Chapter 3

Techniques

This chapter starts off by explaining how the idea for a new video fingerprinting technique came to life and why the techniques that are described in this chapter are used. It then goes on to discuss these techniques in much more detail. The techniques that will be discussed are Scale Invariant Feature Transform (SIFT), Speeded Up Robust Features (SURF), Shazam and Key Frame Detectors (KFDs).

3.1 Introduction

After all the existing video fingerprinting techniques were researched, it was clear that none of the existing techniques could be used for the main objective of this dissertation, namely broadcast monitoring. This meant that a new technique had to be developed that can detect advertisements in a television broadcast in real-time. It was decided that a frame fingerprinting technique was to be developed, instead of a sequence fingerprinting technique, because it looked most promising for the application as explained in Subsection 2.5.7.

Looking at other CBIR techniques, Shazam stood out, as it was quite famous for its very fast audio detection capabilities. Shazam is an audio fingerprinting technique that was patented by Shazam Entertainment, Ltd. and therefore the algorithm isn't available to the public. In [4], Avery Wang describes how the technique works and in [19], Dan Ellis implements his own version in Matlab, based on Wang's description. Throughout this whole dissertation **Shazam** will be used to refer to the algorithm explained by Avery Wang and Dan Ellis.

Shazam makes use of hashes, created from pairs of key points found in the song's spectrogram, to quickly match an unknown audio file to a very large song fingerprint database [4]. The way Shazam uses key points to create hashes that can easily be saved and retrieved from a database to find fingerprint matches quickly is really effective and interesting.

The thought came to mind to create a video fingerprinting technique based on the same basic principles as the Shazam audio fingerprinting technique. With this idea in mind while researching image processing techniques, SIFT looked very promising, because it detects robust, scale and rotation invariant key points in an image. SIFT was originally designed for object recognition by David Lowe [1]. Each SIFT key point consists of a x and y coordinate, a magnitude and a direction. The idea was to use the key points detected by the SIFT algorithm and create hashes from pairs of key points to effectively create a fingerprint for a frame. By using this hashing method the same kind of detection speeds as the Shazam algorithm can be obtained with the video fingerprinting system, as the hashes make it easy to save and retrieve data at specific points in the database.

Therefore SIFT and SURF (alternate key point detector) are described in this chapter, along with the audio fingerprinting technique explained in Wang's paper. At the end of the chapter KFDs are discussed. A KFD is very important for a video fingerprinting system that makes use of frame fingerprinting, because it can select key frames in a video that represent the different shots or scenes. The key frames can then be fingerprinted and added to the database, instead of fingerprinting and adding every single frame of the video to the database. If a KFD is not used, too many frame fingerprints will be generated for videos and the database will get saturated extremely fast.

3.2 Scale Invariant Feature Transform (SIFT)

SIFT is an image processing technique that was developed by David Lowe for object recognition [1] [2] [20]. The algorithm finds key points in an image that are invariant to scale, rotation and noise, based on the local features of the image. Once the key points are calculated, a normalized descriptor is calculated for each key point. A list of descriptors are used to characterize or fingerprint an object.

For image or object detection, the descriptors of reference images are added to a database. To detect a new image, its descriptors are compared/matched to the database of descriptors, using the Euclidean distance as a difference measurement. If a certain number of descriptors are successfully matched, the image or object is detected.

The fact that it is scale and rotation invariant makes it robust to common differences in frames and thus a great candidate to use in video fingerprinting. Another important thing is that key points can quickly be calculated for a frame.

3.2.1 Algorithm

In [1], SIFT is divided into four major computational stages:

- Scale-space extrema detection
- Keypoint localization
- Orientation assignment
- Keypoint descriptor

In the **scale-space extrema detection** stage, a Gaussian pyramid is created from the image. To create a Gaussian pyramid, the original image is used as the base and then pyramid levels are added using two steps: smoothing and down sampling. Gaussian blurring is used to smooth the image (filter size halves each level) and then the image is

Figure 3.1: Example of a Gaussian pyramid.

down sampled to half its size (see Figure 3.1). A scale space is created for each octave of the pyramid through increased blurring of the image at that pyramid level, as shown on the left side of Figure 3.2. The DoG for the entire scale space is then calculated by subtracting adjacent image scales from each other, as shown in Figure 3.2. The equations used for this stage are:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{3.1}$$

where $L(x, y, \sigma)$ is a level in the scale space, $G(x, y, \sigma)$ the variable-scale Gaussian kernel and $I(x, y)$ the input image. It means that a level in the scale space is equal to the first image in the scale space convolved with a Gaussian blurring filter with a certain σ value.

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma) \tag{3.2}$$

where $D(x, y, \sigma)$ is the DoG scale space and k the constant factor between two close scales. This equation is used to calculate a level in the DoG scale space by subtracting two adjacent scale space levels of the original image.

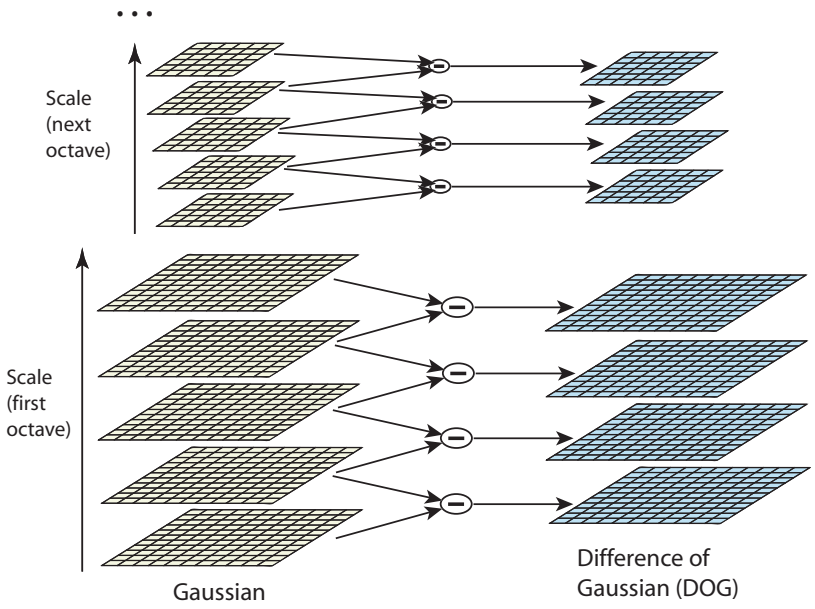


Figure 3.2: A set of scale space images is created for each level of the Gaussian pyramid of the image as shown on the left. The DoG is then calculated by subtracting adjacent scale space images from each other [1].

After the DoG scale space has been built, it is then used to do **keypoint localization**. Each point in the DoG scale space is compared to its 8 neighbours on the same level and the 18 neighbours in the adjacent levels (9 from previous level and 9 from next level) as depicted in Figure 3.3. A key point is found if the point, marked with a X in the figure, is smaller or larger than the neighbouring points, effectively indicating a minimum or maximum in the DoG scale space. Key points that are located near the edges of the image or those that have low contrast are rejected. The remaining key points' positions are refined further by determining the interpolated position of the maximum/minimum.

For the **orientation assignment** of the key points the gradient magnitude, $m(x, y)$, and orientation, $\theta(x, y)$, are calculated for every point of every scale.

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2} \tag{3.3}$$

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y))) \tag{3.4}$$

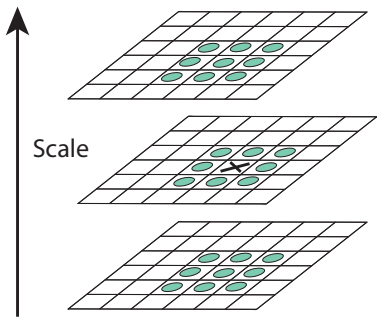


Figure 3.3: A peak is found if the DoG point's value is bigger or smaller than all its neighbouring points [1].

To determine a key point **orientation**, a 36 bin orientation histogram is computed (each bin represents 10°), using the orientation values, weighted by the magnitudes and a Gaussian-weighted circular window, of points in a region around the key point. The highest peak within the histogram is detected as the key point's orientation. If there are other peaks within the histogram that exceed 80% of the main peak's height (±15% of the time), they are also used to create key points with, resulting in multiple key points with the same position and size, but different orientations.

In Figure 3.4, an image is displayed along with its SIFT key points on the left-hand side and a rotated and resized version along with its SIFT key points is displayed on the right. Even with the rotation and size difference, most of the key points correspond in the two images.

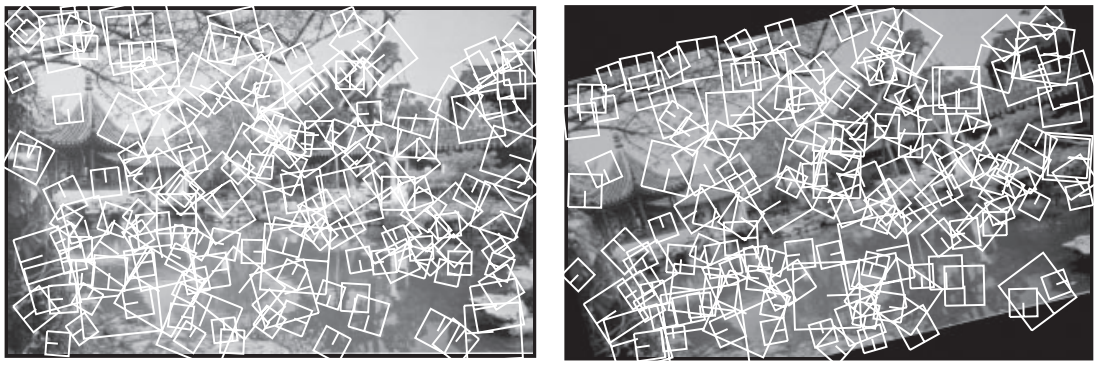


Figure 3.4: Images with SIFT key points displayed on them [2].

Keypoint descriptors are then calculated. This part of the SIFT algorithm will not be discussed, as it is not relevant to this dissertation. For an in depth explanation of the SIFT algorithm, refer to [1].

3.2.2 Speeded Up Robust Features (SURF)

SURF is also a technique that is used for object recognition, like SIFT, and was proposed by Herbert Bay in [3] and [21]. It also finds key points in an image and for this reason it is also mentioned in this dissertation. In Chapter 5 of the dissertation, the effectiveness of the use of SIFT key points will be tested against the use of SURF key points.

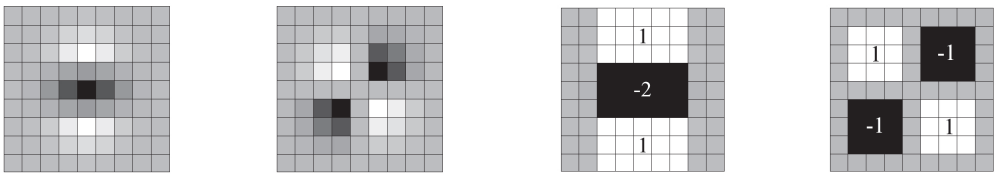


Figure 3.5: Second-order Gaussian partial derivatives on the left and the box filters used by SURF on the right [3].

SURF is also scale and rotation invariant, but was developed with the focus being on reduced processing, while still reliably detecting objects and key points in images. SURF detects its key point in a slightly different way than SIFT. While SIFT uses a image pyramid, filtering each level with Gaussian filters of increasing sigma values and calculating the DoGs, SURF keeps the original image at the same size and uses the idea of integral images and box filters of different sizes to filter the image and quickly detect key points, as illustrated in Figure 3.6. The box filters are approximations of second-order Gaussian partial derivatives and examples can be seen in Figure 3.5.

After the key points' positions are calculated, their orientations are calculated, thus the SURF key points also consist of four values: x position, y position, size and orientation.

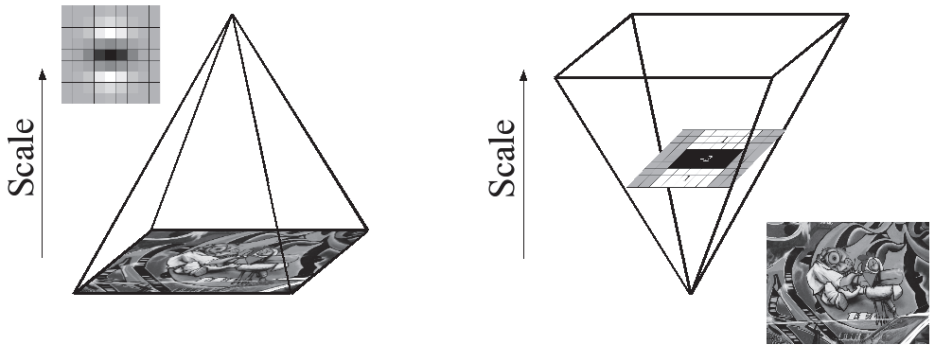


Figure 3.6: SURF keeps the image a constant size and changes the filter's size (right) as opposed to changing the image's size and keeping the filter size constant (left) [3].

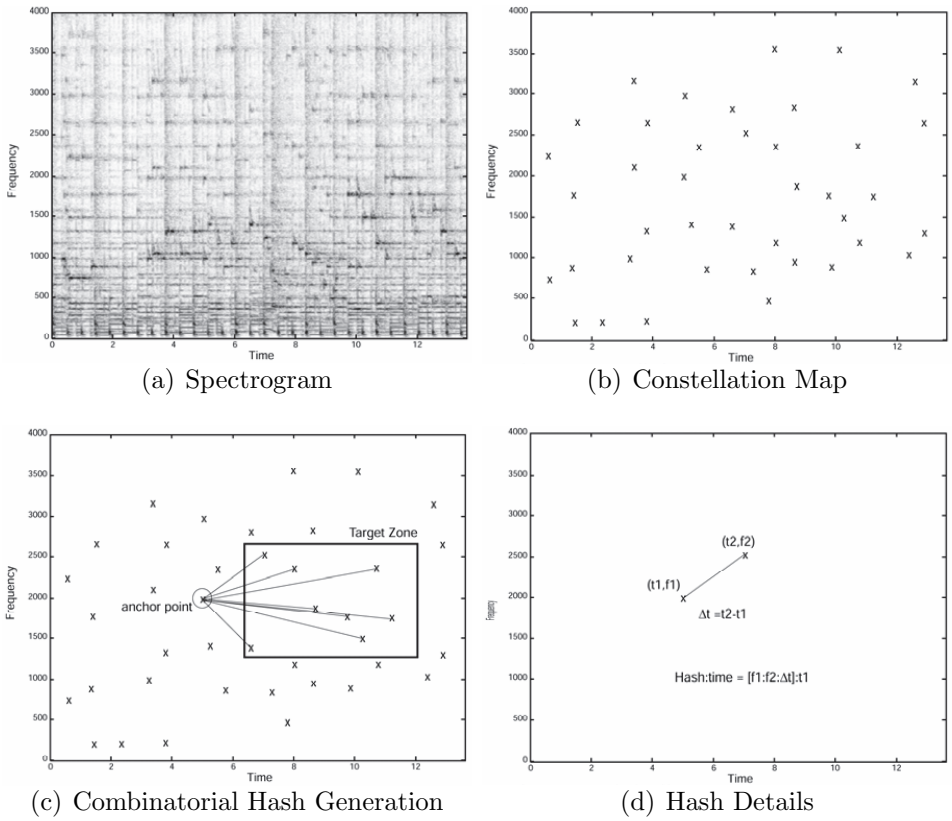
This makes it easy to exchange the use of SIFT and SURF key points within the novel fingerprinting system, as seen later in the dissertation.

SURF calculates a descriptor for each key point, but this will not be discussed in this dissertation, as it is not of importance.

3.3 Shazam

In [4], Avery Wang discusses an audio fingerprinting system that was developed for the Shazam company. The technique calculates the spectrogram of a given audio signal, after which key points are extracted from the spectrogram. These key points are peak locations in the time-frequency data with high amplitude values. In Figure 3.7(a) and Figure 3.7(b) a spectrogram and its resulting set of key points are shown, respectively. This sparse set of time-frequency coordinates is called a "**constellation map**". One of the algorithm's greatest strengths is the fact that the key points are very robust and easily repeatable, because the high amplitude time-frequency peaks of a song is detected even if there are other sounds present.

Fast combinatorial hashing is then implemented by pairing key points within a certain time and frequency range (see Figure 3.7(c)) and then creating a 32 bit hash containing



(a) Spectrogram (b) Constellation Map

(c) Combinatorial Hash Generation (d) Hash Details

Figure 3.7: Shazam algorithm diagrams [4].

the first point's frequency, the second point's frequency and the time difference between the points, as shown if Figure 3.7(d). A 32 bit value is also added for the time offset (first point's time) and the audio track's identification number.

To match unknown audio tracks to the database, the same algorithm is applied to extract the key points and create the list of hashes. For every single hash created in the unknown audio track, the corresponding hashes are extracted from the database. A histogram for each trackID is then created from the values of the difference between the database and unknown audio track's hash time offsets. If the histogram has a significant peak, at a certain time offset, a match is found to the song in the database.

3.4 Key Frame Detectors (KFDs)

A KFD is an algorithm that extracts a set of frames from a video that represents the visual content of that video. It is done by splitting the video into its basic components, namely shots, and selecting a representative frame (or frames) for each shot. A shot is a consecutive set of frames that was filmed with a single camera, therefore the frames are interrelated and contain very similar visual content. The algorithm used to detect the shots in a video is called a Shot Boundary Detector (SBD). Thus, a KFD is basically a SBD with one extra step in the process: selecting key frames from the shots.

This subject has been researched quite a bit and a few techniques have been proposed to solve the problem. In [22], Rainer Lienhart compares some SBDs and it is shown that most of the systems detect hard cuts and fades rather reliably, but fail when they have to detect dissolves. A hard cut is an instant transition between two shots, while a dissolve is used to gradually transition between shots. A fade is a gradual transition of a shot to or from a blank image (usually black).

To detect the different transitions used in videos, a Frame Difference Measurement (FDM) is used. A FDM is a measurement of the difference between a characteristic of two frame images. It can be used on adjacent frames, or it can be applied between a shot start frame and the current frame. A few characteristics used by FDMs to measure the difference between frames are:

- Colour Histogram [23] [24]
- Edge Change Ratio [25]
- Pixel Intensity [26]
- Average Intensity [27]
- Intensity Histogram
- Jensen Shannon Divergence (JSD) [28]

- SIFT [29]

A few other existing techniques include the Ohta Detector [30] and a SBD based on wavelets and Support Vector Machines [31].

Once the differences between the frames have been calculated using a FDM, the information is used to determine shot boundaries. Detection normally happens when a difference measurement exceeds a certain threshold. This threshold can be a set value, or it can change dynamically based on a surrounding difference values.

The most important characteristic of a SBD and KFD is repeatability. If the same key frames or shots can be detected in similar video streams constantly, even if distortions are present, the KFD or SBD can be considered a success.

3.4.1 Jensen Shannon Divergence (JSD)

Looking at the application for this dissertation, namely real-time broadcast monitoring, a KFD is needed that can process and detect key frames in real-time. As the system is using SIFT to create the frame fingerprints, it would also be a good candidate to use in the key frame detection process, as used by Gentao Liu et al. in [29], but that amount of SIFT key points can't be detected in real-time, so another FDM has to be used. A promising FDM is JSD and this measurement can be taken in real time and then used to detect key frames in the video stream.

In 1948, Shannon wrote a paper containing a formula for determining the entropy of a probability distribution [32]. This entropy came to be known as the Shannon entropy and is shown in the following equation:

$$H(P) = -K \sum_{i=1}^{n} p_i log_b p_i \tag{3.5}$$

where $H(P)$ is the Shannon entropy of the the probability distribution, $P = (p_1, p_2, p_3, ..., p_n)$,

and n the number of possibilities in the probability distribution. For the equation, $K > 0$ and $b > 1$.

By combining the Jensen inequality with Shannon entropy, the Jensen-Shannon inequality can be derived [33] and in [28], Qing Xu derives the JSD equation:

$$JSD(f_{i-1}, f_i) = H(\frac{P_{f_{i-1}} + P_{f_i}}{2}) - \frac{H(P_{f_{i-1}}) + H(P_{f_i})}{2} \tag{3.6}$$

where f_i is the current frame and f_{i-1} the previous frame, P_{f_i} and $P_{f_{i-1}}$ the respective probability distributions of the frames' gray scale intensities and $H(\bullet)$ the Shannon entropy function. One can also calculate 3 different JSD values for the red, blue and green channels in colour frames and use the average as the FDM.