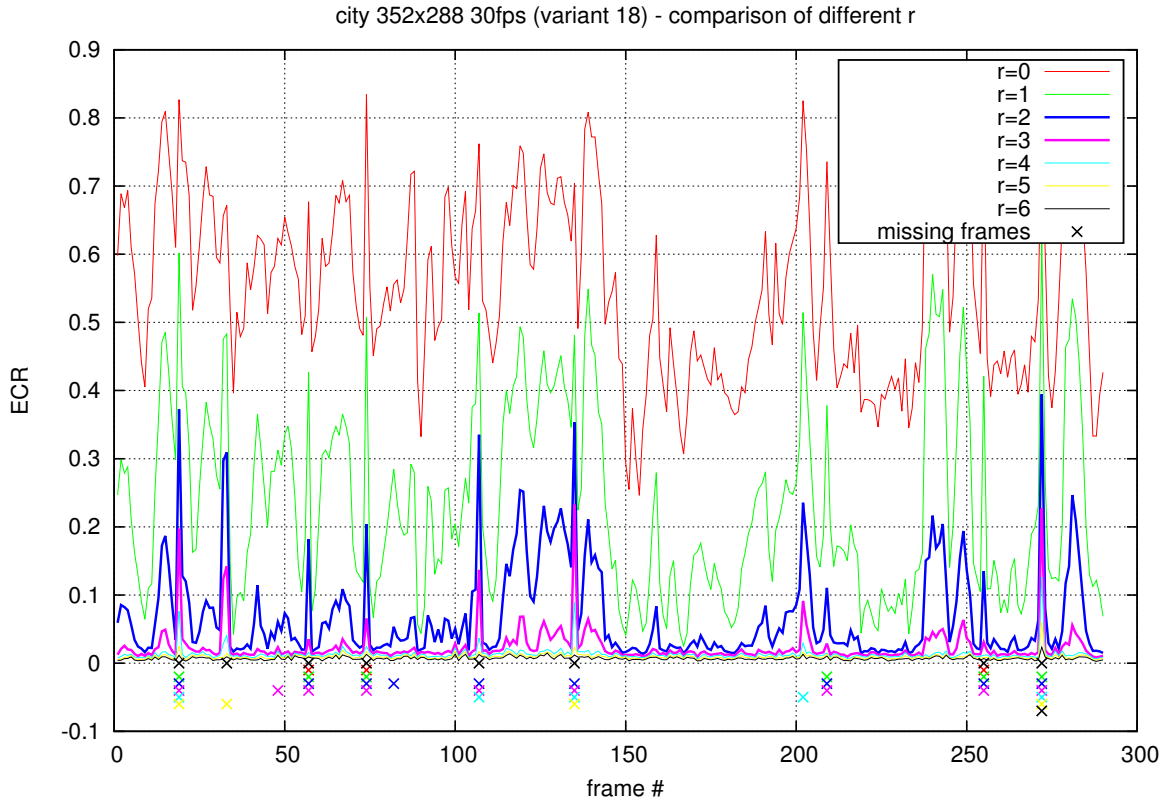


Figure 4.1: ECR graphs of the same video, but different values of r. For $r = 2$ and $r = 3$ the peaks stand out more compared to other values of r, which makes detection more reliable and accurate. Notice that the majority of the ECR values of these two graphs falls below 0.1. For lower values of r there is more oscillation; for higher values the peaks diminish.

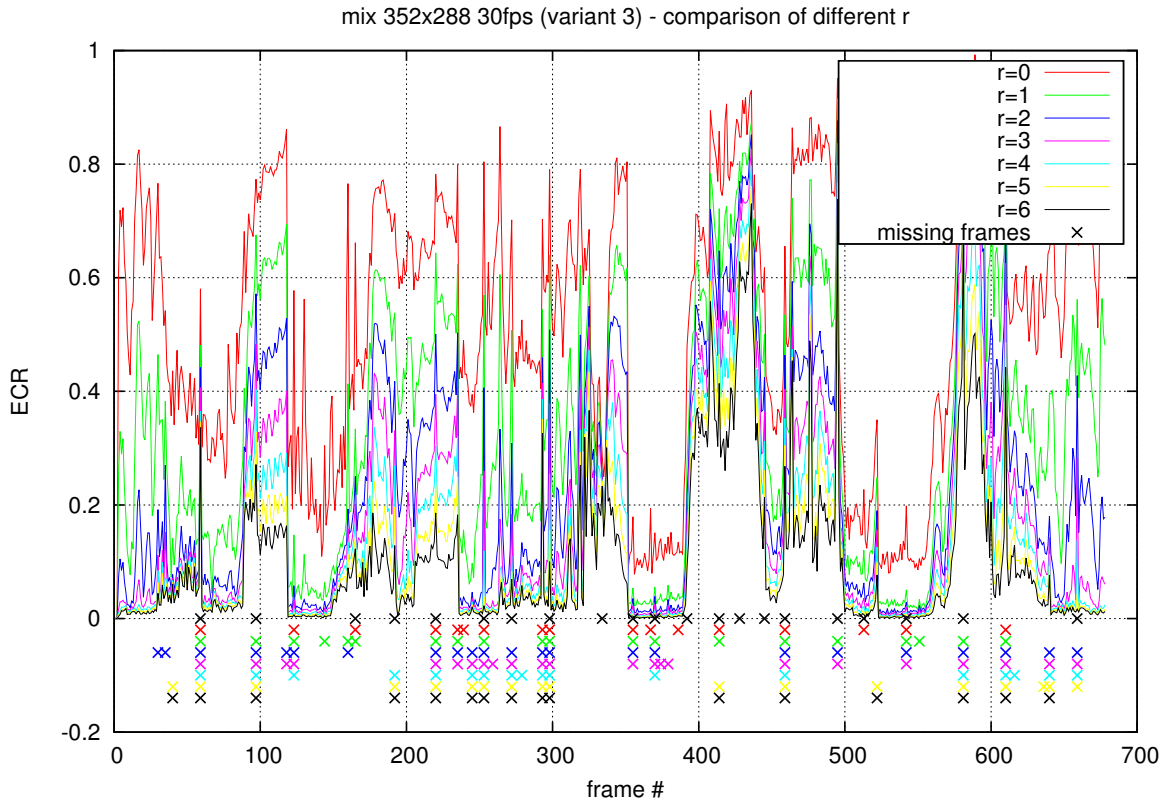
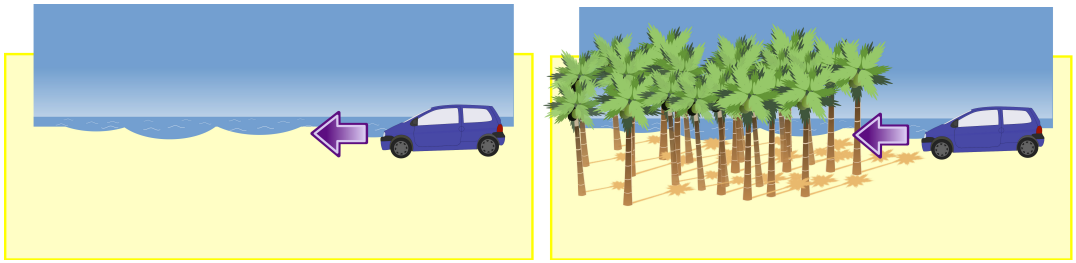


Figure 4.2: ECR graphs of the same video, but different values of r.



(a) (b)

Figure 4.3: In both pictures, only the car is moving. The ECR of the first picture will be much higher than the right picture.

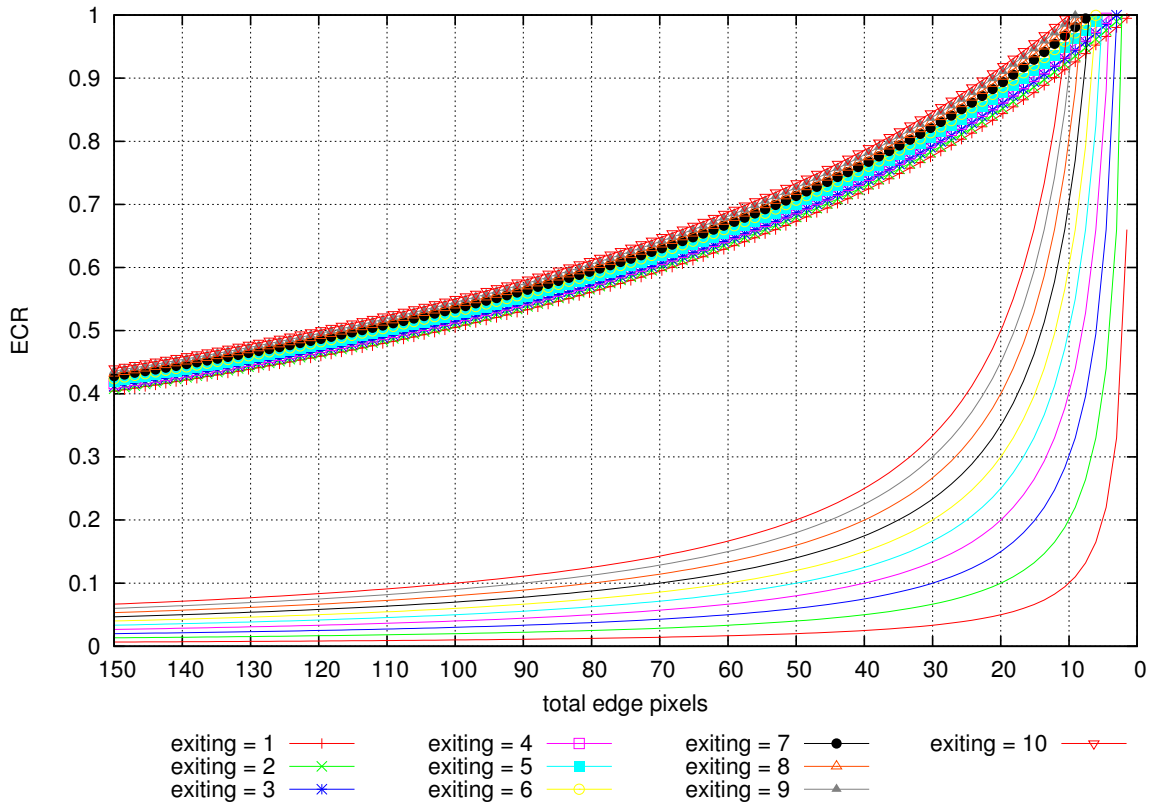


Figure 4.4: Influence of adding a bit of fake motion to the ECR computation. The thin lines show the original, without added fake motion, resulting ECR's; the thick lines show the result of adding 100 pixels to the computation.

The second problem are in fact those very low ECR values. When we get to ECR values of for example below 0.05, a few exiting pixels more or less, combined with only a few more (or less) total edge pixels can have a relatively significant impact on the ECR value.

The first problem can be solved by adding some constant amount of edge pixels to each frame; the second by adding a constant amount to the exiting resp. entering pixels. Combining these two reduces to simply adding it to all of them, compare Equations 4.1 and 4.2 to Equations 3.1 resp. 3.1:

$$\rho_{out} = \frac{\text{number of exiting edge pixels} + some\_constant\_amount}{\text{number of black (edge) pixels in } E + some\_constant\_amount} \tag{4.1}$$

$$\rho_{in} = \frac{\text{number of entering edge pixels} + some\_constant\_amount}{\text{number of black (edge) pixels in } E' + some\_constant\_amount} \tag{4.2}$$

The effect of adding a constant of 100 pixels is shown in Figure 4.4. Note that 'a 100 pixels' is quite vague when dealing with video in different resolutions; instead, we made it a percentage of the resolution (width * height) of the video. As an experiment, we reran the algorithm with this small modification, adding 0.5%, which obtained the results listed in Table 4.6.

Table 4.6: Effects of different r with added 0.5% motion. Listed in red or green are the decrease resp. increase in performance compared to no added motion (Table 4.5.
* = Averages (FP/FN/TP/TN) are worse (compared to no added motion)
** = Averages (FP/FN/TP/TN) are better (compared to no added motion)
*** = Introduces an additional false positive (compared to no added motion)

video	resolution	fps	MCC						
			$r=0$	$r=1$	$r=2$	$r=3$	$r=4$	$r=5$	$r=6$
blackmamba1	480x270	29.97	0.90	0.89	0.86	0.84	$0.86_{+0.02}$	$0.81_{+0.01}$	$0.78_{-0.01}$
	960x540		0.90	0.90	$0.89_{-0.01}$	$0.92_{+0.04}$	$0.94_{+0.04}$	$0.92_{+0.03}$	$0.90_{+0.04}$
	1920x1080		$0.85_{+0.02}$	$0.87_{+0.02}$	$0.86_{+0.01}$	$0.87_{+0.01}$	$0.88_{+0.03}$	$0.91_{+0.05}$	$0.91_{+0.05}$
blackmamba2	480x270	29.97	$0.69_{+0.02}$	$0.77_{+0.01}$	$0.78_{+0.03}$	$0.82_{+0.03}$	$0.81_{+0.02}$	$0.83_{+0.02}$	$0.77_{+0.03}$
	960x540		$0.64_{-0.04}$	$0.66_{+0.03}$	$0.72_{+0.04}$	$0.76_{+0.07}$	$0.76_{+0.01}$	$0.74_{-0.01}$	0.74
	1920x1080		$0.58_{+0.03}$	$0.65_{+0.03}$	0.66	$0.71_{-0.01}$	$0.74_{+0.04}$	$0.71_{+0.04}$	$0.73_{+0.04}$
blackmamba3	480x270	29.97	$0.19_{-0.02}$	$0.38_{+0.02}$	$0.49_{+0.03}$	0.47	$0.48_{-0.03}$	$0.45_{-0.02}$	$0.44_{-0.04}$
	960x540		–	$0.37_{+??}$	$0.36_{-0.01}$	0.40	$0.42_{-0.02}$	$0.48_{+0.03}$	0.48
	1920x1080		–	$-_{-??}$	$-_{-??}$	$0.37_{+0.05}$	$0.38_{+0.04}$	$0.38_{-0.03}$	$0.41_{-0.02}$
bus	176x144	7.5	–	–	–	–	–	–	–
		15	–	–	0.84	$0.59_{-??*}$	–	0.66	0.70
		30	0.89	0.82	0.87	$0.86_{+0.06}$	$0.65_{-0.01}$	0.41	$0.21_{+0.02}$
	352x288	7.5	–	–	0.18	–	–	–	–
		15	–	–	0.84	–	–	–	–
		30	0.81	0.93	0.90	$0.89_{+0.01}$	0.90	$0.94_{-0.01}$	$0.98_{+0.11}$
city	176x144	15	–	–	0.65	$0.71_{+0.03}$	$0.60_{+0.02}$	–	$-_{+??**}$
		30	0.73	$0.81_{+0.04}$	$0.80_{+0.13}$	–	–	$0.12_{+0.01}$	$0.07_{-0.03}$
		60	0.79	0.52	$0.11_{-0.02}$	$0.07_{+0.01}$	$0.08_{+0.02}$	0.04	–
	352x288	15	–	–	$0.19_{+0.01}$	0.40	$0.52_{-0.03}$	$0.64_{+0.04}$	0.70
		30	–	0.69	0.82	$0.83_{+0.03}$	0.74	$0.62_{-0.03}$	–
		60	0.90	$0.94_{-0.01}$	$0.77_{+0.02}$	$0.39_{+0.02}$	$0.16_{-0.02}$	$0.04_{-0.04}$	$0.04_{-0.02}$
	704x576	15	–	–	–	–	0.18	$0.30_{-0.01}$	0.39
		30	–	$0.37_{+??}$	0.50	$0.61_{-0.05}$	0.79	0.78	$0.80_{-0.05}$
		60	0.75	0.96	0.97	$0.93_{-0.01}$	0.74	$0.52_{+0.03}$	$0.34_{-0.02}$
crew	176x144	15	–	–	–	–	–	–	–
		30	$0.71_{+0.12}$	$0.79_{+0.05}$	$0.70_{+0.04}$	$0.57_{-0.07}$	$0.45_{-0.05}$	0.40	$0.31_{+0.03}$
		60	$0.57_{+0.05}$	$0.62_{+0.04}$	$0.46_{+0.02}$	$0.31_{-0.04}$	$0.20_{+0.01}$	$0.10_{-0.04}$	$0.08_{+0.02}$
	352x288	15	–	–	–	–	$0.77_{+0.06}$	$0.79_{+0.04}$	$0.80_{+0.01}$
		30	$0.71_{+0.05}$	$0.76_{+0.02}$	$0.80_{+0.01}$	$0.86_{+0.05}$	$0.81_{+0.05}$	0.70	$0.67_{+0.03}$
		60	$0.77_{+0.06}$	$0.71_{+0.01}$	$0.67_{-0.03}$	0.58	$0.52_{+0.04}$	$0.43_{+0.03}$	$0.42_{+0.07}$
	704x576	15	–	–	–	–	–	–	–
		30	$0.43_{+??}$	$0.53_{-0.05}$	$0.69_{+0.04}$	$0.79_{+0.06}$	0.76	$0.78_{+0.11}$	$0.77_{+0.16}$
		60	$0.72_{+0.02}$	$0.75_{+0.03}$	$0.77_{+0.08}$	$0.68_{-0.03}$	$0.64_{-0.02}$	$0.64_{+0.01}$	0.56
flower	176x144	15	$0.78_{-0.01}$	$0.97_{-0.01}$	0.86	0.86	–	0.33	–
		30	1	$1_{+0.05}$	$0.85_{+0.04}$	–	–	–	–

video	resolution	fps	MCC						
			$r=0$	$r=1$	$r=2$	$r=3$	$r=4$	$r=5$	$r=6$
	352x288	15	–	–	0.77	$0.94_{-0.03}$	$0.86_{-0.01}$	–	–
		30	0.98	0.94	1	$0.92_{-0.01}$	–	–	–
football	176x144	7.5	–	–	–	–	–	–	–
		15	–	–	–	–	–	–	–
		30	$0.46_{+0.06}$	0.63	$0.77_{+0.01}$	$0.64_{-0.06}$	$0.75_{+0.08}$	$0.74_{-0.01}$	0.74
	352x288	7.5	–	–	–	–	–	–	–
		15	–	–	–	–	–	–	–
		30	–	$0.61_{-0.02}$	$0.64_{-0.04}$	$0.71_{+0.06}$	$0.69_{-0.01}$	$0.71_{+0.04}$	$0.64_{-0.01}$
foreman	176x144	7.5	–	–	–	–	–	–	–
		15	–	–	– $_{-??}$	–	–	–	0.12
		30	$0.52_{+0.01}$	0.47	0.40	$0.32_{-0.01}$	$0.31_{+0.03}$	– $_{+??**}$	$0.23_{-0.05}$
	352x288	7.5	–	–	–	–	–	–	–
		15	–	–	–	$0.37_{-0.03}$	0.39	$0.25_{+0.01}$	–
		30	–	$0.62_{+0.01}$	$0.54_{+0.06}$	$0.30_{+0.01}$	0.30	$0.30_{+0.01}$	$0.33_{+0.04}$
greatwhite1	480x270	29.97	0.30	$0.55_{+0.01}$	$0.68_{+0.01}$	0.82	$0.80_{+0.01}$	$0.84_{+0.07}$	$0.75_{-0.01}$
	960x540		0.14	0.23	0.42	0.58	0.69	0.75	$0.77_{-0.01}$
	1920x1080		$0.07_{+0.01}$	0.12	$0.24_{-0.01}$	$0.41_{-0.01}$	0.50	$0.62_{+0.01}$	0.66
greatwhite2	480x270	29.97	0.26	$0.33_{-0.01}$	$0.54_{+0.05}$	$0.61_{+0.02}$	$0.67_{+0.02}$	$0.68_{-0.01}$	$0.64_{-0.04}$
	960x540		–	$0.24_{-0.03}$	$0.32_{-0.01}$	$0.44_{+0.02}$	0.44	$0.56_{+0.03}$	0.60
	1920x1080		$0.12_{-0.02}$	$0.24_{-0.02}$	$0.31_{+0.01}$	$0.37_{-0.02}$	$0.37_{-0.03}$	$0.41_{-0.02}$	$0.47_{-0.01}$
greatwhite3	480x270	29.97	$0.04_{+??}$	0.04	$0.05_{-0.03}$	–	–	–	$0.23_{-0.01}$
	960x540		–	0.05	$0.10_{-0.01}$	$0.10_{-0.02}$	$0.11_{-0.01}$	0.15	$0.16_{+0.02}$
	1920x1080		– $_{-??}$	$0.05_{-0.04}$	$0.08_{+??}$	–	–	–	$0.10_{-0.04}$
greatwhite4	480x270	29.97	– $_{-??}$	0.31	0.36	0.36	$0.45_{+0.01}$	$0.43_{+0.04}$	$0.41_{+0.01}$
	960x540		–	–	$0.31_{+0.01}$	$0.44_{+0.07}$	0.39	$0.34_{+0.01}$	$0.30_{-0.02}$
	1920x1080		–	–	–	–	0.27	$0.40_{+0.04}$	$0.42_{+0.03}$
greatwhite5	480x270	29.97	$0.72_{-0.01}$	0.82	0.87	0.83	0.82	$0.77_{+0.03}$	$0.71_{+0.05}$
	960x540		0.67	0.75	0.83	0.83	$0.85_{-0.01}$	$0.82_{+0.02}$	0.88
	1920x1080		0.48	$0.65_{-0.01}$	0.70	0.77	$0.83_{-0.01}$	0.84	$0.82_{-0.01}$
gtaiv1	320x180	29.97	–	–	–	$0.29_{-0.01}$	$0.40_{+0.08}$	$0.49_{-0.01}$	– $_{+??**}$
	640x360		$0.17_{-0.02}$	$0.13_{-0.02}$	– $_{-??}$	$0.11_{-0.06}$	– $_{-??}$	–	–
	1280x720		0.11	$0.11_{-0.01}$	$0.11_{-0.02}$	0.14	$0.13_{-0.04}$	$0.15_{-0.04}$	$0.11_{-0.03}$
gtaiv2	320x180	29.97	–	–	$0.18_{-0.02}$	–	–	–	–
	640x360		–	–	–	–	–	–	–
	1280x720		–	–	–	–	–	– $_{-??}$	– $_{+??**}$
hallmonitor	176x144	15	–	–	–	–	–	–	–
		30	0.53	$0.50_{+0.02}$	–	–	–	–	–
	352x288	15	–	0.68	–	–	–	–	–
		30	0.36	$0.67_{-0.01}$	$0.56_{-0.01}$	$0.47_{-0.02}$	–	–	–
harbour	176x144	15	$0.43_{+0.02}$	$0.41_{+??}$	–	$0.65_{+0.09}$	–	–	–
		30	0.38	$0.54_{+0.05}$	$0.30_{-0.01}$	$0.13_{-0.03}$	–	–	–

video	resolution	fps	MCC						
			$r=0$	$r=1$	$r=2$	$r=3$	$r=4$	$r=5$	$r=6$
mix	352x288	60	0.47	$0.40_{-0.03}$	$0.10_{-0.02}$	–	–	–	–
		15	–	$0.39_{+0.01}$	0.46	0.45	$0.55_{+0.02}$	$0.57_{-0.02}$	–
		30	$0.38_{-0.01}$	0.65	0.81	0.77	–	–	–
		60	$0.42_{+0.02}$	$0.66_{-0.01}$	0.50	$0.09_{-0.01}$	–	–	–
	704x576	15	0.11	$0.29_{+0.01}$	0.37	0.45	0.59	$0.47_{-0.05}$	$0.42_{-0.01}$
		30	0.30	$0.49_{+0.02}$	$0.48_{+0.01}$	$0.59_{+0.03}$	$0.65_{+0.02}$	$0.69_{+0.01}$	$0.65_{+0.02}$
		60	0.38	$0.67_{-0.01}$	$0.81_{+0.02}$	0.71	$0.30_{-0.02}$	$-_{+??**}$	–
	176x144	7.5	$0.09_{+0.01}$	$0.08_{+0.02}$	$0.02_{-0.02}$	0.07	0.12	0.12	$0.14_{+0.01}$
		15	$0.19_{+0.02}$	$0.21_{-0.02}$	0.31	$0.28_{-0.01}$	$0.21_{+0.01}$	$0.16_{-0.03}$	0.16
		30	$0.41_{+0.01}$	$0.48_{+0.02}$	$0.50_{+0.05}$	$0.38_{-0.02}$	$0.36_{-0.04}$	$0.33_{+0.02}$	$0.30_{-0.06}$
	352x288	7.5	–	$0.11_{+0.02}$	$0.07_{-0.01}$	$0.07_{-0.01}$	$0.05_{-0.05}$	$0.06_{+0.03}$	$0.06_{+0.02}$
		15	$0.12_{+0.02}$	$0.13_{+0.02}$	$0.15_{+0.03}$	$0.16_{-0.03}$	$0.25_{+0.02}$	0.25	$0.27_{+0.02}$
		30	$0.36_{+0.01}$	$0.44_{-0.01}$	$0.52_{+0.03}$	$0.50_{+0.02}$	$0.49_{+0.05}$	$0.43_{-0.01}$	$0.28_{-0.01}$
	704x576	7.5	–	–	$0.11_{-0.01}$	$0.12_{+0.03}$	0.08	0.08	$0.11_{+0.03}$
		15	–	$-_{-??}$	$0.11_{-0.01}$	$0.13_{-0.02}$	$0.16_{-0.03}$	$0.13_{-0.08}$	$0.19_{-0.03}$
		30	$0.25_{+0.02}$	$0.37_{+0.01}$	0.37	$0.45_{+0.04}$	$0.49_{+0.05}$	$0.45_{+0.03}$	0.43
mix-notext	352x288	30	$0.37_{+0.02}$	$0.52_{+0.03}$	$0.61_{+0.04}$	$0.59_{+0.05}$	$0.44_{+0.02}$	$0.41_{+0.01}$	$0.38_{-0.02}$
	704x576		$0.30_{-0.02}$	0.40	$0.43_{+0.01}$	$0.57_{+0.09}$	$0.57_{+0.06}$	$0.58_{+0.10}$	$0.58_{+0.06}$
mobile	176x144	7.5	0.96	0.96	1	0.93	–	–	–
		15	0.85	1	$0.81_{+0.01}$	–	–	–	–
		30	0.95	0.86	–	–	–	–	–
	352x288	7.5	–	–	0.75	0.96	1	1	–
		15	0.89	1	1	0.91	0.78	–	–
		30	$0.96_{-0.04}$	1	0.82	–	–	–	–
motherdaughter	176x144	15	–	–	–	–	–	–	–
		30	$0.43_{+0.01}$	$0.37_{+0.03}$	$0.26_{-0.02}$	0.21	$0.25_{+0.03}$	$0.12_{+0.04}$	$0.09_{+0.02}$
	352x288	15	–	–	0.22	0.16	$0.11_{-0.04}$	–	–
		30	$0.41_{+0.05}$	$0.40_{+0.08}$	$0.40_{+0.03}$	$0.50_{+0.12}$	$0.47_{+0.07}$	$-_{-??}$	–
soccer	176x144	15	–	–	–	–	$0.25_{-0.06}$	–	–
		30	–	$0.50_{+0.02}$	0.60	$0.62_{-0.04}$	$0.62_{-0.01}$	$0.63_{-0.01}$	$0.59_{-0.01}$
		60	$0.93_{+0.01}$	$0.93_{+0.04}$	0.78	$0.75_{-0.02}$	$0.61_{-0.01}$	$0.57_{+0.02}$	$0.45_{-0.03}$
	352x288	15	–	0.20	–	–	–	–	$0.21_{+0.03}$
		30	–	$0.45_{-0.02}$	$0.51_{+0.02}$	0.59	$0.62_{+0.01}$	$0.65_{-0.02}$	$0.65_{-0.03}$
		60	$0.89_{-0.01}$	0.95	$0.96_{-0.01}$	$0.90_{-0.02}$	$0.87_{+0.01}$	$0.85_{+0.01}$	$0.78_{+0.01}$
	704x576	15	–	–	–	–	–	–	–
		30	–	$0.40_{+0.01}$	$0.43_{-0.02}$	$0.48_{-0.03}$	$0.57_{+0.04}$	0.68	$0.72_{-0.04}$
		60	$0.80_{+0.02}$	$0.87_{+0.02}$	$0.92_{-0.02}$	0.90	$0.94_{+0.02}$	$0.94_{+0.07}$	$0.94_{+0.10}$
stefan	176x144	15	–	–	–	–	–	–	–
		30	–	–	–	–	–	–	–
	352x288	15	–	–	–	–	–	–	–
		30	–	–	–	–	–	–	–

video	resolution	fps	MCC						
			$r = 0$	$r = 1$	$r = 2$	$r = 3$	$r = 4$	$r = 5$	$r = 6$
tempete	176x144	15	1	0.88	–	–	–	–	–
		30	$0.78_{+0.01}$	–	–	–	–	–	–
	352x288	15	1	1	0.85	0.53	–	0.03	$-_{-??***}$
		30	0.89	$0.92_{+0.03}$	–	–	–	–	–

Figure 4.5: Frames 470-540 of a roller coaster video sequence (see Figure 4.6). Frames 478-486 are just shots of the sky, causing the edge detector to detect no edges. Frames 495-525 show the camera autocorrecting for illumination. At frame 538 (not shown here), a false positive is sometimes detected.

In most cases, the increased or decreased performance is not very significant, but there are some very interesting observations we can make from it:

- The videos that improved universally generally have about a constant amount of motion
- The few videos on which performance decreased universally performed quite bad to begin with
- On the remaining videos, it's usually the r's that originally performed best that get improved, whereas the others decrease in performance.

Additionally, although the MCC values remain largely the same, on most videos recall was slightly reduced, while precision went up.

The trick of adding some fake motion also solves one particularly nasty edge case: no motion at all. This happened in at least one of the test videos, 'blackmamba2', where a suspended roller coaster with a front-mounted camera went through a loop. The camera usually had a small part of the track visible, as well as surrounding scenery, but as it went through the loop, forces caused the camera to lose the track, and since it was going upwards, only the sky was left. The relevant frames of the video are depicted in Figure 4.5, whereas the resulting ECR values are shown in Figure 4.6.

During this particular sequence, frames 470 until 477 share a common characteristic, namely that from one frame to the next, a more or less constant number of edge pixels disappear. This leads to the following chain of events:

- There are no entering pixels.
- The number of exiting pixels is more or less constant.
- Therefore, the total number of edge pixels decreases at a more or less constant rate.
- As the fraction of exiting pixels compared to the total number of edge pixels grows, so does the ECR.

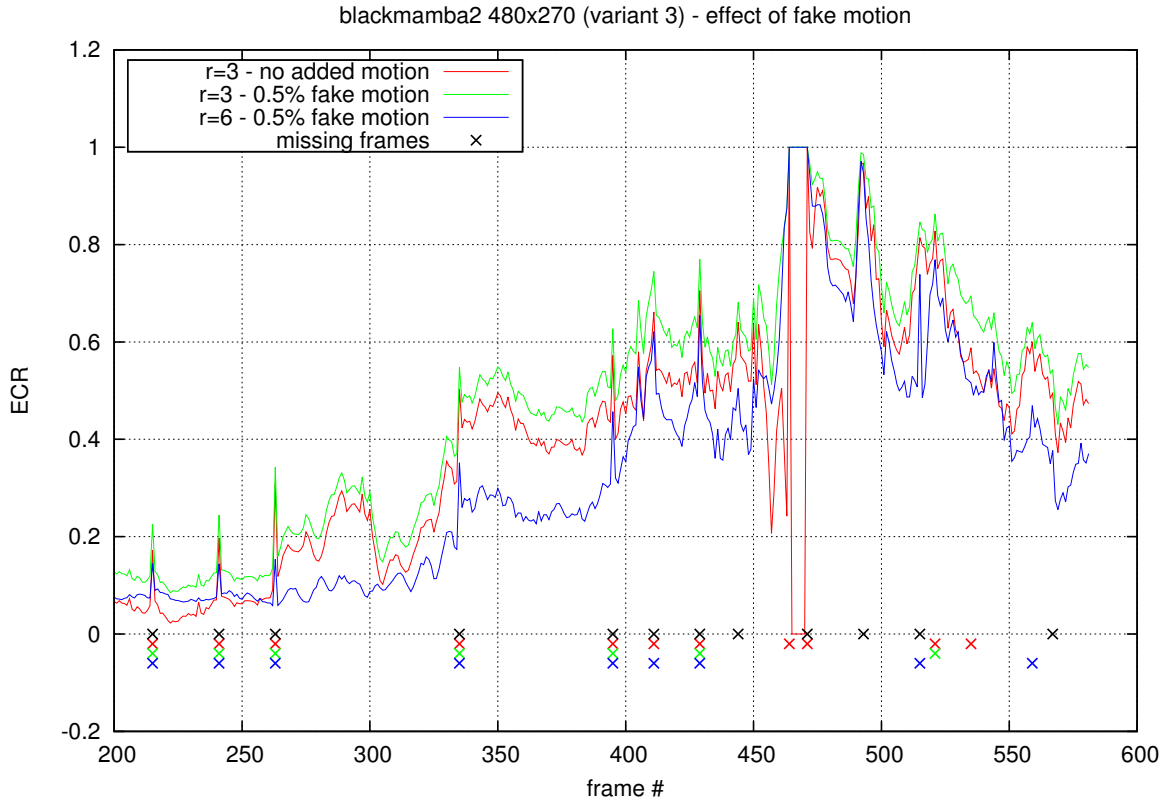


Figure 4.6: ECR graph of a video. Frames 478-485 contain no edges (see Figure 4.5). Frames 538 and 552 are incorrectly classified as a missing frame. Notice that, for the graph with fake motion, the peaks when the ECR is about 0.2 are much larger than later on in the video.

Without the added motion, from frames 470 until 477, the ECR becomes unstable, before suddenly dropping to zero, which causes a false positive. With just a little bit of added motion, the graph climbs slowly to 1, and stays there when there are no edges. The graph also shows the different r's in action: especially in the second part of the graph, the $r = 6$ line is superior to the other two, while in the first half, the green line creates more significant peaks.

We can conclude that a variable r has a positive impact on results. There are certain relations between resolution, framerate, content and r, but the optimal combination(s) only depends on r, since the right choice of r counters effects from resolution and framerate.

4.3.4 False Positives and False Negatives

Remember that in our benchmark, we also annotated the videos with a number of categories (see Table 4.1). Although most videos would have had slightly better ratings if we only took the 'normal' labeled frames into account, there were no clear cases where some type consistently failed. The only exception were peaks when an ambient light was suddenly turned on or off. These created peaks exactly like a missing frame. Camera flashes were generally not a problem, thus an easy solution would be to just split the video there (ie. say that it's a hard cut).

Adding the 0.5% fake motion also eliminated a lot of false positives, especially in parts where there was little motion. Compared to not adding this little bit, ECR values were less fluctuating, and the only additional false positives that were previously detected corrected occured in parts where there was extremely little motion to begin with.

However, by far the most important factor is the right choice of r. Recall Figure 4.2, where we showed the influence of just a different r. Figure 4.7 shows the same graph, but only the parts where the average ECR is around 0.2 (or the closest possible). On this difficult video, of the 21 actually missing frames 16 get detected, with 10 false positives. This means a recall of 0.76, a precision of 0.62, and an MCC of about 0.67 on this particular example. Still not the best numbers, but already significantly better than any of the results with a static r throughout the video (which was an MCC of 0.5 at best). Also notice that the graph only goes up to $r = 6$ but could at some points benefit from a higher r, meaning it could possibly attain an even higher recall and precision.

4.3.5 Conclusion

Our first experiment showed that framerate and resolution influence the outcome. While seemingly trivial, the relation between resolution and framerate, and which combinations perform well is interesting. In the second experiment we looked at essentially the same problem but from a different angle, namely how the optimal combination appears to depend on the content. Finally, the third experiment allowed r to be variable, drastically improving results, which was in line with our subhypothesis. There was a small issue with some false positives under certain circumstances, the majority of which was solved by adding a bit of fake motion. This also had the effect of generally improving already well-performing combinations.

All in all, the experiments show that the Edge Change Ratio algorithm, which was originally developed to detect just scene transitions, can be extended to detect missing frames instead.

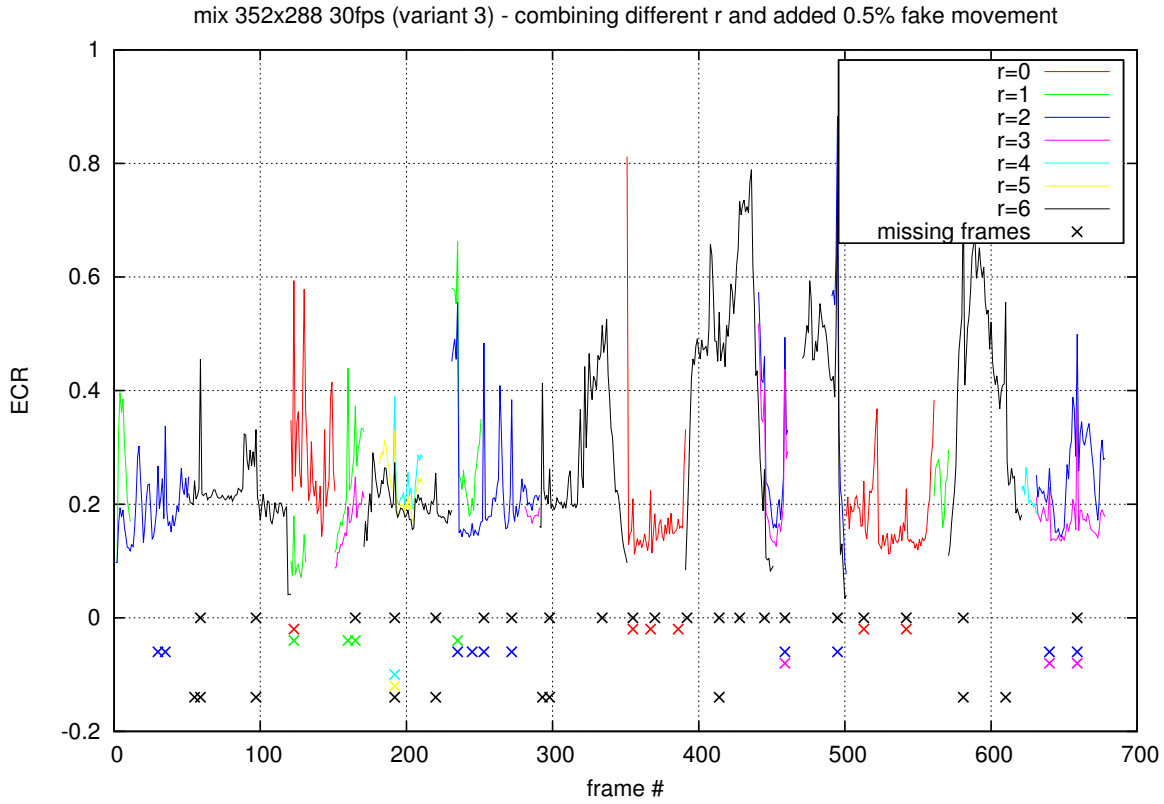


Figure 4.7: Combining 0.5% added fake movement and different values for r. Only the parts closest to ECR's of 0.2 are shown. This graph is not derived from an actual experiment, only to show that we expect overall performance to significantly improve by selecting the right r.

Chapter 5

Concluding remarks

We set out to create a method that automatically detects missing frames, independent of resolution, framerate, or how much actual motion there was in the video. Furthermore, we wanted as little user input as possible, and would prefer precision over recall, since not every missing frame can be detected by definition.

We based our method on the *Edge Change Ratio* algorithm, which was originally developed to detect scene breaks. We first determined whether and how the individual components – resolution, framerate, and motion – influenced the original algorithm. Most videos performed best at a medium resolution with a medium framerate, but also on versions where both resolution *and* framerate were doubled or halved. Some videos, in particular the ones with very little or a lot of motion, performed better at a different combination of framerate and resolution.

We then argued that changing the resolution or framerate is, to some extent, the same as changing the dilation radius r. We ran all the videos with different values of r and found a number of relations between framerate, resolution, motion, and r. In doing this we also discovered a few edge cases in the algorithm that caused it to produce (relatively) fluctuating ECR values, which in turn caused false positives. To counter this, we added a small amount of 'fake motion' to every frame during computations, causing more stable ECR values, and thus (albeit at the cost of some recall) a higher precision. As a bonus, this caused the often best-performing r to produce ECR values averaging around 0.2, thus making automatic selection of r possible in theory. However, that only works for videos of the same sort of content throughout the entire video; more complex videos could benefit from a dynamic r.

As far as computation times are concerned, framerate has a linear effect on it, since it is the same problem as a longer video. Resolution has a quadratic impact on computation time, while the impact of r depends on the implementation of the dilation step – a naive approach would be quadratic. Note that higher resolutions often co-occur with a higher r as well, so it might be worth just downscaling the video first. We only tested up to $r = 6$, but if the results are any indication, some videos – including not-HD ones – could benefit from a higher value.

Because of this, and also because it does not take an optimal r for each frame, but only for an entire video, the method is still incomplete. There are several ways in which the algorithm can possibly be improved, for example:

- Compute per frame the optimal r using a window, not for the entire video. We expect this to increase performance especially on more complex videos dramatically. An intuitive cadidate for computation is starting out with $r = 0$, and only compute higher r's

if necessary.

- Improve the classifier. In our experiment, we used a relatively simple method to decide, after everything had been computed, whether something was a missing frame or not.
- Allow for $r > 6$. We only computed up to $r = 6$, but results indicate that even not-1080p videos could benefit from a higher r.

Appendix A

Benchmark

Key:

- (RS) = Research video. Videos often used in research. Usually 'perfect' in some way, for example super-smooth camera movement, constant type of movement, etc.
- (RW) = Real-World video. Video downloaded off the internet, usually comes with various imperfections such as blocking, changes in illumination, erratic camera movement, etc.
- (S) = Synthetic video. Things you don't see every day. Mostly just consists of very weird transitions.

name
blackmamba1 (RW)

description
POV cam on the 'Black Mamba' roller coaster (part 1). Changes in illumination.

variant	resolution	fps	length	#frames	#missing
blackmamba1_480x270	480x270	29.970	20.09	600	0
blackmamba1_960x540	960x540	29.970	20.09	600	0
blackmamba1_1920x1080	1920x1080	29.970	20.09	600	0

name
blackmamba2 (RW)

description
POV cam on the 'Black Mamba' roller coaster (part 2). Changes in illumination.

variant	resolution	fps	length	#frames	#missing
blackmamba2_480x270	480x270	29.970	20.09	600	0
blackmamba2_960x540	960x540	29.970	20.09	600	0
blackmamba2_1920x1080	1920x1080	29.970	20.09	600	0

name
blackmamba3 (RW)

description
POV cam on the 'Black Mamba' roller coaster (part 3). Changes in illumination.

variant	resolution	fps	length	#frames	#missing
blackmamba3_480x270	480x270	29.970	20.09	600	0
blackmamba3_960x540	960x540	29.970	20.09	600	0
blackmamba3_1920x1080	1920x1080	29.970	20.09	600	0

name
bus (RS)

description
A bus driving by.

variant	resolution	fps	length	#frames	#missing
BUS_176x144_7.5	176x144	7.500	5.33	38	0
BUS_176x144_15	176x144	15.000	5.13	75	0
BUS_176x144_30	176x144	30.000	5.07	150	0
BUS_352x288_7.5	352x288	7.500	5.33	38	0
BUS_352x288_15	352x288	15.000	5.13	75	0
BUS_352x288_30	352x288	30.000	5.07	150	0

name
car360 (RW)

description
A car doing a 360. Contains many missing frames due to a framerate conversion.

variant	resolution	fps	length	#frames	#missing
car360-missingframes	576x360	30.000	5.03	149	26

name
city (RS)

description
Filmed from a helicopter circling a skyscraper.

variant	resolution	fps	length	#frames	#missing
CITY_176x144_15	176x144	15.000	10.07	149	0
CITY_176x144_30	176x144	30.000	10.03	299	0
CITY_176x144_60	176x144	60.000	10.03	600	0
CITY_352x288_15	352x288	15.000	10.07	149	0
CITY_352x288_30	352x288	30.000	10.03	299	0
CITY_352x288_60	352x288	60.000	10.03	600	0
CITY_704x576_15	704x576	15.000	10.07	149	0
CITY_704x576_30	704x576	30.000	10.03	299	0
CITY_704x576_60	704x576	60.000	10.03	600	0

name
crew (RS)

description
People walking by. Contains a zoom-out and a pan, as well as camera flashes.

variant	resolution	fps	length	#frames	#missing
CREW_176x144_15	176x144	15.000	10.13	150	0
CREW_176x144_30	176x144	30.000	10.07	300	0
CREW_176x144_60	176x144	60.000	10.02	599	0
CREW_352x288_15	352x288	15.000	10.13	150	0
CREW_352x288_30	352x288	30.000	10.07	300	0
CREW_352x288_60	352x288	60.000	10.02	599	0
CREW_704x576_15	704x576	15.000	10.13	150	0
CREW_704x576_30	704x576	30.000	10.07	300	0
CREW_704x576_60	704x576	60.000	10.02	599	0

name
flower (RS)

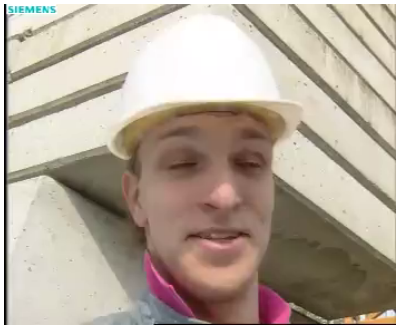
description
A moving camera through a field of flowers.

variant	resolution	fps	length	#frames	#missing
flower_176x144_15	176x144	15.000	8.6	125	0
flower_176x144_30	176x144	30.000	8.4	250	0
flower_352x288_15	352x288	15.000	8.6	125	0
flower_352x288_30	352x288	30.000	8.4	250	0

name
football (RS)

description
Short fragment of a soccer game. Very fast action.

variant	resolution	fps	length	#frames	#missing
FOOTBALL_176x144_7.5	176x144	7.500	8.93	65	0
FOOTBALL_176x144_15	176x144	15.000	8.8	130	0
FOOTBALL_176x144_30	176x144	30.000	8.73	260	0
FOOTBALL_352x288_7.5	352x288	7.500	8.93	65	0
FOOTBALL_352x288_15	352x288	15.000	8.8	130	0
FOOTBALL_352x288_30	352x288	30.000	8.73	260	0



name
foreman (RS)

description
Some guy telling something.

variant	resolution	fps	length	#frames	#missing
FOREMAN_176x144_7.5	176x144	7.500	10.27	75	0
FOREMAN_176x144_15	176x144	15.000	10.13	150	0
FOREMAN_176x144_30	176x144	30.000	10.07	300	0
FOREMAN_352x288_7.5	352x288	7.500	10.27	75	0
FOREMAN_352x288_15	352x288	15.000	10.13	150	0
FOREMAN_352x288_30	352x288	30.000	10.07	300	0

name
greatwhite1 (RW)

description
POV cam on the 'Great White' roller coaster (part 1). Shaky camera.

variant	resolution	fps	length	#frames	#missing
greatwhite1_480x270	480x270	29.970	20.09	600	0
greatwhite1_960x540	960x540	29.970	20.09	600	0
greatwhite1_1920x1080	1920x1080	29.970	20.09	600	0

name
greatwhite2 (RW)

description
POV cam on the 'Great White' roller coaster (part 2). Shaky camera.

variant	resolution	fps	length	#frames	#missing
greatwhite2_480x270	480x270	29.970	20.09	600	0
greatwhite2_960x540	960x540	29.970	20.09	600	0
greatwhite2_1920x1080	1920x1080	29.970	20.09	600	0

name
greatwhite3 (RW)

description
POV cam on the 'Great White' roller coaster (part 3). Shaky camera.

variant	resolution	fps	length	#frames	#missing
greatwhite3_480x270	480x270	29.970	20.09	600	0
greatwhite3_960x540	960x540	29.970	20.09	600	0
greatwhite3_1920x1080	1920x1080	29.970	20.09	600	0

name
greatwhite4 (RW)

description
POV cam on the 'Great White' roller coaster (part 4). Shaky camera.

variant	resolution	fps	length	#frames	#missing
greatwhite4_480x270	480x270	29.970	20.09	600	0
greatwhite4_960x540	960x540	29.970	20.09	600	0
greatwhite4_1920x1080	1920x1080	29.970	20.09	600	0

name
greatwhite5 (RW)

description
Pans and zooms.

variant	resolution	fps	length	#frames	#missing
greatwhite5_480x270	480x270	29.970	20.09	600	0
greatwhite5_960x540	960x540	29.970	20.09	600	0
greatwhite5_1920x1080	1920x1080	29.970	20.09	600	0

name
gtaiv1 (RW)

description
Short gameplay footage. Erratic camera and object movement due to modded physics.

variant	resolution	fps	length	#frames	#missing
gtaiv1_320x180	320x180	29.970	8.07	240	0
gtaiv1_640x360	640x360	29.970	8.07	240	0
gtaiv1_1280x720	1280x720	29.970	8.07	240	0

name
gtaiv2 (RW)

description
Short gameplay footage. Static camera. Interesting because game AI is often 'perfect'.

variant	resolution	fps	length	#frames	#missing
gtaiv2_320x180	320x180	29.970	4.07	120	0
gtaiv2_640x360	640x360	29.970	4.07	120	0
gtaiv2_1280x720	1280x720	29.970	4.07	120	0

name
hallmonitor (RS)

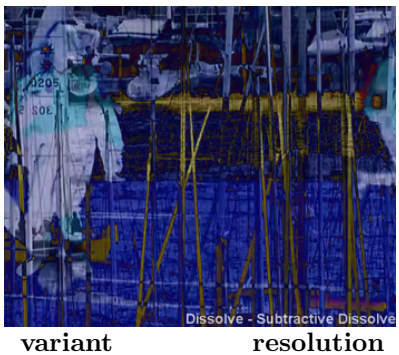
description
Static camera with people walking through an office corridor.

variant	resolution	fps	length	#frames	#missing
hallmonitor_176x144_15	176x144	15.000	10.27	150	0
hallmonitor_176x144_30	176x144	30.000	10.07	300	0
hallmonitor_352x288_15	352x288	15.000	10.27	150	0
hallmonitor_352x288_30	352x288	30.000	10.07	300	0

name
harbour (RS)

description
Static camera. Lots of edge movement due to masts, as well as a few moving boats in the background.

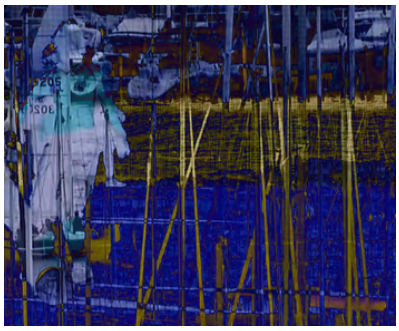
variant	resolution	fps	length	#frames	#missing
HARBOUR_176x144_15	176x144	15.000	10.13	150	0
HARBOUR_176x144_30	176x144	30.000	10.03	299	0
HARBOUR_176x144_60	176x144	60.000	10.03	600	0
HARBOUR_352x288_15	352x288	15.000	10.13	150	0
HARBOUR_352x288_30	352x288	30.000	10.07	300	0
HARBOUR_352x288_60	352x288	60.000	10.03	600	0
HARBOUR_704x576_15	704x576	15.000	10.07	149	0
HARBOUR_704x576_30	704x576	30.000	10.03	299	0
HARBOUR_704x576_60	704x576	60.000	10.03	600	0



name
mix (S)

description
A variety of scene transitions

variant	resolution	fps	length	#frames	#missing
MIX_174x144_7.5	174x144	7.500	23.6	175	0
MIX_174x144_15	174x144	15.000	23.47	350	0
MIX_174x144_30	174x144	30.000	23.4	700	0
MIX_352x288_7.5	352x288	7.500	23.6	175	0
MIX_352x288_15	352x288	15.000	23.47	350	0
MIX_352x288_30	352x288	30.000	23.4	700	0
MIX_704x576_7.5	704x576	7.500	23.6	175	0
MIX_704x576_15	704x576	15.000	23.47	350	0
MIX_704x576_30	704x576	30.000	23.4	700	0



name
mix-notext (S)

description
A variety of scene transitions, but without the transition text

variant	resolution	fps	length	#frames	#missing
MIX-NOTEXT_352x288_30	352x288	30.000	23.4	700	0
MIX-NOTEXT_704x576_30	704x576	30.000	23.4	700	0

name
mobile (RS)

description
Pan with moving background and moving foreground object.

variant	resolution	fps	length	#frames	#missing
MOBILE_176x144_7.5	176x144	7.500	10.27	75	0
MOBILE_176x144_15	176x144	15.000	10.13	150	0
MOBILE_176x144_30	176x144	30.000	10.07	300	0
MOBILE_352x288_7.5	352x288	7.500	10.27	75	0
MOBILE_352x288_15	352x288	15.000	10.13	150	0
MOBILE_352x288_30	352x288	30.000	10.07	300	0

name
motherdaughter (RS)

description
Static camera. Very few moving elements.

variant	resolution	fps	length	#frames	#missing
motherdaughter_176x144_15	176x144	15.000	10.27	150	0
motherdaughter_176x144_30	176x144	30.000	10.07	300	0
motherdaughter_352x288_15	352x288	15.000	10.27	150	0
motherdaughter_352x288_30	352x288	30.000	10.07	300	0

name
soccer (RS)

description
Pans, zoom-out, people moving from and towards the camera.

variant	resolution	fps	length	#frames	#missing
SOCCER_176x144_15	176x144	15.000	10.13	150	0
SOCCER_176x144_30	176x144	30.000	10.07	300	0
SOCCER_176x144_60	176x144	60.000	10.03	600	0
SOCCER_352x288_15	352x288	15.000	10.07	149	0
SOCCER_352x288_30	352x288	30.000	10.03	299	0
SOCCER_352x288_60	352x288	60.000	10.03	600	0
SOCCER_704x576_15	704x576	15.000	10.13	150	0
SOCCER_704x576_30	704x576	30.000	10.07	300	0
SOCCER_704x576_60	704x576	60.000	10.03	600	0

name
stefan (RS)

description
Tennis footage.

variant	resolution	fps	length	#frames	#missing
stefan_176x144_15	176x144	15.000	3.27	45	0
stefan_176x144_30	176x144	30.000	3.07	90	0
stefan_352x288_15	352x288	15.000	3.27	45	0
stefan_352x288_30	352x288	30.000	3.07	90	0

name
tempete (RS)

description
Very slow zooming out of a static scene, but with many randomly moving leaves as well.

variant	resolution	fps	length	#frames	#missing
tempete_176x144_15	176x144	15.000	8.93	130	0
tempete_176x144_30	176x144	30.000	8.73	260	0
tempete_352x288_15	352x288	15.000	8.93	130	0
tempete_352x288_30	352x288	30.000	8.73	260	0

Bibliography

[CSS12] Hyun-Woong Cho, Moon-Kyu Song, and Woo-Jin Song. Unified approach to detect shot transitions in video signals. *International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC)*, July 2012.

[DNR05] Anastasios Dimou, Olivia Nemethova, and Markus Rupp. Scene change detection for h.264 using dynamic threshold techniques. *Proceedings of 5th EURASIP Conference on Speech and Image Processing, Multimedia Communications and Service*, July 2005.

[ffma] FFmpeg mailinglist. http://ffmpeg.org/pipermail/ffmpeg-devel/2012-June/125369.html. Accessed: November 18, 2013.

[ffmb] FFmpeg sourcecode. http://git.videolan.org/?p=ffmpeg.git;a=blob;f=libavfilter/f_select.c. Accessed: November 18, 2013.

[FFR05] Jan Fransens, Fabian Di Fiore, and Frank Van Reeth. The reconstruction of missing frames in histrorical films, a layered approach. *Proceedings of GraphiCon2005, International Conference on Computer Graphics & Vision*, pages 63–69, 2005.

[HM13] Ho Hee-Meng. Digital video forensics: Detecting mpeg-2 video tampering through motion errors. *Technical Report*, May 2013.

[Lie01] Rainer Lienhart. Reliable transition detection in videos: A survey and practitioner's guide. *International Journal of Image and Graphics*, 1(3):469–486, 2001.

[Mat] Johan Mathe. shotdetect. http://johmathe.name/shotdetect.html. Accessed: November 18, 2013.

[TSG12] Manish K Thakur, Vikas Saxena, and J P Gupta. A hybrid video quality metric for analyzing quality degradation due to frame drop. *IJCSI International Journal of Computer Science Issues*, 9(1), November 2012.

[Wol09] Stephen Wolf. A no reference (nr) and reduced reference (rr) metric for detecting dropped video frames. *National Telecommunications and Information Administration (NTIA)*, 2009.

[ZMM95] Ramin Zahib, Justin Miller, and Kevin Mai. A feature-based algorithm for detecting and classifying scene breaks. *ACM Conference on Multimedia*, pages 189–200, November 1995.