Part I

Curve Parameter Detection using the Radon Transform

Chapter 1

Slant Stacking

In this chapter the Radon transform will be introduced. The definition of the Radon transform is, e.g., used within seismics, where it is known as slant stacking. The discrete Radon transform is also introduced and several properties are reviewed. Several examples mostly regarding detection of straight lines from digital images are given [16].

1.1 Why Consider the Radon Transform

First a small example is presented only to motivate that the Radon transform is suited for line parameter extraction even in presence of noise. Fig. 1.1 shows an image containing three lines of which two are very close and the image has been corrupted by additive uniformly distributed noise. Fig. 1.2 shows the discrete Radon transform of the image, i.e., the figure shows a space of possible line parameters. It will be shown that the Radon transform are able to transform each of the lines into peaks positioned corresponding to the parameters of the lines. In this way, the Radon transform converts a difficult global detection problem in the image domain into a more easily solved local peak detection problem in the parameter domain, and the actual parameters can be recovered by, e.g., thresholding the Radon transform.

Note that especially in this noisy case other algorithms in general fail. An alternative could be to use a local detection algorithm, e.g. edge detection filters [17, 18], succeeded by a procedure for linking the individual pixels together, and finally to use linear regression for estimation of parameters. Algorithms of this kind have problems with intersecting lines, and in case of a high noise level, it is difficult to stabilize the edge detection filters. The Radon transform is not limited in the same sense by these problems.

1.2 Defining the (p, τ) Radon Transform

The Radon transform can be defined in different ways. The definition used, e.g., within seismics [19] is perhaps the easiest to comprehend. The Radon transform $\breve{g}(p, \tau)$ of a (continuous) two-dimensional function $g(x, y)$ is found by stacking or integrating values of g along slanted lines. The location of the line is determined from the line parameters; slope p and line offset τ.

$$\breve{g}(p, \tau) = \int_{-\infty}^{\infty} g(x, px + \tau)\ dx \tag{1.1}$$

Within seismics this linear Radon transform is also known as slant stacking or the τ-p transform. Note that different notation exists, and here the notation found in [14] is used.

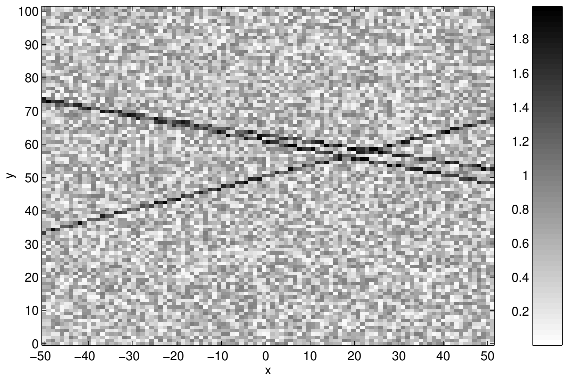

Figure 1.1 An image with two lines where uniformly distributed and uncorrelated noise has been added.

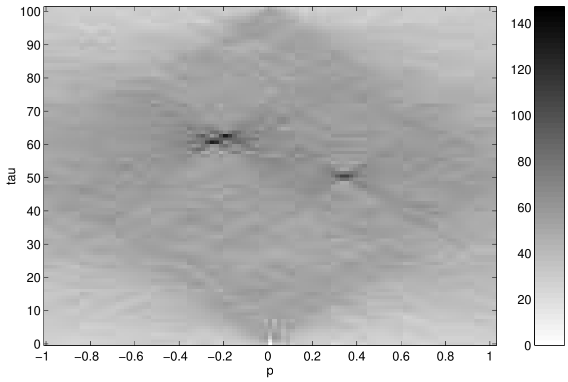

Figure 1.2 The corresponding discrete Radon transform. Each of the peaks correspond to the curves in the image. The line parameters can here be recovered by simple means, e.g., thresholding.

The Dirac delta function, $\delta(\cdot)$ (see Appendix A for basic properties), will be used extensively in the following. Using the delta function implies that slant stacking can be written as

$$\check{g}(p,\tau) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y)\ \delta(y-px-\tau)\ dx\ dy \tag{1.2}$$

The values of $\check{g}(p,\tau)$ is function in two-dimensional (p,τ)-space, i.e., Radon space or parameter domain. In principle, the two parameters do not have lower or higher limits, though, as it will be shown, discrete implementation will use a limited number of samples in both parameter directions.

1.3 Basic Properties of the (p,τ) Radon Transform

From Eqs. 1.1 or 1.2, a set of fundamental properties can be derived. It can also be noted, that it is possible to find the Radon transform analytically of some simple mathematical functions [20].

1.3.1 Linearity

From Eq. 1.1 it is directly found, that the Radon transform of a weighted sum of functions is the same weighted sum of the individually Radon transformed functions.

$$\begin{aligned} h(x,y) &= \sum_i w_i\ g_i(x,y) \ \Rightarrow \\ \check{h}(p,\tau) &= \sum_i w_i \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g_i(x,y)\ \delta(y-px-\tau)\ dx\ dy \\ &= \sum_i w_i\ \check{g}_i(p,\tau) \end{aligned} \tag{1.3}$$

This property is naturally very important.

1.3.2 Shifting

The next property is the Radon transform of a shifted function.

$$\begin{aligned}
h(x,y) &= g(x-x^*, y-y^*) \;\Rightarrow \\
\check{h}(p,\tau) &= \int_{-\infty}^{\infty} g(x-x^*, px+\tau-y^*)\; dx \\
&= \int_{-\infty}^{\infty} g(\tilde{x}, p(\tilde{x}+x^*)+\tau-y^*)\; d\tilde{x}, \text{ where } \tilde{x}=x-x^* \\
&= \check{g}(p, \tau-y^*+px^*)
\end{aligned} \tag{1.4}$$

Thus only the offset parameter is changed by shifting the function. From a geometrical point of view the result is obvious. The slope of a line cannot be altered by a translation, and the offset must be changed as shown in Eq. 1.4.

1.3.3 Scaling

A scaled function is assumed

$$\begin{aligned}
h(x,y) &= g\left(\frac{x}{a}, \frac{y}{b}\right) \text{ where } a>0 \text{ and } b>0 \\
\check{h}(p,\tau) &= \int_{-\infty}^{\infty} g\left(\frac{x}{a}, \frac{px+\tau}{b}\right)\; dx \\
&= a\int_{-\infty}^{\infty} g\left(\tilde{x}, \frac{pa\tilde{x}+\tau}{b}\right)\; d\tilde{x} \text{ where } \tilde{x}=\frac{x}{a} \\
&= a\;\check{g}\left(\frac{pa}{b}, \frac{\tau}{b}\right)
\end{aligned} \tag{1.5}$$

The result can also here be understood from the parameters of a line. It is clear that a compression in the y-direction must be followed by a compression in the line offset τ. And it is not difficult to understand that any slope will be scaled with the ratio between a and b, and so will the Radon transform.

1.3.4 The Point Source

A point source is modelled as a product of two delta functions. Initially the point source is placed in the origin of the coordinate system.

$$\begin{aligned}
g(x,y) &= \delta(x)\;\delta(y) \;\Rightarrow \\
\check{g}(p,\tau) &= \int_{-\infty}^{\infty} \delta(x)\;\delta(px+\tau)\; dx = \delta(\tau)
\end{aligned} \tag{1.6}$$

The point source could easily have been placed at any given position, but now Eq. 1.4 is used to do that

$$g(x,y) = \delta(x-x^*)\;\delta(y-y^*) \;\Rightarrow\; \check{g}(p,\tau) = \delta(\tau-y^*+px^*) \tag{1.7}$$

This result is interesting, because any function can be written as a weighted integral of point sources.

$$g(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x^*,y^*)\;\delta(x-x^*)\;\delta(y-y^*)\; dx^*\; dy^* \;\Rightarrow$$

$$\check{g}(p,\tau) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x^*,y^*)\;\delta(x-x^*)\;\delta(\tau+px-y^*)\; dx^*\; dy^*\; dx \tag{1.8}$$

$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x^*,y^*)\;\delta(y^*-\tau-px^*)\; dx^*\; dy^* \tag{1.9}$$

Eq. 1.7 and especially Eq. 1.9 demonstrates that any point of the function will contribute along an infinitely long line in the parameter domain. This aspect relates to the Hough transform, that will be further analyzed in Chapter 3.

In Fig. 1.3, the result is shown symbolically. Note that the figure shows the two dimensional function $\breve{g}(p, \tau)$ by use of color, white is used to indicate (approximately) zero value and black is a high (in this case infinite) value. This type of graph will be used many times in the following.

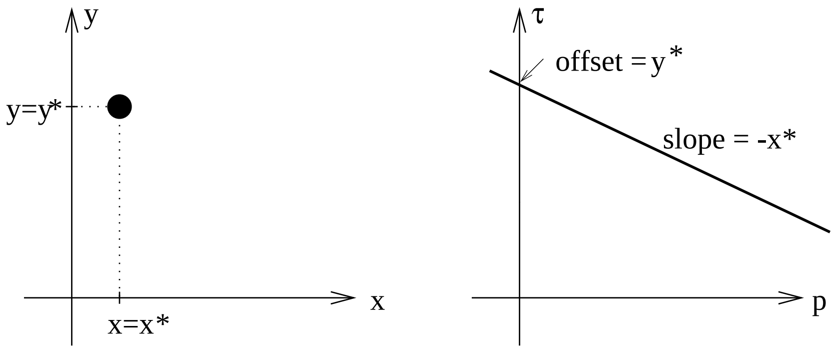

Figure 1.3 Left: A two dimensional function that only is non-zero in the point $(x, y) = (x^*, y^*)$. Right: The corresponding Radon transform (slant stacking result). Only when the Radon domain parameters matches the parameters of the line a non-zero result is found.

1.3.5 The Line

This very important property assumes a function that contains a certain line, here modelled with a delta function.

$$g(x, y) = \delta(y - p^*x - \tau^*) \tag{1.10}$$

hence the function has non-zero values only if (x, y) lies on the line with certain fixed parameters (p^*, τ^*). In this case the Radon transform is given by

$$\begin{aligned} \breve{g}(p, \tau) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \delta(y - p^*x - \tau^*)\ \delta(y - px - \tau)\ dx\ dy \\ &= \int_{-\infty}^{\infty} \delta((p - p^*)x + \tau - \tau^*)\ dx \\ &= \begin{cases} \dfrac{1}{|p - p^*|} & \text{for } p \neq p^* \\ 0 & \text{for } p = p^* \text{ and } \tau \neq \tau^* \\ \displaystyle\int_{-\infty}^{\infty} \delta(0)\ dx & \text{for } p = p^* \text{ and } \tau = \tau^* \end{cases} \end{aligned} \tag{1.11}$$

Note that for $p = p^*$ and $\tau = \tau^*$, the result is written as an infinite function integrated over an infinite interval, hence the result is infinite in that point. If, for now, the finite terms are neglected, the result is that the Radon transform of a line produces an peak (with infinite value) in the parameter domain, and the position of the peak matches the line parameters. This property has naturally been the basis of many curve parameter detection algorithms. In Fig. 1.4 the result is shown symbolically. Again, white color indicates (approximately) zero value and black color high (infinite) values.

It can be noted that in the way slant stacking is defined, there is duality between the two domains. A point in the image domain, i.e., the (x, y)-space, is transformed into a line in the

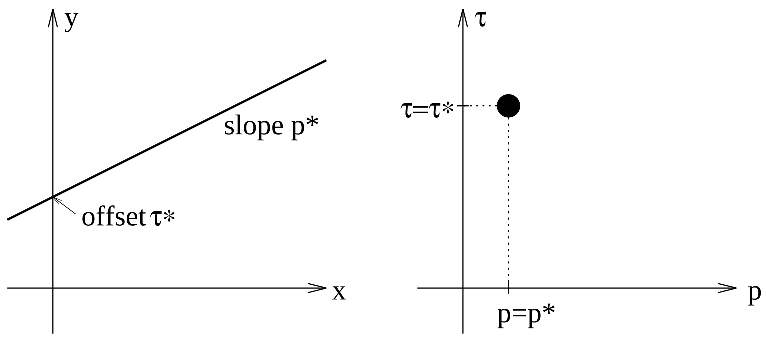

Figure 1.4 Left: A two dimensional function that is only non-zero when on the line. Right: The corresponding Radon transform (slant stacking result). When the Radon domain parameters matches the parameters of the line, a peak is found positioned at the parameters of the line in the image. The finite terms in the parameter domain are here ignored for sake of clarity.

Radon domain, and, if the finite terms are ignored, a line in the image domain will be transformed into a point in the Radon domain. This property is a direct consequence of the form of the integration kernel in Eq. 1.2. This property motivates why the Radon transform can be used for curve detection algorithms.

1.4 Discrete Slant Stacking

Given that only a subset of functions can be Radon transformed analytically, a discrete approximation to the Radon transform that transforms a digital image is very useful. Depending of the aim, e.g. speed, artifacts, or simplicity, there exist nearly as many definitions of the discrete Radon transform as people working with the Radon transform. Though an easy way is to sample the four variables linearly and only work with a limited set of samples.

$$\begin{array}{ll} x = x_m = x_{min} + m\Delta x, & m = 0,1,\ldots,M-1 \\ y = y_n = y_{min} + n\Delta y, & n = 0,1,\ldots,N-1 \\ p = p_k = p_{min} + k\Delta p, & k = 0,1,\ldots,K-1 \\ \tau = \tau_h = \tau_{min} + h\Delta\tau, & h = 0,1,\ldots,H-1 \end{array} \tag{1.12}$$

Here the x_{min} is the position of the first sample, Δx the sampling distance of x, and m is a discrete index used to number the M samples of x. These symbols will be used many times in the following.

By using Eq. 1.1, a simple and common approach to approximate the linear Radon transform is to use a sum approximation

$$\breve{g}(p_k,\tau_h) = \int_{-\infty}^{\infty} g(x, p_k x + \tau_h)\ dx \approx \Delta x \sum_{m=0}^{M-1} g(x_m, p_k x_m + \tau_h) \tag{1.13}$$

Sampling of the function $g(x,y)$ gives a digital image

$$g(m,n) = g(x_m, y_n) \tag{1.14}$$

A new symbol could be assigned to the digital image, though from the indices it should be clear whether the continuous or the discrete version is used. Likewise will the discrete Radon transform be written as

$$\breve{g}(k,h) = \breve{g}(p_k,\tau_h) \tag{1.15}$$

Hence, the discrete Radon transform can and will be presented as a digital image.

1.4.1 Nearest Neighbour Interpolation

Given the digitally sampled image $g(m,n)$ a fundamental problem arises. Eq. 1.13 requires samples not found in the digital image, because linear sampling of all variables implies that $p_k x_m + \tau_h$ in general never coincides with the samples y_n. The problem could be corrected by using, e.g., a nearest neighbour approximation in the y-direction, hence the Radon transform can be approximated by the discrete Radon transform

$$\breve{g}(k,h) = \Delta x \sum_{m=0}^{M-1} g(m, n(m;k,h)) \text{ where } n(m;k,h) = \left[\frac{p_k x_m + \tau_h - y_{min}}{\Delta y}\right] \tag{1.16}$$

where $[\cdot]$ means rounding the argument to nearest integer. Another problem is that the discrete point $(m, n(m;k,h))$ need not lie within the finite image. If the point lies outside the image the value needed could be set to zero, i.e., the point gives no contribution to the discrete Radon transform.

Note that the constant term Δx can be neglected. This term does not carry information, and is only needed if the discrete Radon transform should quantitatively approximate the (continuous) Radon transform.

It can easily be proven that the discrete Radon transform is a linear function, though the other properties shown for the continuous Radon transform can only be used to the discrete Radon transform with approximation.

Before showing a small part of an algorithm to compute the discrete Radon transform, the expression for n should be rewritten to a very simple linear form, in order to reduce the computational cost.

$$n = [n^*] \text{ and } n^* = \frac{p_k x_m + \tau_h - y_{min}}{\Delta y} = \alpha m + \beta \quad \text{where} \begin{cases} \alpha = \frac{p_k \Delta x}{\Delta y} \\ \beta = \frac{p_k x_{min} + \tau_h - y_{min}}{\Delta y} \end{cases} \tag{1.17}$$

The following program is written in pseudo-code, where all array indices start at 0. Comments to the code are shown in C++ style using //.

Algorithm 1.1 : The Discrete Slant Stacking

```
For k = 0 to K-1                                //For all values of p
  For h = 0 to H-1                              //For all values of tau
    alpha = p(k)*Delta_x/Delta_y                //Calculate digital slope
    beta = (p(k)*x_min+tau(h)-y_min)/Delta_y    //Calculate digital offset
    sum = 0                                     //Initialize sum
    For m = 0 to M-1                            //For all values of x
      n = round(alpha*m+beta)                   //Use nearest neighbour approx.
      If n≥0 and n<N                            //Check if pixel lies in image
        sum=sum+g(m,n)                          //If so, then increment sum
      End
    End
    g_radon(k,h)=Delta_x*sum                    //Store Radon value
  End
End
```

In Algorithm 1.1 p_k has been replaced by p(k), and the initialization of the arrays has been removed in order to reduce the size of the pseudo-code. Note that actions have to be taken to assure that pixel going into the sum actually lies in the image. In the time-consuming part of the loop, i.e., within the m-loop, optimization should be done very carefully depending on the platform (computers or similar digital equipment), used for computing the discrete Radon transform. A trick that can boost the performance considerably on most platforms, is to directly calculate which values of m, that gives pixels lying within the image.

$$0 \leq n = [\alpha m + \beta] \leq N-1 \;\Rightarrow \begin{cases} -\frac{\beta+\frac{1}{2}}{\alpha} \leq m < \frac{N-\frac{1}{2}-\beta}{\alpha} & \text{if } \alpha > 0 \\ \frac{N-\frac{1}{2}-\beta}{\alpha} \leq m < -\frac{\beta+\frac{1}{2}}{\alpha} & \text{if } \alpha < 0 \end{cases} \Rightarrow \tag{1.18}$$

$$m_{min} = \max\left\{0, \left\lceil -\frac{\beta+\frac{1}{2}}{\alpha} \right\rceil\right\} \text{ and } m_{max} = \min\left\{M-1, \left\lfloor \frac{N-\frac{1}{2}-\beta}{\alpha} \right\rfloor\right\} \text{ if } \alpha > 0$$

$$m_{min} = \max\left\{0, \left\lceil \frac{N-\frac{1}{2}-\beta}{\alpha} \right\rceil\right\} \text{ and } m_{max} = \min\left\{M-1, \left\lfloor -\frac{\beta+\frac{1}{2}}{\alpha} \right\rfloor\right\} \text{ if } \alpha < 0 \tag{1.19}$$

where $\lfloor\cdot\rfloor$ and $\lceil\cdot\rceil$ rounds to the nearest lower and higher integer, respectively. So instead of using the full interval of m-values, only m from m_{min} to m_{max} should be used, in order to avoid the time consuming testing of valid limits.

Another issue of optimization concern the expression `n=round(alpha*m+beta)`, which could be replaced by an initial `nfloat=beta+m_min*alpha` (if using the alteration described in Eq. 1.19) and then within the loop `n=round(nfloat)` and the line `nfloat=nfloat+alpha`. Hence, multiplications can be avoided in the (time) important part of the program. The extra accumulation of noise will normally only cause problems if using a very poor (small) representation of the floating point numbers or using extremely large images (hardly the case). This alteration might give an additional speedup, but it depends on the platform, i.e., the memory access rate, and the adding/multiplication rate of the platform.

1.4.2 Linear Interpolation

So far only a nearest neighbour approximation has been considered. It can also be chosen to use linear interpolation in the y-direction of the image.

$$\breve{g}(k,h) = \Delta x \sum_{m=0}^{M-1} ((1-w)g(m,n) + wg(m,n+1)) \tag{1.20}$$

$$\text{where } n^* = \frac{p_k x_m + \tau_h - y_{min}}{\Delta y} = \alpha m + \beta,\ n = \lfloor n^* \rfloor,\ \text{and } w = n^* - n \tag{1.21}$$

It can be noted that Eq. 1.20 requires the sum of twice as many samples compared to Eq. 1.16. In general, a better (higher order) interpolation scheme implies more computational work. This tradeoff must be determined by sampling properties and the time available for the algorithm.

As shown in the following equation, linear interpolation does not imply much change to the optimization strategy of computing the range of m-values, corresponding to pixels in the image.

$$0 \leq n = \lfloor \alpha m + \beta \rfloor \leq N-2 \;\Rightarrow \begin{cases} -\frac{\beta}{\alpha} \leq m < \frac{N-1-\beta}{\alpha} & \text{if } \alpha > 0 \\ \frac{N-1-\beta}{\alpha} \leq m < -\frac{\beta}{\alpha} & \text{if } \alpha < 0 \end{cases} \Rightarrow \tag{1.22}$$

$$m_{min} = \max\left\{0, \left\lceil -\frac{\beta}{\alpha} \right\rceil\right\} \text{ and } m_{max} = \min\left\{M-1, \left\lfloor \frac{N-1-\beta}{\alpha} \right\rfloor\right\} \text{ if } \alpha > 0$$

$$m_{min} = \max\left\{0, \left\lceil \frac{N-1-\beta}{\alpha} \right\rceil\right\} \text{ and } m_{max} = \min\left\{M-1, \left\lfloor -\frac{\beta}{\alpha} \right\rfloor\right\} \text{ if } \alpha < 0 \tag{1.23}$$

In Algorithm 1.2, it is shown how to compute the Discrete Radon transform using the linear interpolation approximation. Note also, that Eq. 1.23 should be used to improve the performance.

Algorithm 1.2 : The Discrete Slant Stacking with Linear Interpolation

```
For k = 0 to K-1                                  //For all values of p
  alpha = p(k)*Delta_x/Delta_y                    //Calculate digital slope
  For h = 0 to H-1                                //For all values of tau
    beta = (p(k)*x_min+tau(h)-y_min)/Delta_y      //Calculate digital offset
    Set m_min and m_max using Eq. 1.23            //Not shown
    sum = 0                                       //Initialize sum
    For m = m_min to m_max                        //For all valid values of x
      nfloat =alpha*m+beta                        //Use nearest neighbour approx.
      n=floor(nfloat)                             //Calculate lower index
      w=nfloat-n                                  //Calculate weight factor
      sum=sum+g(m,n)*(1-w)+g(m,n+1)*w             //Increment sum
    End
    g_radon(k,h)=sum*Delta_x                      //Update matrix element
  End
End
```

In Sections 1.5 and 1.6, examples will be given in order to show differences between the simple nearest neighbour approximation, given in Eq. 1.16, the linear interpolation approximation, and a sinc interpolation Radon transform.

1.4.3 Sinc Interpolation

Assuming that the function $g(x,y)$ was sampled according to the Whittaker-Shannon sampling theorem [21], then the function can be recovered from the digital image $g(m,n)$ by convolution with a sinc function,

$$g(x,y) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} g(m,n)\ \frac{\sin(\frac{\pi}{\Delta x}(x-x_m))}{\frac{\pi}{\Delta x}(x-x_m)}\frac{\sin(\frac{\pi}{\Delta y}(y-y_n))}{\frac{\pi}{\Delta y}(y-y_n)} \tag{1.24}$$

It is not quite true that $g(x,y)$ is exactly recovered because the summations should in principle be infinitely long, but these samples are assumed to have value zero.

Instead of approximating the Radon transform to a digital image, a new way of analyzing the Radon transform of a digital image is proposed [16].

Eq. 1.1 will now be applied to Eq. 1.24 in order to gain information of the Radon transform and derive a sampling criterion for the parameter domain

$$\breve{g}(p,\tau) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} g(m,n)\ I(p,\tau,x_m,y_n) \tag{1.25}$$

$$I(p,\tau,x_m,y_n) = \int_{-\infty}^{\infty} \frac{\sin(\frac{\pi}{\Delta x}(x-x_m))}{\frac{\pi}{\Delta x}(x-x_m)}\frac{\sin(\frac{\pi}{\Delta y}(px+\tau-y_n))}{\frac{\pi}{\Delta y}(px+\tau-y_n)}\ dx \tag{1.26}$$

$$= \frac{\Delta x}{\pi}\int_{-\infty}^{\infty}\frac{\sin(t)}{t}\frac{\sin(\alpha t+\gamma)}{\alpha t+\gamma}\ dt \text{ where } \begin{cases} t = \frac{\pi}{\Delta x}(x-x_m) \\ \alpha = \frac{\Delta x}{\Delta y}p \\ \gamma = \frac{\pi}{\Delta y}(px_m+\tau-y_n)\end{cases} \tag{1.27}$$

The last integral can be interpreted as a convolution. Using that the Fourier transform of a sinc-function is a square, and a square multiplied with a square gives a square, implies that the last integral is a sinc function.

$$I(p,\tau,x_m,y_n) = \frac{\Delta x}{\gamma}\sin\left(\gamma\min\left\{1,\frac{1}{|\alpha|}\right\}\right) \quad\Rightarrow \tag{1.28}$$

$$\check{g}(p,\tau) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} g(m,n)\ \Delta x\frac{\sin\left(\gamma\min\left\{1,\frac{1}{|\alpha|}\right\}\right)}{\gamma} \tag{1.29}$$

The result is a sinc-interpolation discrete Radon transform, but right now the expression will be used to analyze the sampling properties of the Radon transform.

1.4.4 Sampling Properties of the Discrete Radon Transform

Assuming that the absolute slope $|p|$ is limited (in Section 1.7, it will be shown that this is reasonable.), i.e., $|\alpha| < 1$, implies that the Radon transform is determined by the function $\sin(\gamma)/\gamma$, which, as a function of γ has an upper limit frequency of $\frac{1}{2\pi}$, thus if γ should be sampled, it should be done with a rate faster than

$$\Delta\gamma = \pi \tag{1.30}$$

A discrete version of the Radon transform demand that the two parameters have to be sampled, and Eq. 1.30 is used for each of the Radon parameters, i.e., p and τ.

$$\Delta\gamma = \left|\frac{\partial\gamma}{\partial\tau}\right|\Delta\tau = \frac{\pi}{\Delta y}\Delta\tau \le \pi \Rightarrow \Delta\tau \le \Delta y \tag{1.31}$$

$$\Delta\gamma = \left|\frac{\partial\gamma}{\partial p}\right|\Delta p = \frac{\pi}{\Delta y}|x_m|\ \Delta p \le \pi \Rightarrow \Delta p \le \frac{\Delta y}{\max|x_m|} \tag{1.32}$$

Note that a conservative evaluation is used to assure that the limit in Eq. 1.32 is valid for all values of x_m. The last equation shows that the sampling distance Δp must be small if $\max|x_m|$ is large. Regarding the sampling of p, it is optimal to choose symmetrical sampling points of x, hence minimizing $\max|x_m|$.

$$x_{min} = -x_{max} \;\Rightarrow\; x_{min} = -\frac{M-1}{2}\Delta x \tag{1.33}$$

Even if x_{min} is fixed by the underlying physics, this little trick can be used due to the shifting property of the Radon transform, shown in Eq. 1.4.

Another reasonable way of defining bounds on the sampling distances in the parameter domain is to demand that changing one of the Radon parameters with its sampling interval, cannot lead to more than a pixel of difference in the image, otherwise some of the pixels might not add weight to the parameter domain, hence information is lost. In other terms

$$p : \max\{|(p+\Delta p)x + \tau - (p\,x+\tau)|\} = \max\{|\Delta p\,x|\} \le \Delta y \tag{1.34}$$

$$\tau : \max\{|p\,x + (\tau+\Delta\tau) - (p\,x+\tau)|\} = \Delta\tau \le \Delta y \tag{1.35}$$

hence, the interesting conclusion is that the two approaches give the same result.

Assuming that the task is to identify line parameters, then the sampling distances Δp and $\Delta\tau$ can be set with Eq. 1.32 and Eq. 1.31, respectively. The two parameters τ_{min} and H can be set from a requirement that, e.g., half of the samples in the discrete Radon transform should lie within the image. If assuming Eq. 1.33, it is found that $H = N$ and $\tau_{min} = y_{min}$, i.e., the sampling of τ follows the sampling of y. The two parameters p_{min} and K must be set to cover the interval of expected slopes, e.g., if the underlying physics can be used to predict the resulting interval of slopes, this should be incorporated. Later, in Section 1.7, it will be shown that there are limitations to this strategy.

The conclusion is that with a given digital image the parameter domain must be sampled sufficiently dense in order to avoid aliasing problems. The limits derived in Eqs. 1.32 and 1.31 will now be tested with discrete implementations of the Radon transform.

1.5 Discrete Radon Transform of a Discrete Line

An digital image was generated with $101 * 101$ samples. The synthetic image contains two lines as seen from Fig. 1.5 and the corresponding discrete Radon transform is shown in Fig. 1.6. Sampling parameters for this example can be found in Table 1.1. Here Δp and $\Delta\tau$ is chosen according to Eqs. 1.31 and 1.32.

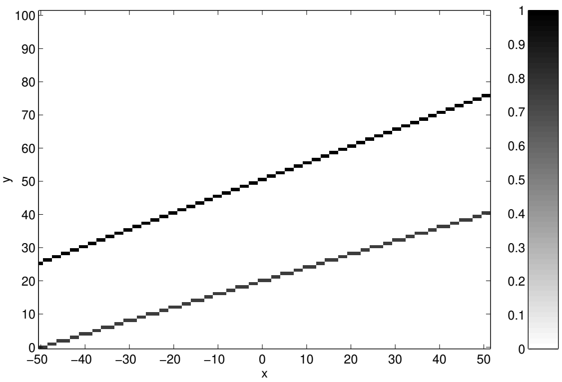

Figure 1.5 The image containing two lines.

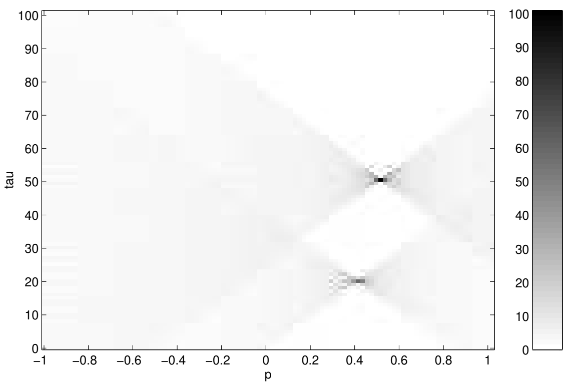

Figure 1.6 The corresponding discrete Radon transform (slant stacking).

Image domain		Parameter domain	
Parameter	Value	Parameter	Value
M	101	H	101
N	101	K	101
Δx	1	Δp	0.02
Δy	1	$\Delta\tau$	1
x_{min}	-50	p_{min}	-1
y_{min}	0	τ_{min}	0

Table 1.1 Sampling parameter settings.

The linearity of the Radon transform implies that two peaks should be expected in the discrete parameter domain, which is shown in from Fig. 1.6. From the position of the peaks, it is clear that the line parameters are $(p_1 = 0.5, \tau_1 = 50)$ and $(p_2 = 0.4, \tau_2 = 20)$.

Now the sampling parameters of the parameter domain are changed, in order to zoom in around one of the peaks. Sampling parameters are shown in left side of Table 1.2 and the Radon Transform is shown in Fig. 1.7. Now the peak is very broad, which suggests that the sampling of the parameter domain is unnecessary dense. It can also be noted that the number of samples in the discrete parameter domain has been maintained so the computational cost is (almost) the same as in Fig. 1.6.

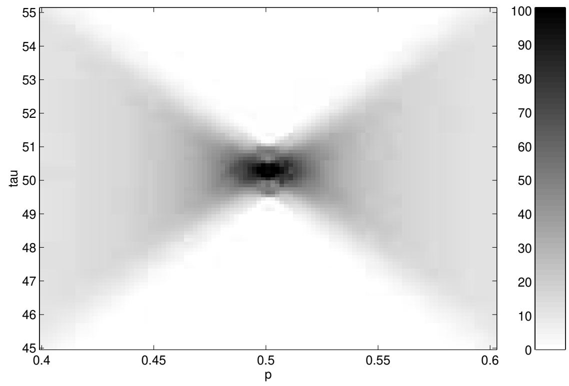

Figure 1.7 The discrete Radon transform of the image shown in Fig. 1.5. Dense sampling of the parameter domain has been used.

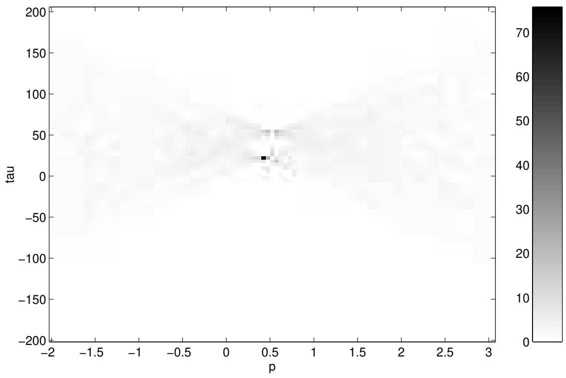

Figure 1.8 The discrete Radon transform of the image shown in Fig. 1.5. Very sparse sampling of the parameter domain has been used.

In Fig. 1.8 the parameter domain has been sampled very sparsely, and note how the upper peak has vanished. The sampling parameters Δp and $\Delta\tau$ are here set to strongly under-sample the parameter domain, i.e., the limits found in Eqs. 1.32 and 1.31 are strongly violated. This example shows how the sampling parameters of the parameter domain should be set. If the parameter domain is sampled too densely, redundant information is acquired. On the other hand sampling the parameter domain very sparsely implies important information *might* be lost. Here information about one of the lines is lost, but the parameters of the other line is maintained.

Parameter Domain (zoom in)		Parameter domain (zoom out)	
Parameter	Value	Parameter	Value
H	101	H	101
K	101	K	101
Δp	0.002	Δp	0.05
$\Delta\tau$	0.1	$\Delta\tau$	4
p_{min}	0.4	p_{min}	-2
τ_{min}	45	τ_{min}	-200

Table 1.2 Sampling parameter settings.

1.5.1 Comparison of Different Interpolation Methods

The sampling of the parameter domain also depends on the type of interpolation being used, and the type of image. If the image $g(m,n)$ does not include very sharp edges with many high frequency components, a nearest neighbour approximation might be sufficient, but an image containing a line, like Fig. 1.9 implies that interpolation method affect the result near the peak in the discrete parameter domain.

In the following the discrete parameter domain is found by use of the three approximations, i.e., the nearest neighbour interpolation in the image (Eq. 1.16), the linear interpolation (Eq. 1.20), and finally the sinc-interpolation (Eq. 1.29). In all three cases the line parameters, corresponding to Fig. 1.9, are found.

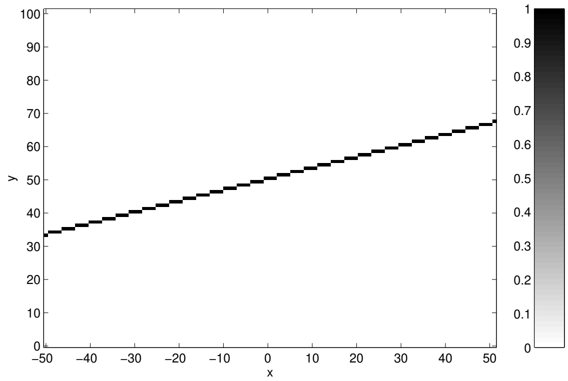

Figure 1.9 An image with one line.

Figs. 1.10, 1.12, and 1.14 use the same color scaling to show the discrete parameter domains. It can be seen that the difference is limited, but the value in the peak is lower in the last two images, compared to the first (nearest neighbour). In Figs. 1.11, 1.13, and 1.15 the discrete parameter domains are shown using a dense sampling in the area of the peak. The images have a common color scaling. It can be seen that the peak in Fig. 1.11 is very narrow, though the largest value (≈ 101) is much higher compared to Fig. 1.13, and 1.15. This indicates that using nearest neighbour approximation requires a dense sampling of the parameter domain in order to assure that the peak is found. Using linear interpolation with this sparse image implies that the peak value will be rather low (≈ 78) but broad. This can be understood from Fig. 1.9, where at least one of the two image points going into Eq. 1.20 is identically zero at any m, so the weight factors will lower the result in the peak.

It is important is that a peak is generated in the discrete parameter domain of a sufficient size, and the adequate sampling of the parameter domain should be compared to the theoretically derived ones. From the Fig. 1.15 a somewhat higher peak value (≈ 89) is found, and the effective size of the peak resembles that found with linear interpolation. For Figs. 1.10, 1.12, and 1.14 the sampling parameters of the image and parameter domain can be found in Table 1.1, and for Figs. 1.11, 1.13, and 1.14 in Table 1.3. Note the special form of the parameter domain in the nearest neighbour case. It is a digital effect, and will be found when the line slope can be written as a fraction with small numbers, e.g., $\frac{1}{3}$, $\frac{1}{2}$, or $\frac{3}{4}$.

The computational complexity of the three approximations is also very interesting. Using the symbols defined in Eq. 1.12 the computational complexity of the three ways of implementing the discrete Radon transform is approximately given by

$$\begin{aligned} \mathcal{O}_{\text{Nearest Neighbour}} &= \mathcal{O}(K\ H\ M) &&\approx \mathcal{O}(M^3) \\ \mathcal{O}_{\text{Linear Interpolation}} &= \mathcal{O}(K\ H\ M) &&\approx \mathcal{O}(M^3) \\ \mathcal{O}_{\text{Sinc Neighbour}} &= \mathcal{O}(K\ H\ M\ N) &&\approx \mathcal{O}(M^4) \end{aligned} \tag{1.36}$$

where $\mathcal{O}(\cdot)$ is the order function, and some lower order terms have been skipped. Assuming M in the order of 100, then the computational complexity of the sinc interpolation is about 100

Parameter	Value
H	101
K	101
Δp	0.0005
$\Delta\tau$	0.02
p_{min}	0.31
τ_{min}	49

Table 1.3 Sampling parameter settings for Figs. 1.11, 1.13, and 1.15.

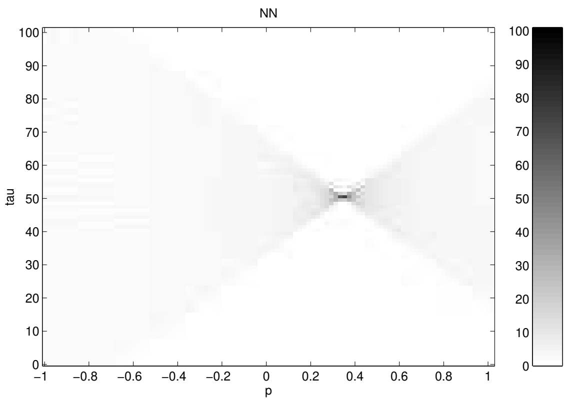

Figure 1.10 The discrete Radon transform using nearest neighbour approximation.

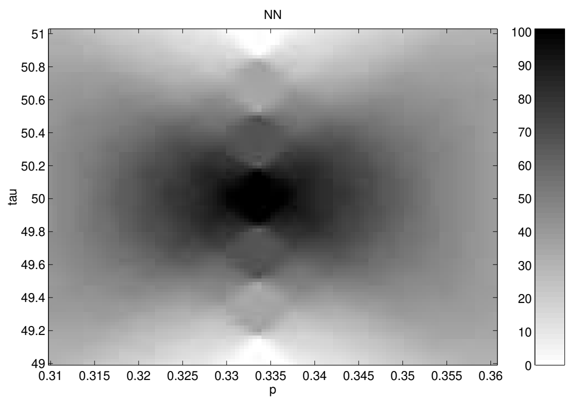

Figure 1.11 The discrete Radon transform in the area of the peak using nearest neighbour approximation.

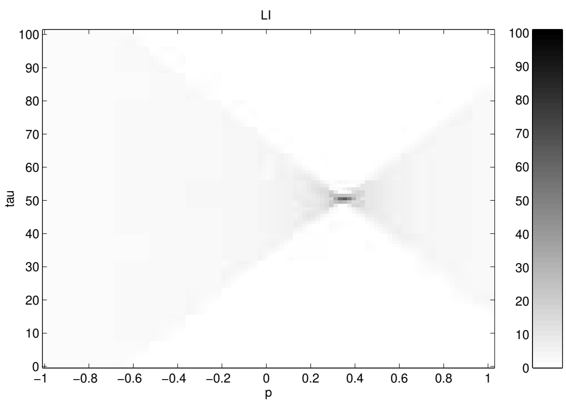

Figure 1.12 The discrete Radon transform using linear interpolation.

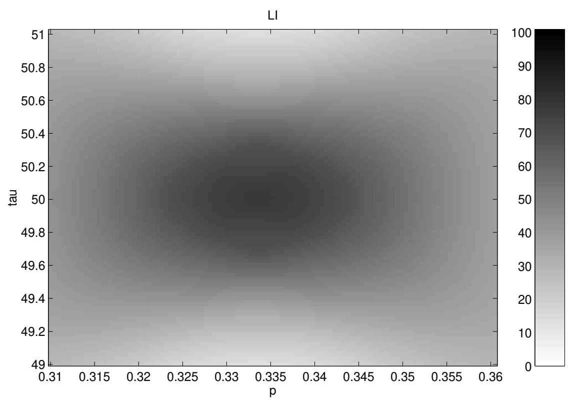

Figure 1.13 The discrete Radon transform in the area of the peak using linear interpolation.

times higher than the other two methods. Note that linear approximation is slower than nearest neighbour approximation, due to the summation of two samples for every m.

In Table 1.4 the time needed for computing the discrete parameter domain is listed, corresponding to the three different approximations. The algorithms have been implemented in the C-language. As it can be seen a speedup-factor of approximately 3.5 has been gained by optimizing the code as described. The time used with linear approximation can be compared to the (not optimized) nearest neighbour time of 1.4, so a minor difference is found there. As could be expected from the Eq. 1.36 the sinc approximation is very slow, due to the fact that this al-

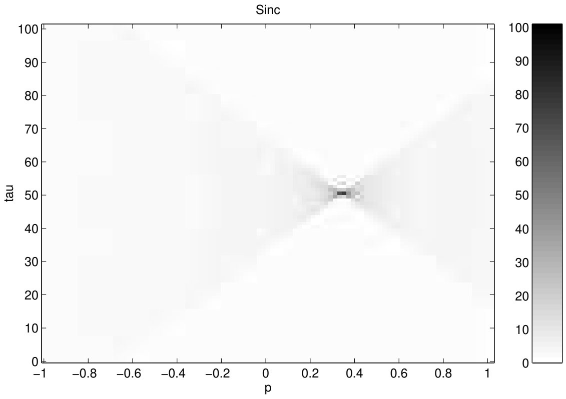

Figure 1.14 The discrete Radon transform using the sinc expansion. Note the ripples away from the peak.

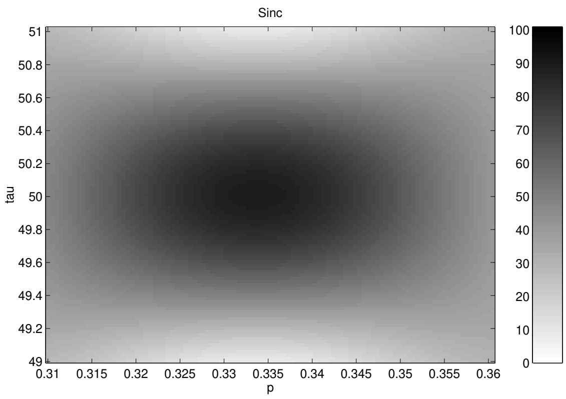

Figure 1.15 The discrete Radon transform in the area of the peak using the sinc expansion.

gorithm has to calculate a sinc function for every (m, n, k, h). No flop-ratings are given because the performance is not totally dominated by the arithmetic statements, but also on the memory access-rate and the number of comparisons needed in the algorithm.

Method	Time
Nearest Neighbour	1.4 sec
Nearest Neighbour (optimized)	0.45 sec
Linear Interpolation	1.7 sec
Sinc Interpolation	287 sec

Table 1.4 Time for calculating the discrete Radon transform for the three types of approximation, corresponding to Figs. 1.11, 1.13, and 1.15. A 120 MHz Pentium PC has been used for the measurements.

One way of analyzing the resolution of the discrete Radon transform is to visualize the shape of the discrete parameter domain with a threshold value corresponding to, e.g., half of the potential max of the parameter domain, i.e, in this case 50. This is a kind of fwhm-measure (full width half max). Figs. 1.16, 1.17, and 1.18 show the parameter domain using a threshold level of 50. From the figures it can be seen that the extension of the peak, which could be interpreted as the resolution of the discrete Radon transform in the τ-direction, is very close to the sampling distances derived in Eq. 1.31, and the resolution in the p-direction is approximately 3 time the one derived in Eq. 1.32. But it should be noted that this kind of analysis depends on the actual threshold level. Regarding this example the conclusion is that the simple approximations actually produce good results and much faster than using the sinc interpolation.

1.6 Discrete Radon Transform of Points

In this subsection, the discrete Radon transform of individual points will be analyzed. Images with an increasing number of points lying on a line will be analyzed in order to see the results of Section 1.5 from another point of view.

Fig. 1.19 is generated from Fig. 1.5 (an image with two lines), and the image only has eight non-zero pixels lying on two lines. The pixel values are now attenuated horizontally across the image (lowest pixel value in the right side of the image). In Fig. 1.20 the corresponding discrete

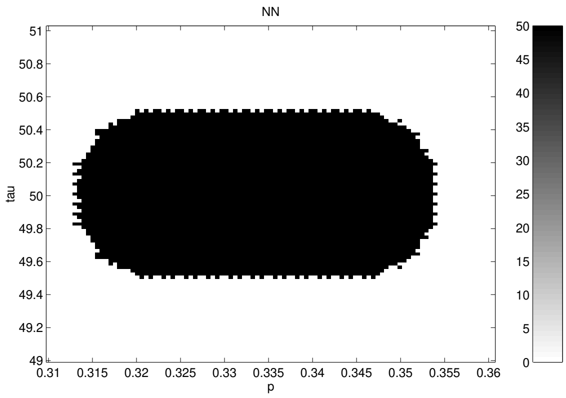

Figure 1.16 The thresholded discrete Radon transform (nearest neighbour).

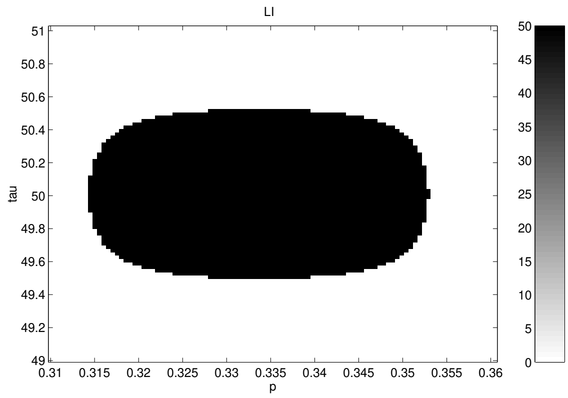

Figure 1.17 The thresholded discrete Radon transform (linear interpolation).

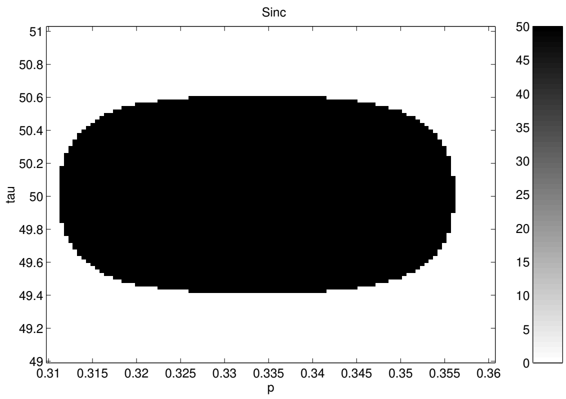

Figure 1.18 The thresholded discrete Radon transform (sinc interpolation).

Radon transform is shown using nearest neighbour interpolation. As could be expected from Eq. 1.7 each of the eight pixels is transformed into a line in the parameter domain. Due to the different values of the non-zero pixels in the image, it is possible to identify which non-zero pixel in the image corresponds to a certain line in the parameter domain. Another important issue is that the lines found in the parameter domain intersect, and amplify in two points, corresponding to the underlying line-parameters. Sampling parameters for this example can be found in Table 1.1.

Fig. 1.21 shows a similar image generated having ten points on each line. From the corresponding discrete Radon transform, shown in Fig. 1.22, the underlying line parameters can be extracted, due to clearly marked position of the peaks in the parameter domain. Finally Fig. 1.23 shows an image with two complete lines, and in Fig. 1.24 are shown the corresponding parameter domain. The images have been scaled individually corresponding to the minimal and maximal pixel value, hence it can be seen that the peak value in the last case is very high compared to the two others.

It has been demonstrated, that each point in the image is transformed into a line in the discrete parameter domain, where the offset and slope (in the parameter domain) vary according to the position of the pixel in the image. This leads to an important issue, namely that the discrete Radon transform of any given pixel cannot be confined in the finite, thus truncated, parameter domain. In this way (perhaps important) information will get lost using discrete slant stacking.

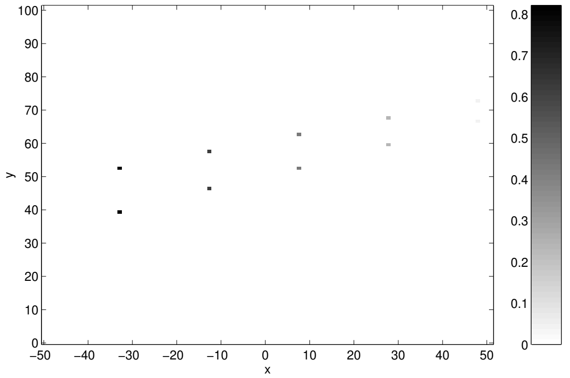

Figure 1.19 Image with ten points lying on two lines.

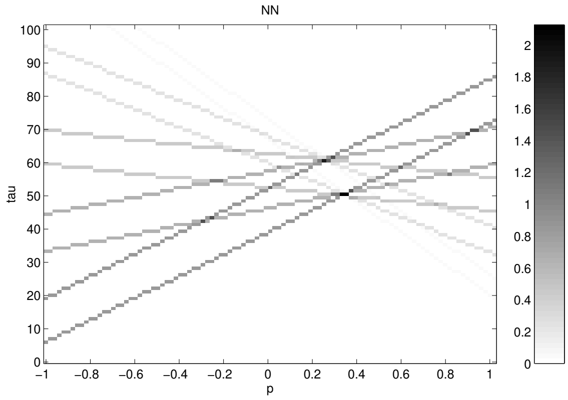

Figure 1.20 The corresponding discrete Radon domain.

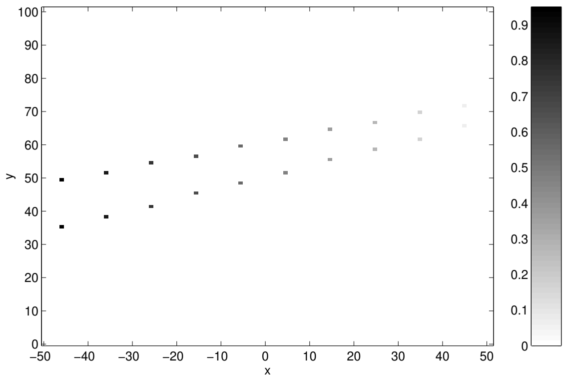

Figure 1.21 Image with 20 points lying on two lines.

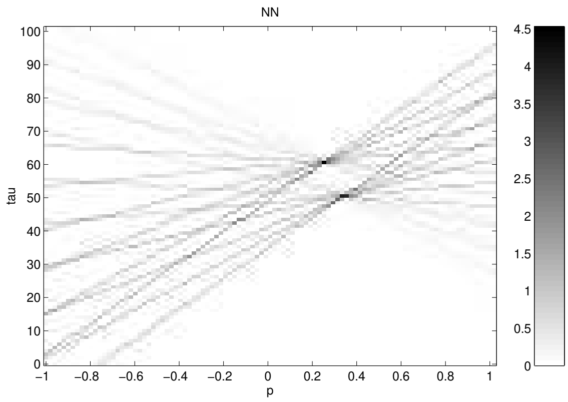

Figure 1.22 The corresponding discrete Radon domain.

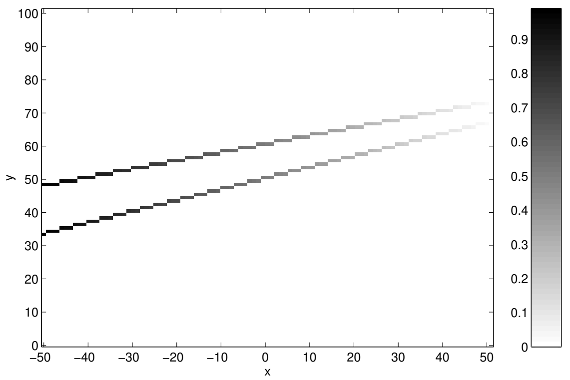

Figure 1.23 Image with 202 points lying on two lines.

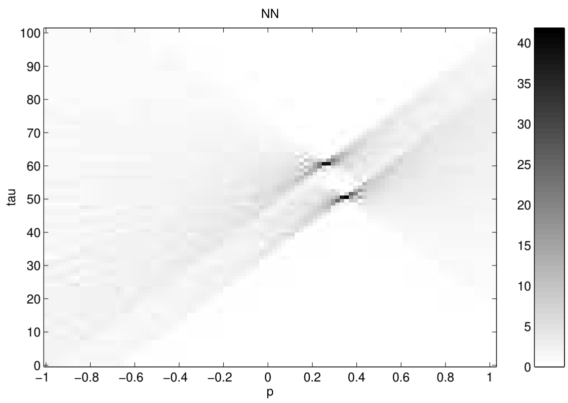

Figure 1.24 The corresponding discrete Radon domain.

1.7 Slant Stacking and Images with Steep Lines

It has been demonstrated that the parameter domain must be sampled sufficiently dense, in order to assure that the discrete approximations to the Radon transform does not cause aliasing problems.

Another problem arises when images include one or more lines with high slope (nearly vertical lines). First assume an image with two lines as shown in Fig. 1.25. One line has a slope of $\frac{1}{9}$ and the other a slope of 9. The corresponding discrete Radon transform is shown in Fig. 1.26, where the parameter p has been limited between -1 and 1. The parameters of the first line is clearly identified, but due to the high slope and the truncated parameter domain, the other line is not detected. This illustrates that only lines with parameters that lie within the limits of the parameter domain can be identified. It is a very general property of the discrete Radon transform that a parameter domain with many samples is required when no prior information is provided. If, e.g., the slopes of the lines are limited between -0.1 and 0.1, then this should be used to limit the discrete parameter domain, in order to reduce the computational work.

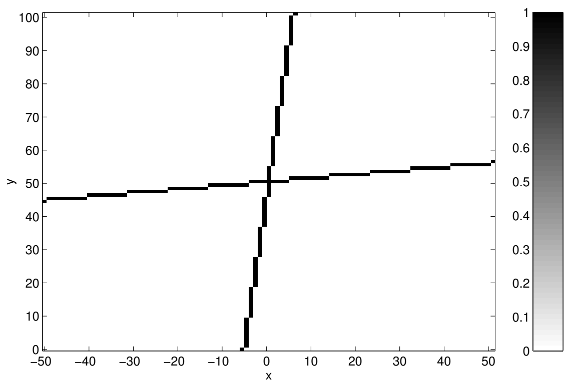

Figure 1.25 An image with two lines having a slope of $\frac{1}{9}$ and 9 respectively.

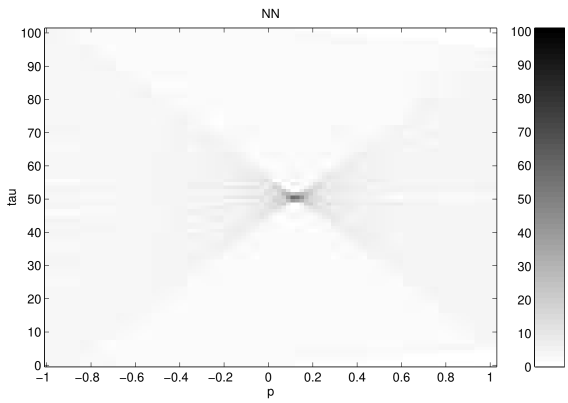

Figure 1.26 The corresponding discrete parameter domain, with the slope parameter p limited between -1 and 1.

One way to enable detection of both lines, is to increase K, i.e., the number of samples in the p-direction, in order to have $p_{max} = p_{min} + (K-1)\Delta p > 9$. This is possible, but the computational cost would be unnecessary high, due to the large number of samples in the discrete parameter domain.

Another problem is that the peak value in the discrete parameter domain should be around 100, (or perhaps somewhat lower as shown in Subsection 1.5.1), due to the fact that the each line is made of 101 non-zero samples with value of 1. Assume that the parameter domain was expanded to include the very steep line, then the value of the discrete Radon transform at the peak would only be in the region of 10. This problem is indicated in Fig. 1.27. The dashed line show the line under which the values are summed. But stepping forward in m, in this case only four samples contribute to the discrete Radon transform, and several samples between the four are skipped. In Fig. 1.25 only around 10 pixels would contribute to the discrete Radon transform in the region of the parameter of the second line parameters, hence the peak value would be much lower than possible from the image.

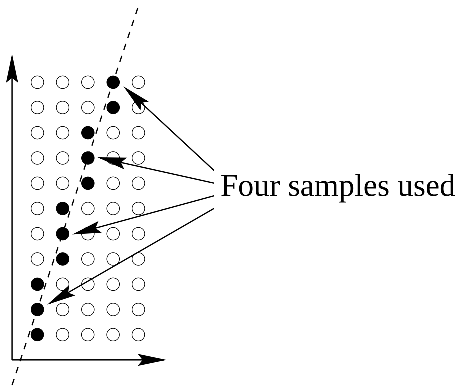

Figure 1.27 Image with steep line. The filled circles are pixels with a value of one, and the others have a value of zero. The dashed line indicate the line of integration. Note how only four samples go into the discrete Radon transform and several pixels are skipped.

Another serious reason for concern, comes from the analysis of functions with vertical lines. The corresponding Radon transform can be found from the analysis of a (continuous) vertical line.

$$g(x, y) = A\,\delta(x - x^*) \;\Rightarrow\; \breve{g}(p, \tau) = A \int_{-\infty}^{\infty} \delta(x - x^*)\; dx = A \tag{1.37}$$

where it should be noted that the Radon transform does not depend on the position of the line x^*. This implies that this information is not maintained in the parameter domain, hence the position of vertical lines cannot be detected using slant stacking.

A way to overcome the described problems is to compute two parameter domains. The first $\breve{g}(p, \tau)$ is restricted to $-p_{lim} < p \leq p_{lim}$, and the other $\breve{g}(r, \eta)$ is used to manage the remaining line orientations.

$$\breve{g}(p, \tau) = \int_{-\infty}^{\infty} g(x, p\,x + \tau)\; dx \tag{1.38}$$

$$\breve{g}(r, \eta) = \int_{-\infty}^{\infty} g(r\,y + \eta, y)\; dy \tag{1.39}$$

The two ways of describing lines are related

$$y = p\,x + \tau \;\Rightarrow\; x = r\,y + \eta \text{ where } \begin{cases} r = \frac{1}{p} \\ \eta = -\frac{\tau}{p} \end{cases} \tag{1.40}$$

Thus if $-p_{lim} \leq p \leq p_{lim}$ then the other slope parameter should be bounded by $-\frac{1}{p_{lim}} \leq r \leq \frac{1}{p_{lim}}$. A reasonable value of the limiting slope is

$$p_{lim} = \frac{\Delta y}{\Delta x} \tag{1.41}$$

This limit assures that the problem illustrated in Fig. 1.27 is eliminated. The change in vertical direction when moving to the next m, during discrete Radon transform, will be less than 1. From Eq. 1.17, this could be interpreted as the digital slope of the line, and $p\frac{\Delta x}{\Delta y}$ lies between -1 and 1. This way of choosing p_{lim} also fulfills the assumption in Eq. 1.30 that $|\alpha| > 1$.

Note that a discrete implementation of the Radon transform defined in Eq. 1.39 does not imply additional programming. By applying the discrete Radon transform to the transpose of the digital image, i.e., $h(m, n) = g(n, m)$, and using Eq. 1.40 to convert the sampling parameters, then the obtained discrete parameter domain estimates the wanted Radon transform.

1.8 Detection of Lines Convolved with a Wavelet

So far only positive valued images have been considered. In this subsection the restriction will be abandoned, by assuming a line convolved with a wavelet. Here a wavelet $\psi(x)$ is defined as a (short) signal centered around $x = 0$. Note that this definition does not agree totally with the one used in wavelet analysis. In wavelet analysis, the wavelet is also restricted to $\int \psi(x)\, dx = 0$. This setup is relevant in connection with filtering of edges, where the filtering output is both positive and negative.

The model considered is

$$g(x, y) = \psi(y - p^* x - \tau^*) \tag{1.42}$$

hence the line has certain parameters (p^*, τ^*). Comparing Eq. 1.42 with Eq. 1.10, it can be noted that the wavelet $\psi(\cdot)$, has been convolved in the y-direction. The wavelet could also have been convolved perpendicularly to the line, and this could be included in the wavelet as a angular dependent scaling parameter. The corresponding Radon transform is given by

$$\breve{g}(p, \tau) = \begin{cases} \displaystyle\int_{-\infty}^{\infty} \psi(\tau - \tau^*)\, dx & \text{for } p = p^* \\ \displaystyle\frac{1}{|p - p^*|} \int_{-\infty}^{\infty} \psi(\tilde{x})\, d\tilde{x} & \text{for } p \neq p^* \end{cases} \tag{1.43}$$

This result is interesting in various ways. In order to have a peak in the parameter domain at the correct parameters, the wavelet should have a global maximum when the argument is zero, i.e., corresponding to the position of the line.

$$\psi(0) > \psi(x)\ \forall x \neq 0 \tag{1.44}$$

Another interesting result is that when $p \neq p^*$ the result only depends on the integral of the wavelet. Consider that the wavelet is chosen to $\psi(x) = \delta''(x)$. The integral (from minus infinity to infinity) of this function is 0, and the value of $\delta''(\cdot)$ is infinite, hence the parameter domain would be infinite at the position of the line parameters $(p, \tau) = (p^*, \tau^*)$, and zero otherwise.

In Fig. 1.28 an image containing two lines is created using the digital wavelet $\{-0.5, 1.0, 0.5\}$. It is found that the difference between the discrete parameter domains obtained by using nearest neighbour, linear interpolation, and the sinc expansion approximation is limited, hence Fig. 1.29 only shows the corresponding discrete Radon transform using the nearest neighbour approximation. Two peaks are found corresponding to the line parameters, and the values in the parameter domain are in fact small away from the peaks. Note also the negative value, just over and under the (positive) peak.

1.9 Summary

Slant stacking or τ-p transform was defined as it is common within seismics. The transform was analyzed as a special case of the Radon transform. Discrete approximations were derived, and sampling properties were derived and analyzed by introducing a new sinc expansion strategy. It was demonstrated for curve detection purposes, that a nearest neighbour approximation is adequate if sampling of the parameter domain is sufficiently dense. It was also demonstrated that linear interpolation gave approximately the same parameter domain as the one found with the sinc interpolation strategy.

It has been shown that slant stacking is capable of detecting curve parameters if the slope of the line is below a certain limit related to the sampling parameters. A simple cure for this limitation was also reviewed.

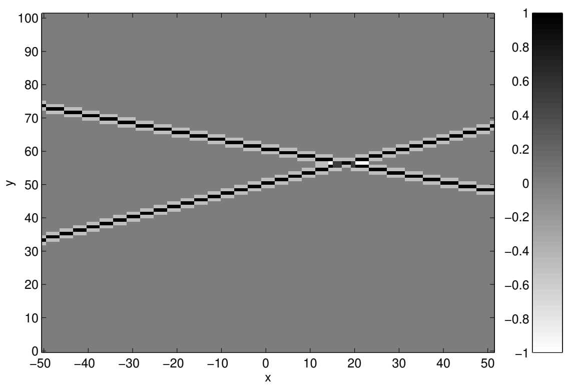

Figure 1.28 Image with two intersecting lines convolved with a wavelet in the vertical direction. Outside the lines the pixels values are zero.

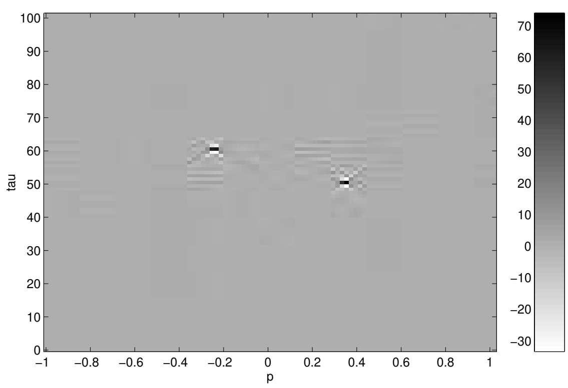

Figure 1.29 The corresponding discrete Radon transform, where the color scaling is determined by the minimum and maximum value, respectively. Besides the areas containing peaks, the values are close to zero.

Chapter 2

The Normal Radon Transform

In this chapter another very popular form of the linear Radon transform are reviewed. Several ways to define a discrete Radon transform is presented and an analysis of the sampling relationships is presented using the same techniques as in Chapter 1. A set of examples are included to illustrate the theory.

2.1 Defining the (ρ, θ) Radon Transform

In Chapter 1 one form of the linear Radon transform has been considered, namely slant stacking. This is merely one linear form of many, and the linear Radon transform can be defined on a more general form as

$$\breve{g}(\xi_0, \xi_1, \xi_2) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) \; \delta(\xi_0 - \xi_1 x - \xi_2 y) \; dx \; dy \tag{2.1}$$

On this form the line is described with (apparently) three degrees of freedom, which is one to many to describe a line, so the three parameters should always have a link, which removes one degree of freedom. In the literature two forms are by far the most common. The first, slant stacking, where the parameters are

$$(\xi_0, \xi_1, \xi_2) = (-\tau, p, -1) \tag{2.2}$$

Another definition of the Radon transform is used in many fields of science, e.g., tomography, astronomy and microscopy [14], where the fundamental function $g(x, y)$ has no preferred orientation. This has lead to describing the line on its normal form

$$\rho = x \cos\theta + y \sin\theta \tag{2.3}$$

i.e., $(\xi_0, \xi_1, \xi_2) = (\rho, \cos\theta, \sin\theta)$, and the normal Radon transform. The term normal Radon transform is not generally accepted, but here it is a very practical term and is only used to be able to distinguish slant stacking from the form shown in Eq. 2.4 can be written as

$$\breve{g}(\rho, \theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) \delta(\rho - x \cos\theta - y \sin\theta) \; dx \; dy \tag{2.4}$$

This definition can be compared to Eq. 1.2, found for slant stacking. Note again that no new symbol for the normal Radon transform has been assigned, but the arguments are used to show which definition is used.

Often another equivalent way of writing Eq. 2.4 is used

$$\begin{aligned}\check{g}(\rho,\theta) &= \int_{-\infty}^{\infty} g(\rho\cos\theta - s\sin\theta, \rho\sin\theta + s\cos\theta)\ ds &(2.5)\\ &= \frac{1}{|\sin\theta|}\int_{-\infty}^{\infty} g\left(x, \frac{\rho}{\sin\theta} - x\cot\theta\right)\ dx &(2.6)\\ &= \frac{1}{|\sin\theta|}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y)\ \delta\left(y - \frac{\rho}{\sin\theta} + x\cot\theta\right)\ dx\ dy &(2.7)\\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y)\ \delta(\rho - x\cos\theta - y\sin\theta)\ dx\ dy &(2.8)\end{aligned}$$

where the s-axis lies along the line. Eq. 2.8 can be compared to Eq. 1.1.

The meaning of the normal parameters used to specify the position of the line are shown in Fig. 2.1. The parameter ρ is the shortest distance from the origin of the coordinate system to the line, and θ is an angle corresponding to the angular orientation of the line.

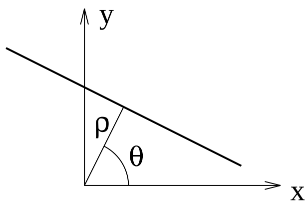

Figure 2.1 The two parameters ρ and θ used to specify the position of the line.

The first thing to notice about the normal Radon transform is that all lines can be described by choosing that $0 \le \theta < 2\pi$ and $\rho \ge 0$, but frequently other limits are used. If negative values of ρ are introduced the parameter domain are bounded by

$$0 \le \theta < \pi \text{ and } -\rho_{max} \le \rho \le \rho_{max} \tag{2.9}$$

where ρ_{max} is positive, and finite when any discrete implementation is considered. Both limitations of the parameter domain are valid, and they are very much related by

$$\check{g}(\rho,\theta) = \check{g}(-\rho,\theta+\pi) \tag{2.10}$$

which is easily found from Eq. 2.4 using the fact that the delta function is even.

The normal Radon transform is (not surprisingly) linked to slant stacking, and the one can be expressed from the other. From Eq. 2.6 it is found that

$$\check{g}(\rho,\theta) = \frac{1}{|\sin\theta|}\int_{-\infty}^{\infty} g\left(x, \frac{\rho}{\sin\theta} - x\cot\theta\right)\ dx = \frac{1}{|\sin\theta|}\check{g}\left(p = -\cot\theta, \tau = \frac{\rho}{\sin\theta}\right) \tag{2.11}$$

The relation is very simple, and the link between the two set of parameters can be found directly from Figs. 2.1 and 1.3. It is clear that Eq. 2.11 impose a problem when θ is close to 0, but in that case a link can be made to the Radon transform defined in Eq. 1.39,

$$\check{g}(\rho,\theta) = \frac{1}{|\cos\theta|}\int_{-\infty}^{\infty} g(ry+\eta, y)\ dy \quad \text{where} \begin{cases} r = -\tan\theta \\ \eta = \frac{\rho}{\cos\theta}\end{cases} \tag{2.12}$$

Using Eq. 2.4 several properties can be derived. Some of these can be found in appendix B. The most important property is that the normal Radon transform is a linear function. It should also be mentioned that several functions can be transformed analytically [14, 22]. As described in Sections B.3 and C.1, a set of those has been implemented in order to, e.g., test numerical approximations to the normal Radon transform or its inverse.

2.1.1 The Point Source

Again the point source is used to gain information about the fundamental behavior of the normal Radon transform. Here an arbitrary position of the point source (x^*, y^*) is assumed.

$$g(x,y) = \delta(x-x^*)\ \delta(y-y^*)\ \Rightarrow \tag{2.13}$$

$$\begin{aligned}\breve{g}(\rho,\theta) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \delta(x-x^*)\ \delta(y-y^*)\ \delta(\rho - x\cos\theta - y\sin\theta)\ dx\ dy \\ &= \delta(\rho - x^*\cos\theta - y^*\sin\theta)\end{aligned} \tag{2.14}$$

Fig. 2.2 illustrates a point source and the corresponding Radon transform.

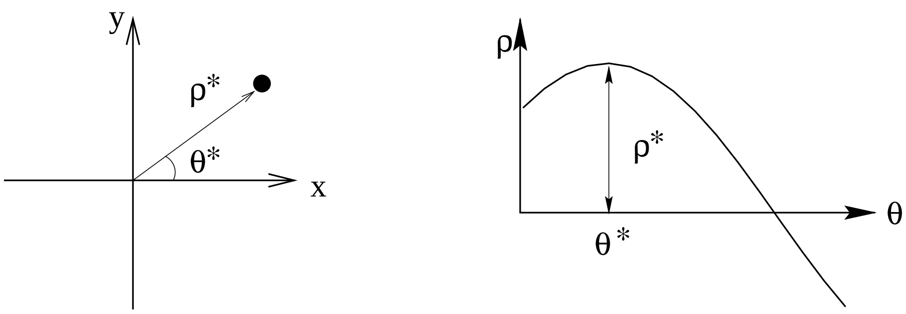

Figure 2.2 To the left is shown a point source, and to the right is shown the corresponding normal Radon transform.

From Eq. 2.14, it is derived that for any function $g(x,y)$ the Radon transform is given by

$$g(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x^*,y^*)\ \delta(x-x^*)\delta(y-y^*)\ dx^*\ dy^* \Rightarrow \tag{2.15}$$

$$\breve{g}(\rho,\theta) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x^*,y^*)\delta(\rho - x^*\cos\theta - y^*\sin\theta)\ dx^*\ dy^* \tag{2.16}$$

$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x^*,y^*)\delta(\rho - \rho^*\cos(\theta-\theta^*))\ dx^*\ dy^* \tag{2.17}$$

$$\text{where } \rho^*\cos\theta^* = x^* \text{ and } \rho^*\sin\theta^* = y^* \tag{2.18}$$

This result is very interesting with respect to numerical algorithms. It shows that a point is transformed into a sinusoid in the parameter domain, and the Radon transform of a point source can be confined in a truncated parameter domain.

$$g(x,y) = 0 \text{ for } \sqrt{x^2+y^2} > \rho_{max}\ \Rightarrow\ \breve{g}(\rho,\theta) = 0 \text{ for } |\rho| > \rho_{max} \tag{2.19}$$

which comes directly from Eq. 2.17. This result should be compared to Eq. 1.7, found for slant stacking.

2.1.2 The (ρ, θ) Radon Transform of a Line

One of the major advantages of the normal Radon transform regards line detection. Modelling a line with certain parameters (ρ^*, θ^*) with the delta function gives

$$
\begin{aligned}
g(x,y) &= \delta(\rho^* - x\cos\theta^* - y\ \sin\theta^*) \ \Rightarrow & (2.20)\\
\breve{g}(\rho,\theta) &= \int_{-\infty}^{\infty} \delta(\rho^* - (\rho\cos\theta - s\sin\theta)\cos\theta^* - (\rho\sin\theta + s\cos\theta)\sin\theta^*)\ ds \\
&= \int_{-\infty}^{\infty} \delta(\rho^* - \rho\cos(\theta-\theta^*) + s\sin(\theta-\theta^*))\ ds \\
&= \int_{-\infty}^{\infty} \frac{1}{|\sin(\theta-\theta^*)|}\,\delta\left(\frac{\rho^* - \rho\cos(\theta-\theta^*)}{\sin(\theta-\theta^*)} + s\right)\ ds, \quad \text{if } \sin(\theta-\theta^*) \neq 0 \\
&= \frac{1}{|\sin(\theta-\theta^*)|} & (2.21)
\end{aligned}
$$

and if $\theta = \theta^*$, i.e., $\sin(\theta - \theta^*) = 0$, it is found that

$$
\breve{g}(\rho,\theta) = \int_{-\infty}^{\infty} \delta(\rho^* - \rho)\ ds = \begin{cases} 0, & \text{if } \rho^* \neq \rho \\ \displaystyle\int_{-\infty}^{\infty} \delta(0)\ ds, & \text{if } \rho^* = \rho \end{cases} \tag{2.22}
$$

Not surprisingly, the result is that a peak is formed, when $\rho = \rho^*$ and $\theta = \theta^*$ (and finite terms will also be found). Furthermore note that there is no limitations on the orientation of the line, which was a problem with the way slant stacking was defined.

2.2 The Discrete (ρ, θ) Radon Transform

A discrete approximation to the continuous slant stacking was straight forward, cf. Eqs. 1.12 and 1.13. Again all parameters are sampled linearly.

$$
\begin{array}{ll}
x = x_m = x_{min} + m\Delta x, & m = 0, 1, \ldots, M-1 \\
y = y_n = y_{min} + n\Delta y, & n = 0, 1, \ldots, N-1 \\
\theta = \theta_t = \theta_{min} + t\Delta\theta, & t = 0, 1, \ldots, T-1 \\
\rho = \rho_r = \rho_{min} + r\Delta\rho, & r = 0, 1, \ldots, R-1
\end{array} \tag{2.23}
$$

Apparently the transform has a lot of free parameters, but normally several of them are fixed. Eq. 2.19 implies that the interesting part of the function $g(x, y)$ should be shifted towards the center of the coordinate system, in order to get the smallest number of samples in the ρ-direction.

Now it will be assumed that the image is square

$$\Delta x = \Delta y \tag{2.24}$$

$$M = N \tag{2.25}$$

which implies that the samples should lie in a symmetrical interval around zero, i.e,

$$x_{min} = -x_{max} = -\frac{(M-1)}{2}\Delta x \tag{2.26}$$

$$y_{min} = x_{min} = -y_{max} = -\frac{(M-1)}{2}\Delta x \tag{2.27}$$

$$\rho_{min} = -\rho_{max} = -\frac{(R-1)}{2}\Delta\rho \tag{2.28}$$

Considering the angular sampling, the angular starting point can be chosen to

$$\theta_{min} = 0 \tag{2.29}$$

Secondly, the sampling interval of θ should be set to span π, i.e.,

$$\Delta\theta = \frac{\pi}{T} \tag{2.30}$$

This leaves only a few free parameters, namely Δx, M, T, R, and $\Delta\rho$. These limitations are assumed in the following.

One way of defining a discrete normal Radon transform could be to approximate Eq. 2.8,

$$\breve{g}(\rho_r, \theta_t) = \int_{-\infty}^{\infty} g(\rho_r \cos\theta_t - s\sin\theta_t, \rho_r \sin\theta_t + s\cos\theta_t)\, ds \tag{2.31}$$

$$\approx \Delta s \sum_{j=0}^{J-1} g(\rho_r \cos\theta_t - s_j \sin\theta_t, \rho_r \sin\theta_t + s_j \cos\theta_t) \tag{2.32}$$

where s_j is a linear sampling of the variable s, done as in Eq. 2.23. This approach have some implications. The most serious is that for a given value of j, the image points found in Eq. 2.32, i.e., $(x, y) = (\rho_r \cos\theta_t - s\sin\theta_t, \rho_r \sin\theta_t + s\cos\theta_t)$ (in principle) never coincide with samples in the (assumed) image $g(m,n) = g(x_m, y_n)$, hence an interpolation in both variables (x, y) is needed. This two-dimensional interpolation can and should be avoided, due to the extra noise added, compared to a one-dimensional interpolation. Another question is how densely should the sampling interval of the parameter s, i.e., how Δs should be adjusted.

Another way of approximating the normal Radon transform and only get a one-dimensional interpolation in the image is to reformulate Eq. 2.8 using the slant stacking definition, like it was done in Eq. 2.11 and Eq. 2.12. That solution only requires an one-dimensional interpolation. Either implemented as a (fast) nearest neighbour approximation like below, or e.g., a linear interpolation.

$$\sin\theta > \frac{1}{\sqrt{2}} : \breve{g}(\rho, \theta) = \frac{1}{|\sin\theta|} \breve{g}\left(p = -\cot\theta, \tau = \frac{\rho}{\sin\theta}\right) \tag{2.33}$$

$$\approx \frac{\Delta x}{|\sin\theta|} \sum_{m=0}^{M-1} g\left(m, [\alpha m + \beta]\right) \tag{2.34}$$

$$\text{where } \alpha = -\cot\theta \text{ and } \beta = \frac{\rho - x_{min}(\cos\theta + \sin\theta)}{\Delta x \sin\theta} \tag{2.35}$$

$$\sin\theta \le \frac{1}{\sqrt{2}} : \breve{g}(\rho, \theta) = \frac{1}{|\cos\theta|} \breve{g}(r = -\tan\theta, \eta = \frac{\rho}{\cos\theta}) \tag{2.36}$$

$$\approx \frac{\Delta x}{|\cos\theta|} \sum_{n=0}^{M-1} g\left([\alpha n + \beta], n\right) \tag{2.37}$$

$$\text{where } \alpha = -\tan\theta \text{ and } \beta = \frac{\rho - x_{min}(\cos\theta + \sin\theta)}{\Delta x \cos\theta} \tag{2.38}$$

where it has been used that $\Delta x = \Delta y$, $M = N$, and $x_{min} = y_{min}$. Note that besides the (projection) factors $\frac{1}{|\sin\theta|}$ and $\frac{1}{|\cos\theta|}$ in front of the integrals, the way to approximate the two integrals follow section 1.4 and especially Section 1.7. The idea is shown in Algorithm 2.1.

Algorithm 2.1 : The Discrete Normal Radon Transform

```
For t = 0 to T-1                                       //For all values of theta
  costheta = cos(theta(t))
  sintheta = sin(theta(t))
  rhooffset = x_min*(costheta+sintheta)                //Common factor
  If sintheta>1/sqrt(2)                                //Project onto x-axis
    alpha = -costheta/sintheta                         //Digital slope is set
    For r = 0 to R-1                                   //For all values of rho
      beta = (rho(r)-rhooffset)/(Delta_x*sintheta)     //Offset is set
      mmin and mmax are set using Eq. 1.19             //Expression omitted
      sum = 0                                          //Initialize sum
      For m = mmin to mmax                             //For all legal values of x
        sum=sum+g(m,round(alpha*m+beta)))              //Increment sum
      End
      g_radon(r,t)=Delta_x*sum/sintheta                //Update matrix element
    End
  Else                                                 //Project onto y-axis
    alpha = -sintheta/costheta                         //Digital slope is set
    For r = 0 to R-1                                   //For all values of rho
      beta = (rho(r)-rhooffset)/(Delta_x*costheta)     //Offset is set
      nmin and nmax are set using Eq. 1.19             //Expression omitted
      sum = 0                                          //Initialize sum
      For n = nmin to nmax                             //For all legal values of x
        sum=sum+g(round(alpha*n+beta),n)               //Increment sum
      End
      g_radon(r,t)=Delta_x*sum/abs(costheta)           //Update matrix element
    End
  End
End
```

As it can be seen in Algorithm 2.1 the program is actually split up in two parts, corresponding to the axis on which the line integral has been projected. Each of the two parts have been optimized like it has been described below Algorithm 1.1. The expressions for mmin and mmax are given is Eq. 1.19, and the expressions for nmin and nmax can be derived from the same equation with some minor changes in variables.

2.2.1 Sampling Properties of the (ρ, θ) Radon Transform

In order to gain knowledge of the sampling relationships, the sinc-expansion procedure, shown in Subsection 1.4.3, is used. But instead of integrating to get the result, Eq. 1.24, i.e., the sinc interpolation result from slant stacking and the relation between the normal Radon transform and slant stacking, Eq. 2.11, are coupled in order to get the result. This shows an important way to gain more knowledge about the Radon transform. Some issues are better understood or easier derived with, e.g., slant stacking, but can easily be translated to the normal Radon transform (and reversely).

$$g(x,y) = = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} g(m,n)\ \frac{\sin(\frac{\pi}{\Delta x}(x-x_m))}{\frac{\pi}{\Delta x}(x-x_m)}\frac{\sin(\frac{\pi}{\Delta y}(y-y_n))}{\frac{\pi}{\Delta y}(y-y_n)} \Rightarrow \tag{2.39}$$

$$\breve{g}(\rho,\theta) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} g(m,n)\ I\left(p=-\cot\theta, \tau=\frac{\rho}{\sin\theta}, x_m, y_n\right) \tag{2.40}$$

$$= \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} g(m,n)\ I(\rho,\theta,x_m,y_n) \tag{2.41}$$

$$I(\rho,\theta,x_m,y_n) = \frac{\Delta x}{\psi}\sin\left(\psi\min\left\{\frac{1}{|\sin\theta|},\frac{1}{|\cos\theta|}\right\}\right) \tag{2.42}$$

$$\text{where } \psi = \frac{\pi}{\Delta x}(\rho - x_m\cos\theta - y_n\sin\theta) \tag{2.43}$$

Using the technique shown in Subsection 1.4.3, the sampling properties of the normal Radon transform is determined by ψ. As a function of θ the function $\min\{1/|\sin\theta|, 1/|\cos\theta|\}$ has values between 1 and $\sqrt{2}$. Using the maximum of $\sqrt{2}$, the sampling interval of the ψ should not be greater than $\frac{\pi}{\sqrt{2}}$. Using this limit for both parameters gives

$$\Delta\psi = \left|\frac{\partial\psi}{\partial\rho}\right|\Delta\rho = \frac{\pi}{\Delta x}\Delta\rho \le \frac{\pi}{\sqrt{2}} \ \Rightarrow\ \Delta\rho \le \frac{\Delta x}{\sqrt{2}} \tag{2.44}$$

$$\Delta\psi = \left|\frac{\partial\psi}{\partial\theta}\right|\Delta\theta \le \frac{\pi}{\sqrt{2}} \ \Rightarrow\ \Delta\theta \le \min_{x_m,y_m}\frac{\Delta x}{\sqrt{2}\sqrt{x_m^2+y_n^2}} = \frac{\Delta x}{2|x_{min}|} \tag{2.45}$$

Note that some rather conservative assumptions are made, in order to make the expressions valid for all values of (x_m, y_n). The results show that for a given image a lower limit of sampling intervals in the parameter domain exist. Violating these limits can lead to aliasing problems in the parameter domain. It can be shown that Eq. 2.44 and Eq. 2.45 corresponds to a demand, that the change of line positions should be below one sample, when changing either ρ or θ with their respective sampling intervals.

With a given image $g(m,n)$ and sampling distance Δx, the angular sampling distance $\Delta\theta$ can be set using Eq. 2.45, hence the angular sampling parameters can be chosen to

$$T = \lceil \pi(M-1)\rceil \quad \text{and} \quad \Delta\theta = \frac{\pi}{T} \tag{2.46}$$

where it has been used that $x_{min} = -\Delta x\,(M-1)/2$. When using Eq. 2.44 and Eq. 2.28, the only remaining parameter to determine is R (or ρ_{min}=-ρ_{max}), which can be determined by assuming $\Delta\rho = \frac{\Delta x}{\sqrt{2}}$, cf. Eq. 2.44, and inserting Eqs. 2.26-2.28 into Eq. 2.19. In total,

$$\rho_{max} = \max_{x_m,y_n}\sqrt{x^2+y^2} = \sqrt{2}\,|x_{min}| \ \Rightarrow \tag{2.47}$$

$$R = 2\frac{\sqrt{2}|x_{min}|}{\Delta\rho} + 1 = 2M-1 \quad \text{and} \quad \rho_{min} = -\frac{R-1}{2}\Delta\rho \tag{2.48}$$

2.2.2 The Discrete (ρ, θ) Radon Transform of Several Lines

Fig. 2.3 shows an image with $101 * 101$ pixels, where seven lines can be found with different orientations and amplitudes μ (pixel value on the line) as shown in Table 2.1.

Fig. 2.4 shows the discrete parameter domain using Algorithm 2.1, i.e., a nearest neighbour approximation. Seven peaks are found and the values in the peaks correspond to the curve amplitudes μ. As seen from Table 2.2, the sampling parameters were chosen according to Eq. 2.44 and Eq. 2.45. Note that a high number of samples is required in the parameter domain compared to the original image.

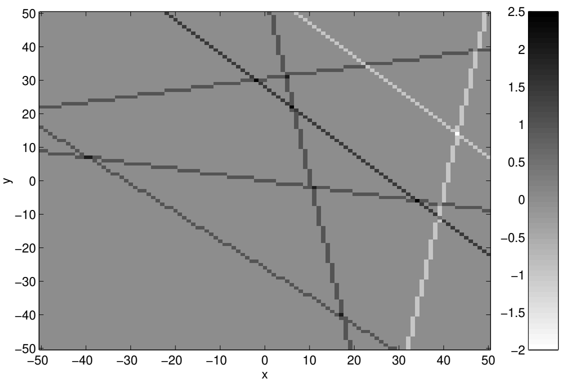

Figure 2.3 An image with seven lines. The parameters are listed in Table 2.1. The background color correspond to a value of zero.

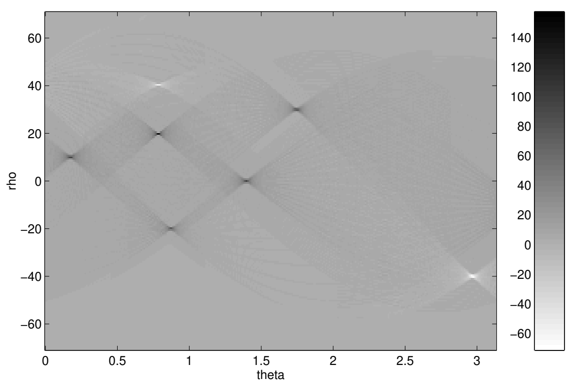

Figure 2.4 The corresponding discrete Radon transform using the nearest neighbour approximation. The background color correspond to a value of zero.

Line number	1	2	3	4	5	6	7
ρ	10	20	40	-20	0	30	-40
θ	0.175	0.785	0.785	0.873	1.396	1.745	2.967
μ	1	1.5	-1	1	1	1	-1

Table 2.1 Line parameters corresponding to Fig. 2.3.

Image domain		Parameter domain	
Parameter	Value	Parameter	Value
M	101	R	201
N	101	T	315
Δx	1	$\Delta\rho$	0.707
Δy	1	$\Delta\theta$	0.01
x_{min}	-50	ρ_{min}	-70.7
y_{min}	-50	θ_{min}	0

Table 2.2 Sampling parameter settings corresponding to Figs. 2.3 and 2.4.

Now the sampling parameters are changed. Firstly, the sampling parameters corresponding to the discrete parameter domain shown in Fig. 2.5, violates Eq. 2.44 and Eq. 2.45. Here the peak corresponding to line number three have disappeared. Secondly, in Fig. 2.6 a discrete parameter domain is shown, where the sampling parameters are set to zoom in the area of parameters of line number five.

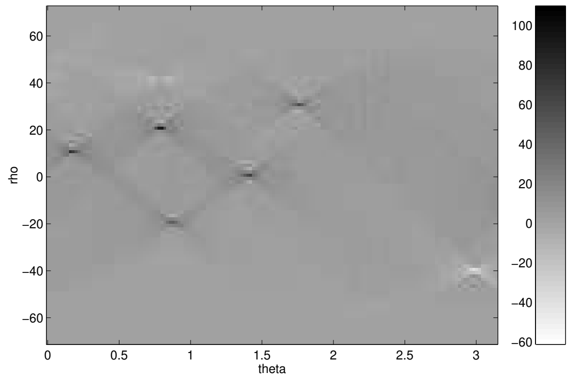

Figure 2.5 The sparsely sampled parameter domain. Note how the peak corresponding to line number three is missing, i.e., at $(\rho = 40, \theta = 0.79)$. The sampling parameters are listed in the left part of Table 2.3.

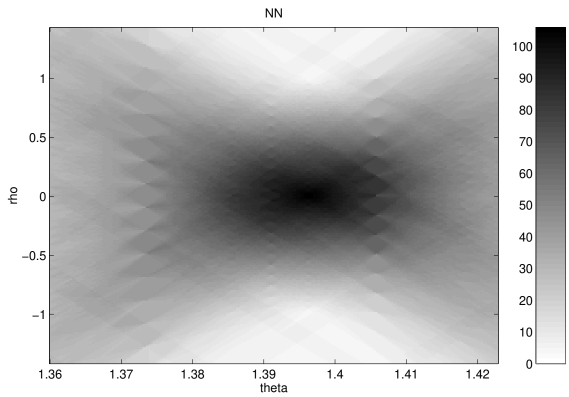

Figure 2.6 The parameter domain with focus in the area of line number five, i.e., at $(\rho = 0, \theta = 1.396)$. The sampling parameters are listed in the right part of Table 2.3.

Parameter domain (zoom out)		Parameter domain (zoom in)	
Parameter	Value	Parameter	Value
R	101	R	201
T	128	T	315
$\Delta\rho$	1.41	$\Delta\rho$	0.0141
$\Delta\theta$	0.02	$\Delta\theta$	$2 \cdot 10^{-4}$
ρ_{min}	-70.7	ρ_{min}	-1.41
θ_{min}	0	θ_{min}	1.36

Table 2.3 Sampling parameter settings corresponding to Figs. 2.5 and 2.6.

Next the discrete parameter domain using linear interpolation and sinc interpolation are displayed in the area of line number five. From Fig. 2.6 it can be seen that the nearest neighbour approximation actually produces a peak, though it is a somewhat different shape compared the peaks found in Fig. 2.7 and Fig. 2.8, which look very much alike. Again for this example, it is found that the peak value is lowest when using linear interpolation.

The evaluation of the results should also incorporate the time used to calculate the three zoomed parameter domains, which is shown in Table 2.4. The complexity of the three approximations are given by

$$\begin{aligned} \mathcal{O}_{\text{Nearest Neighbour}} &= \mathcal{O}(R\,T\,M) &&\approx \mathcal{O}(M^3) \\ \mathcal{O}_{\text{Linear Interpolation}} &= \mathcal{O}(R\,T\,M) &&\approx \mathcal{O}(M^3) \\ \mathcal{O}_{\text{Sinc Interpolation}} &= \mathcal{O}(R\,T\,M^2) &&\approx \mathcal{O}(M^4) \end{aligned} \tag{2.49}$$

which is analogous to Eq. 1.36. And again it should be noted that linear interpolation is somewhat

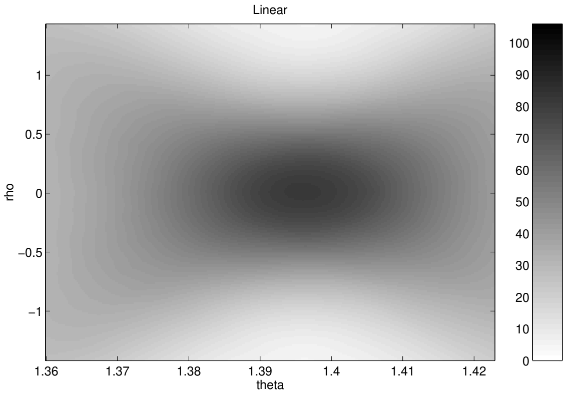

Figure 2.7 The parameter domain with focus in the area of line number five using linear interpolation.

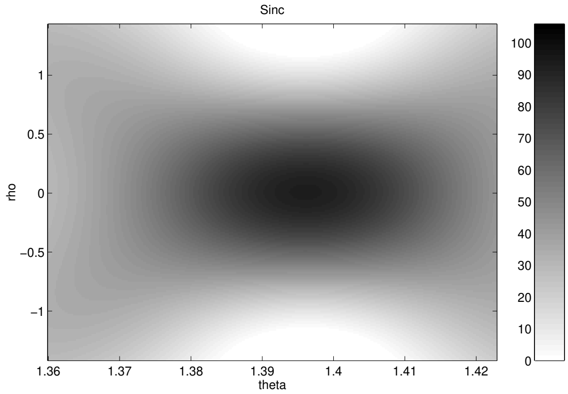

Figure 2.8 The parameter domain with focus in the area of line number five using sinc interpolation.

slower than using nearest neighbour interpolation.

All programs were implemented in C, and for the first two algorithms, the code was optimized and implemented as described below Algorithm 1.1. The huge difference found in Table 2.4 from the first two algorithms to the 1500 seconds used with the sinc interpolation, is due to two things. Firstly, the sinc approximation requires a sinc function, which have to be computed once per iteration. Secondly, the first two algorithms have, cf. Eq. 1.36, a much smaller computational cost, compared to the sinc interpolation strategy, as shown in Eq. 2.49.

Method	Time
Nearest Neighbour	2.9 sec
Linear Interpolation	4.2 sec
Sinc Interpolation	1450 sec

Table 2.4 Time for calculating the discrete Radon transform for the three types of approximation, corresponding to Figs. 2.6, 2.7, and 2.8. All three times are measured on a 120 MHz Pentium.

To visualize the resolution of the discrete normal Radon transform, Figs. 2.9 (nearest neighbour), 2.10 (linear interpolation), and 2.11 (sinc interpolation) show the parameter domain in the area of the peak using a threshold level of 50.

It can be seen that all three implementations give approximately nearly the same resolution, and the resolution in the ρ-direction is a bit over the one derived in Eq. 2.44. The resolution in the θ-direction is approximately five times the one derived in Eq. 2.45, but it must be noted, that this limit was derived by using rather conservative assumptions.

In Fig. 2.12 the discrete parameter domain is shown in the area of the line number three in Table 2.1 by using nearest neighbour approximation, and in Fig. 2.13 the parameter domain has been truncated at the value -50 (note that line number three has amplitude -1, and note that the peak value is approximately -63). It can be seen that resolution here is below the one derived value in Eq. 2.44, in the ρ-direction, which again indicates that the threshold changes the result of this kind of analysis. Likewise Fig. 2.14 shows the same part of the parameter domain found by use of linear interpolation. Finally in Fig. 2.15 the parameter domain has been computed using linear interpolation and then truncated at -50. A significantly smaller region is found, which is to be expected with the high threshold level.

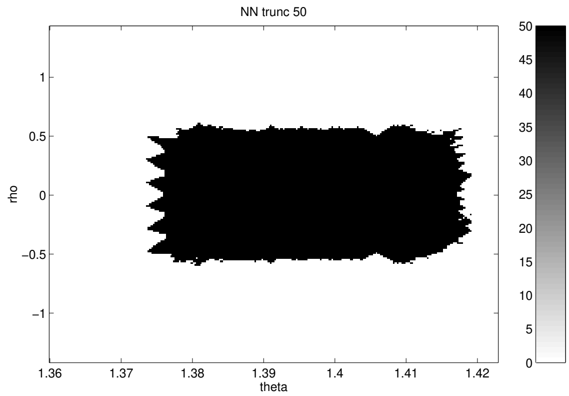

Figure 2.9 The discrete parameter domain using a threshold level of 50 (nearest neighbour).

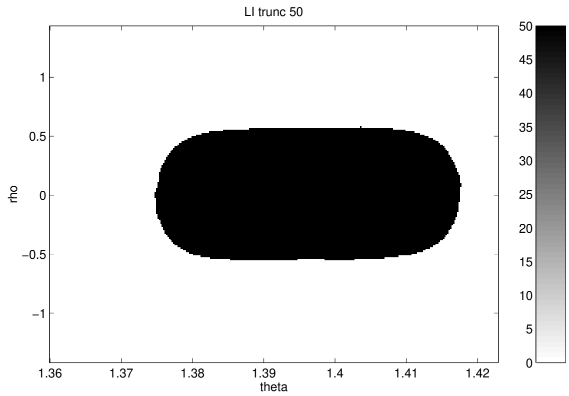

Figure 2.10 The discrete parameter domain using a threshold level of 50 (linear interpolation).

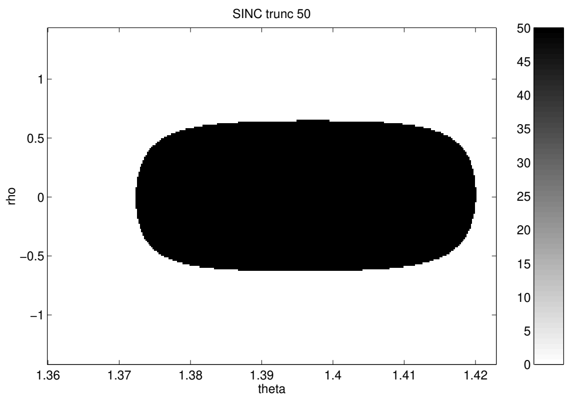

Figure 2.11 The discrete parameter domain using a threshold level of 50 (sinc interpolation).

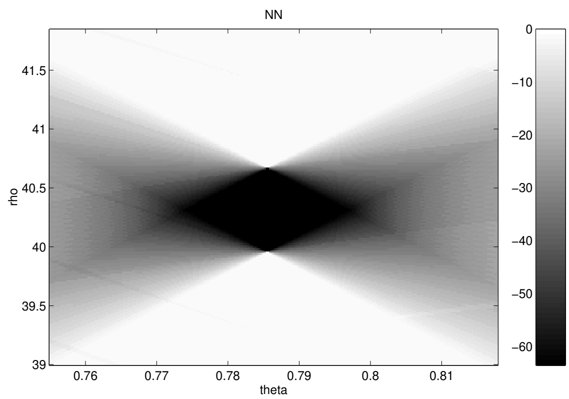

Figure 2.12 The discrete parameter domain (nearest neighbour) in the area of the third line, see Table 2.1.

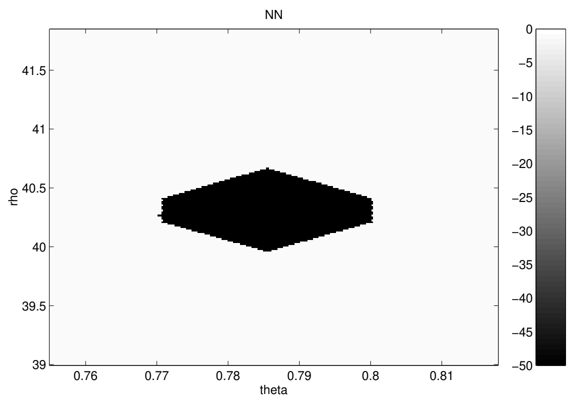

Figure 2.13 The same discrete parameter domain using a threshold level of -50.

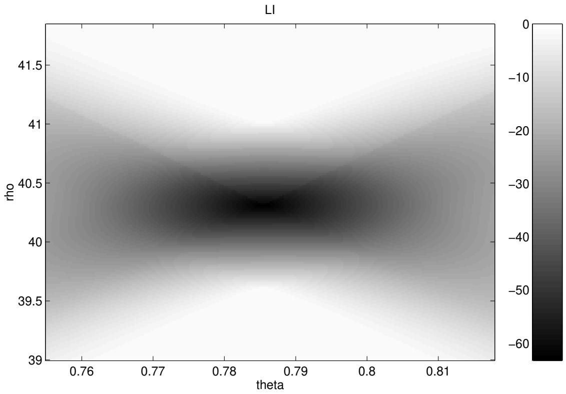

Figure 2.14 The discrete parameter domain (linear interpolation) in the area of the third line, see Table 2.1.

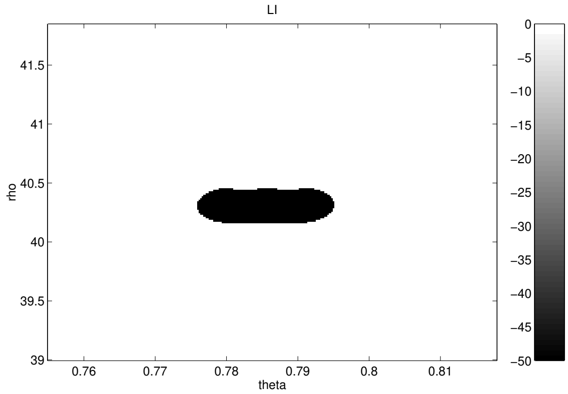

Figure 2.15 The same discrete parameter domain using a threshold level of -50 (linear interpolation).

2.2.3 The Discrete (ρ, θ) Radon Transform of Points

Fig. 2.16 shows an image with only 14 pixels having a value different to zero. The value drops off linearly to the right of the image. The 14 pixels lie on two lines with the same angular orientation, and in Fig. 2.17 the corresponding discrete parameter domain is shown, by use of a nearest neighbour approximation. Sampling parameters can be found in Table 2.2. Each of the 14 points are clearly represented by a sinusoid, and note how the linearity of the Radon transform implies that the sinusoids intersect and peaks are positioned corresponding to the parameters of the two supporting lines.

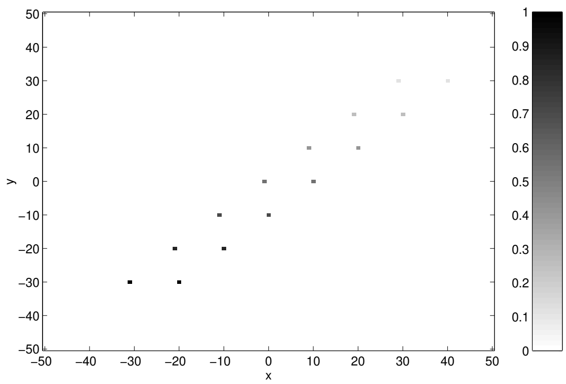

Figure 2.16 An image with 14 pixels, lying on two lines. The values are attenuated towards the right side of the image.

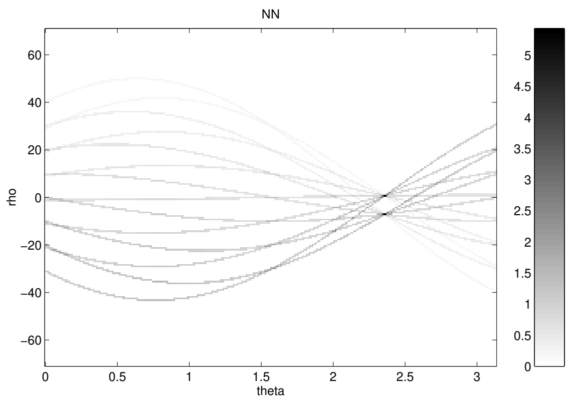

Figure 2.17 The corresponding discrete parameter domain using a nearest neighbour approximation. Note how individual sinusoids form connected curves.

Figs. 2.18 and 2.19 show the discrete parameter domains found by using linear interpolation and the sinc interpolation respectively. Fig. 2.18 shows approximately the same result as Fig. 2.17, but the sinusoids are more smooth. A closer examination of Fig. 2.19 reveals that each of the sinusoids have ringing effects, resulting in undesired non-zero values in the parameter domain also in areas where no sinusoids are found.

Transformation times corresponding to the three approximations are given in Table 2.5.

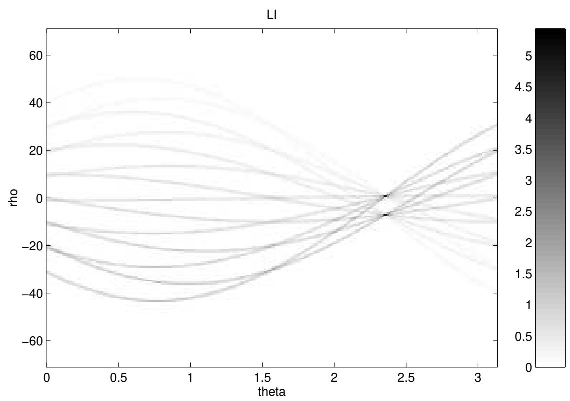

Figure 2.18 The corresponding discrete parameter domain using linear interpolation.

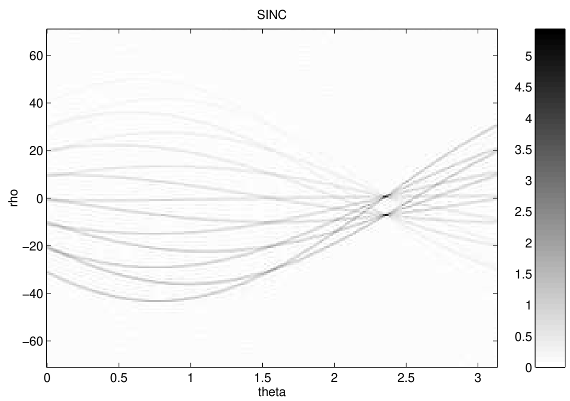

Figure 2.19 The corresponding discrete parameter domain using sinc interpolation.

In Fig. 2.20 is shown the discrete sparsely sampled parameter domain using a nearest neighbour approximation. Sampling parameters are shown in the left side of Table 2.3. It should be noted that the upper peak has a very low value. This is due to the violation of the sampling criterion. It can also be seen that the sinusoids now are broken at certain angles. Using linear interpolation as in Fig. 2.21 only implies that the curves does not break, but note again the low value of the upper peak.

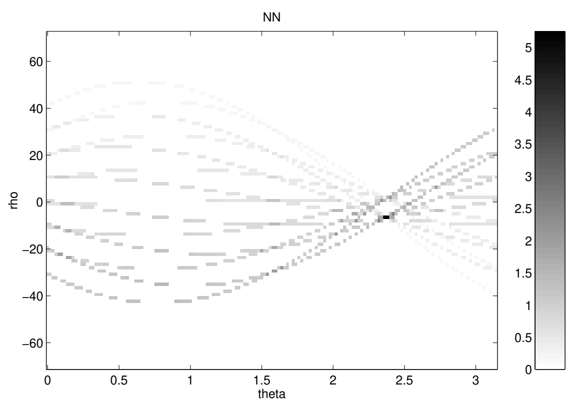

Figure 2.20 The discrete parameter domain using a sparse sampling (nearest neighbour). Note how the curves break up at certain angles.

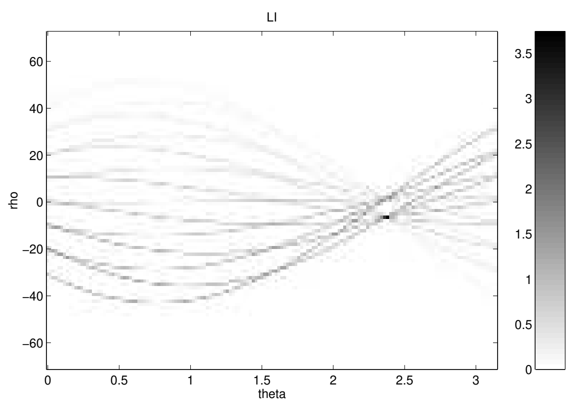

Figure 2.21 The discrete parameter domain using a sparse sampling (linear interpolation). Here the curves nearly break up at certain angles.

Method	Time
Nearest Neighbour	1.9 sec
Linear Interpolation	2.7 sec
Sinc Interpolation	1450 sec

Table 2.5 Time for calculating the discrete Radon transform for the three types of approximation, corresponding to Figs. 2.17, 2.18, and 2.18. All times are measured on a 120 MHz Pentium.

2.3 Summary

The Radon transform was defined on (ρ, θ) form and discrete forms were derived. Both forms were analyzed, and sampling properties were derived and analyzed using a sinc expansion strategy, like it was done in the Chapter 1.

It was demonstrated for curve detection purposes, that a nearest neighbour approximation is adequate if sampling of the parameter domain is sufficiently dense. It was also demonstrated that linear interpolation gave approximately the same parameter domain as the one found with the sinc interpolation strategy.

As expected, it has been demonstrated that the discrete normal Radon transform can detect lines with arbitrary orientation, which could not be fulfilled with slant stacking. The chapter was concerned with quadratic images and the arbitrary line orientation, and that is one of the major differences from the results found in Chapter 1.

Chapter 3

The Hough Transform

So far the Radon transform has been investigated as a tool for line detection. Two different ways of selecting parameters gave mainly the same type of discrete transform, namely, for each position in the parameter domain, sum up pixel values along a line in the image and store the sum with proper scaling at that position.

If the image is very sparse, e.g., a binary image with few non-zero pixels, most of the computer time is spend summing up zeros, that does not contribute to the parameter domain. In one of the most cited patents in the image processing literature, [23], P.V.C. Hough proposed a way of incorporating that prior knowledge into the transform, in order to reduce the computational cost. The idea was initially formulated for lines described with slope and offset parameter (like slant stacking), but many authors have translated the idea into the domain of normal line parameters (ρ, θ), e.g., [24, 25, 26]. It should be mentioned, that there is some confusion in the literature, regarding how the Hough and the Radon transform are related. It is a problem of definition. A common opinion, which here will be used for defining the Hough transform, is that the Hough transform maps the individual pixels from the image domain into a shape in the parameter domain, and the Radon transform transform a shape in the image domain into a single pixel in the parameter domain. As written in [23], "For a given point on a line segment in framelet 108[1], a line is drawn in the transformed plane...", thus the main idea is to transform each pixel in the image individually into the parameter domain.

From a historical point of view it is not clear what motivated the Hough transform. It is likely that it was a new idea, and not motivated by results like the ones shown in Section 1.6. Apparently it was first later on, that it was realized, that the Radon transform and the Hough transform are highly related.

3.1 The (p, τ) Hough Transform

The Hough transform can be derived from the Radon transform [15]. In Eq. 1.9 it was found that

$$\begin{aligned} g(x,y) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x^*,y^*)\ \delta(x-x^*)\ \delta(y-y^*)\ dx^*\ dy^* \ \Rightarrow \\ \breve{g}(p,\tau) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x^*,y^*)\ \delta(\tau-y^*+px^*)\ dx^*\ dy^* \end{aligned} \tag{3.1}$$

Eq. 3.1 actually expresses what Hough meant. For each point in the image $g(x^*, y^*)$, draw a line in the transformed plane, i.e., the parameter domain. The line is found by setting the argument of the last delta function to zero

$$\tau = y^* - px^* \tag{3.2}$$

[1] Reference to an image in the patent.

Eq. 3.1 states that the extra weight, or vote, that the parameter domain should be updated with is proportional to the function value $g(x^*, y^*)$. This is a natural choice, so in this light the Hough transform on a continuous form, c.f. Eq. 3.1, can be viewed as a special case of the Radon transform. But this is only true in the continuous case, and as it will be demonstrated the two transforms are in general not identical in their discrete forms. Note also that the Hough transform commonly is only used in a discrete form.

Eq. 3.1 motivates a new scheme for estimation of the parameter domain. Again in order to reduce the size of the algorithm y(n), is assumed to be set to y_n and so on. Note that the sampling parameters defined in Eq. 1.12 are used again.

Algorithm 3.1 : The Hough Transform

```
Set g_hough(k,h)=0 for all (k,h)               //Initialize Hough space
For m=0 to M-1                                 //For all values of x
  For n=0 to N-1                               //For all values of y
    gvalue = g(m,n)                            //Store in simple variable
    If gvalue≠0                                //Only consider non-zero pixels
      For k=0 to K-1                           //For all values of p
        tau=y(n)-p(k)*x(m)                     //Calculate the tau
        h=round((tau-tau_min)/Delta_tau)       //Find the right sample
        If h≥0 and h<H                         //Check if sample is valid
          g_hough(k,h)=g_hough(k,h)+gvalue     //Update parameter domain
        End
      End
    End
  End
End
```

Algorithm 3.1 describes the core of the Hough transform, though it should be noted that nearly all people working with the Hough transform use the (ρ, θ) form. The key point is that for the non-zero pixels in the image and each value of p_k, a value of τ is calculated, which (here) is rounded to the nearest neighbour. In the following the implications of this strategy will be analyzed and compared to the Radon transform.

The first issue is to compare the computational complexities. Assuming that only some of the image pixels have a non-zero value, then the computational complexity of the Hough transform is given in Eq. 3.4. For easy comparison the first line of Eq. 1.36 is repeated in Eq. 3.3, corresponding to the nearest neighbour discrete Radon transform.

$$\mathcal{O}_{\text{Radon}} = \mathcal{O}(K\,H\,M) \approx \mathcal{O}(M^3) \tag{3.3}$$
$$\mathcal{O}_{\text{Hough}} = \mathcal{O}((M\,N)_r\,K) \approx \mathcal{O}((M\,N)_r\,M) \tag{3.4}$$

where the index r is used to indicate that a reduced number of the pixel has to be transformed. If the image only contains one non-zero pixel then $(M\,N)_r = 1$, and the Hough transform will be the fastest of the two transforms, but a more realistic value of $(M\,N)_r$ lies between M and $M\,N$, depending on the structure of the image. This also indicates that most applications of the Hough transform is in connection with binary images, often rather sparse (many zeros and few ones).

Again the procedure described right after Algorithm 1.1, is applied to reduce the computational cost, i.e., the continuous mapping equation is rewritten using the discrete parameters

$$\tau = y - px \;\Rightarrow \tag{3.5}$$

$$h \approx \frac{y_n - (k\Delta p + p_{min})x_m - \tau_{min}}{\Delta\tau} \;=\; \kappa k + \zeta \quad \text{where} \begin{cases} \kappa = -\frac{x_m \Delta p}{\Delta\tau} \\ \zeta = \frac{y_n - x_m p_{min} - \tau_{min}}{\Delta\tau} \end{cases} \tag{3.6}$$

If a nearest neighbour approximation is applied then Eq. 1.19 can be used to determine the values of h, that corresponds to parameters lying in the truncated discrete parameter domain, i.e.,

$$0 \le h = [\kappa k + \zeta] \le H - 1 \;\Rightarrow \begin{cases} -\frac{\zeta+\frac{1}{2}}{\kappa} < k < \frac{H-\frac{1}{2}-\zeta}{\kappa} \text{ if } \kappa > 0 \\ \frac{H-\frac{1}{2}-\zeta}{\kappa} < k < -\frac{\zeta+\frac{1}{2}}{\kappa} \text{ if } \kappa < 0 \end{cases} \Rightarrow \tag{3.7}$$

$$k_{min} = \max\left\{0, \left\lceil -\frac{\zeta+\frac{1}{2}}{\kappa} \right\rceil\right\} \text{ and } k_{max} = \min\left\{K-1, \left\lfloor \frac{H-\frac{1}{2}-\zeta}{\kappa} \right\rfloor\right\} \text{ if } \kappa > 0$$

$$k_{min} = \max\left\{0, \left\lceil \frac{H-\frac{1}{2}-\zeta}{\kappa} \right\rceil\right\} \text{ and } k_{max} = \min\left\{K-1, \left\lfloor -\frac{\zeta+\frac{1}{2}}{\kappa} \right\rfloor\right\} \text{ if } \kappa < 0 \tag{3.8}$$

In Algorithm 3.2 an implementation of the Hough transform is shown using (p,τ)-parameters and a nearest neighbour approximation in the parameter domain, and for optimization purposes Eq. 3.8 is used to assure that the pixels actually lie in the discrete parameter domain.

Algorithm 3.2 : The Optimized Hough Transform

```
Set g_hough(k,h)=0 for all (k,h)                  //Initialize Hough space
For m=0 to M-1                                    //For all values of x
  For n=0 to N-1                                  //For all values of y
    gvalue = g(m,n)                               //Store in simple variable
    If gvalue≠0                                   //Only consider non-zero pixels
      kappa = -x(m)*Delta_p/Delta_tau             //Calculate digital slope
      zeta = (y(n)-x(m)*p_min-tau_min)/Delta_tau  //Calculate digital offset
      Calculate k_min and k_max from Eq. 3.8
      For k=k_min to k_max                        //For all valid values of p
        h=round(kappa*k+zeta)                     //Find the right tau-value
        g_hough(k,h)=g_hough(k,h)+gvalue          //Update parameter domain
      End
    End
  End
End
```

In the algorithm, it is used that only the image pixels with value different from zero are transformed. For non-binary (floating point valued) images a threshold can be applied, but the threshold level strongly depends on the nature of the image.

3.1.1 Line Detection Using The Hough Transform

In Fig. 3.1 yet another synthetic image containing six line pieces is shown, and the line parameters can be found in Table 3.1. Fig. 3.2 shows the corresponding discrete parameter domain by use of discrete slant stacking. Sampling parameters can be found in Table 3.2. Six clearly marked peaks positioned at the correct line parameters can be found in the discrete parameter domain, even now, where only segments of lines are available. The starting and ending points of the line segments cannot directly be found from the parameter domain, but a more interesting discovery is that the parameter domain found by use of Hough transform in this case is *identical* to the one found with the discrete Radon transform. This is due to the choice of sampling parameters and the way the two algorithms compute the discrete indices.

In Eq. 1.17 it is found that

$$n = \left[\frac{p_k x_m + h\Delta\tau + \tau_{min} - y_{min}}{\Delta y}\right] \tag{3.9}$$

If the sampling parameters have been chosen as $\Delta\tau = \Delta y$ and $\tau_{min} = y_{min}$ as suggested in Subsection 1.4.4, then Eq. 3.9, corresponding to the discrete Radon transform, becomes very simple

$$n = \left[\frac{p_k x_m + h\Delta y}{\Delta y}\right] = h + \left[\frac{p_k x_m}{\Delta y}\right] \tag{3.10}$$

and for the Hough transform with the same choice of $\Delta\tau = \Delta y$ and $\tau_{min} = y_{min}$, the mapping function is given by Eq. 3.6:

$$h = \left[\frac{n\Delta y + y_{min} - p_k x_m - \tau_{min}}{\Delta\tau}\right] \tag{3.11}$$

$$= \left[\frac{n\Delta y - p_k x_m}{\Delta y}\right] \tag{3.12}$$

$$= n - \left[\frac{p_k x_m}{\Delta y}\right] \tag{3.13}$$

which is exactly the same as in Eq. 3.10. So the interesting conclusion is that using slope and offset parameters and nearest neighbour approximation, the two algorithms give *exactly* the same discrete parameter domain, when choosing $\Delta\tau = \Delta y$ and $\tau_{min} = y_{min}$. It should be noted that the Hough transform needs a general multiplicative factor of Δx in order to get the same scaling as the discrete Radon transform.

A major difference of the two algorithms comes with the time needed to compute the discrete parameter domains. Using optimized discrete Radon transform requires 0.42 seconds, and the optimized Hough transform requires only 0.026 seconds, i.e., a factor of 16 in favor of the Hough transform. In this case only 3.34% of the pixels is non-zero. Both times are measured on a 120 MHz Pentium PC.

Line number	1	2	3	4	5	6
p	-0.33	-0.10	0.25	-0.25	0.50	0.00
τ	50	70	60	40	50	20
x_{start}	-41	-11	-50	-11	-50	-41
x_{end}	29	50	9	29	9	9

Table 3.1 Line parameters corresponding to Fig. 3.1 are given along with the starting point x_{start}, and the ending point x_{end}.

Now the sampling parameters of the discrete parameter domain are changed in order to focus on two of the peaks. Sampling parameters are shown in the left side of Table 3.3. In Fig. 3.3 the discrete parameter domain is shown using discrete Radon transform, and likewise in Fig. 3.4

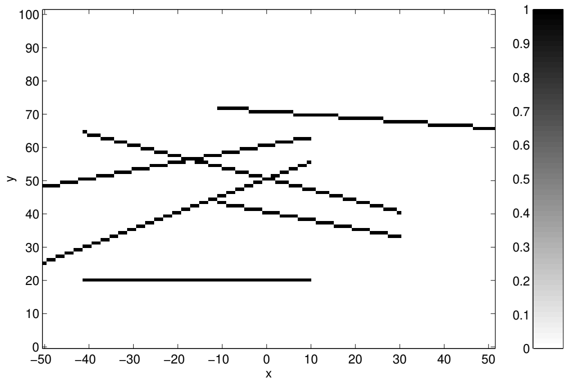

Figure 3.1 An image with 6 line pieces

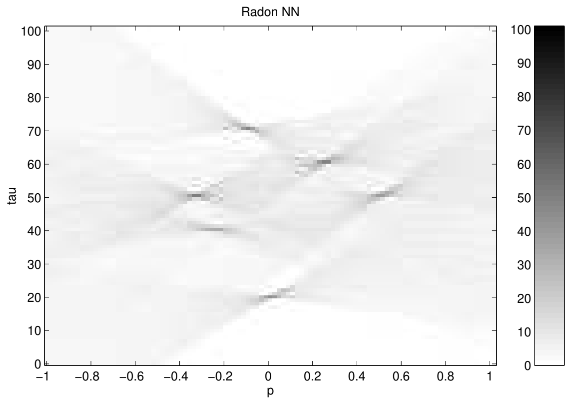

Figure 3.2 The discrete parameter domain found by either the Radon transform or the Hough transform.

Image domain		Parameter domain	
Parameter	Value	Parameter	Value
M	101	H	101
N	101	K	101
Δx	1	Δp	0.02
Δy	1	$\Delta \tau$	1
x_{min}	-50	p_{min}	-1
y_{min}	0	τ_{min}	0

Table 3.2 Sampling parameter settings.

by use of Hough transformation. It can be seen that two clearly marked peaks are found by use of the discrete Radon transform, just as expected from the previous chapters. From Fig. 3.4 it is seen that the Hough transform malfunctions. Firstly, it is found, that the levels are very low. Secondly, the regions of expected line parameters are only weakly marked, hence the parameter domain is useless.

Next, a sparse sampling of the parameter domain in Figs. 3.5 and 3.6 is used corresponding to the right side of Table 3.3. It can be seen that the discrete Radon transform malfunctions, and only two peaks are clearly marked. This is due to the under-sampling of the parameter domain. In contrast to Fig. 3.5, Fig. 3.6 shows that the Hough transform performs well. Six peaks can be found, but it should be noted that the resolution is poor.

Parameter domain (zoom in)		Parameter domain (zoom out)	
Parameter	Value	Parameter	Value
H	101	H	33
K	101	K	51
Δp	0.004	Δp	0.04
$\Delta \tau$	0.15	$\Delta \tau$	3
p_{min}	0.2	p_{min}	-1
τ_{min}	48	τ_{min}	0

Table 3.3 Sampling parameter settings for the zoom figures.

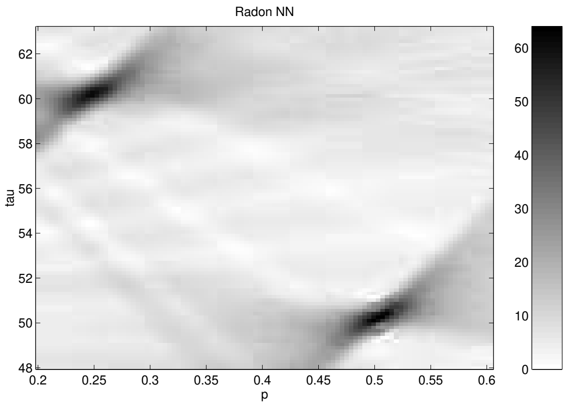

Figure 3.3 The discrete parameter domain found by use of the discrete Radon transform in the area with two line parameters.

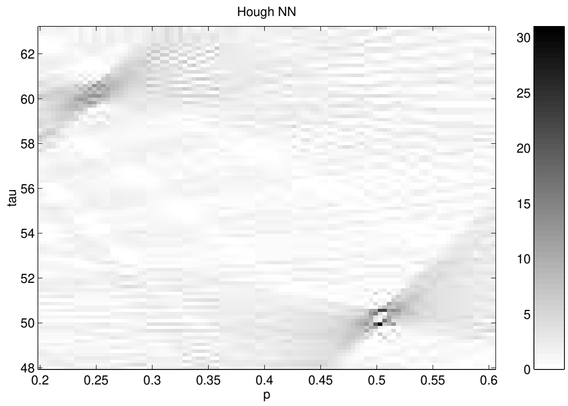

Figure 3.4 The discrete parameter domain found by use of the discrete Hough transform in the area with two of the line parameters.

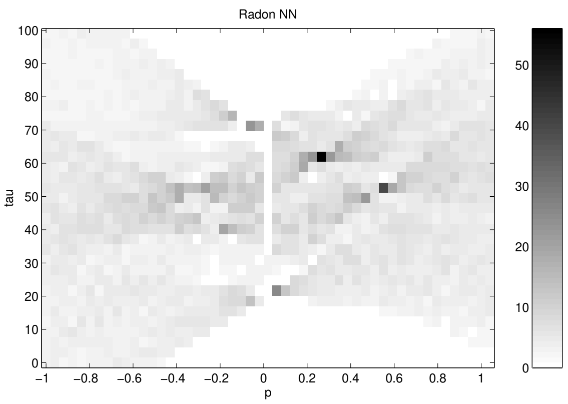

Figure 3.5 The sparsely sampled discrete parameter domain found by use of the discrete Radon transform.

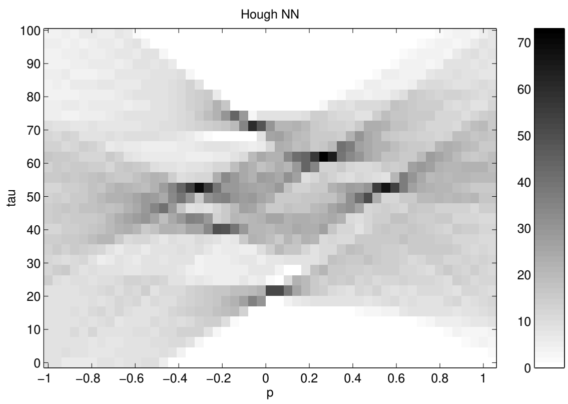

Figure 3.6 The sparsely sampled discrete parameter domain found by use of the discrete Hough transform.

3.1.2 Choosing Sampling Parameters with the (p, τ) Hough Transform

The reason why the Hough transform malfunctions when the parameter domain is sampled densely, and apparently works well with a coarsely sampled sampled parameter domain can be investigated from the mapping procedure using a nearest neighbour approximation. From Eq. 3.6 it will be shown that a band of image points are mapped into the same parameter vector.

$$h = \left[\frac{y_n - p_k x_m - \tau_{min}}{\Delta\tau}\right] \Leftrightarrow \tag{3.14}$$

$$0 = \left[\frac{(n\Delta y + y_{min}) - p_k(m\Delta x + x_{min}) - \tau_h}{\Delta\tau}\right] \Leftrightarrow \tag{3.15}$$

$$|n - \alpha m - \beta| < \frac{1}{2}\frac{\Delta\tau}{\Delta y} \quad \text{where} \begin{cases} \alpha = p_k \frac{\Delta x}{\Delta y} \\ \beta = \frac{p_k x_{min} + \tau_h - y_{min}}{\Delta y} \end{cases} \tag{3.16}$$

The important result is that the way the Hough transform is defined implies that all pixels lying around the (digital) line $n = \alpha m + \beta$ within a band of width $\frac{\Delta\tau}{\Delta y}$, measured in pixels in the n-direction, will be mapped into the same point, (k, h), in the discrete parameter domain. This explains the results shown in Figs. 3.4 and 3.6. Using a densely sampled parameter domain, with, e.g., $\Delta\tau = 0.1\Delta y$ implies that the width of the band in the image domain only amounts to $\frac{1}{10}$ of a sample, hence from a statistical point of view, only every tenth pixel along the digital line contributes to the discrete parameter domain, and a coarsely sampled parameter domain, with, e.g., $\Delta\tau = 5\Delta y$, implies that for a given point in the discrete parameter domain, and a certain value of x_m, five pixels will contribute to the same position in the parameter domain, if their value is different from zero. If no prior knowledge of the image is known, the conclusion is that a reasonable value of sampling distance of τ is, $\Delta\tau = \Delta y$.

Using the Hough transform with a coarsely sampled parameter domain can imply that the pixels need not lie on a perfect line, and still result in a peak in the parameter domain. For the Radon transform, further implications and limitations with respect to noise are discussed in Chapter 5.

It has been shown above, that the Hough transform requires that $\Delta\tau$ is not chosen too small. The parameter Δp can be chosen from different criteria. One criteria is that the absolute (digital) slope of the line in Eq. 3.6 does not exceed 1, in order to avoid that a pixel in the image is mapped into a perforated line in the discrete parameter domain, as shown in Figs. 3.7 and 3.8. This perforation problem can imply that the peak is not found in the discrete parameter domain.

$$|\kappa| = \left| -\frac{x_m \Delta p}{\Delta\tau} \right| \leq 1 \;\Rightarrow\; \Delta p < \frac{\Delta\tau}{\max|x|} = \frac{2\Delta\tau}{(M-1)\Delta x} \tag{3.17}$$

It can be noted that this limit matches Eq. 1.32 derived for the discrete (p, τ) Radon transform, when choosing $\Delta\tau = \Delta y$.

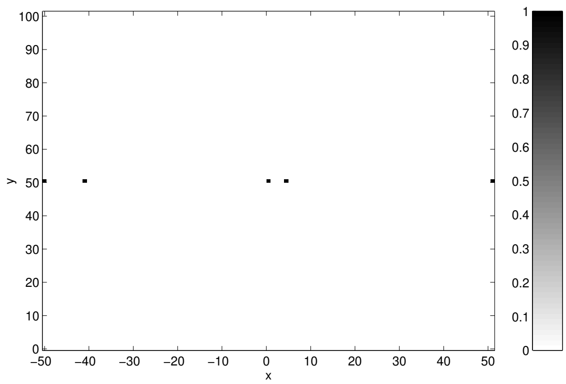

Figure 3.7 An image with 5 points lying on a line.

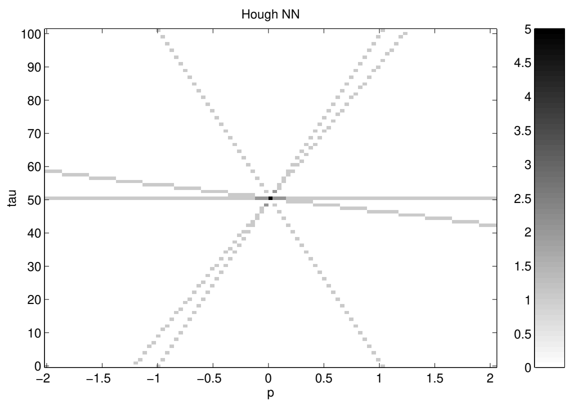

Figure 3.8 The corresponding Hough transform.

3.2 The (ρ, θ) Hough Transform

The Hough transform can be defined using normal parameters (ρ, θ) and Eq. 2.16 motivates the definition of the Hough transform

$$\begin{aligned} g(x,y) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x^*,y^*)\ \delta(x-x^*)\ \delta(y-y^*)\ dx^*dy^* \;\Rightarrow \\ \breve{g}(\rho,\theta) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x^*,y^*)\ \delta(\rho - x^*\cos\theta - y^*\sin\theta)\ dx^*dy^* \end{aligned} \tag{3.18}$$

so the core of (ρ, θ) Hough transform is that each point in the image is transformed into a sinusoid in the discrete parameter domain, and in Algorithm 3.3 the Hough transform is shown using (ρ, θ)

parameters. The easiest implementation distributes the pixel value for every value of θ_t. In the following the image is assumed quadratic, cf. Eqs. 2.24 and 2.25, and the sampling parameters defined in Eq. 2.23 are used.

Algorithm 3.3 : The (ρ, θ) Hough Transform

```
Set g_hough(r,t)=0 for all (r,t)                      //Initialize Hough space
For m=0 to M-1                                        //For all values of x
  For n=0 to N-1                                      //For all values of y
    gvalue = g(m,n)                                   //Store value in a simple variable
    If gvalue≠0                                       //Only consider non-zero pixels
      For t=0 to T-1                                  //For all values of theta
        rho=x(m)*cos(theta(t))+y(n)*sin(theta(t))     //Calculate the rho
        r=round((rho-rho_min)/Delta_rho)              //Find the right sample
        If r≥0 and r<R                                //Check if sample is valid
          g_hough(r,t)=g_hough(r,t)+gvalue            //Update parameter domain
        End
      End
    End
  End
End
```

This algorithm is valid, but the algorithm still needs a lot of checking in the inner parts of the loops, where it is computational expensive, so for increasing the efficiency several parts should be optimized. Ideas for this is shown in Subsection 3.2.2.

3.2.1 Choosing sampling parameters with the (ρ, θ) Hough Transform

Following the procedure shown in Subsection 3.1.2, a similar expression for the nearest neighbour mapping is found

$$r^* = \frac{x_m \cos\theta_t + y \sin\theta_t - \rho_{min}}{\Delta\rho} \text{ and } r = [r^*] \Rightarrow \tag{3.19}$$

$$\left| m\cos\theta_t + n\sin\theta_t - \left(\frac{\rho_{min} - x_{min}(\cos\theta_t + \sin\theta_t)}{\Delta x} \right) \right| < \frac{1}{2}\frac{\Delta\rho}{\Delta x} \tag{3.20}$$

which shows that the Hough transform maps all pixels in the image that lies within a band of width $\frac{\Delta\rho}{\Delta x}$ into the same point in the discrete parameter domain. The width is measured in pixels perpendicular to the line. This implies that $\Delta\rho$ has to be chosen sufficiently large, and not below Δx, i.e.,

$$\Delta\rho \geq \Delta x \tag{3.21}$$

Concerning the choice of $\Delta\theta$, it should be avoided that a pixel in the image is mapped into a perforated sinusoid. Using a first order approximation gives

$$\max_{x_m, y_n} \left| \frac{\partial r^*}{\partial \theta} \right| \Delta\theta < 1 \;\Rightarrow \max_{x_m, y_n} \frac{\sqrt{x_m^2 + y_n^2}}{\Delta\rho}\Delta\theta < 1 \;\Rightarrow \Delta\theta < \frac{\sqrt{2}}{(M-1)}\frac{\Delta\rho}{\Delta x} \tag{3.22}$$

In Fig. 3.9 an image with five non zero pixels is shown, and in Fig. 3.10 the corresponding discrete Hough transform. The sampling parameters have been set from Eq. 3.21 and only using $T = 50$, where Eq. 3.22 and Eq. 2.30 implies that $T = 142$ is necessary.

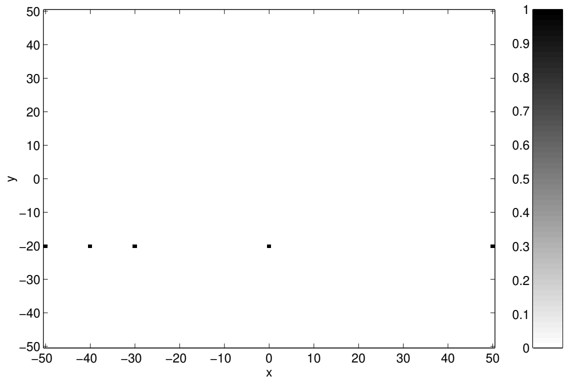

Figure 3.9 An image with 5 points lying on a line.

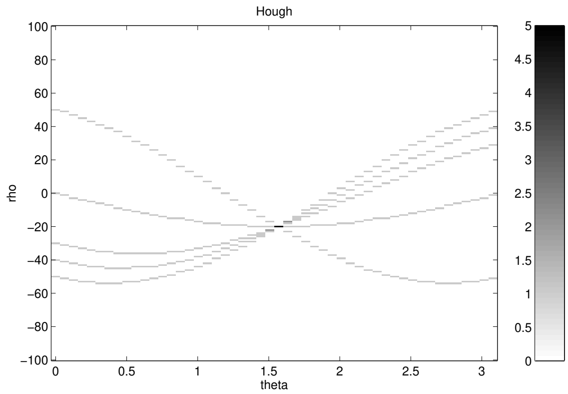

Figure 3.10 The corresponding Hough transform. Note the perforation of the sinusoids.

Note that the discrete Radon and discrete Hough transform can give identical result in the (p, τ) case, but in the (ρ, θ) case the results will always (somewhat) differ. This is also clear from the way sampling of the discrete parameter domain should be chosen.

For further reading on choosing sampling parameters [27] is recommended.

3.2.2 Comparison Between Different Optimization Strategies

Here four different optimizations strategies are compared.

Case 1 No particular optimization as it is shown in Algorithm 3.3.

Case 2 In this case, it is exploited that the fundamental mapping can be written by computing the four vectors containing relative x- and y-values, and cosine and sine to the possible values of θ_t.

$$\text{xrel}(m) = \frac{x_m}{\Delta\rho},\ \text{yrel}(n) = \frac{y_n}{\Delta\rho}, \text{and rhooff} = \frac{\rho_{min}}{\Delta\rho} \tag{3.23}$$

$$\text{costheta}(t) = \cos\theta_t \text{ and sintheta}(t) = \sin\theta_t \tag{3.24}$$

These arrays are computed one time only and used in the t-loop, hence extra calculations of sines and cosines are avoided, as shown in following equation

$$r = [\text{xrel}(m)\text{costheta}(t) + \text{yrel}(n)\text{sintheta}(t) + \text{rhooff}] \tag{3.25}$$

Case 3 In Algorithm 3.3 it is necessary to check that the value of r lies within the image. The cure for this problem can be incorporated in the design of discrete parameter domain, simply by increasing R, i.e., the number of samples in the ρ-direction to fulfill

$$\left.\begin{array}{l} \rho_{min} = -\frac{R-1}{2}\Delta\rho \\ 0 \leq \left[\frac{x_m\cos\theta_t + y_n\sin\theta_t - \rho_{min}}{\Delta\rho}\right] < R \text{ for all } (m,n) \end{array}\right\} \Rightarrow \tag{3.26}$$

$$\max_{x_m, y_n} |x_m\cos\theta_t + y_n\sin\theta_t| = \max_{x_m, y_n} \sqrt{x_m^2 + y_n^2} = \sqrt{2}|x_{min}| < \frac{R}{2}\Delta\rho \Rightarrow \tag{3.27}$$

$$R > \frac{\Delta x}{\Delta\rho}\sqrt{2}(M-1) \tag{3.28}$$

where it has been used that $x_{min} = -\Delta x(M-1)/2$.

Case 3 assumes that Eq. 3.28 is fulfilled and otherwise the implementation is like in Case 2. Normally the expansion of the parameter domain will require extra memory, and in practice no extra computations are needed for that purpose.

Case 4 This case is like Case 3, but for the mapping two matrices a calculated from the vectors shown in Case 2.

$$\mathrm{xc}(m,t) = \mathrm{xrel}(m) * \mathrm{costheta}(t) \text{ and } \mathrm{ys}(n,t) = \mathrm{yrel}(n) * \mathrm{sintheta}(t) - \mathrm{rhooff} \tag{3.29}$$

Hence for the mapping it can simply be used that

$$r = [\mathrm{xc}(m,t) + \mathrm{ys}(n,t)] \tag{3.30}$$

A further reduction of the computational cost can be achieved from the fact that some platforms provides fast truncation of floating-point numbers compared with the round-function. In this case the constant rhooff can be added with 0.5, and truncation is used instead of the rounding function.

This optimization technique will require two matrices and the computational load is lowered significantly if the tables are used many times, i.e., if the image contains few non-zero pixels and the algorithm only has to be used once, then the overhead associated with the computation of the matrices xc and ys will degrade the performance.

Yet another strategy to optimize the calculation of the sinusoid, can be found in [26]. The idea is to use the fact that ρ can be written as $\tilde{\rho}_{m,n}\cos(\theta_t - \tilde{\theta}_{m,n})$, where $x_m + jy_n = \tilde{\rho}_{m,n}e^{j\tilde{\theta}_{m,n}}$. This strategy implies that one table of cosine values is needed, but additional multiplications are needed and interpolation in the table is necessary. The techniques demonstrated in Cases 2, 3, and 4 involve no additional approximation.

First, the times for generating Fig. 3.10 are measured in the four cases shown above. This corresponds to Fig. 3.9 with only five non-zero pixels. Second, the image has been altered by adding a very small bias in order to have all pixel values considered non-zero, hence all 10201 pixels take up time in the transform. From Table 3.4 it can be seen that initialization of the arrays uses too much time in the case of only 5 non-zero pixels, so that the four cases are more or less identical in performance, but in the case of 10201 pixels, it can be seen that the case 3 and 4 outperforms the two other cases. The measurements have been made using a 120 MHz Pentium, hence the ratios look somewhat different with other processor types. Table 3.4 also demonstrates the huge time-difference found when going from 5 to 10201 non-zero pixels in the Hough transform, and in this case, the small additive bias does not imply any visible change of the parameter domain. This also demonstrates that preprocessing, e.g., in the simplest case as a thresholding of the images can imply a huge speedup of the program.

Case	Time (5 Pixels)	Time (10201 Pixels)
1: No optimization	8 msec	1884 msec
2: Vector mapping	10 msec	742 msec
3: Avoid limit check and use vector mapping	8 msec	478 msec
4: Avoid limit check and use matrix mapping	7 msec	454 msec

Table 3.4 Time consumption for the different types of optimization.

3.2.3 Other Hough-like Algorithms

A huge number of authors have proposed other kinds of Radon or Hough-like algorithms. Here only a few of them will be mentioned. Furthermore [25, 28, 29] are recommended for further reading.

3.2.3.1 Random Radon transform

One of the more radically different ideas is the Random Radon transform [30, 31]. The output of the algorithm is still peaks in the parameter domain at the positions of the lines. The main idea can be generalized to handle more general types of curves [32], but here it will only be presented for detection of lines in binary images using normal line parameters.

Basically the idea is to take two non-zero pixels randomly from the image from positions (x_1, y_1) and (x_2, y_2), and then increase the parameter domain with one vote at the line parameter (ρ, θ) matching both pixels, which is given by

$$\tan\theta = \frac{x_2 - x_1}{y_1 - y_2} \text{ and } \rho = x_1 \cos\theta + y_1 \sin\theta \tag{3.31}$$

This procedure is repeated over and over again. Each iteration is very fast, and a lot of refinements are possible with this strategy, e.g., with respect to storage of the parameter domain [33], but the really tricky part is to choose a good stopping criterion. Stopping the iteration too fast a bad estimate of the parameter domain will be obtained, and stopping too late means waste of computing time. The methods works and is potentially very fast, but is iterative.

3.2.3.2 The Gradient Method

In [34] a method for increasing the speed of the (generalized) Radon transform. The idea is here demonstrated for the slant stacking using continuous parameters. If the tangent of the line (curve) can be estimated locally in the image domain, then the computational cost can be lowered significantly. Given an image point with non-zero value at position (x_0, y_0) and that the local estimate of the slope is $\hat{p}$, then the image point only needs to get mapped into a single point in the parameter domain

$$(x_0, y_0)|\hat{p} \ \rightarrow \ (p = \hat{p}, \tau = y_0 - \hat{p}x_0) \tag{3.32}$$

which implies that the computational cost will decrease with an order or magnitude, due to the point to point mapping. The method heavily depends on the quality of the local gradient detector and the noise level.

3.2.3.3 The Hierarchical Hough Transform

In [35] and [25] a hierarchical Hough algorithm for line detection in binary images is presented. The method first uses a very coarse sampling of the parameter domain, to which all non-zero pixels are transformed. Due to the very coarse sampling this operation is very fast. Then the areas of the parameter with zero or almost zero votes are rejected for further investigation. The areas of high votes are very likely to hold line parameters, hence they are divided into smaller pieces (the sampling intervals in the parameter domain are lowered), and the method is repeated until line parameters have been detected with a high accuracy, compared to the initial resolution. The method is potentially fast if large areas of the parameter domain can be rejected in the early stages of the method.

3.2.3.4 A Length Invariant Transform

In some curve detection algorithms the actual length of the curve is without interest, hence the Radon transform using (ρ, θ) as parameters could be redefined as

$$\breve{g}(\rho, \theta) = \frac{1}{L(\rho, \theta)} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) \delta(\rho - x \cos\theta - y \sin\theta) \; dx \; dy \tag{3.33}$$

where $L(\rho, \theta)$ is the length of the line with parameters (ρ, θ) that lies within the image. That length is actually the Radon transform of a square box, so it is given is subsection B.3.2 along with the scaling property given in Eq. B.22.

In that way the parameter domain will show a value close to 1 if the image contains lines with amplitude 1. The price of this is that the Radon transform will be highly dominated by noise when $|\rho|$ gets large, due to a decrease in $L(\rho, \theta)$.

This alteration can very easily be incorporated in the Hough transform as well, but it should solely be used as a post-Hough weighting function of the parameter domain, and not used directly in the mapping, due to a heavy and very unneeded rise in the computational load.

3.3 Summary

It has been shown that the (p, τ) Hough transform can be defined in a way that gives exactly the same discrete parameter domain as found with the nearest neighbour approximation of the discrete Radon transform. The (ρ, θ) Hough transform does not have the same property.

It has also been shown that the Hough transform behaves very differently when changing the sampling intervals in the discrete parameter domain, compared to the discrete Radon transform.

Different optimization schemes has been presented for the (ρ, θ) Hough transform and a set of other Hough-like transforms have been described briefly.

Chapter 4

The FCE-Algorithm

A recurring problem in computer image processing is the identification of curves with specific shapes. The transform developed by Hough [23] for detection of complex patterns in binary digital images has been discussed by several authors, e.g., [24], [25] and generalized for multi-dimensional pattern detection in [35].

As the previous chapters have shown, the Radon transform is a mapping from an image domain to a parameter domain, where the parameters characterize the curves to be identified. The Radon transform can be generalized to handle arbitrary curves [14]. The Hough version of the generalized Radon transform detects specific values of parameters, as spatially extended curves are transformed to produce spatially compact features in the parameter domain. In this way, the generalized Radon transform converts a difficult global detection problem in the image domain into a more easily solved local peak detection problem in the parameter domain.

In recent years, progress has been made to understand and increase the speed of the generalized Radon transform [36, 37, 35]. Investigations concerning both the traditional and more recent generalized Radon transform for curve identification schemes point out two problems with respect to computational cost. First, the estimation schemes are not capable of exploiting image points with zero value (zero image points), and second, estimation is computationally expensive, as it estimates unneeded information. This is a consequence of parameter estimation, in areas of the parameter domain, where curve parameters are very unlikely.

The basis for curve parameter estimation as described in this chapter is a binary image. The binary image can be produced in several ways, e.g., by edge filtering, deconvolution, or mean field annealing.

In this chapter, a new algorithm for fast curve estimation, called the FCE-algorithm, is presented. The key idea of the FCE-algorithm is to pre-condition the parameter domain, creating irregular regions corresponding to the parameter regions of interest. Furthermore, the FCE-algorithm takes advantage of pixels with zero value in generating the pre-condition map. Initially, the FCE-algorithm identifies regions of the parameter domain which contain peaks representing curves in the image. Subsequently, it estimates a traditional generalized Radon transform within these regions.

The identification of hyperbolas is of particular interest within seismic signal processing. Over the years the seismic industry has developed a method of recording seismic signals resulting in images in which the investigated reflections produce curves. The recording geometry of seismic data acquisition results in signals having the same geometrical mid point between source and receiver, see, e.g., [38, 39]. The set of signals corresponding to the same mid point is called a common mid point gather (CMP-gather). A common technique in seismic data analysis for estimation of hyperbola parameters is velocity analysis, e.g., [19]. Within the presented work some features from velocity analysis are incorporated, however, the presented method has less computational cost.

Recently, the multipulse excited speech coding technique, which was originally proposed by Atal *et al* [40], has been suggested for seismic deconvolution by Cookey *et al.* [41]. Basically, the multipulse model for speech assumes the speech signal to be a result of an impulse train transformed by the shape of the vocal tract filter. In seismic deconvolution, the input signal is often modeled as an all-pole impulse and the layered earth reflectivity model is considered as an impulse train, see, e.g., [39]. Thus, the multipulse technique can be used to map seismic signals in a CMP-gather into a binary image, where the image value is set to one in case of a reflection and zero otherwise. The reflections represent information concerning the geological structure of the subsurface.

The generalized Radon transform is described in Section 4.1 with some of its properties, regarding curve detection. Creation of the pre-conditioning map is described in Section 4.2, where the *image point mapping* procedure is developed. In section 4.3, parameter domain sampling is discussed and section 4.4 describes parameter domain blurring, which is a result of discretization. The FCE-algorithm is presented in section 4.5, and is applied to two numerical examples for detection of hyperbolas in Section 4.6. The FCE-algorithm is also presented in [1, 2, 42].

It should be mentioned that many authors have used slant stacking for interpretation of the seismic gathers [43, 44, 45, 46, 47]. Note also that the use of the generalized Radon transform for detection of curve parameters without forming a binary image can be found in [48, 42].

4.1 The Generalized Radon Transform

Let $g(x, y)$ be a continuous signal of the continuous variables x and y and let $\boldsymbol{\xi}$ denote an η-dimensional parameter vector defined as

$$\boldsymbol{\xi} = (\xi_1, \ldots, \xi_i, \ldots, \xi_\eta) \tag{4.1}$$

where $\boldsymbol{\xi}$ spans the parameter domain.

4.1.1 The Continuous Generalized Radon Transform

The generalized Radon transform can be defined in various ways [15, 13]. A very general form is

$$\check{g}(\boldsymbol{\xi}) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y)\ \delta(\phi(x, y; \boldsymbol{\xi}))\ dx\ dy \tag{4.2}$$

Using the definition given in Eq. 4.2, shapes expressed by the parameter form $\phi(x, y; \boldsymbol{\xi}) = 0$ can be detected. However, a subset of Eq. 4.2 will be used in the following, noting that most of the techniques to be presented later in this chapter can be modified to conform to the more general forms of the generalized Radon transform.

Let $\check{g}(\boldsymbol{\xi})$ denote the continuous generalized Radon transform of the function $g(x, y)$, which here is defined as

$$\check{g}(\boldsymbol{\xi}) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y)\ \delta(y - \phi(x; \boldsymbol{\xi}))\ dx\ dy = \int_{-\infty}^{\infty} g(x, \phi(x; \boldsymbol{\xi}))\ dx \tag{4.3}$$

where $y = \phi(x; \boldsymbol{\xi})$ denotes the transformation curve. The generalized Radon transform is the integration of $g(x, y)$ along the transformation curve. A Jacobian-like weighting function in front of the integral can be included if other weighting schemes is required.

Eq. 4.3 can be interpreted as the generalization of the slant stacking technique described in Chapter 1. Note that Eq. 4.3 only allows curves to have one value of y for each value of x and the parameter $\boldsymbol{\xi}$. This excludes, e.g., closed curves such as circles. In this case Eq. 4.2 can be used.

The curve to be detected is now modelled using a delta function following the curve shape.

$$g(x, y) = \delta(y - \phi(x; \boldsymbol{\xi}^*) \tag{4.4}$$

hence the generalized Radon transform is given by

$$\begin{aligned} \breve{g}(\boldsymbol{\xi}) &= \int_{-\infty}^{\infty} \delta(\phi(x; \boldsymbol{\xi}) - \phi(x; \boldsymbol{\xi}^*)) \; dx & (4.5) \\ &= \int_{-\infty}^{\infty} \sum_{i=1}^{I} \frac{\delta(x - x_i)}{\left| \frac{\partial \phi(x;\boldsymbol{\xi})}{\partial x} - \frac{\partial \phi(x;\boldsymbol{\xi}^*)}{\partial x} \right|} \; dx & (4.6) \\ &= \sum_{i=1}^{I} \frac{1}{\left| \frac{\partial \phi(x_i;\boldsymbol{\xi})}{\partial x} - \frac{\partial \phi(x_i;\boldsymbol{\xi}^*)}{\partial x} \right|} & (4.7) \end{aligned}$$

where each of the I values x_i, cf. Eq. A.4, fulfills $\phi(x_i; \boldsymbol{\xi}) = \phi(x_i; \boldsymbol{\xi}^*)$.

Eq. 4.7 shows that the generalized Radon transform will give an infinite peak, when $\boldsymbol{\xi} = \boldsymbol{\xi}^*$, and the parameter domain will also contain some other non-zero values, depending on the value of I and the actual function ϕ. The equation also shows that the transform will malfunction when slope of the curve $\partial\phi/\partial x$ is infinite, i.e., when the tangent to the curve in a (x, y) plane is vertical. In conclusion, Eq. 4.3 is well suited for curves like parabolas, hyperbolas, and other parameterized curves with limited slope. Note that a generalization of the technique described in Section 1.7 can be used to expand the applicability of the transform.

4.1.2 The Discrete Generalized Radon Transform

Let $\boldsymbol{j}$ denote the η-dimensional discrete index parameter vector defined as

$$\boldsymbol{j} = (j_1, \ldots, j_i, \ldots, j_\eta) \tag{4.8}$$

The correspondence between the index vector $\boldsymbol{j}$ and the sampled version of the parameter vector, denoted $\boldsymbol{\xi}_{\boldsymbol{j}}$ can be written as

$$\boldsymbol{\xi} = \boldsymbol{\xi}_{\boldsymbol{j}} = \boldsymbol{\theta}(\boldsymbol{j}), \quad \text{where} \quad \xi_i = \theta_i(j_i) \tag{4.9}$$

where $\theta_i(j_i)$ is a parameter sampling function.

If a uniform sampling of the parameter domain is chosen, the function θ_i can be written

$$\xi_i = \theta_i(j_i) = \xi_{i,min} + j_i \; \Delta\xi_i, \quad j_i = 0, \ldots, J_i - 1 \tag{4.10}$$

where $\xi_{i,min}$ denotes the lower limit and $\Delta\xi_i$ the sampling interval of ξ_i.

The parameter domain and image domain sampling are assumed to be uniform, as defined in Eq. 1.12, i.e.,

$$\begin{aligned} x = x_m &= x_{min} + m \; \Delta x, \quad m = 0, 1, \ldots, M - 1 & (4.11) \\ y = y_n &= y_{min} + n \; \Delta y, \quad n = 0, 1, \ldots, N - 1 & (4.12) \end{aligned}$$

Substitution of Eqs. 4.9, 4.11 and 4.12 into the transformation curve gives

$$y = y_{min} + n \; \Delta y = \phi(x_{min} + m \; \Delta x; \boldsymbol{\theta}(\boldsymbol{j})) \tag{4.13}$$

Using a nearest neighbour approximation of Eq. 4.13, the discrete index transformation curve $\phi(m;\boldsymbol{j})$ can be expressed as (note the change of arguments)

$$\phi(m;\boldsymbol{j}) = n = \left[\frac{\phi(x_{min} + m\ \Delta x; \boldsymbol{\theta}(\boldsymbol{j})) - y_{min}}{\Delta y}\right] \tag{4.14}$$

Let $g(m,n)$ denote the discretized version of the signal $g(x,y)$, i.e., $g(m,n) = g(x_m, y_n)$ and let $\check{g}(\boldsymbol{j})$ denote the discrete generalized Radon transform (GRT) of $g(m,n)$ defined as

$$\check{g}(\boldsymbol{j}) = \sum_{m=0}^{M-1} g(m, \phi(m;\boldsymbol{j})) \tag{4.15}$$

Notice, that the term dx has been omitted in the discrete generalized Radon transform as it is a constant due to the uniform sampling of x. If the discrete generalized Radon transform should quantitatively approximate Eq. 4.3, then Eq. 4.15 must be multiplied by Δx.

The use of rounding instead of, e.g., linear interpolation will normally not lead to problems, when using a sufficient sampling of the parameter domain, but interpolations schemes can easily be incorporated in Eqs. 4.14 and 4.15.

4.2 Image Point Mapping

The first part of Eq. 4.3 can be used to define another way of estimating a discrete parameter domain

$$\check{g}(\boldsymbol{j}) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} g(m,n)\, \delta(n - \phi(m;\boldsymbol{j})) \tag{4.16}$$

where $\delta(\cdot)$ denotes the Kronecker delta function.

Eq. 4.16 shows that each image point (m,n) is transformed into a parameter curve where $n = \phi(m;\boldsymbol{j})$. Thus mapping of image points with zero value is needless, since they will not contribute to $\check{g}(\boldsymbol{j})$. The proposed estimation scheme for the Radon transform accounts for the fact that zero image values do not contribute, and does not include them in the summation in Eq. 4.16. This has the potential to reduce the computational cost considerably.

Now, assume that ϕ is invertible in one of the parameters ξ_i, e.g., ξ_η. The inverse function $\phi_{\xi_\eta}^{-1}$ with respect to ξ_η can be written as

$$\xi_\eta = \phi_{\xi_\eta}^{-1}(x, y; \boldsymbol{\xi}_r), \qquad \boldsymbol{\xi}_r = (\xi_1, \ldots, \xi_{\eta-1}) \tag{4.17}$$

Furthermore, assume that the sampling function θ_η is invertible. Then

$$j_\eta = \left[\theta_\eta^{-1}(\xi_\eta)\right] = \left[(\theta_\eta^{-1}(\phi_{\xi_\eta}^{-1}(x, y; \boldsymbol{\xi}_r)\right] \equiv \Omega(m, n; \boldsymbol{j}_r), \qquad \boldsymbol{j}_r = (j_1, \ldots, j_{\eta-1}) \tag{4.18}$$

Eqs. 4.16 and 4.18, are the basis for an estimation scheme for the generalized Radon transform which here is referred to here as Image Point Mapping (IPM). The key steps of IPM can be summarized as

1. Initialize $\check{g}(\boldsymbol{j}) = 0$ for all $\boldsymbol{j}$
2. For all image points $g(m,n)$ with a value different from zero do
   For all possible $\boldsymbol{j}_r$ do
   $\boldsymbol{j} = (\boldsymbol{j}_r, \Omega(m,n;\boldsymbol{j}_r))$
   $\check{g}(\boldsymbol{j}) := \check{g}(\boldsymbol{j}) + g(m,n)$

Note that IPM is a generalized Hough transform [23], which has been extended to handle a large set of non-linear transformation curves. Like the Hough transform, the objective of IPM is parameter identification, by transforming each pixel in the image into the parameter domain.

One of the fundamental differences between GRT and IPM is, that GRT requires rounding in the image domain, whereas IPM rounds in the parameter domain, as schematically shown in Fig. 4.1.

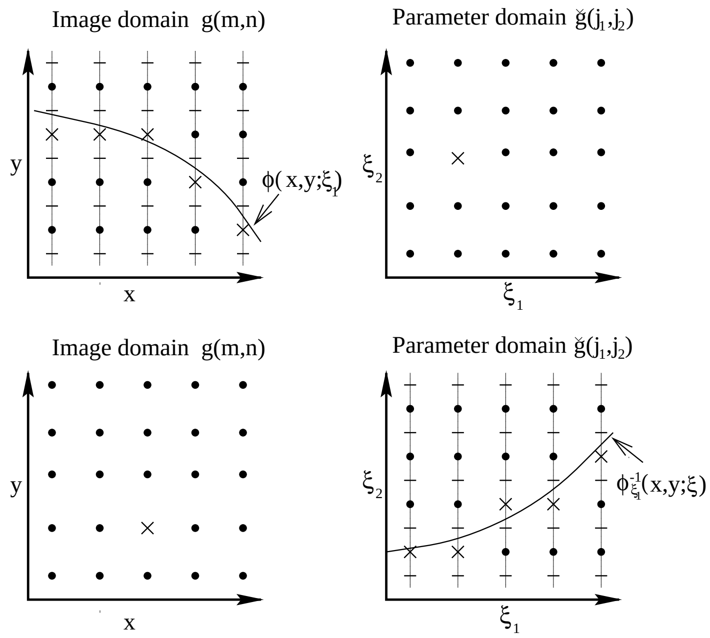

Figure 4.1 Top: Rounding by GRT in the image domain corresponding to a point in the parameter domain. Bottom: Rounding by IPM in the parameter domain corresponding to a point in the image domain.

The computational complexities of GRT and IPM, respectively, are

$$\mathcal{O}_{GRT} = \mathcal{O}\left(M \prod_{i=1}^{\eta} J_i\right) \qquad \mathcal{O}_{IPM} = \mathcal{O}\left((M\,N)_r \prod_{i=1}^{\eta-1} J_i\right) \tag{4.19}$$

where $(M\,N)_r$ indicates that only a reduced number of image points (non-zero values) are to be transformed. In Eq. 4.19, the cost to test for non-zero values is assumed to be negligible.

The major advantage of IPM over GRT is the ability of IPM to ignore image points with a value of zero, making IPM especially well suited for sparse binary images.

If the absolute value between two successive j_η values is greater than one, i.e.,

$$|\Omega(m,n;\boldsymbol{j}_r + \boldsymbol{\varepsilon}) - \Omega(m,n;\boldsymbol{j}_r)| > 1 \tag{4.20}$$

where vector $\boldsymbol{\varepsilon}$ contains zeros at all $\eta - 1$ entries and holds unit value at entry i, i.e., $\varepsilon_i = 1$, it is seen that IPM will map the image point (m,n) into a perforated hyper-curve, i.e., a hyper-curve containing missing sections. Identification of curve parameters can be difficult due to the presence of perforation holes depending on the shape and depth of the holes.

For $\eta = 2$ the perforation problem can be eliminated simply by adding $g(m,n)$ to $\breve{g}(\boldsymbol{j})$ for the parameter values skipped between two successive parameter vectors. Consider, for example,

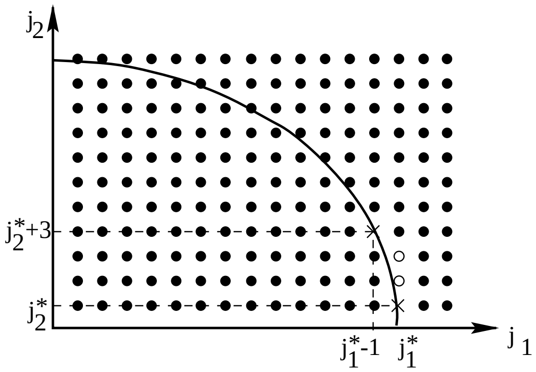

Figure 4.2 Illustration of the perforation problem.

the hyperbolic transformation curve as illustrated in Fig. 4.2, where the parameter domain is described by $(j_1, j_2) = (j_1^*, j_2^*)$.

Fig. 4.2 shows the mapping of an image point (m^*, n^*) to the parameter domain, that is, discretized parameter point accumulators to which the value $d(m^*, n^*)$ of the image point must be added. To investigate the perforation problem, two parameter points (s^*-1, z^*+3) and (s^*, z^*) on the inverse transformation curve are considered. Following the inverse transformation curve from (s^*-1, z^*+3) to (s^*, z^*) along the s axis gives only the two parameter points (s^*-1, z^*+3) and (s^*, z^*). However, following the inverse transformation curve along the z axis also gives the two intervening points (s^*, z^*+2) and (s^*, z^*+1). One way of handling such intervening points is to add the value of the image point to the parameter points skipped between two successive parameter points on the inverse transformation curve. Therefore, in this example, $d(m^*, n^*)$ must be added to the skipped parameter points (s^*, z^*+2) and (s^*, z^*+1) when following the s axis.

4.3 Parameter Domain Sampling

For a given image $g(m, n)$ an exact determination of $\Delta\xi_i$, $\xi_{i,min}$ and J_i, which match the image, is, in general, not possible. However, some guidelines can be stated. First, some of the parameters will be bounded by the underlying physics, e.g., $\xi_{i,min}$ and $\xi_{i,max}$ will normally be limited. Second, the parameter domain can be limited by requiring that at least a fraction β of the image points addressed by the transformation curve $\phi(m; \boldsymbol{j})$ lie inside the image, i.e.,

$$\sum_{m=0}^{M-1} I\left\{y_{min} < \phi(x_m; \boldsymbol{\xi}) < y_{max}\right\} > \beta\, M, \qquad 0 < \beta < 1, \qquad \forall \boldsymbol{\xi} \tag{4.21}$$

where $I\{\cdot\}$ equals 1 when the logical expression is true and 0 otherwise.

Concerning use of the GRT, the sampling intervals $\Delta\xi_i$ can be chosen by requiring that

$$\max\{|\phi(x; \boldsymbol{\xi} + \boldsymbol{\gamma_i}) - \phi(x; \boldsymbol{\xi})|\} \;=\; \Delta y, \qquad \forall\, x, \boldsymbol{\xi}, \boldsymbol{\gamma}_i \tag{4.22}$$

where $\boldsymbol{\gamma}_i$ is a η-dimensional vector containing zeros at all entries except entry i which is $\Delta\xi_i$.

Eq. 4.22 states that two adjacent $\boldsymbol{\xi}$-vectors give $y = \phi(x; \boldsymbol{\xi})$ values which cannot be separated by more than one sample in the y-direction. This design criterion is not optimal as it leads to an unnecessarily dense sampling of certain parts of the parameter domain, however, it can be used as an upper sampling rate limit. Increasing the sampling rate of the parameter domain above a certain limit will not improve the resolution obtained by GRT as adjacent $\boldsymbol{j}$ vectors will result in the same curve $n = \phi(m; \boldsymbol{j})$. For GRT, a coarse sampling cannot guarantee a given curve will be detected, due to the fact that no transformation curve $\phi(m; \boldsymbol{j})$ can be guaranteed to follow the image curve perfectly.

Use of IPM with a dense parameter domain sampling according to Eq. 4.22 produces curves $j_\eta = \Omega_d(m, n; \boldsymbol{j}_r)$ for (m, n) values corresponding to an image curve which do not intersect in a parameter point $\boldsymbol{j}$, but will spread out over several parameter points. The reason for this spread is the individual treatment of image points as regions of zero size and not regions of, e.g., rectangular shape. This spread can be compensated by the use of a coarse sampling of the parameter domain. In addition, a coarse sampling will lead to a lower computational cost.

Summarizing, GRT requires a dense parameter domain sampling, while IPM gives rise to blurring in the parameter domain in the case of dense parameter domain sampling, but works well with a coarse sampling of the parameter domain. This observation is the basis for the curve parameter estimation algorithm, presented in Section 4.5.

4.4 Parameter Domain Blurring

Parameter domain blurring achieved by use of GRT can be interpreted in a way that makes it usable for parameter clustering. According to Eq. 4.3 the continuous transformation curve can be written as

$$y = \phi(x; \boldsymbol{\xi}), \qquad x \in [x_{min}, x_{max}] \tag{4.23}$$

Let Δy denote the sampling interval of y and let $\alpha \Delta y$ be a given uncertainty of y, e.g., $\alpha = 0.5$. For a parameter vector $\boldsymbol{\xi} = \boldsymbol{\theta}(\boldsymbol{j})$ to lie inside a band of width $2\alpha\Delta y$, symmetrically positioned around the image curve $y = \phi(x; \boldsymbol{\xi}^*)$, it must satisfy the following inequality

$$y - \alpha\ \Delta y < \phi(x; \boldsymbol{\xi}) < y + \alpha\ \Delta y, \qquad y = \phi(x; \boldsymbol{\xi}^*), \qquad \forall x \in [x_{min}, x_{max}] \tag{4.24}$$

Several parameter vectors $\boldsymbol{\xi}$ will correspond to image points within the band specified in Eq. 4.24. Therefore, in general, a band in the image around a given curve described by the parameter vector $\boldsymbol{\xi}^*$ will give rise to a region in the parameter space which consists of parameter vectors that correspond to curves inside the image band. These regions are named clusters and the corresponding image curve with parameter $\boldsymbol{\xi}^*$ is called a center curve. An example of a curve band around a center curve and the corresponding parameter cluster is illustrated in Fig. 4.3.

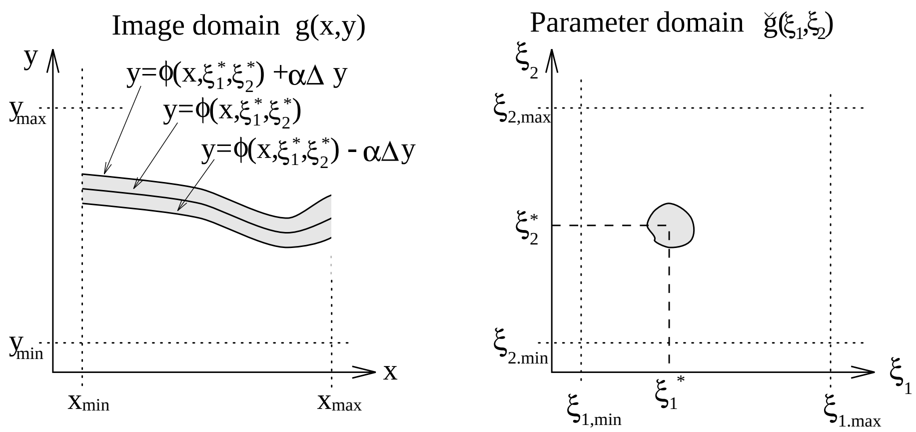

Figure 4.3 Left: Center curve with an uncertainty of $2\alpha\Delta y$ indicated. Right: Cluster corresponding to center curve.

Unfortunately, the center curves are normally unknown, and it is impossible to determine the clusters. However, it is possible to determine whether two parameter sets $\boldsymbol{\xi}_1$ and $\boldsymbol{\xi}_2$ belong to the same cluster. If two parameter vectors belong to the same cluster they must satisfy

$$|\phi(x;\boldsymbol{\xi}_1) - \phi(x;\boldsymbol{\xi}_2)| < 2\ \alpha\ \Delta y, \qquad \forall x \in [x_{min}, x_{max}] \tag{4.25}$$

Parameter domain points may be gathered into regions or clusters, with the guarantee that all possible image curves will be represented by only one cluster in the parameter domain. This partitions the parameter domain into irregular regions, which reflect the information level of the image. Clustering can also be used to estimate parameter uncertainties, e.g., the maximum and minimum parameter value within the cluster can be used to give an estimate of the parameter uncertainty as, e.g., $\frac{1}{2}(\xi_{i,max}^{\text{cluster}} - \xi_{i,min}^{\text{cluster}} + \Delta\xi_i)$. Parameter domain clustering can be performed either before or after the transform, depending on the purpose.

4.5 The Fast Curve Estimation Algorithm

Consider an image where the image values are represented by continuous values. This image can be mapped into a binary image $g(m,n)$, e.g., by edge filtering [17, 49], by deconvolution [50, 41], or by mean field annealing [42, 51]. The proposed algorithm estimates the parameters of curves in the binary image $g(m,n)$, e.g., lines or hyperbolas.

It is well-known that direct use of the GRT is computationally expensive. In the light of the characteristics of GRT and IPM, a fast curve parameter estimation algorithm, the FCE-algorithm, is proposed which simultaneously estimates all parameters of curves having a specific shape, e.g., lines or hyperbolas. The FCE-algorithm uses IPM as a pre-conditioning procedure for GRT by selecting the regions of interest in the parameter domain. IPM is suitable for a rapid determination of regions of interest, as it works well on a coarsely sampled parameter domain and is capable of ignoring image points with a value of zero. Another important circumstance is that GRT relates the uncertainty on the estimated curve parameters to the image domain sampling, while IPM relates it to the parameter domain sampling. The FCE-algorithm is shown in Fig. 4.4, and is summarized as follows:

1. Design the discrete parameter domain, i.e., choose $\xi_{i,min}$, J_i and $\Delta\xi_i$.

2. Design a reduced parameter domain for IPM by choosing

$$J_i' = \left\lceil \frac{J_i}{2v_{\xi_i}+1} \right\rceil, \qquad \xi_{i,min}' = \xi_{min} + v_{\xi_i}\Delta\xi_i, \qquad \Delta\xi_i{}' = (2v_{\xi_i}+1)\Delta\xi_i \tag{4.26}$$

   where $\lceil\cdot\rceil$ rounds to the nearest upper integer, and v_{ξ_i} is an integer related to the resampling (see below). Then use IPM to estimate $\breve{g}_{ipm}(\boldsymbol{j'})$ as

$$\breve{g}_{ipm}(\boldsymbol{j'}) = \text{IPM}\{g(m,n)\} \tag{4.27}$$

3. Use the threshold function T to give

$$\breve{g}_t(\boldsymbol{j'}) = T(\breve{g}_{ipm}(\boldsymbol{j'})) \tag{4.28}$$

   The threshold function T is designed to remove all insignificant parameter combinations and a suitable choice might be

$$T(\breve{g}_{ipm}(\boldsymbol{j'})) = \mu(\breve{g}_{ipm}(\boldsymbol{j'}) - \lambda_1 M) \tag{4.29}$$

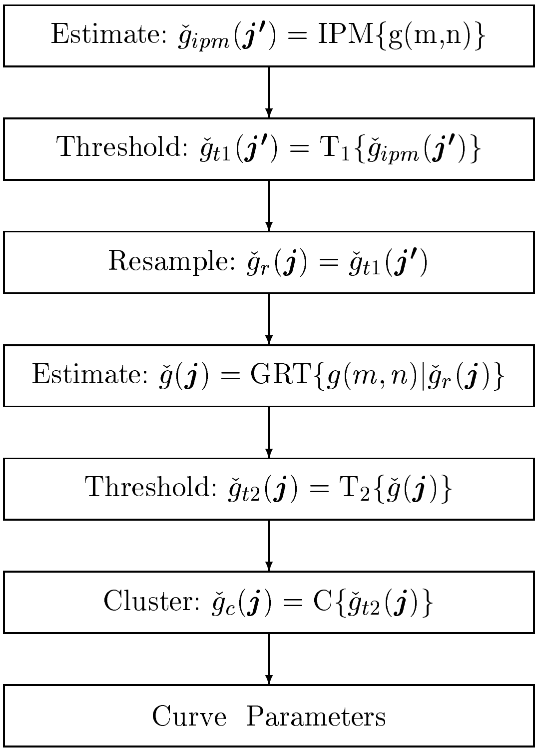

Figure 4.4 Flow diagram of the FCE-algorithm.

where M denotes the number of image lines, λ_1 a fraction defining the significance level, and $\mu(\cdot)$ the Hamilton step function. Having an image with one curve, IPM gives a corresponding parameter domain value of M. Choosing a threshold level of $\lambda_1 M$ allows some deviation for the image curve from the integration curve, e.g., to handle holes.

4. Resample the parameter domain to obtain $\breve{g}_r(\boldsymbol{j})$. The resampling process can be written as

$$\breve{g}_r(\boldsymbol{j}) = \breve{g}_t(\boldsymbol{j'}), \qquad j'_i = \left[\frac{j_i - v_{\xi_i}}{2v_{\xi_i} + 1}\right] \tag{4.30}$$

This quickly fills the entire parameter domain where the interesting regions have value one while other regions have value zero.

5. Use GRT to estimate $\breve{g}(\boldsymbol{j})$ in the regions of the parameter domain where $\breve{g}_r(\boldsymbol{j})$ is not equal to zero.

$$\breve{g}(\boldsymbol{j}) = \text{GRT}\{g(m,n)\,|\breve{g}_r(\boldsymbol{j}) \neq 0\} \tag{4.31}$$

6. As in step three, use the threshold function with a new significance level λ_2 to account for use of the GRT. After the thresholding, all points in the parameter domain having a value different from zero represent curve parameters.

7. Finally, the identified parameters are clustered into common image curves.

The resampling process takes each parameter point corresponding to the sampling interval $\Delta\boldsymbol{\xi}'$ and extends it to several parameter points corresponding to the sampling interval $\Delta\boldsymbol{\xi}$, $\Delta\xi_i < \Delta\xi_i'$. Thus, the FCE-algorithm operates initially in a coarsely sampled parameter domain, using IPM for determination of regions of interest. Subsequently, the sampling is refined to the required level and GRT is applied within the regions of interest.

The significance levels λ_1 and λ_2 must be chosen to reflect the parameter domain values, that can be accepted as peaks corresponding to curves in the image. In general, the precise values of λ_1 and λ_2 are not critical. For the examples, suitable ranges of values for λ_1 and λ_2 are $0.1 - 0.7$ and $0.4 - 0.8$, respectively. There are two reasons for the use of a lower acceptance level in step three. First, IPM has a tendency to blur in the parameter domain, and second, only regions of the parameter domain ensured not to contain image curves must be removed. A higher significance level in step six (λ_2) is justified by the required certainty for correct curve identification.

4.6 The Hyperbolic Transformation Curve

In marine seismic acquisition [39, 52, 19] it is common to use a sound emitter, e.g., an air gun attached to a vessel, which is towing an array of hydrophones (receivers). The setup is shown schematically in Fig. 4.5.

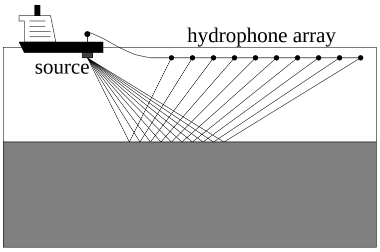

Figure 4.5 A vessel towing an array of hydrophones. A sound source emits a pulse, which is reflected at a layer-boundary.

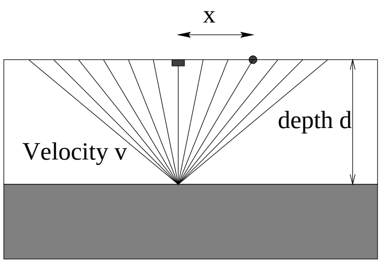

Figure 4.6 Rearranging data corresponding to a common mid point gives a CMP-gather.

As the vessel sails along a line an acquisition is made. With a certain time interval the air gun emits a pulse-like signal, which penetrates the water and top layers of the ground. A simple model is to use assume that the ground is made of homogeneous layers in which the pulse propagates linearly until the pulse is reflected at a layer-boundary and finally is recorded by the hydrophones lying near the water surface. Each of the M hydrophones record a couple of seconds of a signal after the sound emission. The signal is sampled and stored digitally. The digital signal recorded at a single hydrophone is called a trace.

Given that acquisition data is available for a large set of vessel positions relative to the earth, all the measured traces can be rearranged as if the hydrophones had a common mid point [39]. This collection of data is shown in Fig. 4.6 and is known as a CMP-gather [38, 42].

Assume that the layer between the water-level and the reflection boundary has velocity v and depth d. Then it is easy to find the time between the emission of the pulse until it is recorded at the hydrophone lying at the offset x. Using the Pythagorean theorem gives that the two way travel time t follows a hyperbolic curve as a function of x. This is called the normal moveout equation.

$$t^2 = t_0^2 + (\sigma x)^2 \text{ where } \begin{cases} \sigma = \frac{1}{v} \\ t_0 = 2d\sigma \end{cases} \tag{4.32}$$

where σ is known as the slowness, and it should be noted that the zero offset two way travel time t_0 is the value of t, when the offset x is zero.

By assuming a multi-layered earth model, it can be shown that Eq. 4.32 can model the reflection data [53], but the two parameters now depend on several layer parameters corresponding to the actual path of the wave, e.g., [54, 55, 42]. A multi-layered subsurface will generally produce several reflection hyperbolas and in the following the FCE-algorithm is used to detect the parameters of these hyperbolas. It could be mentioned that the parameters afterwards could be used in algorithms trying to reconstruct the individual layer parameters (velocity and thickness).

A digital image $f(m,n)$, containing the measurements, is formed by the finite number of different offset-values normally with a fixed distance between the hydrophones, i.e., the offset x is sampled linearly

$$x = x_m = x_{min} + m\Delta x \tag{4.33}$$

and the index n correspond to the time t, which also is sampled linearly

$$t = t_n = t_{min} + n\Delta t \tag{4.34}$$

Consider a seismic CMP-gather. The multipulse technique or mean field annealing can be used to map the seismic signals in the CMP-gather $f(m,n)$ into a binary image $g(m,n)$, where $g(m,n)$ is set to one if position n of trace number m contains a reflection, and to zero otherwise.

Hence, in this seismic case, the image parameter y is replaced by the time t, the parameter domain are spanned by σ and t_0, and the curve to detect is given by

$$t = \phi(x;\sigma,t_0) = \sqrt{t_0^2 + (\sigma x)^2} \tag{4.35}$$

which easily can be inverted with respect to t_0 to give

$$t_0 = \phi_{t_0}^{-1}(x,t;\sigma) = \sqrt{t^2 - (\sigma x)^2} \tag{4.36}$$

Both t_0 and σ are uniformly sampled and the parameter domain sampling is chosen in agreement with Eq. 4.22. The discretized parameter domain is denoted z and s corresponding to t_0 and σ, respectively,

$$t_0 = t_{0,low} + z\ \Delta t_0, \qquad z = 0,\ldots,Z-1 \tag{4.37}$$

$$\sigma = \sigma_{low} + s\ \Delta\sigma, \qquad s = 0,\ldots,S-1 \tag{4.38}$$

The curve perforation described in section 4.2 occurs in the present case of a hyperbolic transformation curve. It is, however, eliminated by adding $g(m,n)$ to $\breve{g}(s,z)$ for the parameter values skipped between two successive parameter vectors.

4.6.1 Clusters in the Hyperbolic case

Assume the parameter set (σ^*, t_0^*) corresponds to a center curve. The parameter sets (σ, t_0) corresponding to image curves within the band of width $2\alpha\Delta t$, symmetrically positioned around the image curve $t = \phi(x;\sigma^*,t_0^*)$, can according to Eq. 4.24 be written as

$$t - \alpha\ \Delta t < \sqrt{t_0^2 + (\sigma x)^2} < t + \alpha\ \Delta t, \qquad t = \sqrt{t_0^{*2} + (\sigma^* x)^2}, \qquad \forall x \in [x_{min}, x_{max}] \tag{4.39}$$

Let q denote a binary variable, $q \in \{-1, +1\}$. Thus, the lowest and highest curve within the image band specified by Eq. 4.39 can be written as

$$\sqrt{t_0^{*2} + (\sigma^* x)^2} + q\,\alpha\,\Delta t = \sqrt{t_0^2 + (\sigma x)^2}\ \Leftrightarrow t_0^2 = \left(\sqrt{t_0^{*2} + (\sigma^* x)^2} + q\,\alpha\,\Delta t\right)^2 - (\sigma x)^2 \tag{4.40}$$

In [42] it is shown, that x_{min} and x_{max} give the cluster limiting elliptical curves, the parameter cluster corresponding to the center curve (σ^*, t_0^*) can be written as

$$\sqrt{\left(\sqrt{t_0^{*2} + (\sigma^* x_{min})^2} - \alpha\, \Delta t\right)^2 - (\sigma x_{min})^2} < t_0 < \sqrt{\left(\sqrt{t_0^{*2} + (\sigma^* x_{min})^2} + \alpha\, \Delta t\right)^2 - (\sigma x_{min})^2} \quad (4.41)$$

and

$$\sqrt{\left(\sqrt{t_0^{*2} + (\sigma^* x_{max})^2} - \alpha\, \Delta t\right)^2 - (\sigma x_{max})^2} < t_0 < \sqrt{\left(\sqrt{t_0^{*2} + (\sigma^* x_{max})^2} + \alpha\, \Delta t\right)^2 - (\sigma x_{max})^2} \quad (4.42)$$

The left part of Fig. 4.7 shows a hyperbolic center curve (σ^*, t_0^*) with an uncertainty of $2\alpha\Delta t$. The right part of the figure shows the shape of the corresponding cluster, which is bounded by four elliptical curves.

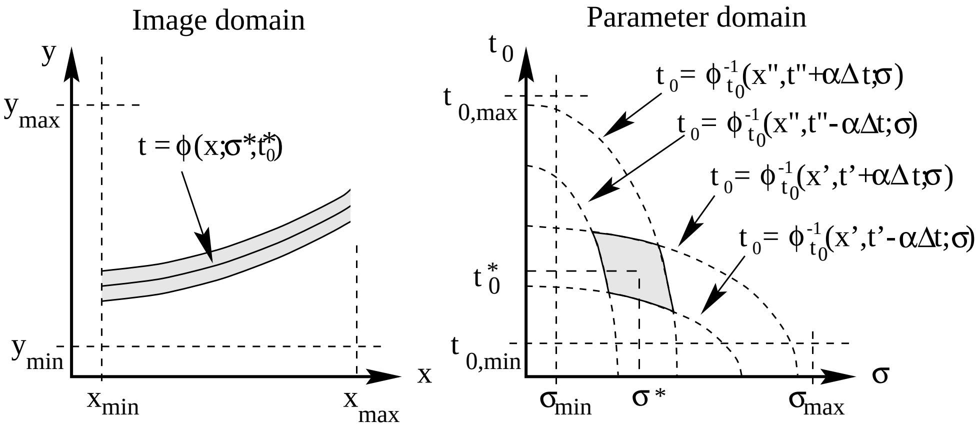

Figure 4.7 Left: A hyperbolic center curve (σ^*, t_0^*) with an uncertainty of $2\alpha\Delta t$. Right: The shape of the corresponding cluster, which is bounded by four elliptical curves, where $x' = x_{min}$, $x'' = x_{max}$, $(t')^2 = t_0^{*2} + (\sigma^* x_{min})^2$, and $(t'')^2 = t_0^{*2} + (\sigma^* x_{max})^2$.

Fig. 4.8 illustrates four clusters in the parameter domain. The four clusters correspond to four center curves with an uncertainty of $10\Delta t$, i.e., $\alpha = 5$. It should be noted that the cluster shape is highly dependent on the center curve (σ^*, t_0^*).

4.6.2 An Example with Eight Hyperbolas

In this example the potential of the FCE-algorithm is illustrated. A synthetic image is composed of eight hyperbolas, assembled into four groups of two hyperbolas each, as shown in Fig. 4.9. In order to get a binary gather, where the pixels are set to one, when a reflection is found the gather has been deconvolved, by finding the maximum cross-correlation between the gather and the shape of the reflection wavelet a number of times. The result is shown in Fig. 4.10. Here only 2.5% of the pixels has been set to 1. It can be seen that several pixels have erroneously been identified as reflections.

Using GRT according to the settings in Table 4.1 leads to eight peaks in the parameter domain as shown in Fig. 4.11. Thus, although the eight curves intersect and lie very close, GRT is able to separate the curves in the parameter domain into eight separate peaks. The figure can be compared to Fig. 4.12 showing the actual hyperbola parameters.

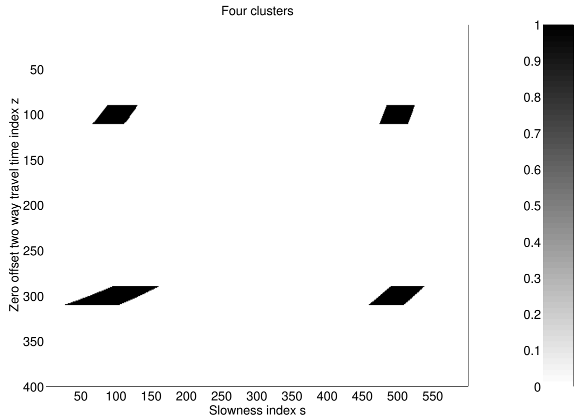

Figure 4.8 Four clusters in the parameter domain corresponding to an uncertainty of $10\Delta t$, i.e., $\alpha = 5$.

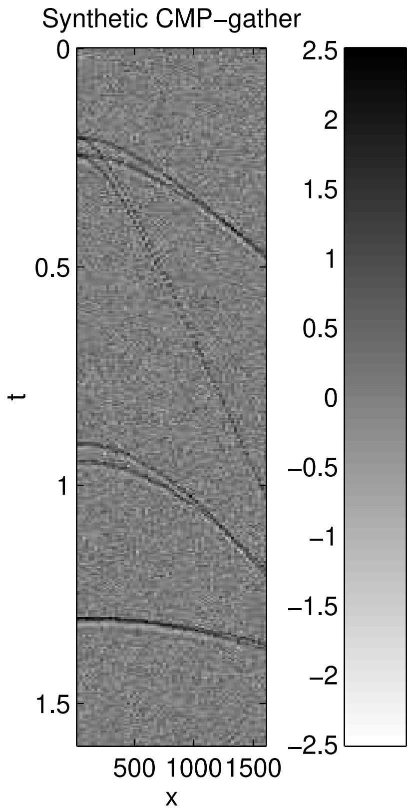

Figure 4.9 Synthetic CMP-gather composed of eight hyperbolas.

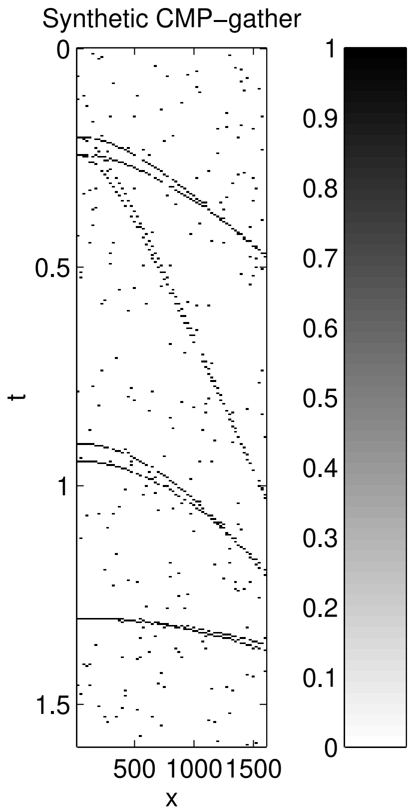

Figure 4.10 The binary deconvoluted CMP-gather.

The FCE-algorithm is used for fast detection of the interesting regions in the parameter domain resulting from GRT. The following figures show the steps of the FCE-algorithm. Fig. 4.13 shows the coarsely sampled parameter domain obtained using IPM with $v_\sigma = 4$ and $v_{t_0} = 2$.

Image domain		Parameter domain	
Parameter	Value	Parameter	Value
M	50	S	300
N	400	Z	400
Δx	$50\ m$	$\Delta\sigma$	$1.569 \cdot 10^{-6}\ s/m$
Δt	$4 \cdot 10^{-3} s$	Δt_0	$4 \cdot 10^{-3} s$
x_{min}	$100\ m$	σ_{min}	$2.3 \cdot 10^{-4}\ s/m$
t_{min}	$0\ s$	$t_{0,min}$	$0\ s$
α	1.5	λ_1	0.5
		λ_2	0.75
		v_σ	4
		v_{t_0}	2

Table 4.1 Image and parameter domain settings.

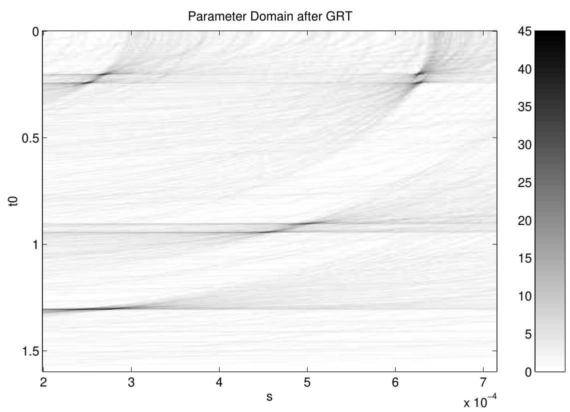

Figure 4.11 Full GRT estimated parameter domain.

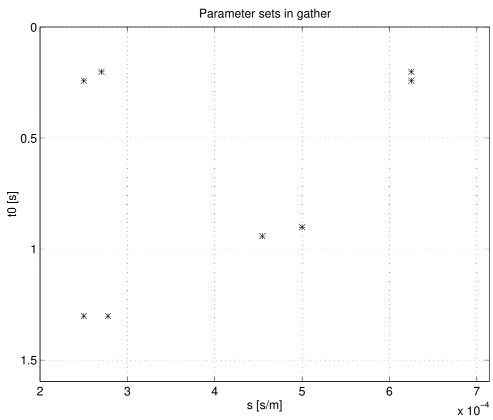

Figure 4.12 Parameters of the hyperbolas shown in the parameter domain.

The parameter domain is then thresholded using a significance level $\lambda_1 = 0.5$, and the resulting binary parameter domain is shown in Fig. 4.14.

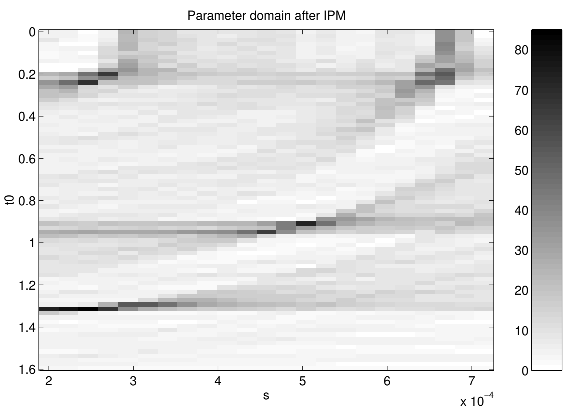

Figure 4.13 Parameter domain estimated by use of IPM.

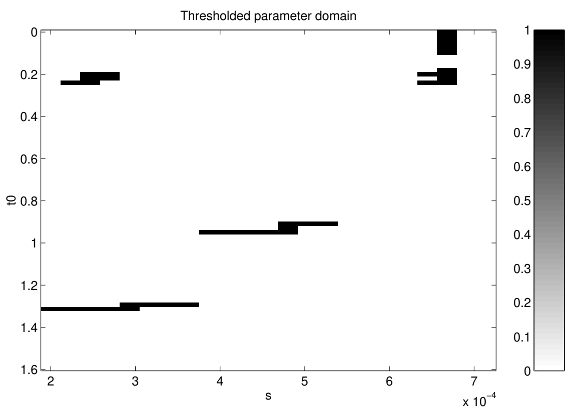

Figure 4.14 The IPM estimated parameter domain after thresholding using $\lambda_1 = 0.5$.

The binary parameter domain is then resampled to a full size parameter domain, which looks exactly like Fig. 4.14, but now the parameter domain has 27 times the number of samples used for IPM.

Comparing this pre-condition map with Fig. 4.11, shows that all regions containing curves are contained within the pre-condition map. Next, Fig. 4.15 shows the parameter domain obtained by using GRT within the regions specified by the pre-condition map. Finally, the pre-conditioned GRT parameter domain is thresholded using a threshold level $\lambda_2 = 0.75$, and the result is shown in Fig. 4.16.

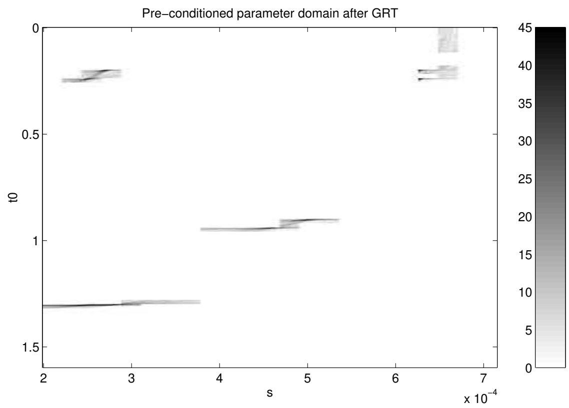

Figure 4.15 Parameter domain obtained using GRT within the regions specified by the resampled parameter domain.

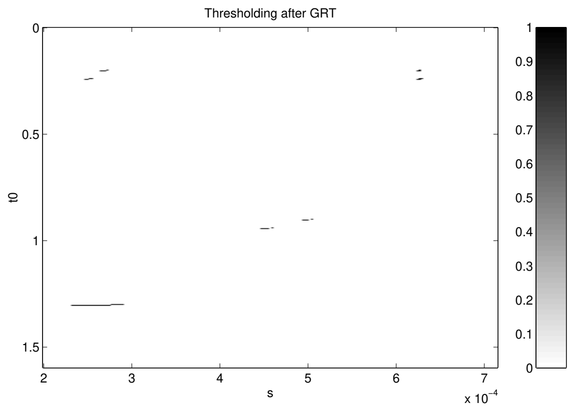

Figure 4.16 The preconditioned GRT estimated parameter domain after thresholding using $\lambda_2 = 0.75$.

The estimated parameter vectors are given in Table 4.2, where the clustering process has used a value of $\alpha = 1.5$. As seen from Table 4.2, eight groups of curve parameters are found, and the estimated curve parameters are rather close to the true parameters.

Curve Number	True parameter	Estimated parameter
1	$(0.202\ s, 2.7027\ 10^{-4}\ s/m)$	$(0.204\ s, 2.6832\ 10^{-4}\ s/m)$
2	$(0.202\ s, 6.2500\ 10^{-4}\ s/m)$	$(0.201\ s, 6.2567\ 10^{-4}\ s/m)$
3	$(0.242\ s, 2.5000\ 10^{-4}\ s/m)$	$(0.242\ s, 2.5517\ 10^{-4}\ s/m)$
4	$(0.242\ s, 6.2500\ 10^{-4}\ s/m)$	$(0.242\ s, 6.248\ 10^{-4}\ s/m)$
5	$(0.902\ s, 5.0000\ 10^{-4}\ s/m)$	$(0.902\ s, 4.994\ 10^{-4}\ s/m)$
6	$(0.942\ s, 4.5455\ 10^{-4}\ s/m)$	$(0.943\ s, 4.530\ 10^{-4}\ s/m)$
7	$(1.302\ s, 2.5000\ 10^{-4}\ s/m)$	$(1.303\ s, 2.571\ 10^{-4}\ s/m)$
8	$(1.302\ s, 2.7778\ 10^{-4}\ s/m)$	$(1.304\ s, 2.803\ 10^{-4}\ s/m)$

Table 4.2 The results of the FCE-algorithm along with the true curve parameters. The width of the cluster band is set to 3, i.e. $\alpha = 1.5$. The parameters are estimated by the center of the cluster.

The FCE-algorithm has been implemented in MATLAB and C (the loop oriented functions are made in C), and using a 120 MHz Pentium, as previously, result in the time measurements shown in Table 4.3. The table shows the individual parts of the FCE-algorithm and the total cost. It can be seen that the FCE-algorithm here outperforms the use of GRT with a factor of 10, and all eight hyperbola parameters have been estimated without any additional errors.

IPM	0.04 sec
Threshold 1	0.01 sec
Resampling	0.24 sec
Pre-Conditioned GRT	0.14 sec
Threshold 2	0.05 sec
Clustering	0.16 sec
The FCE-algorithm	0.64 sec
Full GRT estimation	6.35 sec

Table 4.3 Time measurements on a 120 MHz Pentium.

Next a small example of the sensitivity of the FCE algorithm with respect to the choice of v_σ and v_{t_0} is given. A binary image containing the same eight hyperbolas as above was created. No noise is present in the image. Fig. 4.17 shows the time used by the FCE-algorithm to estimate the hyperbola parameter as a function of v_σ and v_{t_0}, and in Fig. 4.18 in shown the number of hyperbolas identified by the FCE-algorithm. From the figures it can be seen that an optimum with respect to time is found at $v_{t_0} = 0$ and $v_\sigma = 2$, but for $v_{t_0} = 0$ Fig. 4.18 shows that one or two of the hyperbolas can be missed. This is due to the fact that IPM requires that the parameter domain is coarsely sampled especially in the t_0-parameter, cf. Eq. 4.36.

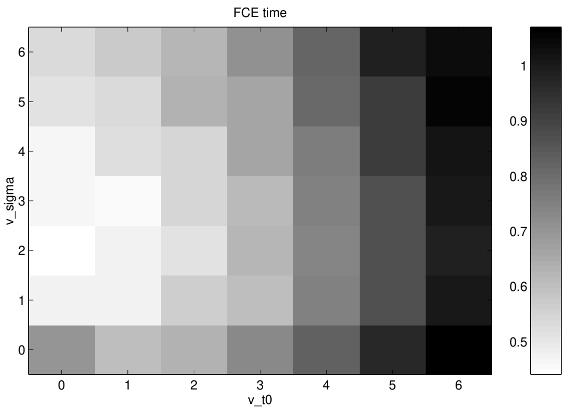

Figure 4.17 The time used by the FCE-algorithm to estimate the parameters of the hyperbolas shown as a function of v_σ and v_{t_0}. For the time measurements a 120 MHz Pentium has been used.

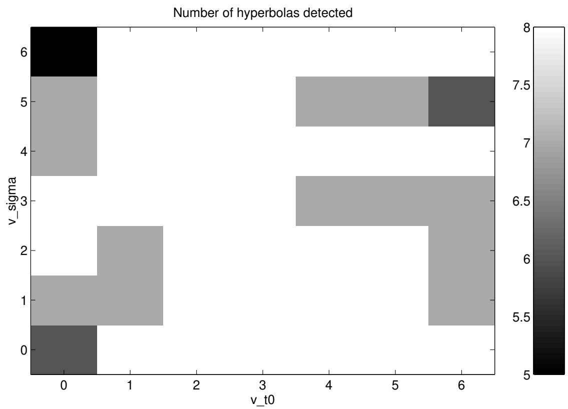

Figure 4.18 The number of hyperbolas detected by the FCE-algorithm shown as a function of v_σ and v_{t_0}.

4.6.3 An Example with a Noise Corrupted Synthetic CMP-gather

To demonstrate the performance of the FCE-algorithm with many curves, an example based on a noise corrupted synthetic CMP-gather is given. The subsurface model is a pure acoustic (compressional waves only) horizontally layered subsurface consisting of four finite layers and infinite top and bottom layers. The synthetic CMP-gather is produced by use of ray tracing [38]. The resulting CMP-gather is composed of 24 distinct reflection curves, shown in Fig. 4.19, where each vertical line represents a trace. The binary signal has been generated by adding the hyperbolas and noise is here simulated by switching the values of the individual binary pixels with a probability of 5%. The parameters of the hyperbolas are shown in Fig. 4.22 and the parameters for both image and parameter domains are given in Table 4.4.

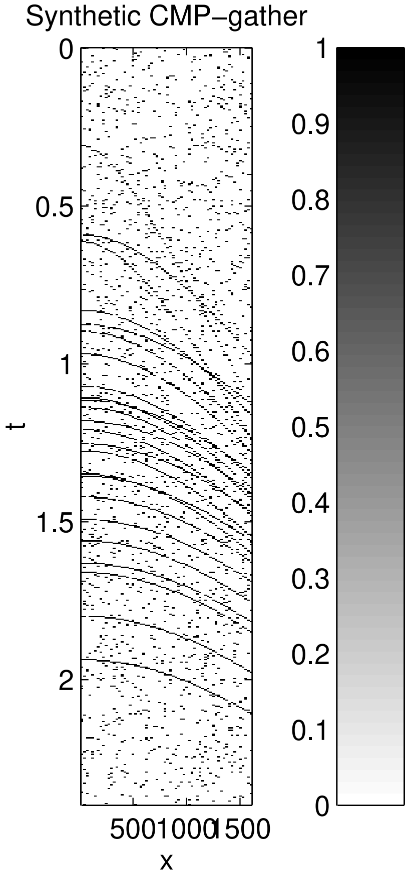

Figure 4.19 Synthetic CMP-gather composed of 24 distinct reflection curves.

Image domain		Parameter domain	
Parameter	Value	Parameter	Value
M	64	S	153
N	600	Z	600
Δx	$25\ m$	$\Delta\sigma$	$2.5 \cdot 10^{-6}\ s/m$
Δt	$4 \cdot 10^{-3} s$	Δt_0	$4 \cdot 10^{-3} s$
x_{min}	$25\ m$	σ_{min}	$3.33 \cdot 10^{-4}\ s/m$
t_{min}	$0\ s$	$t_{0,min}$	$0\ s$
α	1.5	λ_1	0.5
		λ_2	0.5
		v_σ	4
		v_{t_0}	2

Table 4.4 Image and parameter domain settings.

Applying the FCE-algorithm to the binary image gives the IPM estimated parameter domain shown in Fig. 4.20, using $v_{t_0} = 2$ and $v_\sigma = 4$. Applying GRT on the parameter domain region specified by the resampled and thresholded IPM estimated parameter domain, with $\lambda_1 = 0.5$, results in the GRT parameter domain shown in Fig. 4.21.

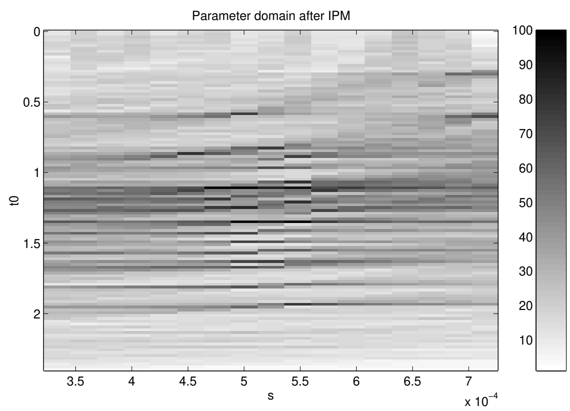

Figure 4.20 IPM estimated parameter domain.

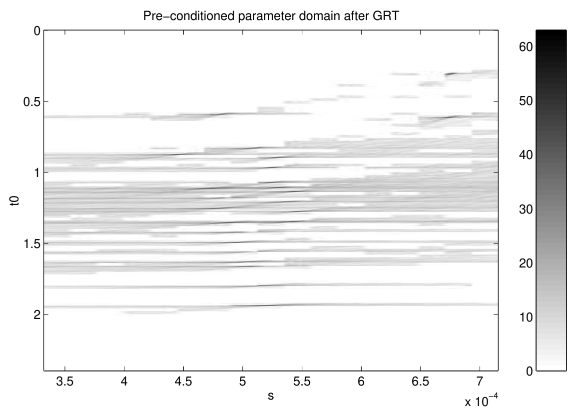

Figure 4.21 The preconditioned GRT.

Finally, Fig. 4.23 shows the estimated curve parameters, where a significance level of $\lambda_2 = 0.5$ in the final threshold process and $\alpha = 1.5$ in the clustering process have been used. In this case hyperbolas have been identified matching the original parameters very well, but two additional hyperbolas have also been identified. This is partly due to the noise level in the initial image, and that the given sampling of the parameter domain is a tradeoff between the certainty of detection and limiting the number of samples in the parameter domain. In this case the FCE-algorithm is approximately 2.6 faster than the GRT. The decrease in speed, compared to the previous example, is due to the higher number of hyperbolas in this example, covering a larger part of the discrete parameter domain.

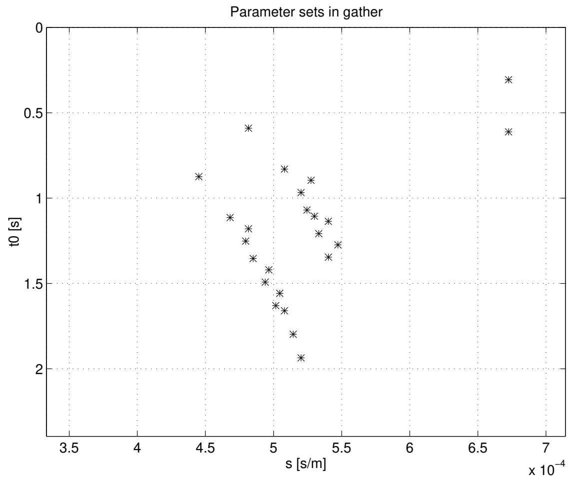

Figure 4.22 The parameters of hyperbolas shown in the parameter domain.

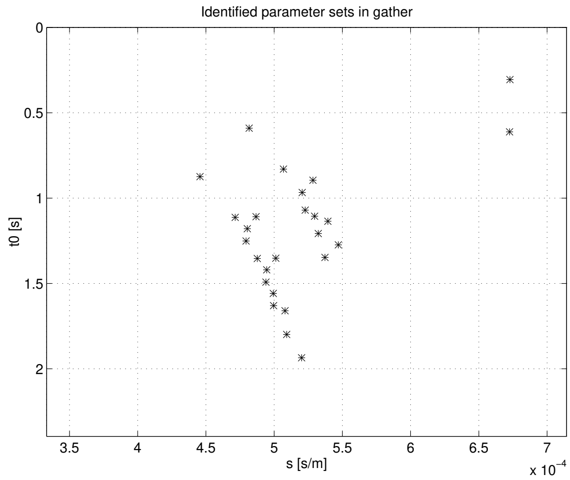

Figure 4.23 The estimated parameters of hyperbolas shown in the parameter domain.

IPM	0.14 sec
Threshold 1	0.01 sec
Resampling	0.24 sec
Pre-Conditioned GRT	2.08 sec
Threshold 2	0.07 sec
Clustering	0.17 sec
The FCE-algorithm	2.71 sec
Full GRT estimation	7.10 sec

Table 4.5 Time measurements on a 120 MHz Pentium.

4.7 Summary

A new algorithm for fast curve parameter estimation, named the FCE-algorithm, has been presented. The algorithm identifies curve parameters by operating on a binary image, obtained, e.g., by edge filtering, deconvolution, or mean field annealing.

The fundamental idea of the algorithm is the use of pre-conditioning to reduce the computational cost of the traditional generalized Radon transform. The pre-conditioning map determines regions of the parameter domain which contain peaks, and the generalized Radon transform is applied only in these regions. As the size of the regions is less than the full parameter domain, the pre-conditioning map reduces the computational costs when applying the generalized Radon transform. For fast generation of the pre-conditioning map, a generalization of the Hough transform named *image point mapping* has been developed. Image point mapping is computationally efficient by taking account of image points with value zero. The required parameter domain sampling and the resulting parameter domain blurring have been investigated.

The FCE algorithm was successfully applied to the identification of hyperbolas in seismic images and two numerical examples have been presented. One example demonstrates the potential of the algorithm for fast and accurate parameter estimation, and the other example illustrates the robustness of the algorithm with respect to noise.

Chapter 5

Curve Parameter Estimation in Noisy Images

So far it has been assumed, that the lines in the images are straight within the (digital) resolution. A very reasonable question regards the situation where lines have wiggles. In Section 5.1 it is analyzed, whether the discrete linear Radon transform still be used to detect line parameters [4]. Analytical expressions are given to quantify the theory.

In Section 5.2 it is shown that lines with wiggles can be incorporated in the Radon transform, but it is also shown that the alteration can be transformed back to the original image as a simple non-linear filter.

Finally in Section 5.3 the implications of additive noise in the image, is analyzed with respect to curve parameter detection [3]. Here a very general form of the generalized Radon transform is assumed, and the theory is illustrated in the linear Radon transform case. Simple formulas are derived, quantifying whether a curve in a noisy image can be detected or not.

5.1 Lines with Wiggles

It is assumed that the digital line in the image can be modelled by

$$g(m,n) = \delta\left(n - [\alpha^* m + \beta^* + \lambda]\right) \text{ where } \lambda \in \mathcal{N}(0,\sigma^2) \tag{5.1}$$

where $\delta(\cdot)$ is the Kronecker delta function, and $[\cdot]$ rounds to the nearest integer. The Gaussian distributed noise term λ determines the change in position of the line in the y-direction, and it will be assumed that the noise terms λ are uncorrelated as a function of m. The constants α^* and β^* correspond to actual slope and offset values as in Eq. 1.17, i.e., $\alpha = \frac{p_k \Delta x}{\Delta y}$ and $\beta = \frac{p_k x_{min} + \tau_h - y_{min}}{\Delta y}$.

What is of interest now, is to find the value and shape of the peak (if any is found) in the discrete parameter domain as a function of the noise deviation σ. Eq. 5.1 implies that the probability of the sample $g(m,n)$ being 1 is given by

$$\begin{aligned} P\{g(m,n)=1\} &= P\left\{n = [\alpha^* m + \beta^* + \lambda]\right\} & (5.2)\\ &= P\left\{0 = [n - \alpha^* m - \beta^* - \lambda]\right\} & (5.3)\\ &= P\left\{-\frac{1}{2} < n - \alpha^* m - \beta^* - \lambda < \frac{1}{2}\right\} & (5.4)\end{aligned}$$

and assuming that the sample point (m,n) is given from the nearest neighbour mapping in the discrete Radon transform, $n = [\alpha\, m + \beta]$, then the rounding function is modelled as an additive (noise) term ω

$$P\{g(m,n)=1\} = P\left\{-\frac{1}{2} < [\alpha\, m+\beta] - \alpha^* m - \beta^* - \lambda < \frac{1}{2}\right\} \tag{5.5}$$
$$= P\left\{-\frac{1}{2} < \alpha\, m+\beta+\omega - \alpha^* m - \beta^* - \lambda < \frac{1}{2}\right\} \tag{5.6}$$
$$= P\left\{-\frac{1}{2} < \zeta+\omega-\lambda < \frac{1}{2}\right\} \text{ where } \zeta = (\alpha-\alpha^*)m+\beta-\beta^* \tag{5.7}$$
$$= \Phi\left(\frac{\frac{1}{2}+\zeta+\omega}{\sigma}\right) - \Phi\left(\frac{-\frac{1}{2}+\zeta+\omega}{\sigma}\right) \tag{5.8}$$

where $\Phi(\cdot)$ is the Gaussian probability function, and ζ is a displacement between the true line parameters (α^*, β^*) and the parameters (α, β) at a given position m.

Due to many values of m used in the discrete Radon transform, it is a reasonable approximation to model ω as a uniformly distributed variable between the limits $-1/2$ and $1/2$, i.e., $\omega_m \in \mathcal{U}(-1/2, 1/2)$. Eq. 5.8 implies that the average contribution from the pixel $g(m,n)$ to the discrete Radon transform is given by

$$E\{\breve{g}(k,h)|g(m,n)\} = \int_{\omega=-\frac{1}{2}}^{\omega=\frac{1}{2}} P_m(\omega)\; d\omega \tag{5.9}$$
$$= \int_{\omega=-\frac{1}{2}}^{\omega=\frac{1}{2}} \Phi\left(\frac{\frac{1}{2}+\zeta+\omega}{\sigma}\right) - \Phi\left(\frac{-\frac{1}{2}+\zeta+\omega}{\sigma}\right)\; d\omega \tag{5.10}$$

In order to get an analytical expression for $E\{\breve{g}(k,h)|g(m,n)\}$ the Gaussian probability function is approximated

$$\Phi(x) \approx \frac{1}{2}\left(1+\tanh\left(\frac{x}{\gamma}\right)\right) \text{ where } \gamma = \sqrt{\frac{\pi}{2}} \tag{5.11}$$

i.e., the integral of $\Phi(\cdot)$ can be approximated by

$$\int \Phi(x)\; dx \approx \frac{\gamma}{2}\left(\frac{1+x}{\gamma} + \log\cosh\left(\frac{x}{\gamma}\right)\right) \tag{5.12}$$

where $\log(\cdot)$ is the natural logarithm.

If Eq. 5.12 is inserted into Eq. 5.10, and doing some rearrangements of the expressions, it is found that

$$E\{\breve{g}(k,h)|g(m,n)\} = \frac{\gamma\sigma}{2}\log\left(\frac{\cosh\left(\frac{2\zeta}{\gamma\sigma}\right)+\cosh\left(\frac{2}{\gamma\sigma}\right)}{\cosh\left(\frac{2\zeta}{\gamma\sigma}\right)+1}\right) \tag{5.13}$$

In Fig. 5.1 the average weight to the discrete Radon transform is shown as a function of σ and ζ.

A special case of interest regards $\zeta = 0$, i.e., where the coordinates of the line in the image matches exactly a sample in the discrete parameter domain.

$$\zeta = 0 \;\Rightarrow\; E\{\breve{g}(k,h)|g(m,n)\} = \sigma\sqrt{\frac{\pi}{8}}\log\left(\frac{1}{2}\left(1+\cosh\left(\frac{\sqrt{8}}{\sigma\sqrt{\pi}}\right)\right)\right) \tag{5.14}$$

In Fig. 5.2 the average weight to the discrete Radon transform is shown as a function of σ, when the displacement is assumed negligible. It can be seen that the function agrees well with a simulated result shown in Fig. 5.3, that is found by generating 10000 images with the correct noise

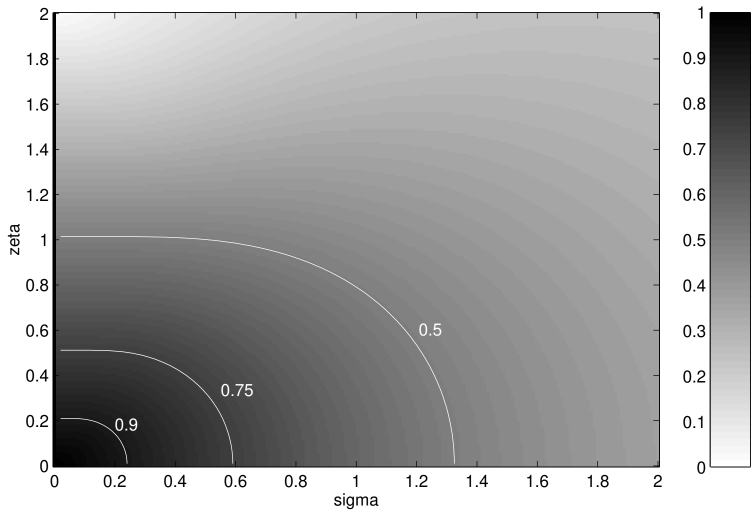

Figure 5.1 The average weight to the discrete Radon transform $E\{\check{g}(k,h)|g(m,n)\}$ as a function of the noise deviation σ on the first axis, and the displacement ζ on the second axis.

amplitude σ and computing the discrete Radon transform only for the parameter set matching the true parameters. An even better agreement can be found using a numerical integration of Eq. 5.10, but Eq. 5.12 provides an simple analytical result that approximates the simulated results well, though it can also be seen that the theoretical model has a small bias, when $\sigma \approx 0.1$.

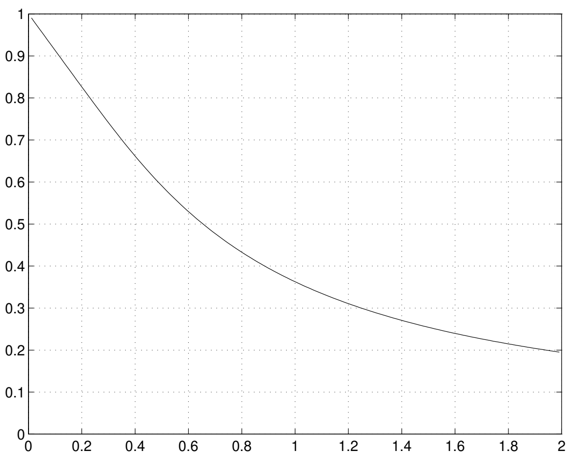

Figure 5.2 The theoretical result of the average weight to the discrete Radon transform, found in Eq. 5.14, as a function of σ when the displacement ζ is negligible.

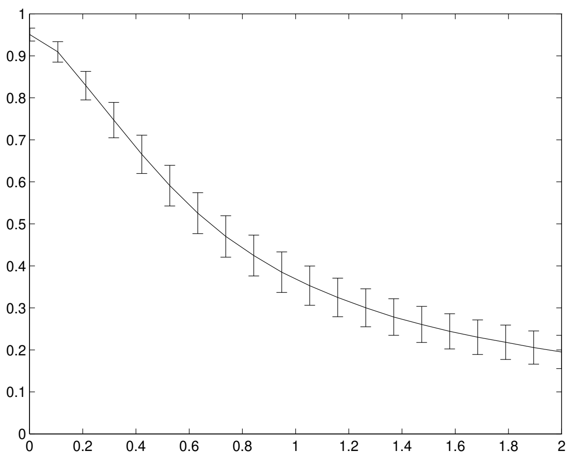

Figure 5.3 The simulated average value of the peak using 10000 images for each noise level. The vertical error bars show the measured deviation.

Fig. 5.2 or Eq. 5.14 can be used to predict whether line parameters might be estimated given that the lines have wiggles. If, e.g., demanding that the peak value cannot decrease more than 50% due to wiggles, then σ cannot exceeds approximately 0.7. And a 25% decrease of the peak value is found if $\sigma \approx 0.25$. In Fig. 5.4 is shown an image with one line without noise, and Fig. 5.5 shows the discrete parameter domain in the area of the peak. Using the same sampling parameters, Fig. 5.6 shows an image where $\sigma = 0.25$, and Fig. 5.7 shows the corresponding discrete parameter domain. The images are scaled individually according to the minimal and maximal values, and it can be seen that the maximal value here is around 80, and the shape is approximately the same as seen from Fig. 5.5. Next $\sigma = 0.5$ gives an image like it is shown in Fig. 5.8, and it can be seen that the noise level is high. In Fig. 5.9 is shown the discrete parameter domain, and it can be seen that the peak here covers a larger part of the field of view and the peak is somewhat scattered.

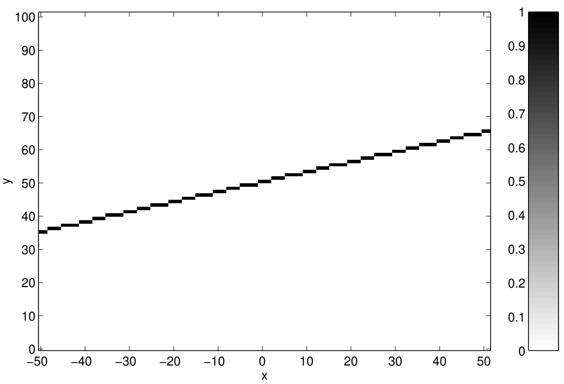

Figure 5.4 An image without noise.

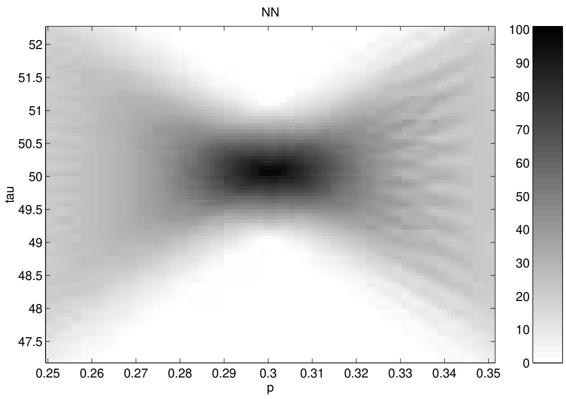

Figure 5.5 The corresponding discrete Radon transform.

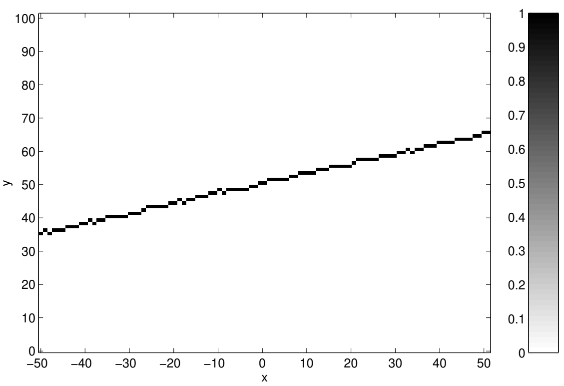

Figure 5.6 An image using $\sigma = 0.25$.

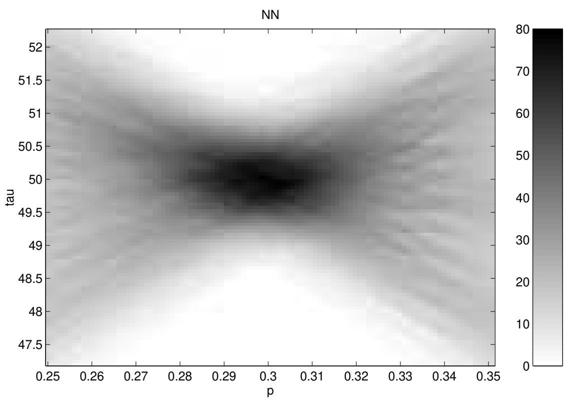

Figure 5.7 The corresponding discrete Radon transform.

Eq. 5.13 can also be analyzed in the case where $\sigma = 0$, in order to focus on the importance of the displacement. First, the expression for ζ is rewritten

$$\zeta = (\alpha - \alpha^*)m + \beta - \beta^* = (p^* - p_k)\frac{x_m}{\Delta y} + \frac{\tau^* - \tau_h}{\Delta y} \tag{5.15}$$

where (p^*, τ^*) is the line parameters corresponding to (α^*, β^*), and (p_k, τ_h) correspond to (α, β).

Assuming that $p^* = p_k$, the displacement can be analyzed in the τ-direction. From Fig. 5.1 it can be seen that $\zeta \approx 1$, correspond to $E\{\breve{g}(k,h)|g(m,n)\} \approx 0.5$, i.e., that the peak value has decreased to approximately 50% of the possible maximum. This can be used to restrict the sampling interval $\Delta\tau$, due to the quantization of τ_h

$$E\{\breve{g}(k,h)|g(m,n)\} \geq \frac{1}{2} \;\Rightarrow\; \frac{|\tau - \tau_h|}{\Delta y} < 1 \;\Rightarrow\; |\tau - \tau_h| < \Delta y \;\Rightarrow\; \Delta\tau < \Delta y \tag{5.16}$$

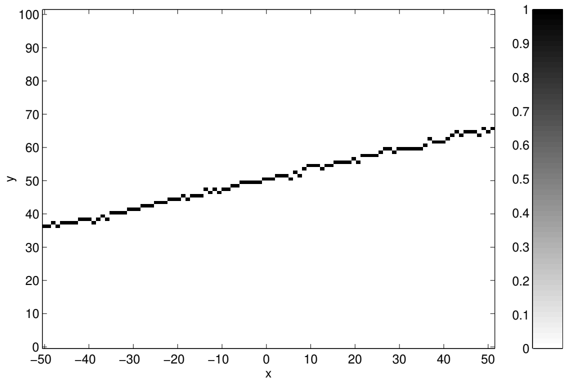

Figure 5.8 An image using $\sigma = 0.5$.

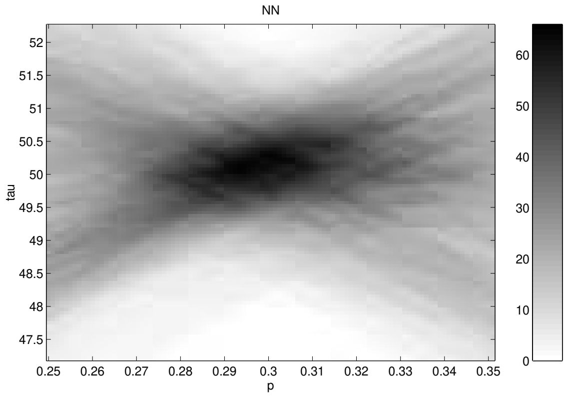

Figure 5.9 The corresponding discrete Radon transform.

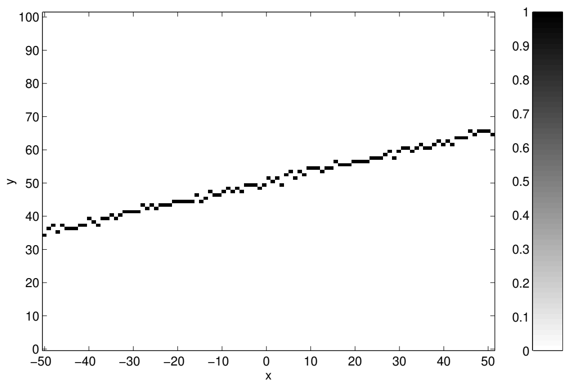

Figure 5.10 An image using $\sigma = 0.7$.

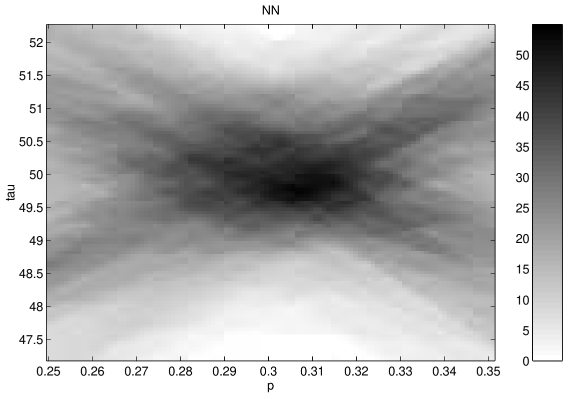

Figure 5.11 The corresponding discrete Radon transform.

This result agrees with Eq. 1.31, but here the result comes with a very important information: Using Eq. 1.31 can imply that the peak value is reduced by 50% of the possible maximum. This implies a compromise between the number of samples in the discrete parameter domain, i.e., the sampling intervals, and the time used to compute the discrete parameter domain.

It should be mentioned, that a simple and intuitively model can be proposed to analyze the case $\zeta = 0$, but the results is not as good as obtained by the model described above. The discrete Radon transform will due to the rounding have a window in the n-direction, like it is shown in Fig. 5.12, of length 1. Hence assuming that the offset ω is negligible, then the probability of detecting a 1 is given by

$$\mathrm{P}\{g(m,n) = 1\} = \mathrm{P}\left\{-\frac{1}{2} < \lambda < \frac{1}{2}\right\} = \Phi\left(\frac{1}{2\sigma}\right) - \Phi\left(-\frac{1}{2\sigma}\right) = 2\Phi\left(\frac{1}{2\sigma}\right) - 1 \qquad (5.17)$$

which can be interpreted as the reduction factor of the peak. The probability is shown as a function of σ in Fig. 5.13.

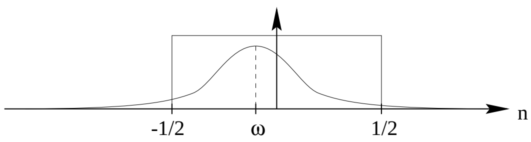

Figure 5.12 A Gaussian function and the windowing function found by using a nearest neighbour approximation of the discrete Radon transform.

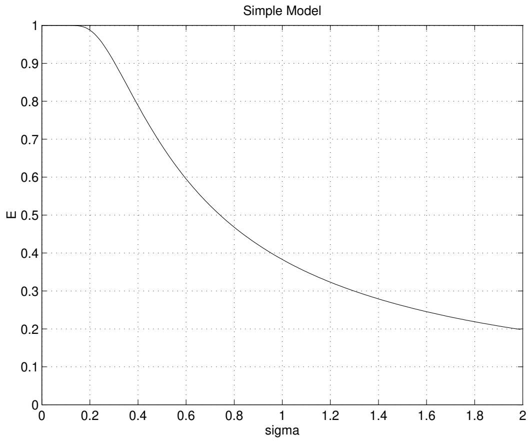

Figure 5.13 The simple model.

The conclusion on this section is that a peak is generated in the discrete parameter domain corresponding to each line in the image, and the value of the peak will be lowered corresponding to the noise level, which in this case is due to the wiggles. In Section 5.3 additive image noise will also be considered, and it could be of interest to combine the analysis of these two problems.

5.2 A "Fuzzy" Radon Transform

Instead of using the pixel positions designated by the discrete Radon transform, a search strategy could be used to incorporate wiggly lines into the curve detection algorithm. Assuming that the lines have wiggles as shown in Section 5.1, one strategy to incorporate this prior knowledge could be to allow the discrete Radon transform to search for the maximum image value within a fixed window.

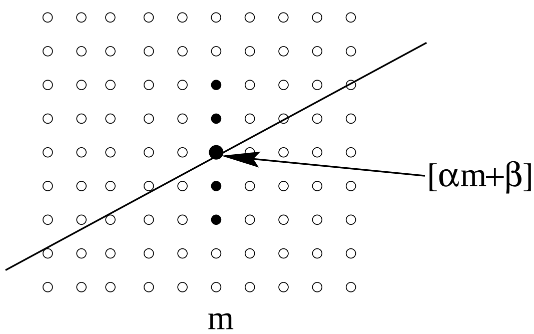

Figure 5.14 Instead of adding the nearest vertical pixel to the sum the maximum is used within a window of here five pixels.

The strategy is demonstrated for discrete slant stacking, but can be adapted to, e.g., the normal Radon transform or the discrete generalized Radon transform. Assuming that the curve amplitude is positive and additive noise is corrupting the image, Fig. 5.14 shows how a search window of five pixels is used for each value m on the horizontal axis. Within the window one apparent strategy is to update the sum in the discrete Radon transform by the maximum of the five pixels. Besides that this strategy implies a big increase in the processing time compared to the use of discrete slant stacking, it is not optimal due to the fact that, this strategy is identical to filtering the input image with a median type filter for each value of m. The filter should return the maximum of the pixels within the window, hence the strategy is equivalent to an initial image domain filtering and then ordinary discrete slant stacking. This demonstrates that the alteration should not be used, but the local filtering should only be used, if the maximum filter can select the correct pixel within the window.

The following six figures illustrate the problem with the strategy. Fig. 5.15 is a noisy image containing two intersecting lines, and Fig. 5.16 shows the corresponding discrete Radon transform (nearest neighbour approximation). Two peaks are very clearly marked. Next Fig. 5.17 shows the image, where a maximum filter with window width of 3 pixels has been used for each value of m. Fig. 5.18 shows the corresponding discrete Radon transform, and it is seen that the peaks are still detectable, but the resolution has become worse. Finally Figs. 5.19 and 5.20 uses a filter window of length 5, and the result is more blurring of the peaks, and that the background level in the discrete parameter domain has gone up.

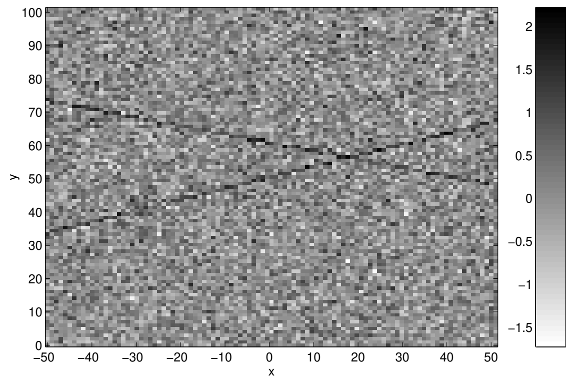

Figure 5.15 The original image containing two intersecting lines covered with additive noise.

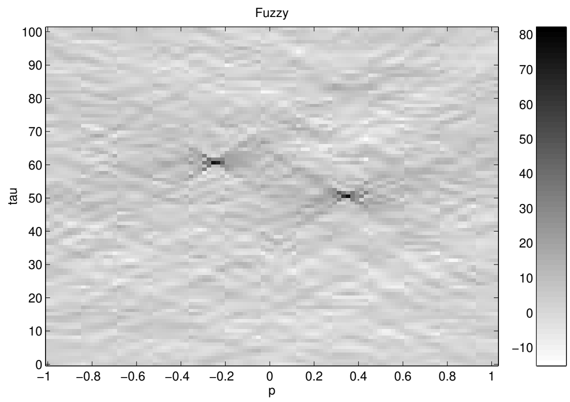

Figure 5.16 The corresponding discrete parameter domain.

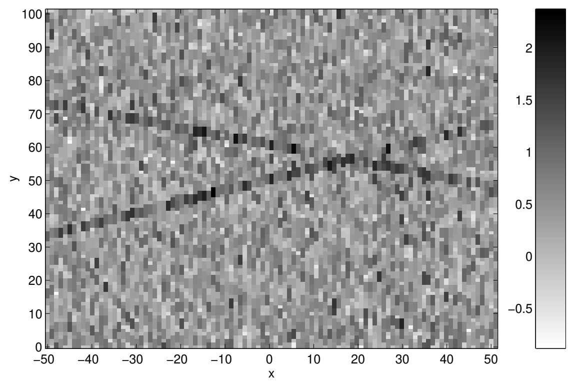

Figure 5.17 The filtered image containing two intersecting lines covered with additive noise.

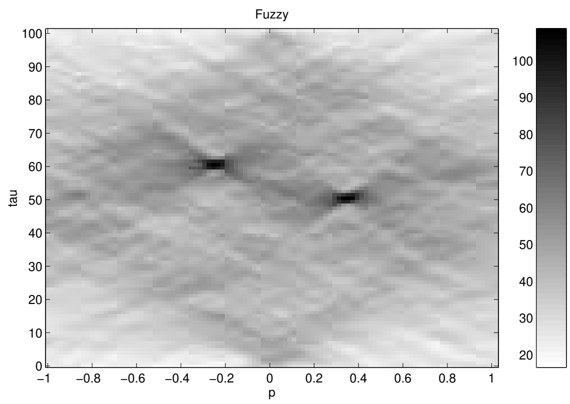

Figure 5.18 The corresponding discrete parameter domain.

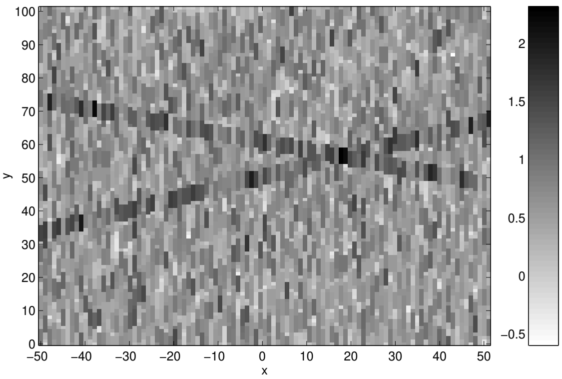

Figure 5.19 The filtered image containing two intersecting lines covered with additive noise.

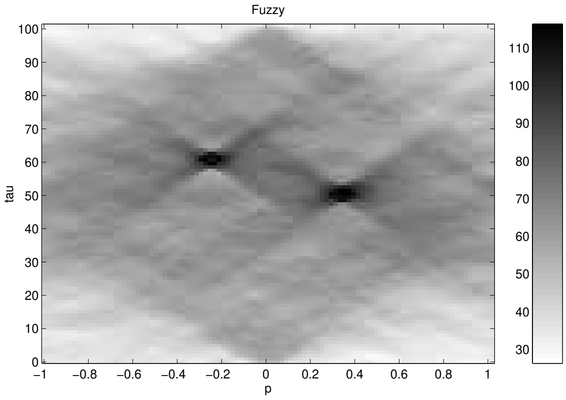

Figure 5.20 The corresponding discrete parameter domain.

5.3 Detection of Curves in Noisy Images

A natural expansion of the Radon transform is the (discrete) generalized Radon transform (GRT) [36, 1, 2]. As demonstrated in Chapter 4 a major advantage of the GRT is that curves are allowed to intersect. Another major advantage that will be demonstrated in this Section, is that the GRT is very robust to noise.

In this section a probabilistic approach is used to show that the GRT can be used for curve detection if the noise in the image is below a certain level compared to the signal values on the curves. An analytical expression is given quantifying the limits of using GRT for curve detection especially with respect to very noisy images. If noise is added to an image containing curves, the problem is that peaks in the parameter domain may or may not correspond to actual curve parameters. A threshold level, based on the noise level, is derived and applied for separation of noise and curve information in the parameter domain.

A numerical example is provided to illustrate the presented theory. Note that the theory can be used to reestimate the image containing curves [56].

5.3.1 The Generalized Radon Transform

The generalized Radon transform, GRT, of a digital image can be defined in many ways. One way is

$$\breve{g}(\boldsymbol{j}) = \sum_{l=0}^{L-1} g(\phi_m(l,\boldsymbol{j}), \phi_n(l,\boldsymbol{j})) \tag{5.18}$$

where $\breve{g}$ denotes the GRT of the image $g(m,n)$ and $\boldsymbol{j}$ is a multi dimensional vector containing the curve parameters. The two curve functions $\phi_m(l,\boldsymbol{j})$ and $\phi_n(l,\boldsymbol{j})$ define the curve type and are (in principle) arbitrary and an interpolation scheme is assumed implicitly; e.g., by rounding the functions $\phi_m(l,\boldsymbol{j})$ and $\phi_n(l,\boldsymbol{j})$ to the nearest sample point. A popular choice is the linear curve functions; e.g., normal parameters $\boldsymbol{j} = (\rho, \theta)$. Another frequent choice is the (τ, p)-parameters, cf. Chapter 1, where $\phi_m(l,\tau,p) = l$ and $\phi_n(l,\tau,p) = p\ l + \tau$. Choosing only $\phi_m(l,\boldsymbol{j}) = l$ the definition will match the one used in Chapter 4.

Even though the GRT can be applied to any given image, the main feature of the GRT is that an image, which contains a discrete curve matching the curve functions at one parameter vector $\boldsymbol{j}^*$ implies that the parameter domain $\breve{g}(\boldsymbol{j})$, will show a peak at that specific parameter vector $\boldsymbol{j} = \boldsymbol{j}^*$. The linearity of the GRT implies that each curve in the image will be transformed into a peak in the parameter domain. In this section a curve in an image is defined by large image values of the same sign on the curve and otherwise (approximately) zero.

Initially only two values of the GRT will be considered. The first, $\breve{g}(\boldsymbol{j}^*)$, corresponds to a curve in the image. Another, $\breve{g}(\boldsymbol{j}^-)$, corresponds to a parameter vector, that does not match a curve in the image. It is assumed that $\breve{g}(\boldsymbol{j}^*)$ is the sum of a mean signal value μ over all L samples, and $\breve{g}(\boldsymbol{j}^-)$ covers (approximately) no samples of the curve(s) in the image. Both values of $\breve{g}$ are contaminated with noise, which is due to noise in the image. Assume the noise in the image is nearly uncorrelated with zero mean (e.g., by subtracting a DC-value from the image) and variance σ^2.

5.3.2 Curve Detection using the Generalized Radon Transform

A classical curve detection algorithm is to determine the parameter vectors from the positions of peaks in the parameter space

$$\boldsymbol{j}^* = \arg \{|\breve{g}(\boldsymbol{j})| \geq L\ \mu^*\} \tag{5.19}$$

The reason for choosing the significance level in this way is that Eq. 5.18 consists of a summation over L samples, and μ^* is a lower positive bound on the mean signal level on the curve; e.g, found by estimation. The purpose of the following is to estimate whether curves having the signal level μ can be detected using Eq. 5.19, if the image is contaminated with the described noise.

Due to the linearity, $\breve{g}$ consists of a curve part and a noise part. If $L \gg 1$ the sum of the noise terms $\breve{g}_{\text{noise}}$ will approximately be Gaussian distributed with zero mean and variance $L\sigma^2$ due to the Central Limit Theorem. This implies that the two considered values of the GRT are distributed as $\breve{g}(\boldsymbol{j}^*) = \mu L + \breve{g}^*_{\text{noise}} \in \mathcal{N}(\mu L, L\sigma^2)$ and $\breve{g}(\boldsymbol{j}^-) = \breve{g}^-_{\text{noise}} \in \mathcal{N}(0, L\sigma^2)$. Since Eq. 5.19 selects the large values in the parameter domain, an important issue is the probability of detecting the correct parameter vector of the two considered

$$\mathrm{P}_{\text{det }2} \equiv \mathrm{P}\{|\breve{g}(\boldsymbol{j}^*)| > |\breve{g}(\boldsymbol{j}^-)|\} \tag{5.20}$$

Assume two uncorrelated stochastic variables

$$A \in \mathcal{N}(\mu_A, \sigma^2) \quad \text{and} \quad B \in \mathcal{N}(\mu_B, \sigma^2) \tag{5.21}$$

hence the joint probability distribution function is given by

$$f(a,b) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{1}{2\sigma^2}\left((a-\mu_A)^2+(b-\mu_B)^2\right)} \tag{5.22}$$

The probability $P\{|A| > |B|\}$ will now be found by rotating the coordinate system: $x = \frac{1}{\sqrt{2}}(a-b)$ and $y = \frac{1}{\sqrt{2}}(a+b)$. In the new coordinates the joint probability density function is

$$g(x,y) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{1}{2\sigma^2}\left((x-\mu_x)^2+(y-\mu_y)^2\right)} \tag{5.23}$$

where $\mu_x = \frac{1}{\sqrt{2}}(\mu_A - \mu_B)$ and $\mu_y = \frac{1}{\sqrt{2}}(\mu_A + \mu_B)$.

$$P\{|A| > |B|\} = \int_{x=-\infty}^{0} \int_{y=-\infty}^{0} g(x,y)\ dy\ dx + \int_{x=0}^{\infty} \int_{y=0}^{\infty} g(x,y)\ dy\ dx \tag{5.24}$$

The two integrals can be separated and it is easily found that

$$P\{|A| > |B|\} = \frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{\mu_A - \mu_B}{2\sigma}\right) \operatorname{erf}\left(\frac{\mu_A + \mu_B}{2\sigma}\right)\right) \tag{5.25}$$

where erf(x) is the error function. Inserting $\mu_A = \mu L$, $\mu_B = 0$, and replacing σ with $\sigma\sqrt{L}$ gives

$$\mathrm{P}_{\mathrm{det}\ 2} = \frac{1}{2}\left(1 + \left(\mathrm{erf}\left(\frac{\lambda}{2}\right)\right)^2\right) \quad \text{and} \quad \lambda = \frac{\mu\sqrt{L}}{\sigma} \tag{5.26}$$

Note that in this case $\mathrm{P}_{\mathrm{det}\ 2}$, shown in Fig. 5.21, only depends on one parameter λ. Note that $|\lambda| > 4$ gives an almost certain detection. This is the case if $\sigma \to 0$ or $\mu \to \infty$.

When using the GRT to detect curves then the discrete parameter domain will not only have two, but $\mathcal{J}$ different parameter vectors, where $\mathcal{J}$ is the number of samples in the parameter domain. It is assumed that all the noise sources in the parameter domain are independent and in the following, the detection of a single curve is analyzed. Selecting the position of the highest peak in the parameter domain, the probability of the selected parameter vector being correct, can be approximated by

$$\mathrm{P}_{\mathrm{det\ all}} \cong \prod_{i=2}^{\mathcal{J}} \mathrm{P}_{\mathrm{det}\ 2} \cong \mathrm{P}_{\mathrm{det}\ 2}^{\mathcal{J}-1} = \left(\frac{1}{2}\left(1 + \left(\mathrm{erf}\left(\frac{\lambda}{2}\right)\right)^2\right)\right)^{\mathcal{J}-1} \cong 1 - \frac{2\mathcal{J}}{\lambda\sqrt{\pi}}\,\mathrm{e}^{-\lambda^2/4} \tag{5.27}$$

The last simple approximation is valid if the detection probability is close to 1 as seen from Fig. 5.22. Several characteristics can be noted: The figure shows a narrow transition from low to high detection probability as a function of λ, and $\mathcal{J}$ does not change the shape of $\mathrm{P}_{\mathrm{det\ all}}$ significantly. If demanding a high detection probability $\mathrm{P}_{\mathrm{det\ all}}$ then Eq. 5.27 and Fig. 5.22 demonstrate that $\mathcal{J}$ should be held low; i.e., by reducing the number of samples in the parameter domain to a minimum. It should be noted that this will involve a compromise on the range of possible parameter vectors.

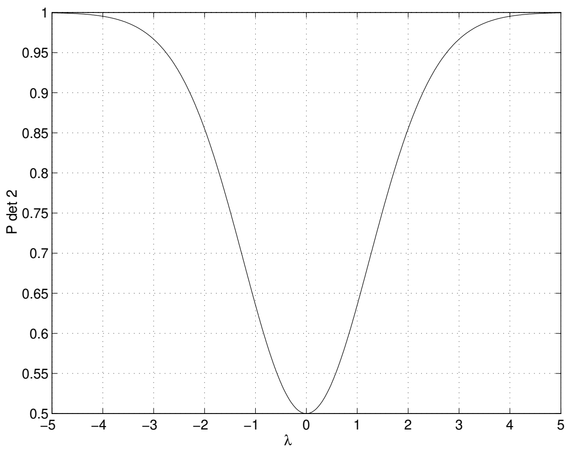

Figure 5.21 The probability of detecting the right curve parameters of two possible as a function of λ.

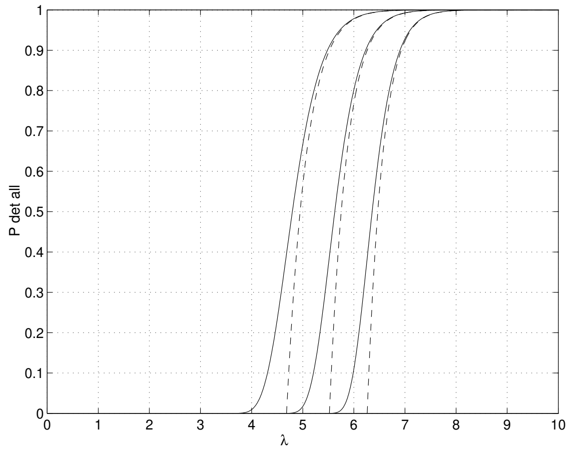

Figure 5.22 Solid lines show the probability of detecting the right curve parameter as a function of λ, and dashed lines show the simple approximation. From left to right $\mathcal{J}$=1000, $\mathcal{J}$=10000 and rightmost $\mathcal{J}$=100000.

Eq. 5.27 can also be used to set requirements on, e.g., the absolute mean signal level μ^* of the curve(s) to be detected. Demanding a detection probability $\mathrm{P}_{\mathrm{det\ all}}$ greater than P^* implies that $\mu^* = \sigma/(\lambda^*\sqrt{L})$, where λ^* can be found by from Eq. 5.27 with a given detection probability, summation length L, number of samples in the parameter domain $\mathcal{J}$, and the standard deviation σ (e.g., by found by estimation). Any $|\mu|$ less than the threshold level, μ^*, can be considered as noise. In this way it is possible to give a statistically based estimate on the thresholding level in Eq. 5.19.

5.3.3 Discussion

Even though the above theory is developed by analyzing one curve in the image, the theory can be used if the image contains few curves. Instead of having one peak in the parameter domain representing one curve in the image, each of the Z curves in the image, where $Z \ll \mathcal{J}$, will give a peak in the parameter domain, even if the curves cross each other. With Z curves each of corresponding Z $\breve{g}$-values must be larger than the rest in the parameter domain. If only a few curves are present, the rest of the parameter domain is dominated by noise, and the probability of detection for each of the Z curves can be found from Eq. 5.27.

The theory used to derive Eq. 5.27 is somewhat pessimistic in estimation of the influence of the noise. This is partly due to the assumption that all the GRT-values include summing noise over L samples. Normally some of the GRT-values will require summing up over a curve partially outside the image, where the image must be assumed equal to zero. Furthermore some correlation must be expected in the parameter domain, especially if the number of dimensions in the parameter domain is higher than two. Depending on the sampling parameters, this implies that an effective $\mathcal{J}$ (less than the number of samples in the parameter domain) must be used in Eq. 5.27.

5.3.4 An Example of Line Detection in a very Noisy Image

A noise free image containing eight lines with limited slope is created. The image, shown in Fig. 5.23, has $101 * 101$ samples. The curve sampling functions are chosen to $\phi_m = l - 50$ and $\phi_n = p\,(l - 50) + \tau$ and L is set to 101. The offset is made in order to lower the sampling requirements in the parameter domain, cf. Subsection 1.4.4. The sampling distances in the parameter domain is set to $\Delta\tau = 1$ and $\Delta p = 0.01$. The line parameters are listed in Table 5.1.

No	p	τ	μ	No	p	τ	μ
1	0.10	60	1.5	5	0.00	30	1.0
2	0.25	50	-1.5	6	0.40	45	-1.0
3	-0.10	80	-1.0	7	-0.36	52	-1.0
4	-0.25	30	1.0	8	0.10	80	0.5

Table 5.1 Line parameters. p is the slope, τ is the offset, and μ is the curve amplitude.

To illustrate the potential of the GRT a very noisy image is generated, by adding Gaussian noise to the noise free image with zero mean and standard deviation $\sigma = 1$. It can be seen from Fig. 5.24, that the lines are hard to identify. Choosing $\mathrm{P}_{\text{det all}} = 0.95$, Eq. 5.27 gives $\lambda^* = 6.53$; i.e., only lines with $|\mu| > 0.65$ should be detectable. This implies that all but line number eight should be detectable. The absolute value of the parameter domain obtained by the use of the GRT to the noisy image is shown in Fig. 5.25.

Since the noisy image contains few lines with absolute curve amplitude $|\mu|$ being of the same order of magnitude as σ and has approximately zero mean, σ was estimated from the image using the ordinary central variance estimator, which gave $\hat{\sigma} = 1.04$. Setting $\mathrm{P}_{\text{det all}}$ to 0.95, Eq. 5.27 results in $L\,\mu^* = 65.7$. This is used for thresholding of the parameter domain as shown in Fig. 5.26. Seven of the eight line parameters are found despite the poor signal to noise ratio in the image. Note that some of the lines will be represented by a few neighbour parameter vectors, which can be corrected by clustering neighbour parameter vectors. The error is due to the sampling of the parameter domain and the finite image size.

The theory predicted that only seven lines could be detected. The eighth line can be detected if the curve length can be increased or the noise variance can be reduced. If the theory is used with $P_{\text{det all}}$ very low, $L\mu^*$ get lower and noise peaks will appear in the parameter domain along with parameters of the eighth line. In Fig. 5.27 the threshold level has been reduced to, e.g., $0.7L\mu^* = 46.0$. As it can be seen, noise will now give parameter vectors which do not represent a curve. As seen from Fig. 5.28 a further reduction of the threshold level to, e.g., $0.5L\mu^* = 32.9$ gives a parameter domain, where all eight lines are present. Due to the noise level many false parameter vectors can also be observed.

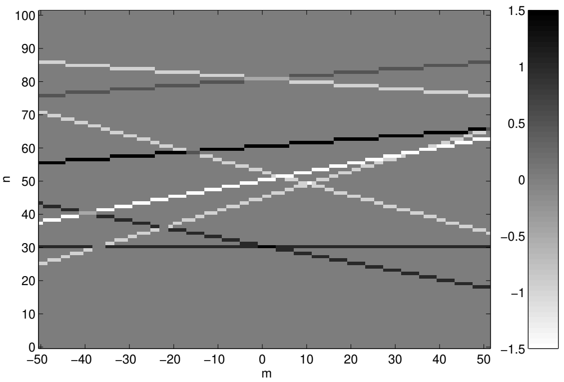

Figure 5.23 Noise free image with eight lines.

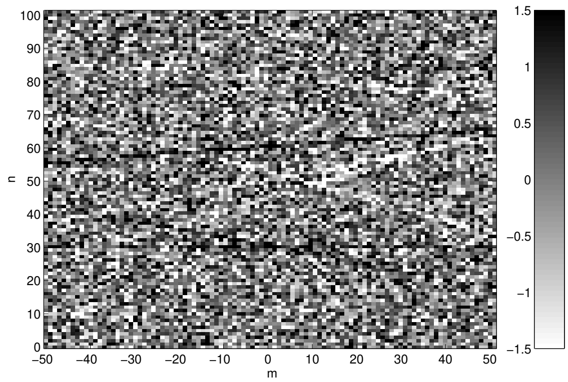

Figure 5.24 The same image contaminated with additive Gaussian noise.

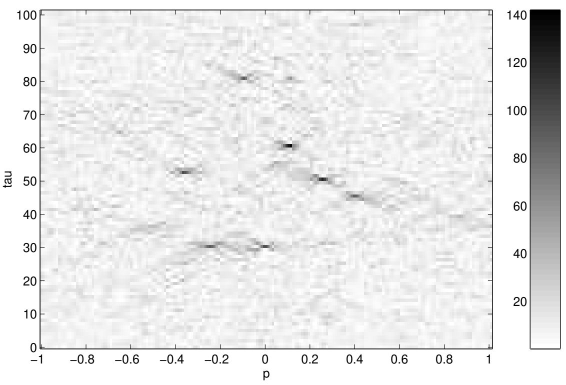

Figure 5.25 The absolute GRT of the noisy image. Note the peaks corresponding to the curves.

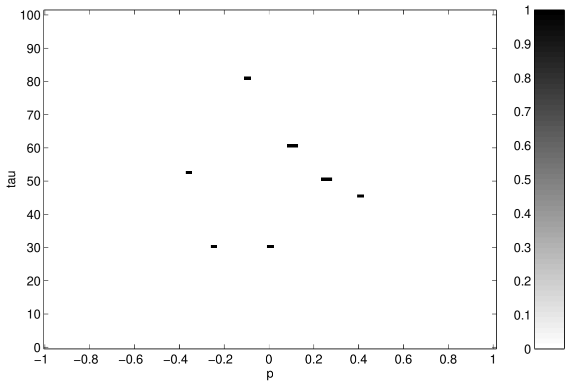

Figure 5.26 Threshold of the absolute GRT using the estimated threshold level.

5.4 Summary

This chapter demonstrated the possibilities and limitations of using discrete Radon transform with respect to noise. In Section 5.1 detection of wiggly lines has been analyzed, and analytical expressions have been derived that models the behavior of the discrete Radon transform with respect to wiggly lines. In was shown in Section 5.2 that the simple incorporation of wiggles can be perceived as a maximum filter used in the image domain.

Finally Section 5.3 presented a statistically based noise analysis of the generalized Radon transform. The analysis was used to derive a threshold level in order to separate curve information and noise in the parameter domain. A numerical example was provided to illustrate the theory.

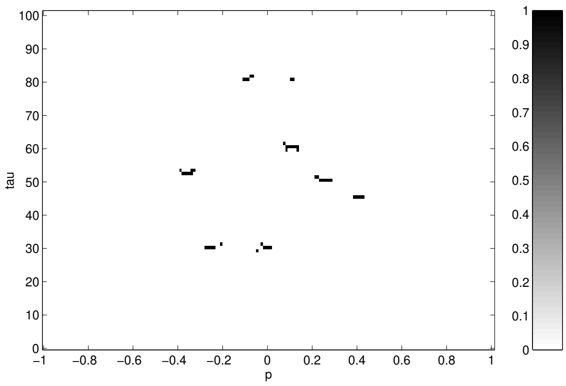

Figure 5.27 Threshold of the absolute GRT using 0.7 times the estimated threshold level.

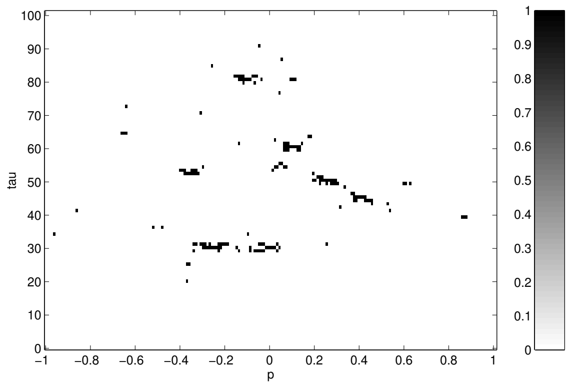

Figure 5.28 Threshold of the absolute GRT using 0.5 times the estimated threshold level.