Feature Extraction for Very Low Bit Rate Video Coding

P.R. Meulemans, Dr. R. Wilson
Department of Computer Science,
University of Warwick,
Coventry

August 31, 1995

Abstract

The work described in this report is part of a project aimed at the design of an image sequence (video) coding method for very low bit-rates. The coding method will use multiresolution affine motion estimation combined with extracted image features to make accurate inter-frame predictions.

This report is mainly concerned with extracting linear features (i.e lines and edges) from an image and combining them into more complex features that represent object boundaries in the image. The linear features are extracted in a multiresolution scheme from local frequency estimates provided by the Multiresolution Fourier Transform (MFT).

Contents

1 Introduction

The use of video images in many areas of everyday life is very much increasing at the moment and will probably continue to do so in the foreseeable future. With the rise of multimedia, people are starting to see the value of adding moving images to areas of communication that were traditionally done just in print or by voice. In communications that until recently could only be done face-to-face (i.e. meetings), video links can now be used to conduct these meetings long-distance, thus saving travelling time and costs.

The advantages of video images are clear, but so is their main problem: the huge amount of information necessary to transmit or store the video images. For instance, transmitting a grey level 8 bit/pixel image of size 256x256 pixels at 25 frames per second takes over 13 million bits/sec. If this image was for instance used for a video-phone than the actual bit-rate available would be 64 thousand bits/sec (on an ISDN-line), which is less than one two-hundredth of the needed rate. Clearly there is a need for compression of the information.

Although modern technology is capable of sending information over at vast bit-rates, these bit-rates are usually not available to the users for a number of reasons. Firstly, there is the simple rule: the higher capacity a lines has, the more it costs to rent or buy. So compression to low bit-rates will reduce costs. Secondly, the latest technology is not always available to everyone. A lot of non-commercial communications will at the moment be done over existing telephone lines, using a modem that can typically reach 28800 bits/sec. Thirdly, there is a trend towards wireless communication which also has highly restricted bit-rates.

Most video compression methods used to date operate on the principle of making inter-frame predictions using block matching motion estimates and coding the prediction errors using a DCT. These methods were originally developed for higher bit-rates and simply applying them to bit-rates under 64 kbit/sec does not give the best possible results. Two main streams of coding methods have emerged that are specially aimed at these lower bit-rates. The distinction between the two is that one is 'model-based' and the other one isn't. A model-based method extracts information about the contents of the image and uses this in the coding. Non model-based methods do not analyse the image in this way. Gains in coding rates in these methods over the conventional ones are obtained by for instance using better transformations or quantisation schemes, and reducing overhead information.

Model-based methods are in general more complex then non model-based methods, but they also are thought to have a potential for higher compression. This is especially true when judging the result subjectively, because model-based methods can 'shape' their errors in an observer-friendly way since they have information on the contents of the image. This potential for better compression has however not yet been converted into an actual method that clearly outperforms the non model-based methods.

This report describes the start of a project which is aimed at creating a new model-based video coding method. As in conventional video coding methods, this one is based on making a prediction of the next image in the sequence from the previous ones and coding the hopefully small prediction error. The main differences from conventional methods are:

- Use of affine motion parameters instead of just translation motion vectors. This improves modelling 3-D motion of objects in the image.
- Use of linear features extracted from the image. These features (lines and edges) are used to enhance the motion estimates and to identify parts of the image that belong to the same object and therefore move together.

Both above aspects are dealt with within a multiresolution context, which means that the image is segmented into blocks of variable size, according to its local contents, so that large structures can be dealt with at a large and efficient scale and finer details in an appropriate finer scale. The calculation of both the affine parameters and the linear features is done using the Multiresolution Fourier Transform(MFT), which provides local frequency spectra at multiple scales.

The main topic of the research described in this report is how the linear features that will be used in the proposed compression method can be extracted from the images in the sequence and how the separate extracted features can be joined together to form a useful set of boundary representations.

1.1 Video Coding

Conventional video coding methods based on block transform coding using temporal prediction and motion estimation (e.g. MPEG, H.261 [12] and H.263 [22]) seem to be reaching a plateau in performance. The low-level processing used by these methods, such as transforms, filtering and correlations, is by now well understood and major leaps in performance are therefore not to be expected. Still, improvement can be achieved by using for instance different transformation schemes, like subband coding [14][20]. In order to get major improvements in video coding, a lot of research therefore focused on compression methods that use a higher level of processing, which deals with higher level concepts such us global motions, surfaces, regions, boundaries, textures, etc. These concepts can be referred to as mid-level concepts [1] because they are sophisticated enough to produce powerful representations, and yet simple enough to be computed.

This higher level of processing can come in many different forms. At one side of the range there are relatively simple methods like applying motion vectors to regions instead of blocks, where the regions are formed on the basis of quantised grey level values [9]. On the other side of the range there are knowledge based model methods which recognise highly specific objects, e.g. a head-and-shoulder model, that can be represented almost totally by parameters [6][13]. These highly specific coding methods are likely to get the best performance, but are only useful in highly restricted applications. In methods for general video compression a balance has to be found between the generality and the performance of the method.

A very basic aspect of the contents of almost all images is that it is a 3-D scene that is projected on to the 2-D image. This knowledge can be used in compression schemes in several ways. One consequence is that the motion in the image is actually a projection of the 3-D motion in the scene. So unless an object in the scene moves parallel to the image plane, its motion in the image will not be a 2-D translation. The 2-D motion estimation used in conventional video coding can therefore be improved upon by using more appropriate motion parameters. In [17] three types of 2-D transformations are used that model the projection of the 3-D motion: affine, perspective and bilinear transformation. It is shown that these motion estimations result in better compression ratios than just translation motion estimation. The parameters of the transformation are in that work found by block matching, which presents a problem because the transformations do not have just two degrees of freedom but six (affine) or nine (perspective). This increases the search space and therefore the computational load dramatically.

Adelson and Wang in [1][21] use the notions of regions and 3-D depth in their work by representing moving scenes with layers. The scene that is shown in the sequence is segmented into layers: one background layer and a number of occluding layers, ordered by depth. As the sequence goes along, the layers are segmented by analysis of affine motion and the image intensities of the layers are filled in. The image is reconstructed by warping the filled in layers according to the affine motion parameters.

One of the advantages of this method is its ability to deal with occlusions, which is an important aspect of video coding. Especially, the way that reappearing parts of the image that were temporarily occluded can be synthesised by virtue of the memory function of the layer representation is a significant gain. A problem for this coding scheme could be how to deal with 3-D rotating motion, because layers are inherently flat, or with local small scale movements which are to small to be represented with layers.

The coder being developed in this project will combine several ideas used in model based coding. The aim is to produce a coder that is generally applicable without knowledge of the contents of the image, yet approaches the performance of a 3-D model knowledge based coder. A multiresolution technique will be used to estimate affine motion parameters for 'planar patches' at an appropriate scale. A '$2\frac{1}{2}$-D' image model can then be used in which surfaces are treated as assemblies of planar patches moving in 3-D.

The affine motion estimation of the planar patches is implemented using a scheme consisting of two components:

1. A block based affine motion estimation in which a block in this frame is modelled as an affine transform of a block in the previous frame.

2. A linear feature extraction in which piecewise linear parts of object boundaries in the image are found.

Both these aspects are done within a multiresolution framework to ensure that an appropriate scale is used in all parts of the image. To synthesise the predicted frame from the previous frame, the estimated motion parameters are applied to the planar patches whose boundaries are constrained by both the blocks and the linear features.

The use of the linear features has several more advantages. One of these is that their position is established again for each frame. When inter-frame predictions are made based on only motion estimates, parts of the image often have the tendency to 'drift off' because of accumulating motion errors. Using the accurate position of the linear features, this effect can be reduced. Another advantage is that the features indicate likely occlusion boundaries. This knowledge can be used to improve predictions for newly unoccluded parts of the image.

It has been mentioned that finding affine motion parameters can be a time consuming job when done by block matching. Therefore, in this work the motion parameters are calculated in a more efficient way within the framework of the MFT. The estimation is based on a method developed by Hsu [10] in the context of texture analysis and synthesis.

1.2 Feature Extraction

One of the novelties of the proposed video compression method is that it makes use of linear features. So the question arises of how best to extract these features from the images in the sequence.

The simplest kind of line and edge detectors are those that generate a map of the image which indicates whether the corresponding pixel in the image is on an edge and possibly what the strength and orientation of that edge is. This category includes: line and edge masks, gradient operators, and detectors of zero crossings of second order derivatives of the image [11].

The low-level representation of edges that these methods produce is hard to use directly in a coding application. In order to use feature information in the proposed coding scheme, it needs to be represented in a higher-level form. A suitable form of representation would be a set of feature types of interest, whose exact form (e.g. position and orientation) can be represented by a number of parameters. The extraction methods would then produce a list of all features present in the image, with a set of parameters for each feature.

One well known method for doing this is the Hough Transform.

1.2.1 Hough Transform

The Hough transform [8] is used to go from a low-level representation of edges to a higher level parametric representation of features of a chosen type. The input of the Hough transform is the edge map produced by applying one of the above mentioned edge detection methods to the image. All of the high intensity pixels in this map lie on (potential) features. The idea behind the Hough transform is that for each combination of parameter values it gives an indication of how well the corresponding feature is supported by the input edge map.

The Hough transform is a transformation from the spatial domain to a parameter domain, whose parameters correspond to the chosen feature type. To demonstrate how the Hough transform works, consider the extraction of linear features.

To represent a line in an x, y-plane, two parameters are needed. Several possible sets of parameters can be used, for instance the ordered pair (a, b) which represents the line given by:

$$y = ax + b. \tag{1}$$

The problem with this representation is that the values of a and b are not restricted to a finite interval. Therefore the Normal representation of a line is used, which has parameters (θ, p) and represents the line:

$$p = x \cos\theta + y \sin\theta. \tag{2}$$

So to extract linear features from the edge map, the map is transformed from the spatial (x, y) domain to the parameter (θ, p) domain.

Consider a point (x_i, y_i) on the edge map. Through this point, a infinite number of lines can be drawn, with varying parameters θ and p. Therefore this point (x_i, y_i) transforms to an infinite number of points in the parameter space, which together form a curve in that space. Transforming a second point (x_j, y_j) will produce a different curve in the parameter space. The coordinate in the parameter space where these two curves cross, indicates the parameter set for the straight line in the spatial domain that both points lie on. Similarly, all points of the edge map whose value exceeds a given threshold can be transformed, resulting in a parameter space filled with curves. If there is a coordinate in this space through which a lot of these curves pass then this indicates that a lot of pixels in the edge map lie on the line that corresponds to the parameter values of this coordinate, which can therefore be considered a feature in the image.

In practice, finding these points in the parameter space can be done by discretising the parameter space. Each discrete coordinate than represents a small range of parameter values, and the transformation of a point on the edge map increases an accumulative counter of all appropriate discrete coordinates. A linear feature in the image can then be found by looking for peaks in these counter values.

1.2.2 Frequency spectrum method

A different way in which parameterised features can be extracted from an image is by analysing its frequency spectrum. Calway [5] and Davies [7] have used a method of extracting linear features directly from the Fourier spectrum of an image.

Given that an image contains a single linear feature, certain properties of the spectrum are known: 1) the magnitude spectrum will have a preferred orientation, perpendicular to the orientation of the linear feature; 2) The phase of the spectrum will be linear. By checking the spectrum of an image for these two properties, it can be established whether the image has a single linear feature. Also, the orientation and position of the feature can be obtained from the spectrum.

Davies extended the method to detecting multiple linear and arced features.

The feature extraction method that will be used for the proposed video compression scheme is based on this frequency domain method. It will be extensively described in a later section.

1.2.3 Hierarchical feature extraction

Using one of the methods described above, single linear features or other simple geometric features can be extracted from an image. However, most images contain far more than one feature and of more complex forms. It would not be possible to extract all these features in a global manner using the described methods. To overcome this, a 'divide and conquer' strategy can be adopted, dividing the image into smaller parts to each of which the feature extracting method can be applied.

The simplest way to divide the image is to use fixed size square blocks. However, this presents the problem of choosing one block size. The blocks should be small enough that they will not contain more features than the feature extraction method can detect. On the other hand, making the block size too small results in a very inefficient representation of larger features, which will be divided over a large number of small blocks, or even in missing large features that are not sharp enough to be detected in a small window.

It is clear that the optimal block size for an image depends on the contents of that image, and will usually be different for different parts of the image. It is therefore an improvement to use different sized blocks in different parts of the image. This can be done using a *hierarchical method*. Hierarchical methods usually use a quad-tree-like representation of the image. The process starts out with a big block size and decides for each block in the image either to keep this big block, or split it up into smaller blocks. This decision process is then repeated for all smaller blocks, until the optimal block size for each part of the image is reached. Alternatively, a bottom up approach can be used which starts out with all small blocks that can be merged into bigger ones.

The clear advantage of hierarchical methods is that they adapt to the image contents so that each feature can be detected and represented using the best window size or *scale*. This is especially important in compression applications, because representing a feature at a larger scale takes fewer (but bigger) blocks and therefore fewer feature parameters.

To extract the feature parameters of the local features in each block, and to decide whether or not the block needs to be split again, the non-hierarchical feature extraction methods described in the previous sections can be used. Complex features, such as boundaries of objects in the image can be represented by combining the simple features in the separate blocks.

1.3 Image Representations

The feature extraction in this work will be done in the frequency domain. One of the reasons for this is that in the video coding process the feature extraction needs to be combined with other aspects like the affine motion estimation and texture analysis. All these aspects are preferably dealt with within one unified framework and it has been established [5] that the Multiresolution Fourier Transform (MFT) is a suitable frequency domain image representation to act as a basis for this framework. This section will discuss some much used frequency domain image representations, and compare them to the MFT.

The standard frequency domain representation is the two-dimensional Fourier Transform (FT) [11]. The FT has good properties for extracting feature and affine motion parameters. Orientation information is obtainable from the magnitude and position information from the phase of the spectrum. Furthermore, the FT basis vectors are closed under affine transformation. Consequently, a small affine transformation in the image results in a small predictable change in the frequency coefficients. The major problem of the FT is its inability to deal with locality. So if there are several features at different areas of the image, it is hard to distinguish them in the spectrum.

To avoid this problem of locality, *space-frequency* representations are used. The best known of these is the Short Time Fourier Transform (STFT) [2]. The STFT applies a window to the image to ensure locality, before transforming the image. By shifting the window over the image, a set of local spectra is obtained. In the STFT, the size of the window is fixed, so one appropriate size has to be chosen for the entire image. Therefore, when using a STFT for feature detecting, all features will be analysed at one scale. As already mentioned in the previous section, it is an advantage to analyse each region of the image at its own appropriate scale, therefore an image representation is needed that supports multiple scales.

Representations that do support multiple scale analysis are called *multiscale* or *multiresolution* representations. For frequency domain representations the choice of scale, i.e. the choice of the size of the analysis window, is a trade-off between frequency and spatial resolution. Due to Heisenberg's uncertainty principle, which gives an upper bound for the product of the spatial and frequency resolutions, it is not possible to get simultaneous arbitrarily high spatial and frequency resolutions [24].

One popular multiresolution representation is the Wavelet Transform (WT) [16][19]. The WT is a *space-scale* representation which means that each coefficient of the WT is identified by a position and a scale. The WT decomposes the signal onto a set of analysis functions that are all obtained from one basis function by a translation (according to the position) and a dilation (according to the scale). The frequency associated to a WT coefficient is inversely related to its scale, because the dilation of the analysis function of a large scale will reduce its frequency. In terms of the trade-off between frequency and spatial resolution this means that at low frequencies the WT has a high frequency and low spatial resolution, and at high frequencies it has a low frequency and high spatial resolution.

The WT has the advantage over the STFT that it does represent the image at a range of scales, but it also has some disadvantages compared to the STFT when used for feature extraction. The STFT inherits the pleasant properties of the FT for feature extraction that have been mentioned earlier. The WT lacks some of these properties, e.g. the useful phase information to determine position and a 'nice' reaction to small translations in the image [18]. Another reason why the WT is not very useful for feature extraction is that its notion of scale is quite limited. For each frequency, only one fixed scale is available which means that there is no freedom to choose the optimal scale or to use several scales in the detection process. For these reasons, the WT is not used.

The image representation that is used in this work is called the Multiresolution Fourier Transform (MFT). This representation satisfies all of the above indicated problems. The discrete MFT can be seen as a stack of STFT's of which each level uses a different window size. It therefore retains all the good properties of the STFT and provides different scales through the levels. Because the MFT is an over-complete representation, it provides all scales for all positions and frequencies.

2 The Multiresolution Fourier Transform

The MFT [5] is a linear transform, that provides spatially localised frequency spectra of a signal, over a range of scales. A coefficient of the continuous MFT is identified by three parameters: the spatial position $\vec{\xi}$, the frequency $\vec{\omega}$ and the scale parameter σ. For a signal x the MFT $\hat{x}$ is defined by [7]:

$$\hat{x}(\vec{\xi}, \vec{\omega}, \sigma) = \sigma^{1/2} \int_{-\infty}^{+\infty} w(\sigma(\vec{\chi} - \vec{\xi}))x(\vec{\chi}) \exp[-j\vec{\chi} \cdot \vec{\omega}]d\vec{\chi}, \tag{3}$$

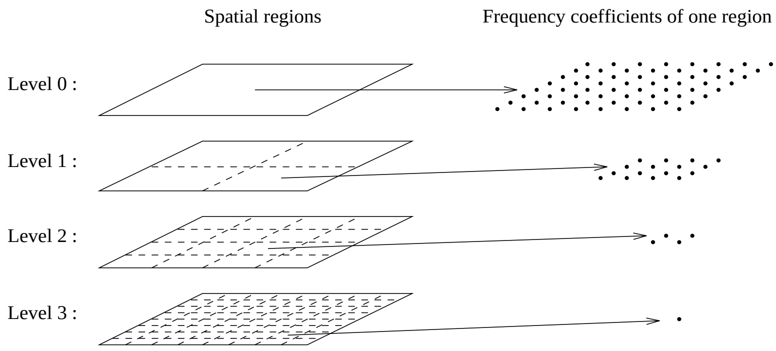


Figure 1: *MFT structure of an 8×8 image.*

where $j = \sqrt{-1}$, $w(\cdot)$ is a window function and '$\cdot$' denotes the inner product. It can be seen that for a fixed scale σ this definition is identical to that of a STFT, where the window size is determined by σ. By varying σ the coefficients of various scales of STFT can be obtained, thus changing the frequency/spatial resolution trade-off.

In this work, a 2-dimensional discrete version of the MFT is used, which is applied to a discrete image. The discrete MFT consists of a number of levels, each of which provides local spectra for a number of spatial regions of the image, as shown in figure 1. At the lowest level (level 0), there is only one region (the whole image) and it is therefore comparable to the DFT of the image. At each level up, the number of regions is doubled in each dimension, thus increasing the spatial resolution. However, the number of frequency coefficients is halved in each dimension, which results in a reduced frequency resolution. The highest level, at which each pixel of the image is an individual region, is simply the spatial representation of the image.

In its simplest form, the local spectrum of an image region is obtained by taking the 2-dimensional DFT of that region. This corresponds to using a spatially limited rectangular window of the size of the region. In this case, at any given level l, the number of spatial regions in one dimension, S_l, and the number of frequency coefficients per region in one dimension, N_l, satisfy the equation:

$$S_l \times N_l = M, \tag{4}$$

where $M \times M$ is the size of the image. It can be shown that this is the minimal requirement for the product of S_l and N_l in order for any given level of the MFT to fully represent the image.

However, for the purpose of image coding, this simplest form of the MFT is not very suitable, for the use of non-overlapping image regions will lead to a 'blocky' reconstructed image. Therefore, in this work a 'relaxed' MFT [7], with overlapping windows, is used. In this case, each MFT-level has two times as many coefficients in each direction as there are in the original image:

$$S_l \times N_l = 2M, \tag{5}$$

The windows used are still spatially limited, but now have a 50% overlap on each of the neighbouring windows, as shown in figure 2. The local spectrum of a region is now obtained by performing a DFT on the window after weighting the coefficients of the window with the window function. For efficiency reasons, the origin of the DFT is chosen to be on one of the four centre coefficients of the region, rather than in between them as done by Calway and Davies. The use of a spatially limited window function instead of a frequency limited and therefore spatially infinite window function is one of the main differences between this work and previous work by Calway and Davies.

The image regions at a given level l are indexed from $(0,0)$ to (S_l-1, S_l-1). Let $\Lambda_{l,\vec{\lambda}}$ be the region at level l with index $\vec{\lambda} = (\lambda_1, \lambda_2)$. The distance between the centre positions of two consecutive regions, or in other words, the spatial sampling interval at level l is given by:

$$\Xi_l = \frac{M}{2^l}. \tag{6}$$

The position of the lower-right pixel of the four centre pixels of region $\Lambda_{l,\vec{\lambda}}$ is given by:

$$\vec{\xi}_{l,\vec{\lambda}} = (\xi_{l,\lambda_1}, \xi_{l,\lambda_2}) = (\Xi_l(\frac{1}{2}+\lambda_1), \Xi_l(\frac{1}{2}+\lambda_2)). \tag{7}$$

The local spectrum of region $\Lambda_{l,\vec{\lambda}}$ can now be defined by:

$$\hat{x}(l,\vec{\lambda},k_1,k_2) = \sum_{n_1}\sum_{n_2} w_l(n1,n2)x(\xi_{l,\lambda_1}+n_1, \xi_{l,\lambda_2}+n_2)\exp[-j\frac{2\pi}{N_l}(n_1k_1+n_2k_2)], \tag{8}$$

where

$$-\frac{N_l}{2} \leq k_1, k_2 < \frac{N_l}{2} \tag{9}$$

are the discrete frequency coefficients. The window used $w_l(\cdot)$ is a Cartesian separable cosine window centred on one of the centre coefficients of the region:

$$w_l(n_1,n_2) = \begin{cases} \cos(n_1)\cos(n_2) & \text{, if } -\frac{N_l}{2} \leq n_1, n_2 < \frac{N_l}{2} \\ 0 & \text{, otherwise} \end{cases} \tag{10}$$

The discrete MFT is very well suited for the hierarchical feature detection process because its structure of levels with doubling block size supports the quad-tree structure of that process. Each image region can be checked on features at a certain scale by analysing the local spectrum from the appropriate level and spatial position.

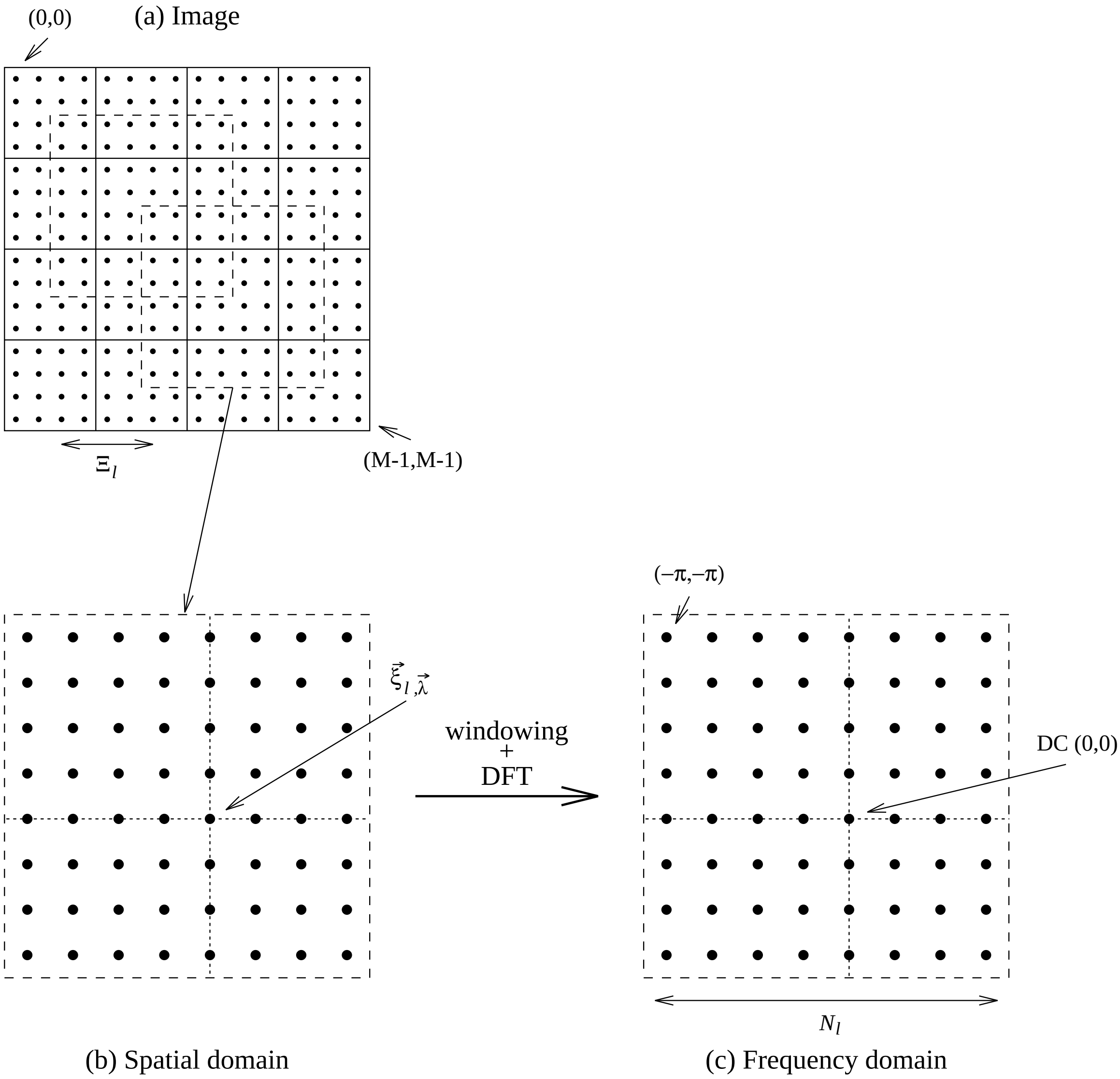


Figure 2: *(a) The regions of a* 16×16 *image at level* $l = 2$*. The overlapping windows of two of the regions are shown. (b) A close-up of one of the windows. The centre coefficient for the DFT is shown. (c) The window after transformation.*

3 Feature Extraction - Outline

The process of getting a parameter based description of the features in an image can be divided into three parts:

- Extracting local feature parameters from local spectra supplied by the MFT.
- Hierarchically determining the optimal division of the image into variable sized feature blocks.
- Linking the found single features together to form meaningful object boundaries in the image.

The following three sections will describe these three aspects of the feature extraction process, discuss the implementation and show the results of applying them to a small sequence of images.

In the local feature extraction, only single linear features will be looked for.

4 Local feature extraction

The task of deciding if an image region contains a single linear feature, and if so estimating its parameters, is done by analysing the local spectrum given by the MFT. In order to do this, a model of the spectrum of a linear feature is needed.

4.1 Linear feature model

As in [7], a straight line or edge of infinite length is considered to start with. Such a feature, as shown in figure 3, can be represented in the continuous spatial domain by

$$x(\vec{\xi}) = x(\vec{\xi} \cdot \vec{v}_\theta), \tag{11}$$

where

$$\vec{v}_\theta = \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix} \tag{12}$$

is the orientation unit vector and the one dimensional function $x(\cdot)$ represents the profile of the feature. The Fourier spectrum $\hat{x}(\vec{\omega})$ of such an image is concentrated in the orientation perpendicular to that of the feature:

$$\hat{x}(\vec{\omega}) = \hat{x}(\vec{\omega} \cdot \vec{v}_\theta)\delta(\vec{\omega} \cdot \vec{v}_{\theta_\perp}), \tag{13}$$

where $\vec{v}_{\theta_\perp}$ is the unit vector perpendicular to $\vec{v}_\theta$ and $\hat{x}(\cdot)$ is the one dimensional Fourier spectrum of $x(\cdot)$. Furthermore, it can be shown [15] that the spectrum has a linear

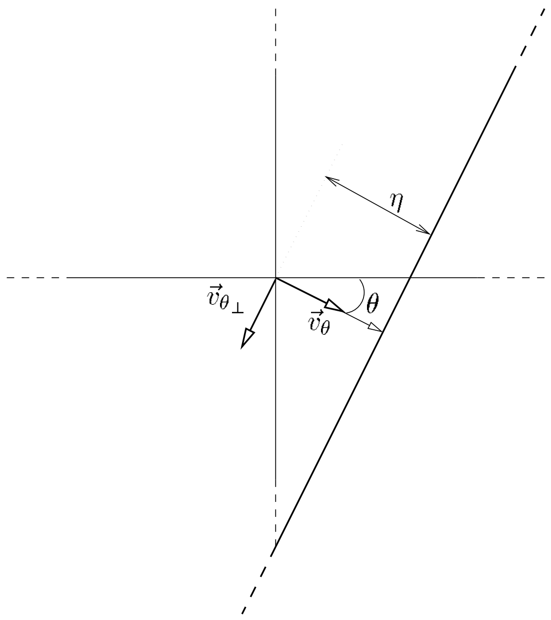


Figure 3: *An infinite lenght linear feature.*

phase, which relates to the position of the feature relative to the image's origin in orientation θ. The phase $\phi(\vec{\omega})$ of $\hat{x}(\vec{\omega})$ is given by:

$$\phi(\vec{\omega}) = \phi(\vec{\omega} \cdot \vec{v}_\theta) = \eta\vec{\omega} \cdot \vec{v}_\theta + \varepsilon(\vec{\omega} \cdot \vec{v}_\theta) \tag{14}$$

where η is the centroid of the one dimensional function $x(\cdot)$. The function $\varepsilon(\omega)$ is dependent on the profile of $x(\cdot)$. If the feature is a line, then

$$\varepsilon(\omega) = 0. \tag{15}$$

If the feature is an edge, then

$$\varepsilon(\omega) = \begin{cases} \pi/2 & \text{if } \omega < 0 \\ -\pi/2 & \text{if } \omega < 0 \end{cases} \tag{16}$$

However, the features in the spatial regions of the MFT are not infinitely long, but always of finite length, because of the spatially finite cosine window. The infinite feature becomes finite when it is multiplied in the spatial domain with the window function $w(\cdot)$:

$$x_w(\vec{\xi}) = x(\vec{\xi})w(\vec{\xi}) \tag{17}$$

In the frequency domain, this windowing results in a convolution of the linear feature spectrum with the Fourier transform of the window:

$$\hat{x}_w(\vec{\omega}) = \hat{x}(\vec{\omega}) * \hat{w}(\vec{\omega}). \tag{18}$$

The spectrum $\hat{w}(\vec{\omega})$ of the cosine window is real. The effect of the convolution of the spectrum $\hat{x}(\vec{\omega})$ with $\hat{w}(\vec{\omega})$ can be separated in two orientations. In the orientation of $\vec{v}_{\theta_\perp}$, perpendicular to the main direction of the spectrum, the convolution causes the spectrum of the feature to be spread out. This means that the spectrum changes from a perfect line to something more like an ellipse shape.

In the direction of $\vec{v}_\theta$, along the main axis of the spectrum, the phase of the spectrum can be affected by the convolution. The convolution in this direction can be seen as a weighted averaging of the neighbouring coefficients. Now, if the spectrum has a linear phase and the magnitude of the coefficients is about the same, then this averaging will not change the phase much. From (14) it is seen that the spectrum indeed has linear phase in the direction of $\vec{v}_\theta$, so the assumption could be made that the convolution will not have too much effect on the phase of the spectrum. However, there are two problems with this assumption. First, at the origin of the spectrum the sign of $\varepsilon(\omega)$ changes, which means that there is a phase jump. Therefore the phase around DC will be affected by the convolution. In the case of a line, this problem does not occur, since $\varepsilon(\omega) = 0$. The second problem is that if the feature is an edge, the magnitude of the coefficients will increase quite rapidly around DC. This can also cause the convolution to change the phase. These two problems cause the actual spectrum to deviate from the linear phase model, especially in the case of an edge (rather than a line) and especially around DC.

However, in this model the assumption is made that the windowing of the infinite linear feature has an effect only in the orientation of $\vec{v}_{\theta_\perp}$, perpendicular to the main axis of the spectrum. The spectrum can then be written as:

$$\hat{x}_w(\vec{\omega}) = \hat{x}(\vec{\omega} \cdot \vec{v}_\theta)\hat{w}(\vec{\omega} \cdot \vec{v}_{\theta_\perp}). \tag{19}$$

The phase $\phi_w(\vec{\omega})$ can now be written as:

$$\phi_w(\vec{\omega}) = \vec{\omega} \cdot (\eta \vec{v}_\theta + \eta' \vec{v}_{\theta_\perp}) + \varepsilon(\vec{\omega} \cdot \vec{v}_\theta), \tag{20}$$

where η' is the centroid of the linear feature, after windowing, in the direction along the feature, as shown in figure 4.

In this formula the phase is given relative to the axes parallel and perpendicular to the feature orientation θ. For practical use, it is helpful to transform it so that it is given relative to the normal $\vec{\xi}$-axes:

$$\phi_w(\vec{\omega}) = \vec{\xi}_0 \cdot \vec{\omega} + \varepsilon(\vec{\omega} \cdot \vec{v}_\theta), \tag{21}$$

where $\vec{\xi}_0$ is the spatial point that is associated with η and η', i.e. the centroid of the feature.

This model of the spectrum has two properties that can be used in identifying a linear feature:

1) The form of the spectrum is concentrated in one direction, which is perpendicular

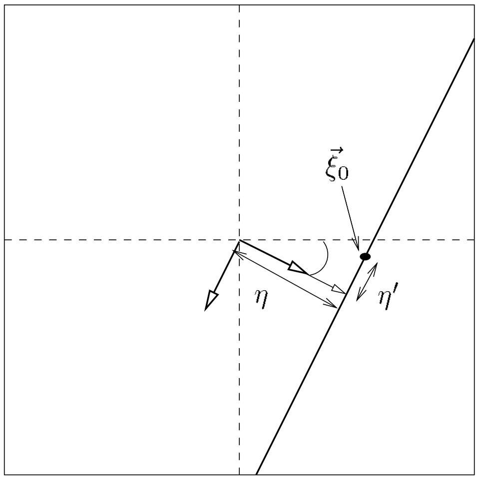


Figure 4: *An windowed linear feature.*

to the direction of the feature in the spatial domain.
2) The centroid of the feature can be derived from the first order partial derivatives of the linear phase of the spectrum.
In the next sections, it will be explained how those properties are used for the parameter estimation.

The model given here is based on a continuous and infinite frequency domain. However, the local spectra of the MFT in the work are calculated using a discrete DFT. One problem that is encountered when using a discrete transformation is a 'wrap-around' artifact in the spectrum. In the case of the spectrum of a linear feature, this artifact can be seen clearly if the magnitude of the spectral coefficients does not decrease much at higher frequencies along the main axis of the ellipse. Figure 5 shows the spectrum of an image displaying a line with gradient 1/2. It can be clearly seen that the ellipse in which the energy is concentrated 'wraps around' the border of the spectrum and continues on the other side. This is not predicted by the model and causes some problems, especially with the orientation estimation.

As mentioned before, the actual spectrum can deviate from the linear phase model due to the windowing, especially around DC. To minimise this deviation, a high pass version of the image is used to estimate the feature parameters from. The low frequencies that deviate most from the linear phase model will be reduced in this high pass version, so the parameter estimation will be more accurate. The high pass version of the image is obtained by subtracting a low pass version from the image. This low pass version is created by constructing a Gaussian pyramid of the image, from which a low pass sub-sampled level is interpolated back to the original size; a procedure described in [4]. The amount of low pass signal that is removed has to be chosen so that on the one hand, enough is removed to improve the accuracy of the

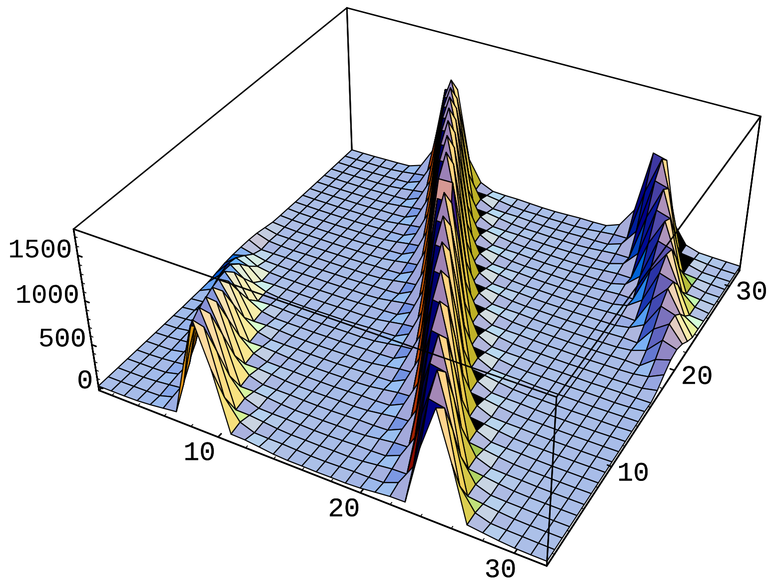


Figure 5: *Magnitude of local spectrum showing 'wrap-around' artifact.*

parameters, but on the other hand, enough signal energy is left compared to the noise energy to detect the feature. The optimal amount of low pass signal to be removed depends on the size of the block. Therefore, for different levels of the MFT different low pass versions are used. These different low pass versions can be obtained by using different levels of the Gaussian pyramid to interpolate the low pass image from. For a finer regulation of the amount of low pass energy, particular linear combinations of the different levels of the Gaussian pyramid can be used, as described in [23].

4.2 Feature parameters

Given the model of the spectrum of a linear feature, a check can be made whether a region of the image contains one linear feature by comparing its local spectrum with the model. There are four parameters of the model that will be calculated from the spectrum:

The orientation θ. This parameter gives the principal orientation of the spectrum.

The dispersion measure ζ. This parameter gives a measure of the preference of one orientation of the spectrum. From the model it is seen that a region with a linear feature has a very distinct preferred orientation.

The spatial position $\vec{\eta}_0$. This is the estimate for the $\vec{\xi}_0$, the spatial position of the centroid of the linear feature.

The correlation measure γ. This parameter gives a correlation measure of the real spectrum, and a spectrum of which the phase is synthesised from the estimated orientation and position parameters. This measure is used as a check whether the spectrum is well represented by the parameters.

4.3 Orientation and dispersion estimation

If an image contains a linear feature, the energy of the spectrum of this image can be modelled as an ellipse, with its major axis in the orientation perpendicular to that of the feature. The orientation of the feature can therefore be estimated by calculating the orientation of this major axis. Furthermore, by calculating the ratio between lengths of the major and the minor axis of this ellipse, a good indication of the dispersion of the spectrum energy can be derived.

Following [7] the calculation of the axes is done by defining the moment of inertia matrix of the spectrum. Of a region $\Lambda_{l,\vec{\chi}}$ of size $N_l \times N_l$ the inertia matrix is given by:

$$I = \begin{bmatrix} I_{11} & I_{12} \\ I_{21} & I_{22} \end{bmatrix} \tag{22}$$

where

$$
\begin{aligned}
I_{11} &= \sum_{k_1}\sum_{k_2} |\vec{x}(l,\vec{\lambda},k_1,k_2)|^2 k_1^2 && (23)\\
I_{22} &= \sum_{k_1}\sum_{k_2} |\vec{x}(l,\vec{\lambda},k_1,k_2)|^2 k_2^2 && (24)\\
I_{12} = I_{21} &= -\sum_{k_1}\sum_{k_2} |\vec{x}(l,\vec{\lambda},k_1,k_2)|^2 k_1 k_2 && (25)
\end{aligned}
$$

with

$$-\frac{N_l}{2} \le k_1, k_2 < \frac{N_l}{2}. \tag{26}$$

The orientations of the major and minor axes of the spectral energy ellipse are now represented by the eigenvectors corresponding to the largest and smallest eigenvalues, respectively [3]. The eigenvalues λ_1 and λ_2 of I are the solutions to the characteristic equation of I, and are given by:

$$\lambda_1 = \frac{I_{11} + I_{22} + \sqrt{I_{11} + I_{22} - 4(I_{11}I_{22} - I_{12}^2)}}{2} \tag{27}$$

$$\lambda_2 = \frac{I_{11} + I_{22} + \sqrt{I_{11} - I_{22} - 4(I_{11}I_{22} - I_{12}^2)}}{2} \tag{28}$$

The estimation of θ can now be found by calculating the orientation of an eigenvector corresponding to λ_1:

$$\theta = \arctan\left(\frac{\lambda_1 - I_{11}}{I_{12}}\right). \tag{29}$$

An indication of the dispersion of the energy spectrum is obtained by taking the ratio of the eigenvalues:

$$\zeta = \frac{\lambda_2}{\lambda_1}. \tag{30}$$

Since λ_1 is the largest eigenvalue, the value of ζ will range from 0 to 1. A region without any linear features, will not have an energy concentration at any particular orientation so the two eigenvalues will be roughly the same, therefore ζ is close to 1. A region with two (or more) linear features of different orientation will have energy concentrations along two (or more) orientations. In this case, ζ will still be rather large. Only if a region contains one or more linear features in one orientation, will ζ assume values closer to zero.

During experiments it was found that the orientation estimation as described above was in some cases inaccurate because of the 'wrap around' problem described in section 4.1. The inertia matrix will clearly be 'polluted' by the spectral energy of the wrap around parts of the spectrum. To solve this problem, a second stage was added to the orientation estimation. In this stage, only coefficients in a given area around

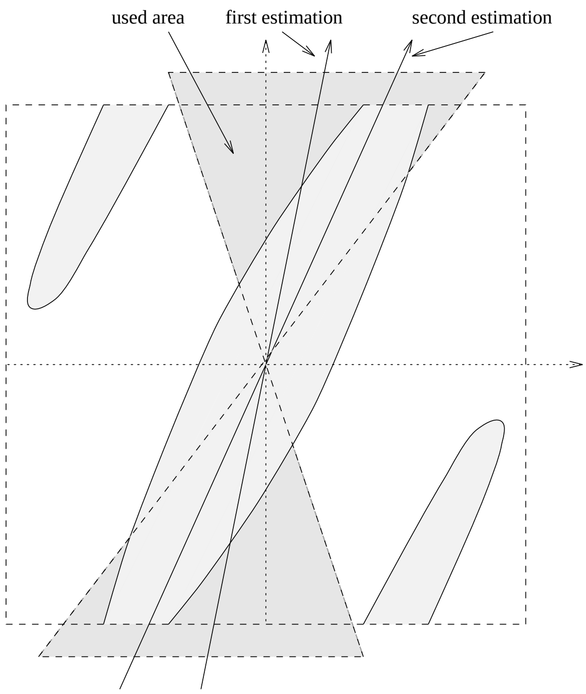


Figure 6: *Second stage orientation estimation.*

the orientation estimated in the first stage, are used to calculate a new inertia matrix. In Figure 6 this area is shown. The second estimation of the orientation, based on the new inertia matrix should be more accurate, as the experiments confirmed.

4.4 Position estimation

The estimation of the spatial centroid of the linear feature in a region can be made from the phase of the spectrum. According to the model, the spectrum of a linear feature can be written as:

$$\hat{x}(\vec{\omega}) = \begin{cases} |\hat{x}(\vec{\omega})| \exp[-j(\vec{\xi_0} \cdot \vec{\omega} + \varepsilon)] & \text{if } \vec{\omega} \cdot \vec{v}_\theta > 0 \\ |\hat{x}(\vec{\omega})| \exp[-j(\vec{\xi_0} \cdot \vec{\omega} - \varepsilon)] & \text{otherwise} \end{cases} \tag{31}$$

where θ is the feature orientation that by now is known, and $\vec{\xi_0}$ is the centroid that needs to be estimated. Rewriting this formula in an appropriate form for an MFT region $\Lambda_{l,\vec{\lambda}}$ gives:

$$\hat{x}(l, \vec{\lambda}, k_1, k_2) = \begin{cases} |\hat{x}(l, \vec{\lambda}, k_1, k_2)| \exp[-j(\frac{2\pi}{N_l}(\xi_{01} k_1 + \xi_{02} k_2) + \varepsilon)] & \text{if } k_1 v_{\theta 1} + k_2 v_{\theta 2} > 0 \\ |\hat{x}(l, \vec{\lambda}, k_1, k_2)| \exp[-j(\frac{2\pi}{N_l}(\xi_{01} k_1 + \xi_{02} k_2) - \varepsilon)] & \text{otherwise} \end{cases} \tag{32}$$

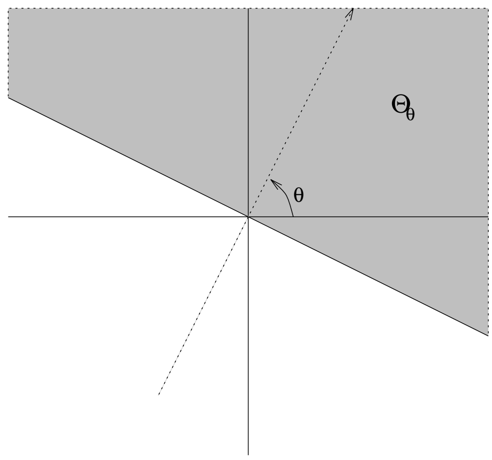


Figure 7: *Shaded area indicates Θ_θ. A phase shift occurs at the border of this area. (Copied from [4])*

The formula shows that the phase is linear in both dimensions, and that the position of the feature centroid in each dimension is proportional to the phase difference of two adjacent samples in that dimension. As in [7] a forward conjugate product between neighbouring samples in each dimension is defined as:

$$d_1(k_1, k_2) = \hat{x}(l, \vec{\lambda}, k_1, k_2)\hat{x}^*(l, \vec{\lambda}, k_1 + 1, k_2) \tag{33}$$

$$d_2(k_1, k_2) = \hat{x}(l, \vec{\lambda}, k_1, k_2)\hat{x}^*(l, \vec{\lambda}, k_1, k_2 + 1) \tag{34}$$

with $-\frac{N_l}{2} \leq k_1, k_2 < \frac{N_l}{2}$. If the spectrum is linear phase, then all $d_m(k_1, k_2)$ will have the same argument, $\frac{2\pi}{N_l}\xi_{0m}$, except for those frequencies that are along the line where the sign of the inner product $k_1 v_{\theta 1} + k_2 v_{\theta 2}$ changes. At frequencies along this line, which is shown in figure 7, a phase shift of 2ε occurs. In particular, the frequencies around DC will always be on this line, since it goes through the origin.

The position $\vec{\xi_0}$ of the centroid of the feature can be estimated by calculating an average phase difference over all frequencies in each direction. This can be done by taking the sum of all forward conjugate products:

$$R_m = \sum_{k_1} \sum_{k_2} d_m(k_1, k_2), \tag{35}$$

for $m = 1, 2$. However, the conjugate products which involve the DC-component, i.e. $k_m = -1$ and $k_m = 0$, should be excluded from this sum, for the 2ε phase shift at DC will negatively affect the accuracy of the estimation.

The phases of R_1 and R_2 now give the estimate for $\vec{\xi_0}$:

$$\vec{\eta_0} = \frac{N_l}{2\pi} \begin{bmatrix} \arg R_1 \\ \arg R_2 \end{bmatrix} \tag{36}$$

4.5 Correlation check

Once the orientation and position parameters have been estimated, it is necessary to check whether the model with these parameters is accurate.

Again following [7], the check is done by synthesising the local spectrum using the estimated parameters, and then comparing this with the real spectrum, by means of a correlation function. The synthesised spectrum $\tilde{x}(l, \vec{\lambda}, k_1, k_2)$, is generated using the magnitude of the real spectrum and the location parameters to estimate the phase:

$$\tilde{x}(l, \vec{\lambda}, k_1, k_2) = |\hat{x}(l, \vec{\lambda}, k_1, k_2)| \exp[-j(\frac{2\pi}{N_l}(\eta_{01}k_1 + \eta_{02}k_2)] \tag{37}$$

The comparison of the synthesised and the real specta is done by calculating inner product of the real and synthesised coefficients. However, the sum is only taken over those frequency coefficients (k_1, k_2) for which $k_1 v_{\theta 1} + k_2 v_{\theta 2} > 0$, i.e. for which the sign of the phase constant ε is positive. This set of coefficients is named Θ_θ and is shown in figure 7.

The correlation sum γ can now be given by:

$$\gamma = \sum_{(k_1,k_2) \in \Theta_\theta} \tilde{x}(l, \vec{\lambda}, k_1, k_2) \hat{x}^*(l, \vec{\lambda}, k_1, k_2) \tag{38}$$

where '*' denotes the complex conjugate.

If the spectrum $\hat{x}(l, \vec{\lambda}, k_1, k_2)$ is linear phase and therefore given by (32), γ can be rewritten as:

$$\gamma = \exp[j\varepsilon] \sum_{(k_1,k_2) \in \Theta_\theta} |\hat{x}(l, \vec{\lambda}, k_1, k_2)|^2 \exp[j((\xi_{01} - \eta_{01})k_1 + (\xi_{02} - \eta_{02})k_2)] \tag{39}$$

This formula shows that γ has a maximum value of $\gamma_{\max}$,

$$\gamma_{\max} = \sum_{(k_1,k_2) \in \Theta_\theta} |\hat{x}(l, \vec{\lambda}, k_1, k_2)|^2 \tag{40}$$

and that it reaches this maximum value when the estimation $\vec{\eta}_0$ for $\vec{\xi}_0$ is accurate. As described in [7], the ratio $\frac{\gamma}{\gamma_{\max}}$ can be used as in indicator of the accuracy with which the model and the estimated parameters describe the real spectrum.

Experiments showed that very low intensity features, which were really a kind of structured noise, could result in a high correlation ratio. These features do not actually correspond to any object boundaries in the image and should therefore be suppressed. A simple way to reduce their correlation ratio is to add a suitable small constant γ_c to $\gamma_{\max}$. The ratio $\frac{\gamma}{\gamma_{\max}+\gamma_c}$ will be reduced for the very low energy features, while the ratio for 'real' higher intensity features will hardly be affected.

Figure 8: *Original missa077 image.*

4.6 Results

The local feature extraction is applied to one greyscale frame out of the 'Miss America' sequence, which is shown in fig 8. The resulting features for the levels 3 to 6 are shown in fig 9. The threshold values for the dispersion measure ζ and correlation measure $\frac{\gamma}{\gamma_{\max}+\gamma_c}$ used here are the same as those that will be used for the hierarchical feature detection, as is the constant γ_c. It can be seen that most boundaries are found at some level. At level 3 only very clear large scale features exeed the threshold values. At level 4 a good outline of the head and shoulder is found, except for where the hair meets the background or the dark shoulders. At level 5 in addition to the outlines more smaller scale features are added in the face. At level 6 the details in the face are better found, but the larger scale features are less clear and are often found double. Clearly these should be represented at one of the larger scale levels.

In none of the levels is the boundary between the hair and the background found. Although the grey level difference between the two regions is very low, one would hope that it can be detected. The reason that it is not found is that the Miss America sequence contains a global noise in one particular orientation (vertical). This structured noise can be seen when looking at the images on the display. Because of the directional structure, the noise shows up clearly at particular frequencies of the local spectra. At places where the boundary between two regions is very weak, this noise has significant energy in the spectrum compared to the feature energy. When calculating the preferred orientation of the spectrum this noise causes a bias in the orientation estimate in the best case and totally dominates the estimate in the worst case. No solution has been found for this problem.

5 Hierarchical feature extraction

In this second part of the feature extraction process, a top-down hierarchical scheme is used to find the optimal division of the image into variable sized blocks. The optimal division in the work would be that in which each block represents at most one linear feature, at its largest possible scale, without missing any relevant features.

The local spectra are provided by the MFT, using an appropriate high pass version of the image for each level.

In finding the right block size for a particular region of the image, the hierarchical algorithm makes use of a check between two consecutive levels of the MFT. This check, the scale consistency check, is described first.

5.1 Scale consistency check

When a region, on the grounds of its spectral parameters, is assumed to contain exactly one linear feature, it is subjected to the scale consistency test. The objective

Figure 9: *Local linear features in missa077 image at levels 3 to 6.*

of the check is to test once more if this is the right scale for this region of the image, by comparing the information found at this scale with that of a smaller scale, one level down in the MFT. Therefore this region is compared to its four child regions in the quad tree. This can be done in two ways.

The first way is to compare the parameters of the feature found in the parent region to the parameters of those that are found in the child regions. The idea in this case is that if the parent region really contains only one feature, then the parameters of its child regions should not indicate incompatible features. The way in which the parameters are compared is as done in [5]. For each child region, the parameters of the model are calculated as usual. If the parameters of a child region indicate that there is no linear feature present in this region then this child region is not in conflict with the parent. If however the parameters do indicate a linear feature, then the orientation and location parameters of the child are compared with those of the parent, and a consistency measure is calculated. The parent region passes the scale consistency check if none of its children have a consistency measure below a chosen lower threshold value.

The consistency measure used in this work is defined in [5]. It gives a result between 0 and 1, becoming larger when both the orientation and the position of the parent and child features are similar. If either or both of the parameters are significantly inconsistent, then the result will be lower.

The second way in which scale consistency can be checked is by using a spectrum correlation as described in section 4.5. If there is only one feature in the region, then the phase of the spectra of the child regions can be modelled from the position of the centroid of this feature relative to the centre of each child region. For each child region the centroid of the feature is calculated from the position of the feature in the parent region. Then the phase of the child spectrum is synthesised and the correlation between this synthesised spectrum and the actual spectrum of the child region is calculated. If a child region contained more than just this one feature, the correlation will be low, which indicates that the parent region should be split into smaller regions.

This second method is less ad hoc and more consistent with the rest of the work than the first one. However, the correlation can be quite sensitive to small errors in the position and orientation.

At the moment, the first method is used.

5.2 Hierarchical algorithm

The quad-tree structure representing the image at its best scale is constructed using this algorithm:

Starting from a given start level l_s the following analysis is done at each node:

1. The total energy of the spectrum is calculated and checked against a threshold

value dependent on the mean energy of the image. If the energy is below the threshold the region is classified as low pass. The quad-tree is truncated at this node, which thus becomes a leaf node.

2. Otherwise, the orientation θ and the dispersion ratio ζ of the spectrum are calculated from the inertia matrix.
   If ζ is above a given upper threshold, it is decided that the region does not contain one linear feature. It might contain either no, or more than one feature. The algorithm proceeds to the next level to process all four child regions.

3. Otherwise, the location estimation $\vec{\eta}$ is calculated and from this, the correlation measure γ.
   If γ is below a given lower threshold, it is decided that the model and parameters do not accurately represent the spectrum. The algorithm proceeds to the next level to process all four child regions.

4. Otherwise, the scale consistency test is performed.
   If there is an inconsistency with one or more of the children, it is decided that the region has to be split up to accommodate the found feature in the child region. The algorithm proceeds to the next level to process the four child regions.

5. Otherwise, the region is classified as containing one linear feature. The quad-tree is truncated at this node, which thus becomes a leaf node. All calculated parameters are stored for use in the reconstruction step.

If the recursion reaches a given end level l_e, then an alternative action is taken: If the spectrum parameters at this last level indicate that there is a linear feature present, then it will be accepted without a scale consistency check, since there are no child regions. If, however, no linear feature is present at this last level, then this region is marked as empty and the recursion is stopped. Wherever there are no features in the image, the quad-tree will be fully extended to the level l_e, and all regions will be classified as empty. These parts of the tree are pruned back as far as possible, by joining four empty leaf regions into one bigger, empty leaf at one level up the tree.

The threshold used in the algorithm all have to be set before starting the algorithm. The thresholds can be set differently at different scales.

5.3 Results

Applying the hierachical feature extraction scheme to the missa077 image in figure 8 yields the result shown in figure 10. The shoulder outline is mostly represented in large level 3 and 4 features, as is most of the clear neck and collar. The facial features are almost all represented in small level 5 and 6 features. There are several places where the image is not represented optimally. For instance a part of the right collar

Figure 10: *Multiscale linear features in missa077 image (overlay).*

boundary is represented in small level 5 and 6 features although it is a larger scale feature that was found on level 4. This is caused by the fact that the boundary at that point is less sharp, which at the smaller scales causes several parallel features to be found. During the scale consistency check, the larger feature is discarded because of these parallel features in the child regions. In general this will happen for less sharp large scale features. A possible solution could be to switch to the second described scale consistency check based on the spectrum correlation.

6 Feature linking

The hierarchical feature extraction algorithm yields a division of the image into blocks that all contain at most one linear feature. Most of these features will be part of a boundary of an object in the image, but almost none of them will represent the whole object boundary by itself. The features are extracted as single linear features because that is the easiest way to extract them, but when the features are used to help the video coding, it is desired that a feature represents a complete boundary in the image, rather then a single linear piece of boundary. Therefore the features need to be linked together to form the more complex boundaries of objects. This needs to be done in such a way that all the single features belonging to one boundary are linked and features belonging to different object boundaries are not linked.

The basic way to check whether two found linear features should be linked together is again based on a spectrum correlation. Given two features that represent two parts of the same object boundary, they should have the same edge profile and therefore similar spectra. By adapting the spectrum of one block for the difference in position and orientation of the features in the blocks, an approximation of the spectrum of the other block can be synthesised. Correlating this synthesised spectrum with the actual spectrum of the block gives a measure of similarity between the spectra of the blocks. If this measure indicates that the spectra are similar it can be concluded that the blocks represent the same object boundary and the features can be linked.

Before this spectrum correlation test is done, the two features are first subjected to a co-linearity test. This test, based purely on the location and orientation parameters of the features, ensures that the features are reasonably well aligned, as might be expected from two parts of the same object boundary. The co-linearity test is only used to stop obviously incompatible features being linked and, once passed, does not influence the final decision based on the spectrum correlation.

Apart from linking two found linear features there are two more types of linking processes. First there is 'gap filling'. Two features can be linked even though the blocks that they are in do not border each other. In this case, there is a gap between the linked features, which one would expect the object boundary that the features represent to go through. An attempt can be made to fill these gaps by examining the spectrum for evidence of the feature where it is expected.

The other kind of linking is 'extending' a feature. If a found feature has no found neighbouring features at all, it can be that the actual boundary in the image continues anyway, but is for some reason not detected in the extraction process. Assuming that the boundary continues in the neighbouring block roughly where the feature ended in this block, the spectrum of the neighbouring block can be checked for evidence of a feature at that position.

6.1 Linking algorithm

All features in an image are linked into more complex features in the following way: In a structured way the algorithm picks a starting region. If the region contains a feature then the two endpoints of that feature at the border of the non-overlapping part of the block are calculated. First, one of the feature ends is going to be linked. A group of neighbouring features is considered and the one with the best fit is chosen. If a suitable link is found, it is checked if there is a gap and if so an attempt is made to fill it. Then the process is continued with the new feature that has been linked. In this way the object boundary that is represented by the features is followed. When at some point no further feature can be linked, it will be tried to extend the feature into the neighbouring block. If this succeeds, the linking process continues from this new found feature. If the feature can not be extended then this end of the link comes to an end. Now the second feature end of the starting feature will be linked in the same way. When this end comes to and end as well, a new starting feature that not yet has been linked is chosen and a new object boundary is traced. When no unprocessed features are left the linking process is done.

The following procedure is used to find the best feature to link the start feature to: Three sets of candidate features are made, indicating an order of preference (See figure 11). Candidate set 1 contains just the block that borders immediately to the feature end. Candidate set 2 contains all other blocks that border to the side of the block on which the feature ends, including those at the corner. Set 3 contains all blocks within an area of the size of the feature block on the side that the feature ends in, including corner blocks.

First the block in set 1 is checked for a suitable link. If this block contains a feature and it passed the co-linearity test and the spectrum correlation test then the features are linked. If they can not be linked, all features in set 2 are considered. If more features pass the tests, the one with the best result in the spectrum correlation test is chosen. If again no link is found, the features in set 3 are checked. Obviously, different and more candidate sets can be used.

Figure 12 shows a sketch of the co-linearity test. Consider two features with centroid positions $\vec{\xi}_1$ and $\vec{\xi}_2$ respectively. The vector $\vec{d}$ pointing from $\vec{\xi}_1$ to $\vec{\xi}_2$ is calculated. If the features are co-linear, the differences between the orientation of the

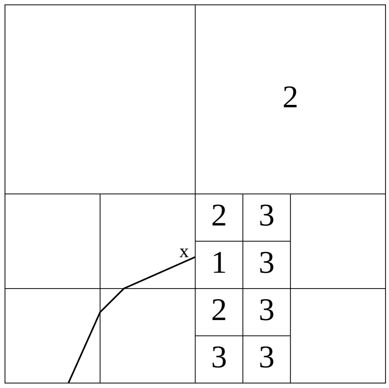


Figure 11: *The candidate sets 1 to 3 for linking of feature end x.*

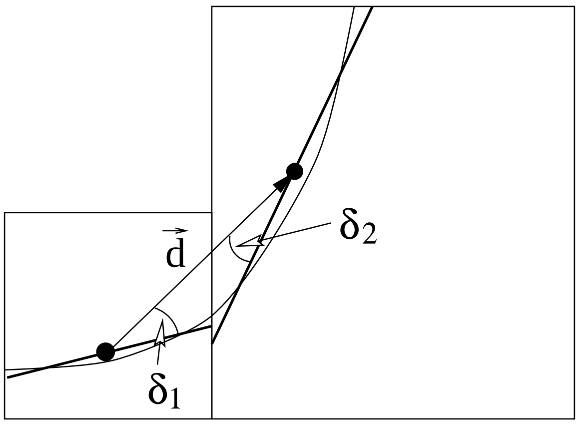


Figure 12: *Sketch of co-linearity test.*

features and this vector $\vec{d}$, named δ_1 and δ_2, are both small. When the two features are both part of the same *curved* boundary, δ_1 and δ_2 will deviate from zero, but within limits. When a δ_i becomes too big, it can be concluded that the two features are not of the same boundary. This is tested by comparing $\cos(\delta_1)$ and $\cos(\delta_2)$ with a threshold value.

As described in [7], the spectrum correlation between two regions $\Lambda_{l,\vec{\lambda}_1}$ and $\Lambda_{l,\vec{\lambda}_2}$ is calculated as follows: From the spectrum $\hat{x}(l, \vec{\lambda}_1, \vec{\omega})$ the linear phase is removed to compensate for the position of the centroid $\vec{\xi}_{01}$ of the feature relative to the region centre,

$$\hat{x}'(\vec{\omega}) = \hat{x}(l, \vec{\lambda}_1, \vec{\omega}) \exp[+j(\vec{\xi}_{01} \cdot \vec{\omega})]. \tag{41}$$

Then the spectrum is rotated over the difference in orientation, $\Delta\theta$, of the two features,

$$\hat{x}''(\vec{\omega}) = \hat{x}'(R_{\Delta\theta}\vec{\omega}), \tag{42}$$

where $R_{\Delta\theta}$ is the rotation matrix:

$$R_{\Delta\theta} = \begin{bmatrix} \cos(\Delta\theta) & -\sin(\Delta\theta) \\ \sin(\Delta\theta) & \cos(\Delta\theta) \end{bmatrix}. \tag{43}$$

Finally, the linear phase corresponding to the feature centroid $\vec{\xi_0}_2$ is introduced to the spectrum,

$$\tilde{x}(l, \vec{\lambda}_2, \vec{\omega}) = \hat{x}''(\vec{\omega}) \exp[-j(\vec{\xi_0}_2 \cdot \vec{\omega})]. \tag{44}$$

This gives the approximation $\tilde{x}(l, \vec{\lambda}_2, \vec{\omega})$ of spectrum $\hat{x}(l, \vec{\lambda}_2, \vec{\omega})$.

In practice, the spectra are discrete rather then continuous, which presents a problem when rotating the spectrum. If (k_1, k_2) is an integer frequency coordinate, then $(l_1 l_2)^T = R_{\Delta\theta}(k_1, k_2)^T$ is usually not a integer coordinate. The value of the spectrum at coordinate (l_1, l_2) is therefore not directly available and should be estimated from neighbouring coefficients. This is done by bilinear interpolation from the four surrounding coefficients. The bilinear interpolation should give reasonable results because it is applied to a spectrum from which the phase component has been removed. Another practical aspect of the spectra is that they are finite and square. When rotated, some coordinates will be mapped to coordinates outside the actual spectrum. These coordinates are set to zero.

The estimation $\tilde{x}(l, \vec{\lambda}_2, k_1, k_2)$ of the discrete spectrum $\hat{x}(l, \vec{\lambda}_2, k_1, k_2)$ is performed as described above. The correlation r between the synthesised and actual spectrum is now calculated by:

$$r = \frac{\sum_{k_1,k_2} \tilde{x}(l, \vec{\lambda}_2, k_1, k_2)\hat{x}^*(l, \vec{\lambda}_2, k_1, k_2)}{\sqrt{(\sum_{k_1,k_2} |\tilde{x}(l, \vec{\lambda}_2, k_1, k_2)|^2)}\sqrt{(\sum_{k_1,k_2} |\hat{x}(l, \vec{\lambda}_2, k_1, k_2)|^2)}}, \tag{45}$$

which gives a value between -1 and 1. A decision to link the two features is now made by comparing r with a given threshold.

One aspect of the spectrum transformation that has not been mentioned yet is that the width of the spectrum, i.e. the spread in the direction perpendicular to the main axis, depends on the length of the feature in the block. If the feature passes through a corner of the block it will be shorter, which results in a wider spectrum. This may cause the correlation value to drop when the feature crosses through the middle of one block but through the corner of the other. A way to reduce this bias is to restrict the summations in (45) to a narrow band of coefficients along the main axis of the spectrum, where both spectra can be assumed to have much energy.

When linking two found features, the correlation measure is only applied to blocks on the same level. When two features of different scales need to be linked, this will be done at the level of the smallest of the two blocks. In this case the bigger of the two regions is represented by its appropriate child region of the right size.

6.2 Gap filling

A gap between the ends of two linked features can span one or more blocks. It is logical to assume that the object boundary that the two features belong to does not cease to exist in the gap, but rather that it has been missed in the feature extraction process. Filling in the gap by looking for these missed features is done using the spectrum correlation method.

For each block in the gap, the expected position of the centroid and expected orientation are found by simply assuming a straight line between the ends of the two linked features. Using these estimated feature parameters, a spectrum correlation can be calculated with one of the two linked features, in exactly the same way as done when linking two existing features. Of the two available spectra, the one that is closest in scale to the gap block is used. Again the result of the correlation is checked against a threshold value. This value is usually set lower than the value used for normal linking, because of the strong evidence that there should be a feature there.

6.3 Feature extending

When a feature end cannot be linked, two cases can be distinguished: 1) The object boundary in the image has come to an end. 2) The object boundary has not come to an end, but the next bit of it has not been detected. Quite often, the second case applies. To find as much of the object boundaries as possible, it can be attempted to extend this feature, the end-feature, into the region into which it crossed, the extend-region.

To do this, a set of possible positions and orientations in which the end-feature can be continued is compiled. The basic position and orientation are those by which the extend-feature will be in a straight line from the end-feature. The other combinations of the set are variations in position and orientation around the basic ones, as shown in figure 13. For all the combinations in the set, the spectrum correlation with the end-feature is calculated in the by now conventional manner. If the maximum found correlation is higher than the set threshold then the feature with the corresponding parameters is assigned to the extend-block. The linking process now continues from this feature.

The range of orientations over which the extended feature should be looked for can be quite large. Two features on a curved object boundary can easily have a 45 degree difference in orientation, which makes the search interval at least 90 degrees. Choosing the step-size to cover this interval with, is a trade-off between performance and computational load of the algorithm. It is also necessary to look at the behaviour of the correlation function of two spectra when there is an error in the estimation of the orientation of the feature. Figure 14 shows the value of the correlation function applied to the spectrum of a single line and the rotated version of that same image. It can be seen that the correlation has already dropped to 70% of the optimal value when the features are 5 degrees out of line. This drop-off is a bit less drastic when it concerns an edge or less sharp feature, but it can be concluded that using steps of more then 5 degrees might cause features to be missed.

This spectrum correlation method can be used as long as the extend-block is of the same size or smaller than the end-block. If it is smaller, then the appropriate child region of the end-block is used for the correlation. If however the extend-block is larger than the end-block, the same method cannot be applied. Taking a child region of the extend-block to do the correlation with the end-block is not a good solution because then a feature would be assigned to the whole extend-region based on examining only a part of the region while having no knowledge of the rest of it.

A way out of this problem is to derive a reduced size spectrum from the full size spectrum of the extend-block, as done in [7]. The reduced size spectrum should only have those frequency coefficients that a spectrum of the smaller block size has. With the definition of the MFT used in this work, reducing the spectrum by half the size is very easy, because the bigger spectrum has coefficients at all the frequency coordinates that the smaller size spectrum has. Reducing the size of the spectrum is therefore just a question of sub-sampling the spectrum by a factor 2. This smaller spectrum, that still represents the whole bigger region, can now be used for calculating a spectrum correlation. Extending into regions more than twice the size of the end-region is not done in this work.

6.4 Results

The feature linking process is applied to the set of features found by the hierarchical feature detection process. In figure 15 the result for the features in figure 10 from the image in fig 8 is shown. The little circles indicate end points of links. It could not be explicitly indicated which features connect to which, but it can hopefully in most cases be derived from the end points. To clean the image up at bit, all single unlinked features are removed from the image, although this doesn't make much difference, since most features are linked to others.

It can be seen that the linking algorithm works reasonably well. Especially the larger features like the shoulders, the neck and collar and the wrinkles in the cardigan are well linked up. In the facial features, the linking is less clear but again larger features like the eyebrows and the outer contours of the mouth are properly linked.

In several places, like in the wrinkles in the cardigan, gaps can be seen to be filled up by the linking process.

There are problems in the places where the boundaries in the image are less sharp, especially in shaded areas where there are no real boundaries. In these places features are sometimes found and linked, but it is not clear what they actually represent.

Another problem is that there are places where found features seem perfectly to connect to each other, but are not linked by the linking process. For instance, the lower part of the left side of the neck is not connected to the upper part of the neck, but to the left ear. This is probably caused by the fact that the ear-feature tried to link to the lower-neck-feature and succeeded because, although not part of the same boundary, they have the same profile. The lower-neck-feature is then no longer available to be linked to the upper-neck-feature which it should be linked to. A possible solution to this would be to allow more links to one feature end, thus resulting in some kind of branched graph of connected features instead of a single ordered row. By some post-processing step this graph must then be reduced to a representation that is useful for the coding application. This would also solve the problem that the boundary following method can be 'led astray' by linking to a small dead-end feature instead of following the boundary.

Also apparent are multiple parallel features around boundaries that are less sharp. A post processing step should be used to remove these.

In figure 16 the feature extraction results for four frames of the Miss America sequence are shown. The frames are two frames apart to increase the motion between them. It can be seen that the larger features can be identified in each frame and can be seen moving in a reasonably stable way.

However, it can also be seen that the hierarchical division in the frames varies quite a bit. This would indicate that the scale selection process is quite sensitive to slight variations in the image and to the position of features relative to the block boundaries. This would again indicate that experimenting with a different scale consistency measure might improve results. Another problem that can be seen from the results is that features indicating less sharp boundaries, like the wrinkles in the cardigan, seem to vary in position more then the actual image does at those places. This can possibly cause problems when these features are used for correction of the motion estimation.

6.4.1 Comparing results

In order to see if the feature detection algorithm misses any features that are detectable, the results are compared with the result of a simple edge detection algorithm which uses a 3×3 Sobel edge detection mask [11]. Figure 17 shows both the result of the Sobel edge detection(a) and the feature detection(c) algorithm applied to the missa077 image. It can be seen that some very low intensity object boundaries, in particular almost all of the outline of the hair, are not found by the feature detection, but do show up in the edge detection. Although the hair boundary is very faint, it might none the less be important in the video coding, because it is visible to a human viewer and errors in it can therefore be disturbing.

The reason why the hair boundary is not found is, as has been mentioned earlier, the noise that is present in the image. This noise can also clearly be seen in the Sobel edge image, but the actual hair boundary is still prominent. To show that the boundary can be separated from the noise, a simple multiresolution edge detection process was applied. In this process, the same Sobel edge detection was applied to a Gaussian filtered and subsampled version of the image (i.e. one level up in the Gaussian pyramid). The multiresolution result was then obtained by setting values in the original edge image to zero, if the corresponding value in the subsampled edge image failed to exceed a given threshold. The result, shown in figure 17 (b), clearly shows the hair boundary.

The conclusion that can be drawn is that the feature detection process misses certain low intensity boundaries that can actually be found. Ways of improving these results are being looked at.

7 Conclusions and future work

A novel image sequence coding scheme has been proposed, which makes use of affine motion parameters and extracted linear features. It has been pointed out that the Multiresolution Fourier Transform provides a suitable basis for a consistent framework in which the different aspects of the scheme can be dealt with. The focus of the report has been on the extraction of single linear features and the linking of them into complex features representing object boundaries, all within the multiresolution framework. A model for the spectrum of an image containing a linear feature and algorithms to estimate the model parameters from the image have been presented. Furthermore a method for linking the single features into relevant boundary representations has been given.

All described algorithms are implemented and results are shown. It was found that the resulting features were quite reasonable, but not yet ideal for the coding task. Several problems indicated that it might be worth to trying improve the scale selection process. The linking algorithm gave reasonable results, but can also be improved upon. By allowing feature-ends to be linked to more than one other feature, longer features can probably be found. The graph like structure that results from that should be processed to find the optimal feature links. Also a problem are small garbage features and parallel double features that represent the same boundary. These problems should be taken care of in a 'cleaning up' post processing step.

Looking beyond the stage of feature extraction, the question of how exactly to use these features in the coding scheme will be the next step in the project. This will involve several different aspects:

- Identifying corresponding features in consecutive frames, that represent the same object boundary.
- Combining the affine motion estimates and the features to improve on both the motion estimation and the feature linking.

- Incorporate low-pass information to arrive at a division of the image into the mentioned planar patches.

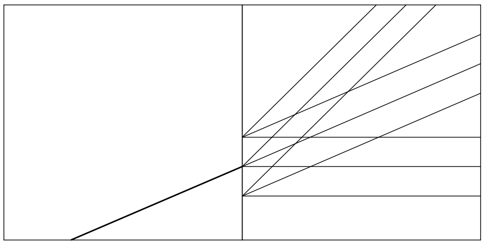


Figure 13: *Possible extended feature positions.*

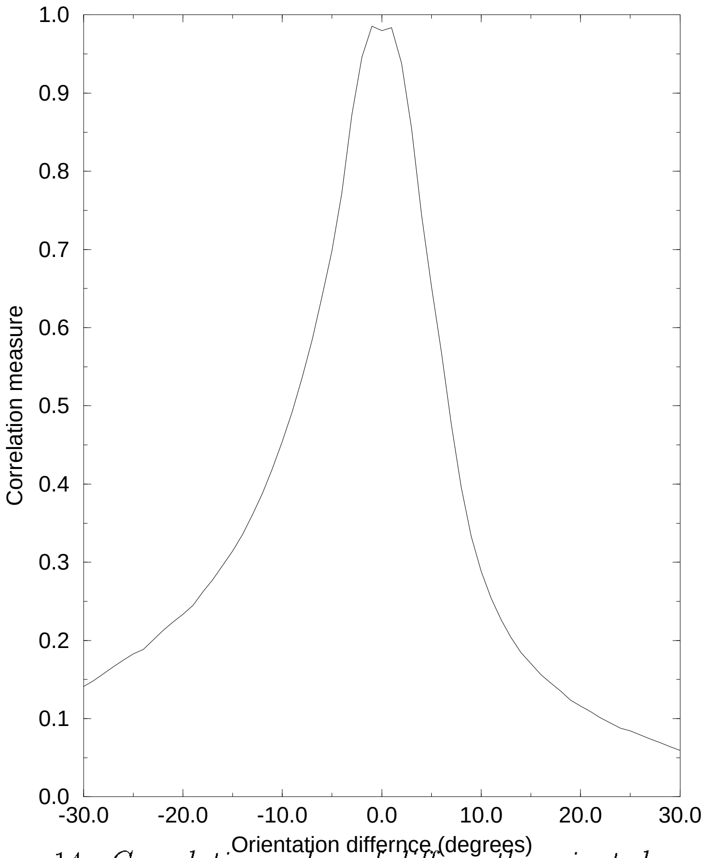


Figure 14: *Correlation value of differently oriented spectra.*

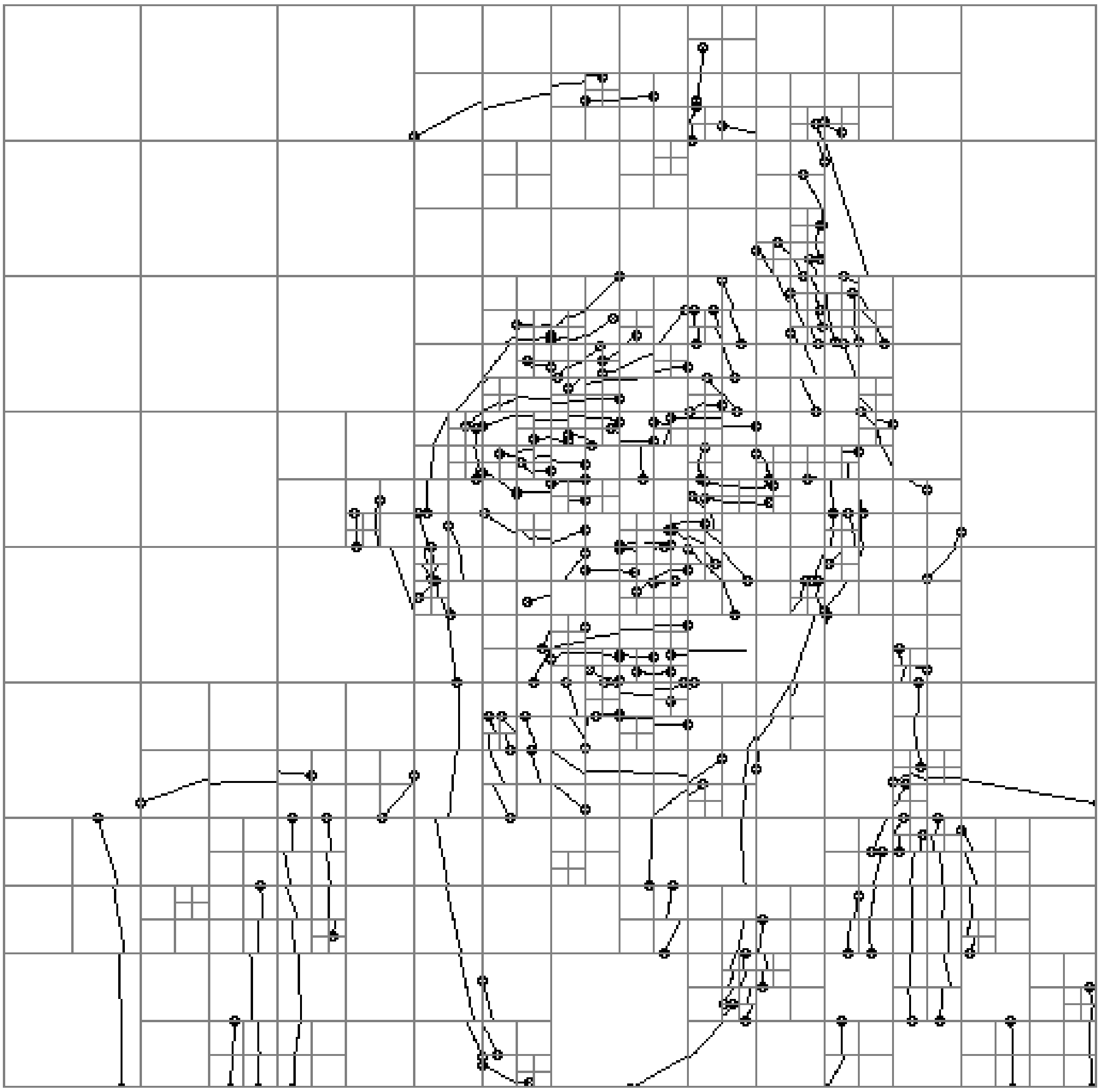
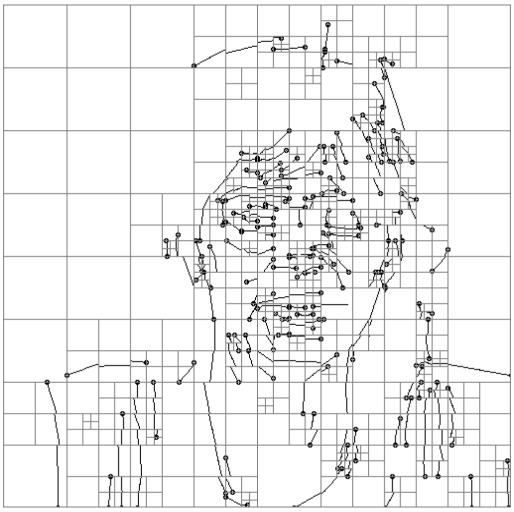
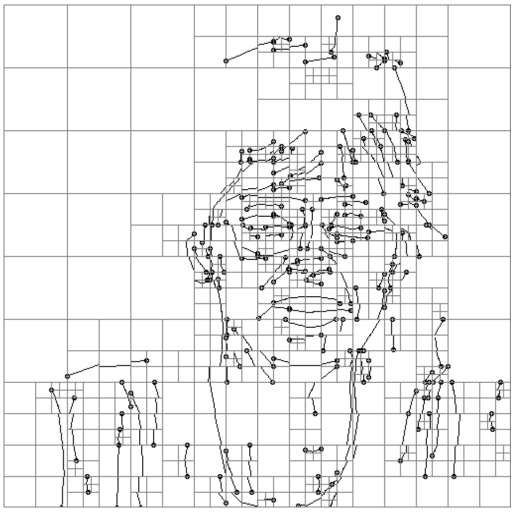


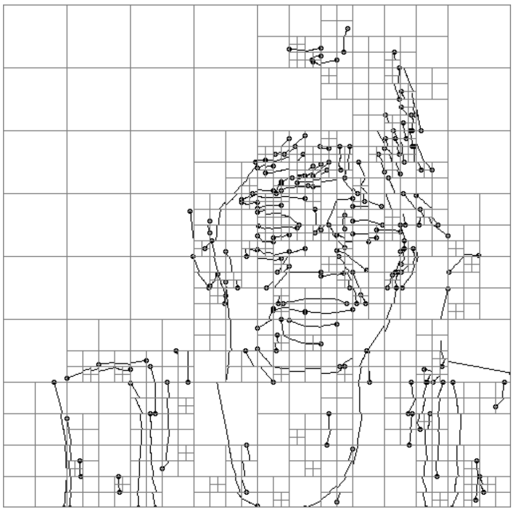
Figure 15: *Linked features of image missa077*



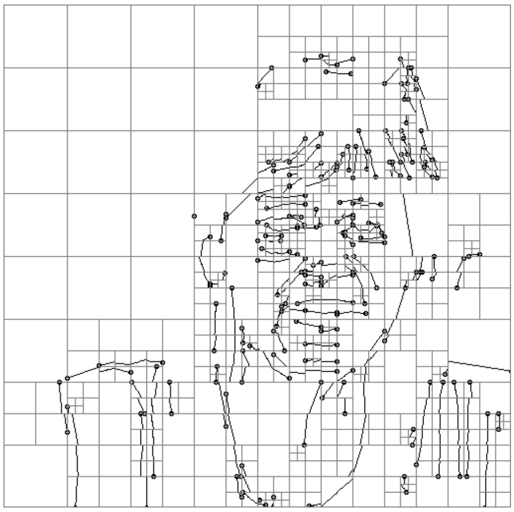
missa077



missa079



missa081



missa083

Figure 16: *Linked features of images missa077, 079, 081 and 083.*

(a)

(b)

(c)

Figure 17: *(a) Result of Sobel edge detection. (b) Result of simple multiresolution Sobel edge detection. (c) Result of feature detection (after linking process).*

References

[1] E. H. Adelson, J. Y. A. Wang, and S. A. Niyogi. Mid-level vision: new directions in vision and video. In *Proc. IEEE International Conference on Image Processing.*, volume II, pages 21–25, Austin, Texas, 1994.

[2] J. B. Allen and L. R. Rabiner. A unified approach to short-time Fourier analysis and synthesis. *Proceedings of the IEEE*, 65(11):1558–1564, November 1977.

[3] A. I. Borisenko and I. E. Tarapov. *Vector and Tensor Analysis.* Dover Publications, Inc., New York, 1979.

[4] P. J. Burt and E. H. Adelson. The laplacian pyramid as a compact image code. *IEEE Trans. Communications*, 31(4):337–345, April 1983.

[5] Andrew Calway. *The Multiresolution Fourier Transform: A general Purpose Tool for Image Analysis.* PhD thesis, Department of Computer Science, The University of Warwick, UK, September 1989.

[6] C. S. Choi, K. Aizawa, H. Harashima, and T. Takebe. Analysis and synthesis of facial image sequences in model-based image coding. *IEEE Trans. Circuits and Systems for Video Technology*, 4(3):257–275, June 1994.

[7] A. R. Davies. *Image feature analysis using the Multiresolution Fourier Transform.* PhD thesis, Department of Computer Science, The University of Warwick, UK, August 1993.

[8] R. O. Duda and P. E. Hart. Use of the Hough transform to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15, January 1972.

[9] N. W. Garnham and M.K. Ibrahim. Classified subregion motion estimated interframe coding. In *Proc. IEE Colloquium on Low Bit Rate Image Coding*, pages 3/1–3/6, London, 1995.

[10] T. Hsu. *Texture analysis and synthesis using the Multiresolution Fourier Transform.* PhD thesis, Department of Computer Science, The University of Warwick, UK, September 1994.

[11] A. K. Jain. *Fundamentals of Digital Image Processing.* Prentice Hall, 1989.

[12] D. LeGall. MPEG: A video compression standard for multimedia applications. *Communications of the ACM*, 34(4):47–58, April 1991.

[13] H. Li, P. Roivainen, and R. Forchheimer. 3-D motion estimation in model-based facial image coding. *IEEE Trans. P.A.M.I*, 15(6):545–555, June 1993.

[14] King N. Ngan and Weng L. Chooi. Very low bit rate video coding using 3D subband approach. *IEEE Trans. Circuits and Systems for Video Technology*, 4(3):309–316, June 1994.

[15] A. Papoulis. *Signal Analysis.* McGraw-Hill, 1984.

[16] O. Rioul and M. Vetterli. Wavelets and signal processing. *IEEE SP Magazine*, pages 14–38, October 1991.

[17] V. Seferidis and M. Ghanbari. General approach to block-matching motion estimation. *Optical Engineering*, 32(7):1464–1474, July 1993.

[18] E. P. Simoncelli, W. T. Freeman, E. H. Adelson, and D. J. Heeger. Shiftable multiscale transforms. *IEEE Trans. Information Theory*, 38(2):587–607, March 1992.

[19] M. Vetterli and C. Herley. Wavelets and filter banks: Theory and design. *IEEE Trans. Signal Processing*, 40(9):2207–2232, September 1992.

[20] S. P. Voukelatos and J. J. Soraghan. An ASBVQ codec for VLBR video coding. In *Proc. IEE Colloquium on Low Bit Rate Image Coding*, pages 1/1–1/5, London, 1995.

[21] J. Y. A. Wang and E. H. Adelson. Representing moving images with layers. *IEEE Trans. Image Processing*, 3(5):625–638, September 1994.

[22] M. W. Whybray and W. Ellis. H.263 - video coding recommendation for PSTN videophone and multimedia. In *Proc. IEE Colloquium on Low Bit Rate Image Coding*, pages 6/1–6/9, London, 1995.

[23] R. G. Wilson. Scale in transient detection. Technical Report 18/1993, IMSOR The Institute of Mathematical Statistics and Operations Research, The Technical University of Denmark, 1993.

[24] R. G. Wilson and G. H. Granlund. The uncertainty principle in image processing. *IEEE Trans. P.A.M.I*, 6(6):758–767, November 1984.