The Motion Detection Transform and Its Applications

Leo Blume
Sarah A. Rajala

CENTER FOR COMMUNICATIONS AND SIGNAL PROCESSING

Electrical and Computer Engineering
North Carolina State University
Raleigh, NC

December 1983

CCSP-TR-83/8
1.0 INTRODUCTION

The motion detection transform (MDT) maps an image acquired by a sensor into two one-dimensional data vectors which preserve the motion characteristics. Thus, one very important property of this transform is that it significantly reduces the dimensionality of the velocity detection process.

Let \( f(x,y,t) \) represent an image sequence defined over \( 0 < t < T \). The motion detection transform is a procedure which attempts to uniquely separate the time-varying and time-invariant parts of an image sequence. The MDT associates sinusoids with the time-varying portions of the image sequence, such that higher motion is associated with higher frequency sinusoids and the time-invariant portions map to constant levels.

This technical report presents the analysis of the motion detection transform (MDT) starting with a derivation of the continuous-time form of the algorithm in section 2.0. Section 3.0 presents the discrete-time formulation of the algorithm and section 4.0 presents the analytical results for several example cases.

2.0 A VELOCITY DETECTION ALGORITHM

Consider a one-dimensional time sequence \( f(x,t) \) where \( x \) and \( t \) are continuous, real variables. Define a transformation

\[
g(k,t) = \int_{-\infty}^{\infty} f(x,t)w(x,k) \, dx
\]

where \( w(x,k) \) is a weighting function of order \( k \). A properly chosen weighting function will allow us to use this weighted function \( g(k,t) \) to calculate the velocity of a moving object while discriminating against a stationary background.

One appropriate weighting function would be, \( w(x,k) = \cos(2 \pi kx/a) \) and equation (1) then becomes
If the signal $f(x,t)$ is multiplied point by point along the $x$-direction with the amplitude of a sinusoid and the result is summed over all spatial variables $(0,a)$, the result is a single-valued function $g(k,t)$. When the signal $f(x,t)$ is stationary with respect to time, $g(k,t)$ will have a constant value, i.e. $g(k,t) = C$. When $f(x,t)$ is time-varying $g(k,t)$ will vary with time. More specifically, if $f(x,t)$ contains an object moving with constant velocity, $g(x,t)$ will trace out a sinusoid.

If spectral analysis is now performed on $g(k,t)$, the result is a function which has a peak at a location that corresponds to the velocity of the moving object. Taking the Fourier transform of (2) yields

$$G(f) = \frac{1}{T} \int_{0}^{T} g(k,t) \exp(-j2\pi ft) \, dt$$

and

$$G(f) = \frac{1}{T} \int_{0}^{T} \int_{0}^{a} f(x,t)\cos(2\pi kx) \exp(-j2\pi ft) \, dx \, dt$$

where

$$k = \frac{f_{\text{max}}}{v_{\text{max}}}$$

and

$$0 < |k| < \frac{a}{2}.$$ 

A zero value of $k$ implies no sinusoid will be created by a moving object, and a $k$ value greater than $a/2$ produces an aliased spectral response. The actual value of $k$ that is chosen is derived from the frame rate, $f_{\text{max}}$ and the maximum expected target speed, $v_{\text{max}}$, according to equation (4).

This method can be easily generalized for use on time-sequences of two-dimensional spatial images. Let the sequence be represented by $f(x,y,t)$ with $0<x<a$, $0<y<b$ and $0<t<c$. Any resulting velocity vector $v = [v_x, v_y]$, can be extracted as follows:
The velocity components are then extracted by taking the Fourier transform of \( g(k_x,t) \) and \( g(k_y,t) \), [1].

\[
\{G(k_x,f)\} = \frac{1}{T} \int_0^T \{ g(k_x,t) \} \exp(-j2\pi ft) \, dt
\]

The velocity detection algorithm just described can be generalized so that Fourier transforms may be utilized in both phases of the algorithm. If the \( \cos(2\pi kx/a) \) is replaced by \( \exp(-j2\pi kx/a) \) the algorithm becomes

\[
G(f) = \begin{bmatrix} G(k_x,f) \\ G(k_y,f) \end{bmatrix}
\]

where

\[
G(k_x,f) = \frac{1}{T} \int_0^T g(k_x,t) \exp(-j2\pi ft) \, dt
\]

\[
G(k_y,f) = \frac{1}{T} \int_0^T g(k_y,t) \exp(-j2\pi ft) \, dt
\]

and

\[
g(k_x,t) = \frac{1}{a} \int_0^b \int_0^a f(x,y,t) \exp(-j2\pi k_x x/x_f) \, dx \, dy
\]

\[
g(k_y,t) = \frac{1}{b} \int_0^b \int_0^b f(x,y,t) \exp(-j2\pi k_y y/y_f) \, dx \, dy
\]

where \( 0<x<x_f \) and \( 0<y<y_f \) is the extent of the image. In this two-dimensional case, the original image data is first projected onto the x and y spatial axis to permit the utilization of one-dimensional Fourier transforms. This projection does not interfere with the detection of the velocity components.

2.1 Example

To best understand the mechanics of this velocity detection algorithm, consider the following simple example. The image sequence contains a single
point-target moving with velocity \((v_x, v_y)\) over a zero background. The model to be used for \(f(x,y,t)\) is
\[
f(x,y,t) = A \delta(x - v_x t, y - v_y t). \tag{11}
\]
This may be considered to be a spot of grey-level \(A\) moving over a black background. Applying the algorithm, equation (10) becomes
\[
g(k_x,t) = A \exp(j2\pi k_x v_x t / x_f) \tag{12a}
\]
\[
g(k_y,t) = A \exp(j2\pi k_y v_y t / y_f). \tag{12b}
\]
Note that (12a) and (12b) are complex exponentials rotating with angular frequencies determined by the velocities \(v_x\) and \(v_y\). The extraction of the velocity information is straightforward and is implemented using equation (9).
\[
G(k_x,f) = \frac{1}{T} \int_0^T A \exp(j2\pi f_x t) \exp(-j2\pi ft) \, dt \tag{13a}
\]
\[
G(k_y,f) = \frac{1}{T} \int_0^T A \exp(j2\pi f_y t) \exp(-j2\pi ft) \, dt \tag{13b}
\]
where
\[
f_x = k_x v_x / x_f \tag{14a}
\]
\[
f_y = k_y v_y / y_f \tag{14b}
\]
Evaluating (13) yields
\[
G(k_x,f) = A \exp(-j2\pi (f - f_x)T) \text{sinc}((f - f_x)T) \tag{15a}
\]
\[
G(k_y,f) = A \exp(-j2\pi (f - f_y)T) \text{sinc}((f - f_y)T) \tag{15b}
\]
where
\[
\text{sinc}(x) = \sin(\pi x) / \pi x \tag{16}
\]
The sampling functions in equation (15) produce maxima for frequencies \(f_x\) and \(f_y\). Thus, a simple velocity estimation algorithm would find the peaks of \(G(f)\) for each component and then compute the corresponding velocity according to equation (14).

If this velocity detection procedure is to be effectively used on discrete imagery an extension of the analysis must be made in the discrete
domain. The discrete implementation of the motion detection transform (MDT) is given in the next section.

3.0 THE MOTION DETECTION TRANSFORM

The transition to the discrete case is straightforward. Analogous to equations (8-10) are the following:

$$G(f) = \begin{bmatrix} G_x(f) \\ G_y(f) \end{bmatrix}$$

where

$$G_x(f) = \frac{1}{P} \sum_{p=0}^{P-1} g_x(t_p) \exp(-j2\pi f_q t_p)$$

(18a)

$$G_y(f) = \frac{1}{P} \sum_{p=0}^{P-1} g_y(t_p) \exp(-j2\pi f_q t_p)$$

(18b)

and

$$g_x(t_p) = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} f(m,n,t_p) \exp(j2\pi k_x m/M)$$

(19a)

$$g_y(t_p) = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} f(m,n,t_p) \exp(j2\pi k_y n/N)$$

(19b)

In equations (17) - (19), P is the total number of frames processed and the extent of each image is M columns (the x-direction) by N rows (the y-direction). Additionally, the following notation is used in (17) - (19):

$$t_p = p\Delta t$$

(20a)

$$f_q = q\Delta f,$$

(20b)

where \(t\) is the frame sampling time, i.e., the reciprocal of the frame rate, and

$$\Delta f = \frac{1}{P\Delta t}$$

(21)
This parameter is related to the velocity resolutions, $v_x$ and $v_y$, by using (14):

$$\Delta x = M \frac{\Delta f}{k_x} = M(\frac{k_x P \Delta t}{2})$$
$$\Delta y = N \frac{\Delta f}{k_y} = N(\frac{k_y P \Delta t}{2})$$

The properties of the MDT for velocity estimation are best seen by analyzing several specific cases. Further, since (18a) and (18b) are identical except for the difference in coordinate, we may simplify the analysis somewhat by considering only the estimates for the $x$-coordinate parameters. The extension to the $y$-coordinate parameters is straightforward.

4.0 EXAMPLE OF VELOCITY ESTIMATION VIA THE MDT

Several examples will be used to illustrate the properties of the MDT for differing target and noise environments. The examples are taken in order of complexity and are as follows:

1. Subpixel target, constant background;
2. Multiple sub-pixel target, with constant background;
3. Multiple targets, platform jitter, and scintillating background.

4.1 Subpixel Target, Constant Background

The subpixel target is defined as having an extent less than the resolution area of one pixel, such that the target intensity is effectively limited to and constant over a single pixel. This produces a target model

$$f_t(m,n,t_p) = A \delta(m-m_0-v_xt_p, n-n_0-v_yt_p)$$

Where in (23), $A$ is the target intensity, the location $(m_0,n_0)$ is the initial position of the target at time $t_p$, and $(v_x,v_y)$ is the vector velocity in pixels/frame. Note, that the form of (23) defines an impulse function progressing across the image with intensity

$$f(m,n,t_p) = f_T(m,n,t_p) + f_B(m,n,t_p)$$
where
\[ f_B(m, n, t_p) = \begin{cases} 0, & (m, n) = (m_0 + v_x t_p, n_0 + v_y t_p) \\ B, & \text{elsewhere.} \end{cases} \] (25)

Substituting (23) through (25) into (19) produces
\[ g_X(t_p) = A \exp(j2\pi k_x(m_0 + v_x t_p)/M) \]
\[ + B \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \exp(j2\pi k_x m/M) \left[ 1 - \delta(m-m_0-v_x t_p, n-n_0-v_y t_p) \right]. \] (26)

This expression may be simplified considerably using the relation
\[ \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \exp(j2\pi k_x m/M) = \sum_{n=0}^{N-1} \left[ \frac{1 - \exp(j2\pi k_x)}{1 - \exp(j2\pi k_x/M)} \right] \] (27)

Note that for \( k \) equal to any integer the summation in (27) vanishes. This is equivalent to the case of summing \( M \) phasors, all of magnitude \( B \) and equally-spaced in angle, which produces a zero resultant. Using this result in (26) and recognizing the sifting property of the impulse produces
\[ g_X(t_p) = (A-B)\exp(j2\pi k_x(m_0 + v_x t_p)/M) \] (28)

The quantity \( (A-B) \) is the difference between the target pixel and the background pixel grey level. Thus, all that is required for detection is that the target and pixel grey levels differ by one grey level.

One may then compute the spectral estimates via the discrete Fourier transform of (18a). For simplicity, it will be assumed that the target is present in all \( P \) frames, although the analysis to be presented is not limited to this situation. Assuming then that \( g_X(t_p) \) in (28) has nonzero values for \( 0 < p < P-1 \), the frequency estimate from (18) becomes
\[ G_X(f_q) = \frac{1}{P} \sum_{p=0}^{P-1} (A-B)\exp(j2\pi k_x(m_0+v_x t_p)/M)\exp(-j2\pi f_y t_p) \] (29)

Employing the substitution from (14), and factoring terms independent of the time argument \( p \) produces
where \( f \) is the "frequency" of the target (corresponding to the velocity \( v_x \)).

Using the definition of \( t \) from (20) allows the summation in (30) to be evaluated in closed form, giving

\[
G_x(f_q) = \left[ \frac{A-B}{P} \exp(j2\pi kx_m \omega_0/M) \right] \sum_{p=0}^{P-1} \exp(j2\pi(f_x-f_q)tp) \tag{30}
\]

This expression may be simplified using Euler's relationships and, after some straightforward algebra, produces

\[
G_x(f_q) = (A-B) K(m_0,f_q) \frac{sinc (f_x - f_q) \Delta t}{sinc (f_x - f_q) \Delta t} \tag{32}
\]

where

\[
K(m_0,f_q) = \exp(j2\pi kx_m \omega_0/M) \exp(j\pi(f_x-f_q)(P-1) \Delta t). \tag{33}
\]

An analysis of the properties of (32) allows one to unambiguously estimate the target velocity as well as target initial position. These techniques are illuminated below.

(1) Unambiguous Velocity Estimate:

The magnitude of the spectral estimate, \( |G_x(f_q)| \), given by

\[
|G_x(f_q)| = |A-B| \left[ \frac{sinc (f_x - f_q) \Delta t}{sinc (f_x - f_q) \Delta t} \right], \tag{34}
\]

has the form of an aliased spectrum which is periodic in frequency every \( 1/\Delta t \) hertz. This is a direct result of the discrete sampling of the input frames every \( t \) seconds and manifests itself mathematically as shown in (34). Given a true target frequency of \( f \) in (34) there will indeed be a spectral peak of magnitude \( |A-B| \) at this frequency. This may be seen by applying l'Hopital's Rule to (34) as \( f_q \to f_x \). However, again using l'Hopital's Rule for \( f_q \to f_x n/\Delta t \) (\( n \) an integer), one may show that the spectral magnitude also has peaks for \( f_q = f_x n/\Delta t \) and, in general, has a \( 1/\Delta t \) periodicity. Thus, one must limit the spectral range to

\[-1/2\Delta t < f_q < 1/2\Delta t\]
to insure an unambiguous frequency estimate. This equivalency limits the unambiguous velocity range (using (22)) to
\[ \frac{-M}{2k_x\Delta t} < v_q < \frac{M}{2k_x\Delta t} \] (35)

(2) Initial Position Determination:

Once the unambiguous spectral amplitude peak \( f_x \) has been determined from (34), the phase of the spectrum at the target frequency \( f_x \) allows us to find the target initial position, \( m_0 \). Note that the phase at \( f_x \), denoted by \( \Phi[G_x(f_x)] \), is given from (32) as
\[ \Phi[G_x(f_x)] = \Phi[K(m_0,f_x)] \]
\[ = 2\pi k_x m_0 / M \] (36)
since the phases of the sinc (\( * \)) functions cancel. Thus, from (36) it is easily seen that the phase at the target frequency is dependent only upon initial position \( m \), since \( k \) and \( M \) are fixed. Thus,
\[ m_0 = \frac{M}{k_x} \cdot \frac{\Phi[G(f_x)]}{2\pi} \] (37)

4.2 Multiple Subpixel Targets, with Occlusion and Constant Background

The extension of the MDT to the velocity resolution of multiple targets, which may occlude one another is straightforward. It is instructive first, however, to examine the case where there is no mutual target occlusion. Assuming distinct subpixel targets in the image, the model of (23) and (24) produces

\[ f(m,n,t_p) = \sum_{i=1}^{J} f_i(m,n,t_p) + f_B(m,n,t_p) \] (38)
where
\[ f_i(m,n,t_p) = A_i \, \delta(m-m_i-v_{ix}t_p, n-n_i-v_{iy}t_p), \] (39a)
and
\[ f_i(m,n,t_p) = \begin{cases} 0, & (m,n) = (m_i+v_{ix}t_p, n_i+v_{iy}t_p) \\ B, & \text{elsewhere}. \end{cases} \] (39b)
As in the previous example, only the x-component, \( G_x(f) \), will be examined, since the y-component evaluations are identical. For the case of no target occlusions, the analysis is straightforward, following the method of the previous example, giving the spectral estimate \( G_x(f_q) \) as follows:

\[
G_x(f_q) = \sum_{i=1}^{J} (A_i - B) K(m_i, f_q) \frac{\text{sinc}(f_i - f_q) P \Delta t}{\text{sinc}(f_i - f_q) \Delta t}
\]  

(40)

where

\[
K(m_i, f_q) = \exp(j2\pi k_x m_i/M) \exp(j\pi [f_i - f_q] (P-1) \Delta t)
\]  

(41)

If the properties of (40) are analyzed as was done with (4-10), the target velocity and initial position can be estimated unambiguously.

4.3 Subpixel Target, Fluctuating Background

We now examine the case in which a subpixel target, modeled as in (23), has motion over a fluctuating background. In this case, the general image model of (24) holds, with \( f_B(m, n, t_p) \) now being given by

\[
f_B(m, n, t_p) = \begin{cases} 
0, & (m, n) = (m_0 + v_x t_p, n_0 + v_y t_p) \\
B + b(m, n, t_p), & \text{elsewhere},
\end{cases}
\]  

(42)

where \( b(m, n, t_p) \) is assumed to be a zero mean, uncorrelated (in time and spatial variables) Gaussian sequence. This case might model a background which was approximately constant, but subject to minor fluctuations (such as foliage or wave-phenomena on the ocean). Further, let the variance of \( b(\cdot) \) be given by \( \sigma^2 b \).

It is desired to categorize the second-order statistics of the velocity estimator \( Q(f) \) for the stochastic background case of (38). In general it is desired to compute the mean and variances of \( Q(f) \), \( 0 < \lambda < (P-1) \), for the case of a single subpixel target. This is of interest for our simple peak-picking velocity selection criteria, since the situations for which spectral variance is comparable to spectral mean can lead to an extraneous (nontarget) peak exceeding the true target frequency peak.
The remaining sections of this paper present the results of the simulation of this technique and its potential for specific applications.

5.0 SIMULATION OF THE MDT

5.1 Introduction

Having formulated the MDT from an analytic viewpoint, this section will explain the results of simulations performed on the MDT. Simulation experiments on the MDT had two goals. The first was to fully characterize the properties of the basic MDT. The second objective was to apply the MDT (or variants thereof) to the task of reconstructing the moving parts of an image sequence.

The next subsections detail the results of the simulations by applying the MDT to increasingly complex image sequences, illustrating and explaining the output of the MDT as image complexity is gradually increased. First, the MDT used for velocity detection will be discussed. Then, the MDT used to isolate and reconstruct the moving parts of an image will be covered.

All simulations were coded in Pascal. "C" language subroutines were used for image I/O, performing the FFT's, and plotting the results. The image format consists of 512 by 512 pixels, each eight bits deep, having intensity values ranging from 0 to 255.

5.2 Operation of the Basic MDT

The equations presented in Section 3 were simulated in the following way. The projections over a square image window yield two projection arrays. One array represents the projection of the window along rows, the other represents the projection of the window along columns. A projection, or summation, is thus performed for every row and every column. See Fig. 5.2.1 for a diagram depicting the projection process for a single frame in a sequence.
Thus we first take orthogonal image projections of an image window by performing summations across rows and columns so that each row and column has a single projection element - one in the row projection array and the other in the column projection array.

The next step is to multiply these projections by a weighted exponential. The weighting factor, \( k \), determines the number of periods of the exponential that will be multiplied point-for-point by the projection arrays.

For example, a 32 by 32 image window will have 32 elements in its row projection array and 32 elements in its column projection array. These projection arrays are multiplied point for point by a 32 point discrete exponential. The weighting factor \( k \), will determine the frequency of the exponential.

This product of the weighted exponential and orthogonal projection array is summed into a single scalar value. This effects the second summation in equation 19.

After performing the first summation, the complex multiplication, and the second summation, we are left with two values - one generated from the row projection, the other from the column projection. Figure 5.2.2 depicts the second summation for a given image in an image sequence.

For \( n \) frames we thus generated \( 2n \) points. The \( n \) points generated by the row projections and the \( n \) points generated by the column projections are treated separately because they give, when processed by the next step, the velocity information for the vertical and horizontal directions respectively.

Taking the \( n \) points generated by the row projection process, we perform an \( n \) point FFT. As shown in Section 3, this Fourier transform yields the velocity information.
Thus, we have two processes (one for the rows and one for the columns) consisting of: a projection, a complex multiplication, and a second summation - all performed for each of n images. The result of this is then Fourier transformed. The results of the FFT will be illustrated graphically in plots of the resulting frequency spectra. In viewing the plots that accompany the following simulations it should be noted that the plotting routines connect the actual plotted points with straight line segments. The actual data points are indicated by the small squares on the plot.

5.3 Characterization of the Basic MDT

5.3.1 Simple Moving Objects

The velocity detection abilities of the MDT will first be demonstrated. In the first cases to be examined, we perform velocity detection on synthetically generated objects. These objects move in discrete steps. The object undergoes whole-body linear translation of an integral number of pixels per frame. The objects stay at the same "distance" from the viewpoint, and thus are not subject to perspective scaling. Also, the objects do not rotate. This is a slight idealization of the general case in which nonintegral velocities may exist and in which edges may be blurred with the background (these synthetic objects have well-defined edges). Most of the synthetically generated objects are either point objects or squares having edges parallel and perpendicular to the image plane edges.

Unless otherwise stated, velocity is the same in both the horizontal and vertical directions. Likewise, weighting factors will be taken to be the same in both dimensions. In general, we'll look at the MDT output for only one dimension, the other being identical.
In this first case, we have a single pixel object of intensity 200 (in a possible range of 0 - 255), moving against a zero background. The object moves at one pixel per frame in both the horizontal and vertical directions. The length of the image sequence examined was eight.

This is clearly the most trivial experiment that one could perform. This sequence contains no noise, no background (on which the object is superimposed), and no camera motion. The object moves at uniform velocity and is a single pixel in size. This best-case image sequence will provide a benchmark against which we can compare the results when conditions are not so favorable.

In this first experiment, the exponential weighting factor, $k$, is 1.

Figure 5.3.1 (a) shows the resulting MDT spectrum for the horizontal direction. The vertical velocity spectrum is identical since the object is moving at the same velocity in both directions and the background against which it moves is isotropic (all zero in this case). The effect of a nonisotropic background will be explained later.

The output of the Fourier transform of equation 18 (implemented as a FFT), will be presented graphically. For an N point FFT, positive frequencies appear in the range 0 to $N/2 - 1$. Frequencies from $N$ down to $N/2 + 1$ represent the negative half of the spectrum. The frequency $N/2$ is the foldover frequency.

A sharp peak is evident at the $f = -1$ frequency. Calculating the velocity as

$$|\text{vel}| = \frac{|f|}{k}$$

we get the absolute velocity to be $1/1 = 1$. The velocity is positive as the peak occurs in the negative portion of the spectrum. Negative velocities will appear in the positive frequency range of the spectrum (frequencies one to four).
Figure 5.3.1 (b) shows the results of performing the MDT on a sequence containing the same object, this time moving at a positive velocity of 2 pixels per frame in both the horizontal and vertical directions. The spectral peak occurs at $f = -2$ indicating that the velocity is $f = 2k$.

Thus we have seen that in this case of a bright point object on a zero background, the peak is clearly defined - there is no problem detecting the velocity generated spectral component.

In the next example, the single pixel object is again moving at a velocity of 1 pixel/frame but this time the background is nonzero. Instead, it is a uniform field (intensity = 100, half the object intensity).

The resulting spectrum in Fig. 5.3.2 shows again a single peak produced by the moving object. Since the background is uniform, the product of a single period exponential and the background gives the single period of the exponential multiplied by some constant scaling factor. When this product is summed, by the the second summation, the result is zero.

In general, the effect of any uniform background will always be eliminated by being multiplied by an integral number of periods of an exponential in the manner shown above.

In the next experiment, the MDT was performed on a 32 by 32 window of a complex image sequence, of the street scene pictured in Fig. 5.3.3, that contains both noise and camera motion but no moving object. The results are shown in Fig. 5.3.4 (a). As can be seen, there is a high d.c. peak and content at all other frequencies as well. As there is no moving object in this window, this spectral content is entirely generated by what we shall soon consider as merely background effects (as opposed to moving object effects).
The next case examined is that of the same single pixel object moving on the same complex background as before. We see in the spectra of Fig. 5.3.4 (b) that the velocity peak is accompanied by both a high amplitude d.c. peak and by content at the other frequencies as well. The large d.c. value indicates that the time sequence of summed projection products has a high time invariant content.

In comparing this plot with the previous one, it is noted that the velocity generated peak is higher in amplitude than all other peaks except the d.c. peak. This result is characteristic of superimposing any moving object on a complex background. Velocity determination by simple peak detection must now discriminate against the d.c. frequency peak. The velocity spectra for the velocity in the vertical direction is shown in Fig. 5.3.4 (c). Though similar to the plot in (b), the difference is accounted for by the fact that the background image being nonisotropic will generate different time-varying orthogonal projections for both the vertical and horizontal directions.

As will be demonstrated in the section titled MDT analysis, to obtain impulsive velocity spectra, the relationship between \( m \), the number of frames, \( n \), the window size, and \( v \), the velocity had to be

\[
\frac{m \times v \times k}{n} = \text{an integer.} \tag{44}
\]

An impulsive spectrum is one in which the velocity of the object results in an impulse at a frequency proportional to the velocity. When the above relationship does not hold, the resulting spectra contain sinc or other nonimpulsive functions. In all the experiments examined thus far, this relationship has been maintained even though \( m = 8 \), \( v = 1 \), \( k = 1 \), and \( n = 32 \). Note that these values should give

\[
\frac{8 \times 1 \times 1}{32} = \frac{1}{4} \Rightarrow \text{not an integer.}
\]
The program automatically adjusts the frequency $k$ of the multiplicative exponential to insure that the relationship $(m \ast k) / n$ holds, (the program of course can not know what $v$ is in advance).

The adjustment has the form:

$$k' = k \ast \frac{n}{m}.$$  

So here,

$$k' = 1 \ast \frac{32}{8} = 4.$$  \hspace{1cm} (45)

Put in words, this equation says that the number of periods $k$ of the complex exponential in the image window is increased to $k'$ so that there will still be $k$ periods of the exponential in the first $m/n$ th part of the window.

This adaptation is necessary in order to compute the velocity as

$$v = \frac{|f|}{k}$$

whenever the window size and the number of frames are not equal.

Now, $(m \ast k')/n = 8 \ast 4 / 32 = 1$. This guarantees an impulsive spectrum.

An in depth discussion of the requirements for obtaining impulsive spectra follows in Sect 5.4.

Note that for an 8 point FFT, the nonaliased velocity range is constrained to be in the range of $-4, -3, \ldots, 3, 4$.

5.3.2 **K Value Determination**

For a $n$ by $n$ image window, the frequency $k$ of the multiplicative exponential can range from $-n/2$ to $n/2$. The value of $k$ selected determines the range of velocities detectable. As indicated by the relationship

$$v = \frac{f}{k},$$

the $k$ value and the velocities detectable have a reciprocal relationship, the larger the value of $k$, the smaller the lowest detectable velocity. For

\[\text{\ldots} \]
example, with \( k=1 \), the lowest detectable velocity is clearly one pixel per frame. This velocity generates an impulse at \( f = -1 \) which is the lowest non-dc value \( f \) can assume. The highest detectable unaliased velocities are \(+4\).

With \( k = 2 \), a peak at \( f = 1 \) would, by equation 43, indicate a velocity of \( 1/2 \). For this value of \( k \), the highest unaliased velocities detectable are \(+2\) (corresponding to a spectral peak at \( f = 4 \)). The range of detectable velocities has been reduced but the resolution has been increased to velocities separated by half a pixel per frame. Velocities that produce aliased spectra have peaks that cross the foldover frequency. These peaks are indistinguishable from those generated by velocities of opposite sign and are thus ambiguous.

It should be noted that some basic a priori information about the velocity of the object is implicit in the selection of \( k \) and the interpretation of the resulting spectrum. For example, in an eight image sequence, if a peak at a frequency of \(-1\) is generated with a \( k \) value of 1, this could indicate an unaliased velocity of \(+1\) or an aliased velocity of \(-7\) or any number of higher magnitude aliased velocities such as \(-15\), \(+9\), \(-23\), \(+17\), and so forth.

5.3.3 Nonmapping Velocities

Those velocities that do not map directly to a given frequency in the MDT spectrum for some value of \( k \) will present problems for velocity determination based on simple peak detection. Though certain aspects of this situation are discussed in depth in Section 5.4, an example will serve to illustrate the point. Given a sequence of 8 images, and a \( k \) value of 1, an object moving at 3/2 pixels per frame will fall between the resolved velocities of one and two. The resulting spectrum will be nonimpulsive regardless of window size. With a
window of 8 pixels, a 3/4 period exponential is traced by the object. With a window size of 16, 1.5 periods of an exponential are traced out. The Fourier transform of neither exponential will result in an impulsive spectrum. There are two possible remedies to the situation. First, velocity may be determined by selecting a value of \( k \) that results in an appropriate range of detectable velocities. In effect, we try another value of \( k \) to see if it results in an impulsive spectrum that will work with our simple peak detecting velocity determination technique. Here, a \( k \) value of two generates a spectrum with an impulse at a frequency of negative three. This, by equation 43, indicates a velocity of 1.5.

Alternately, velocity determination based on a more sophisticated technique, one not requiring impulsive spectra, could be performed.

5.3.4 Multipixel Objects

Having examined several cases where the object size was a single pixel, we will now examine the output of the MDT as we increase the size and complexity of the object.

First, we shall perform the MDT on a sequence having the following parameters:

- length = 8
- \( k_x = k_y = 1 \)
- object size = 4 by 4 pixels
- intensity = 200 and uniform
- window size = 32 by 32
- vertical velocity = horizontal velocity = 1
- background = uniform field, intensity = 0

The results are show in Fig. 5.3.5. Clearly, the velocity is easily detected.
One of the situations that may degrade performance for multipixel moving objects is when the object may enter or exit the window being examined in stages. This degradation is illustrated in Fig. 5.3.6, where we have the output of the same sequence examined above, except this time, the window is only 8 by 8 pixels. Consequently, the box starts moving out of the window during frame 5, and by frame 8 there is only the upper left corner of the box (a single pixel) left in the window.

Text, a multipixel moving object is overlaid on a time-varying background as was done in Fig. 5.3.4. The object is this time 6 pixels by 6 pixels, and the background is a time-varying street scene. The window size is 16 by 16 insuring that the object does not leave the area of examination. Again, the frequency of the weighting exponential is one. The results shown in Fig. 5.3.7 show that the correct velocity is clearly detectable, yet the background time variation (induced by noise and camera motion, primarily) contributes spurious velocity spikes.

The relative level of these spurious spikes is increased further when the object leaves the window of examination. This result is seen in Fig. 5.3.8 where the sequence is the same as in Fig. 5.3.7 except that the window size has been reduced to 8 by 8 pixels. This causes the box to start exiting the scene at frame 3.

Another problem encountered with multipixel objects is that it is possible for the size and intensity of the object to interact with the multiplicative exponential in such a way that the exponential is 'nulled out.' In this case, the MDT fails.

For example, if an 8 by 8 pixel box of uniform intensity were moving at 1 pixel per frame, if the window size were set to be 32 by 32, and if the number of frames examined was 8, then for \( k = 1 \), the effective period of the multiplicative exponential, given by equation (2), would be 8 pixels. Thus,
when multiplied by the projection of the 8 pixel box, which is oriented so that its principal axes are parallel with that of the projections, the resulting product when summed is zero.

Changing the weighting factor to \( k = 2 \) will not help. The period of the exponential is now 4 pixels in length, but the uniform intensity projection of the box still spans an integral number of periods of the multiplicative exponential thus nulling it out upon performing the second summation (the summation across the projection array).

Similarly, seeking out other values of \( k \) to get one that will yield velocity information will not work unless \( k \) creates periods that are either greater that 8 pixels in length or less than 8 pixels but such that the period length is not an integer divisor of the box size 8.

The incorrect results that can be generated by this effect are seen in Fig. 5.3.9. The parameters of this sequence are

- length = 8
- window = 32 by 32
- \( k = 1 \)
- object: uniform, intensity of 125
- velocity of object = 1
- object superimposed on 8 frames of street scene.

As can be seen, the highest non-dc peak occurs at the frequency corresponding to a velocity of +2. This indicates a velocity of +2 and is clearly incorrect.

In some applications, it may be argued that such objects are not likely to occur. In teleconferencing environments, however, rectangular objects are
likely to be passed around a conference room—thus raising the possibility of encountering this effect.

To illustrate the subtle interaction of effects, the next sequence examined contained the same 8 by 8 box moving at the same velocity of 1 pixel/frame. This time, however, the box was placed in an 8 by 8 window. As a result, the box starts leaving the window from the second frame. Previously, the fact that the multipixel object was leaving the scene had a degrading effect on the MDT output. Here, the fact that the object leaves the scene a pixel at a time, eliminates the exponential nulling for frames 2 through 8—thus preserving the velocity information. The MDT output for this case is shown in Fig. 5.3.10.

The sequences examined thus far in which a multipixel object is overlaid on a time-varying background exhibit considerable realism in several respects, and yet are still idealistic in others. The level of camera motion and noise present is probably representative of any typical video scene. No teleconferencing scene is likely to be worse in these regards. However, the object itself and its motion is idealistic. The object moves at constant velocity in a straight line, and experiences no rotation. The object itself is a single uniform intensity box.

In the following paragraphs we will attempt to remove some of these idealizations and also elaborate on the noise performance of the MDT.

5.3.5 Complex Moving Objects

In order to test the velocity detection characteristics of the MDT on a more complex moving object, a vaguely car-shaped object consisting of several rectangles of different sizes and intensities was created. This object as superimposed on the street scene is seen in the photograph in Fig. 5.3.11.
For this sequence, the object was given a velocity of 2 pixels/frame. The MDT spectrum is shown in Fig. 5.3.12. Clearly, the velocity of this more complex object is easily detected.

5.3.6 Noise

The street scene on which the synthetically generated objects have been placed contains noise and camera motion as was noted earlier. The exact level of these perturbations is hard to estimate however. Also, the objects superimposed on these images were not, themselves, corrupted by noise. To more adequately test the noise immunity of the MDT, a series of experiments was performed in which noise at various levels was added to images. The results indicated that the MDT was relatively immune to high levels of noise.

For example, a nonstationary street scene sequence that contained the superimposed car-shaped object was corrupted with pseudo-random gaussian noise with mean intensity 70 (intensity range 0-255) and an intensity variance of 20. A closeup of the "car" in Fig 5.3.13 reveals the extent of the degradation. As seen in the spectral output of Fig. 5.3.14, however, the velocity peak is still easily differentiated from the other spectral components.

Fig 5.3.13 reveals the extent of the degradation. As seen in the spectral output of Fig. 5.3.14, however, the velocity peak is still easily differentiated from the other spectral components.

5.3.7 Nonconstant Velocities

The MDT velocity detection performance for objects moving at non-constant velocity is variable and tends to be poor. Velocity determination based on simple peak detection will in general fail for even simple cases of nonconstant velocity.
If it is desired to get some velocity measure such as the average velocity for an object moving at nonconstant velocity, then a more elaborate velocity determination technique is needed.

### 5.3.8 Multiple Moving Objects

The MDT is able to detect the velocities of multiple objects moving within the area of examination. As a consequence of the weighted exponential multiplication, each moving object generates a sinusoid whose frequency is proportional to its velocity. The Fourier transform of this sinusoid gives a frequency impulse from which we calculate the velocity. Multiple moving objects will thus generate multiple sinusoids whose sum will be Fourier transformed.

As long as the multiple objects have differing velocities, the sinusoids generated by the objects will have different frequencies. Upon Fourier transforming, we will be able to tell the number of moving objects by counting the velocity spikes. We can tell the velocity of the objects by the frequency of these velocity generated spikes. We will not be able to associate a given velocity with a particular object as such information is lost in the compression caused by the two summations in the MDT. The above assumes $m^*v = n$ holds. If not, we could generate sincs where sinc lobes would look like lower intensity objects of differing velocities.

Multiple objects moving at the same velocity will generate sinusoids of the same frequency. These sinusoids will Fourier transform to a spike at the same frequency. In this case, it is not possible to determine the number of moving objects from an examination of the frequency spectrum.

As an illustration of these concepts, Fig. 5.3.15 shows the spectra produced for a sequence containing two moving objects. Both objects had a vertical velocity of $+1$. In the horizontal direction, however, one of the
objects had a velocity of +1, while the other had a velocity of -1. Fig. 5.3.15 (a) shows the spectrum for the vertical velocity components. Clearly there is only one peak, and this peak corresponds to a velocity of +1. Here both of the objects have contributed to this one peak because of their identical vertical velocities.

In Fig. 5.3.15(b), the spectrum for the horizontal velocity components is shown. Here it is seen that we have peaks for a velocity of +1 and for a velocity of -1. Thus, the differing horizontal velocities of the two objects show up in the spectrum.

5.4 MDT Analysis

In this section, we shall examine the effect that window size, frame sequence length, velocity, object characteristics, and background characteristics have on the ability to detect object velocity as a function of k, and the ability to separate object-generated information from that information generated by other effects for this projection-based MDT.

The purpose here is to formalize the relationship between these variables and show how they interact.

Here, instead of performing the projection, complex multiplication, and sum, we will generalize and replace the multiplication and sum by a Fourier transform. Thus instead of multiplying the projection by a single exponential of frequency k and then summing, we are multiplying by all exponentials of frequency -n/2 to n/2 and then summing.

In the sections that follow, three cases will be examined. Each case defines a relationship between m, the number of frames, v, the object velocity, and n, the window size. It will be shown how the relationship between the above parameters interact with characteristics of the object to affect MDT output.
5.4.1 Case 1: \( m * v = n \)

Figure 5.4.1 shows the frequency space results for a single pixel object moving at 1 pixel per frame for 8 frames. In accordance with the relation
\[
m * v = n,
\]
the window size is 8 pixels in both dimensions. This is the "trivial" case that was examined first in the Section 5.3 "Characteristics of the Basic MDT." It is noted that all impulses are of equal amplitude. The process creating this spectrum will be described next.

The projection of the single pixel object, \( f(x) \), is analogous to an intensity impulse which is shifted in time with each frame. The spectrum of a single impulse is \( F(k) \) and is constant regardless of the value of \( k \). The spectrum for frame 2 is the spectrum for frame 1, \( F(k) \), shifted in phase by a multiplicative exponential that accounts for the position shift of the impulse with time.

Thus for frame 2, the spectrum would be \( F(k)*\exp(-j2\pi k dx/N) \) where \( k \) is the spatial frequency variable, \( dx \) is the frame to frame displacement of the object, and \( N \) is the window size. For this 8 frame sequence where the frame-to-frame displacement \( dx \) is 1 pixel, we have:

Frame n:

\[
F(k) * \exp \left( \frac{-j2 \pi k dx}{N} \right)
\]

\( F(k) = \text{FFT of projection}, \)
the projection here is an impulse.
\( k \) = spatial frequency variable
\( dx \) = displacement by frame n \( (n * \text{vel}) \)
\( N \) = window size

frame 1: \[ F(k) \]
frame 2: \[ F(k) * \exp \left( \frac{-j2 \pi k * 1}{N} \right) \]
frame 3: \[ F(k) * \exp \left( \frac{-j2 \pi k * 2}{N} \right) \]
frame 4: \[ F(k) * \exp \left( \frac{-j2 \pi k * 3}{N} \right) \]
frame 5: \[ F(k) * \exp \left( \frac{-j2 \pi k * 4}{N} \right) \]
frame 6: \[ F(k) * \exp \left( \frac{-j2 \pi k * 5}{N} \right) \]
frame 7: \[ F(k) * \exp \left( \frac{-j2 \pi k * 6}{N} \right) \]
frame 8: \[ F(k) * \exp \left( \frac{-j2 \pi k * 7}{N} \right) \]

\(- (\text{window size}) / 2 \leq k \leq (\text{window size}) / 2\)
Now take the second transform to go from k-t space to k-f space, where k is the spatial frequency variable output by the first FFT, t is time, and f is the frequency variable in the domain produced by taking the FFT of the spatial frequency.

We note that the function F(k) is not a function of time and thus can be taken outside the Fourier transform integral. What remains is a sequence of exponentials (frame 1 may be thought of as an exponential with a zero argument). It is noted that the above sequence of exponentials will trace out one complete period for k = 1, two periods for k = 2, and so on.

It is also noted that the Fourier transform of an exponential of period 1 is an impulse at frequency f = -1. The Fourier transform of an exponential of period 2 is an impulse at f = -2.

Thus, for k = 1, upon taking the second Fourier transform, we get

\[ F( k=1 ) \ast \text{an impulse at } f = -1. \]

For k = 2, we get

\[ F(k = 2) \ast \text{an impulse at } f = -2. \]

This effect is seen in Fig. 5.4.1 where each triangular peak is the plotted representation of an impulse.

Since, the transform of the image intensity impulse, F(k), is d.c, it is itself independent of k. Thus, we get equal amplitude impulses for all values of k.

To illustrate the behavior of the transform for velocities greater than 1 and m * v still equal to n, the MDT was performed on a sequence of images containing a single pixel object moving at a velocity of 2 pixels per frame. Thus, to obey the proper relationship, the number of frames is 8 and the window size was 16. The output is given in Fig. 5.4.2.
It is seen in this figure that the slope of the impulse vector is the same as in Fig. 5.4.1. The wrap around effect visible in the vector of impulses is explained by examining the output of the first Fourier transform. Taking the discrete Fourier transform of the single pixel object as it moves with a displacement of 2 pixels per frame yields for the 8 frames:

Frame 1: $F(k)$
Frame 2: $F(k) \times \exp\left(-j2\pi k^2/16\right)$
Frame 3: $F(k) \times \exp\left(-j2\pi k^4/16\right)$
Frame 4: $F(k) \times \exp\left(-j2\pi k^6/16\right)$
Frame 5: $F(k) \times \exp\left(-j2\pi k^8/16\right)$
Frame 6: $F(k) \times \exp\left(-j2\pi k^{10}/16\right)$
Frame 7: $F(k) \times \exp\left(-j2\pi k^{12}/16\right)$
Frame 8: $F(k) \times \exp\left(-j2\pi k^{14}/16\right)$

As a function of $k$, the spectrum of the second Fourier transform taken on the above sequence is

$k = 0$: $F(0) \times \text{impulse at } f = 0$. An impulse at $f = 0$ is generated by taking the F.T. of a d.c. level of one (the value of the exponential when its argument is zero).

$k = 1$: $F(1) \times \text{impulse at } f = -1$. An impulse at $f = -1$ is generated by taking the F.T. of an exponential of a single period.

$k = 2$: $F(2) \times \text{impulse at } f = -2$. An impulse at $f = -2$ is generated by taking the F.T. of an exponential of two periods.

$k = 8$: $F(8) \times \text{impulse at } f = 8$. An impulse at $f = 0$ is generated by taking the F.T. of an a d.c. level. This d.c. level is generated because the argument to the exponential is an integral multiple of $2\pi$ for $k = 8$.

Note that previously a $k$ value of one would generate an impulse at a frequency of minus two for this object. Here, however, we do not make the $k$ to $k'$ adjustment discussed in Section 5.3 for the situation when the number of frames is not equal to the window size. In this situation, to calculate the velocity by Equation 43, the value of $k$ used to generate a spectrum must be divided by two before being used in the equation. For example, the
spectrum generated by a \( k \) value of one created an impulse at a frequency of minus one. Dividing this \( k \) value by one half and substituting it into Equation 43 gives a correct velocity of two.

5.4.1.1 Constant Backgrounds

The cases examined thus far have had a single pixel object on a zero background. Placing the moving object on a uniform nonzero background causes the background to map to the \( f = 0, k = 0 \) point in \( k-f \) frequency space. The reason for this is that the background is spatially invariant and so it maps to \( k = 0 \) after the first, spatial, Fourier transform. The resulting sequence of d.c. impulses is itself time invariant and thus maps to \( f = 0 \) when the second FFT is performed.

Thus the time sequence consisting of the output of the spatial FFT's for each of the image projections will have an impulse at \( k = 0 \) produced by the uniform background.

The uniform nonzero background appearing as a large magnitude impulse at \( f=0, k=0 \) in transform space, tends to wash out the remaining parts of the spectrum (those parts generated by the moving object itself). This is seen in Fig. 5.4.3 (a). This figure illustrates the spectrum for the same object as depicted in Fig. 5.4.1, but this time with a uniform nonzero background.

As can be seen, the d.c. peak is sufficiently high so as to make invisible the rest of the spectrum. Fig. 5.4.3 (b) is identical to Fig. 5.4.3 (a) except that this time a log scale has been used to bring out the features of the entire spectrum.

Note that the impulse at \( f=0, k=0 \) has contributions both from the uniform background and from the moving point object.
5.4.1.2 Nonconstant background

Fig. 5.4.4 illustrates the spectrum that results from superimposing a point object with velocity = 1, on 8 identical frames of a street scene. The window size was also eight.

Again, there is a large d.c. term from the nonzero background and so a dB scale has been used.

Here, because the background is no longer uniform (isotropic) but is spatially variant, spectral components of the background appear at values of k other than simply k = 0. Since the same background image was used for all 8 frames, the spatial spectrum is time invariant and thus those components of the final spectrum in Fig 5.4.4 that were generated by the background, all appear along the f = 0 axis (these components are circled in the plot).

The difference in the x and y plots in Fig. 5.4.4 arise from the fact that the spatial variation of the background scene differs in the x and y dimensions. Thus, the spectral components along the f = 0 axis have different magnitudes in one dimension than in the other.

In some sense, velocity determination may be thought of as the task of isolating velocity generated impulses in k-f space from those impulses produced by other effects such as noise. As we have seen, there are some values of k which, depending upon object characteristics, object velocity, and background, will produce spectra in which the object's velocity impulse is not distinguishable form the spectral content produced by other scene components.

Any real world scene will have a nonuniform background creating spectral components all along the f = 0 axis. These components will interfere with isolating the object from the background whenever the spectrum of the object crosses the f = 0 axis.
Additionally, real world scenes will not have time invariant backgrounds (though they may be nearly time invariant). As a result, there will be spectral components of the background at points other than along the f = 0 axis - further complicating velocity peak isolation.

5.4.2 Case 2: \( m \cdot v < n \)

5.4.2.1 Single Pixel Objects

The behavior of the MDT for cases where the number of frames multiplied by the velocity is less than the window size is now examined.

Again, we will start with a single pixel object moving at a velocity of 1 pixel per frame for 8 frames. This time, however, the window size is taken to be 16 pixels wide. So the window size is twice the number of frames. The resulting spectrum is shown in Fig. 5.4.5. As can be seen, there are impulses at \( k = 0, 2, 4, 6, 8, 10, 12, \) and \( 14 \). The impulses are circled on the plot. For the other values of \( k \), the spectra have several components being characterized by broad flat peaks.

After the first Fourier transform we have:

\[
\begin{align*}
(1) & \\
\text{Frame 1:} & \quad F(k) \\
\text{Frame 2:} & \quad F(k) \cdot \exp \left( (-j2 \cdot \pi \cdot k \cdot 1) / 16 \right) \\
\text{Frame 3:} & \quad F(k) \cdot \exp \left( (-j2 \cdot \pi \cdot k \cdot 2) / 16 \right) \\
\text{Frame 4:} & \quad F(k) \cdot \exp \left( (-j2 \cdot \pi \cdot k \cdot 3) / 16 \right) \\
\text{Frame 5:} & \quad F(k) \cdot \exp \left( (-j2 \cdot \pi \cdot k \cdot 4) / 16 \right) \\
\text{Frame 6:} & \quad F(k) \cdot \exp \left( (-j2 \cdot \pi \cdot k \cdot 5) / 16 \right) \\
\text{Frame 7:} & \quad F(k) \cdot \exp \left( (-j2 \cdot \pi \cdot k \cdot 6) / 16 \right) \\
\text{Frame 8:} & \quad F(k) \cdot \exp \left( (-j2 \cdot \pi \cdot k \cdot 7) / 16 \right)
\end{align*}
\]

\( F(k) \) is again constant for all \( k \). For \( k = 1 \), we take the FFT of the above sequence (1). Note however, that with \( k = 1 \), the argument to the exponential ranges from only 0 to \( 2 \cdot \pi \cdot 7 / 16 \). In other words, only half a period of the exponential is traced out. Taking the second FFT of this half period exponential results in the broad spectrum that is seen for \( k = 1 \).

For \( k = 2 \), we again take the FFT of the above sequence (1). This time, the
argument of the exponential is such that the exponential has values over an entire period. Taking the second FFT of this exponential results in $G(f,1)$ having the expected impulse at $f = -1$ ($f = 7$).

In fact, all even values of $k$ will produce spatial exponentials that trace out an integral number of periods and thus produce impulsive spectra upon again being Fourier transformed.

All odd values of $k$ will produce spatial exponentials that trace out incomplete periods and thus do not produce impulsive spectra upon again being Fourier transformed. Specifically, for this case, odd values of $k$ produce

$$0.5, 1.5, 2.5, \ldots$$

periods of the spatial exponential.

This result leads to the following conclusion. We get an impulsive spectra for a value of $k$ whenever:

$$(k \times \text{vel} \times \text{number of frames}) / \text{window size} = \text{an integer} \quad (46)$$

Also evident is the fact that the number of impulsive spectra we get will equal the number of frames over which we are taking the second FFT.

Using the above relation, let us predict the values of $k$ for which we will get an impulsive spectrum for a single pixel object with velocity = 1, number of frames = 4, and window size = 16.

$$(k \times 1 \times 4) / 16 = \ldots -2, -1, 0, 1, 2, \ldots$$

where $-8 \leq k \leq 8$

or $k = 0, 1, \ldots , 15$

Note: the range of permissible $k$ values are determined by the window size since $k$ is the spatial frequency variable.

This says we will get impulses at $k = 0, 4, 8, 12$. This result is verified by the output spectrum in Fig 5.4.6, where impulses are once again circled.
5.4.2.2 Multipixel objects

Equation 46 describes the values of \( k \) for which impulsive spectra will be produced for single pixel objects only. Next we will see what happens when the object is a uniform intensity multipixel object.

Fig. 5.4.7 is the resulting spectrum for a 4 x 4 uniform intensity box moving at 1 pixel per frame. The sequence consists of 8 frames. In order to keep all or part of the object from moving out of the scene, the window size must be at least 12 x 12. A window size of 16 by 16 has been selected (since the program uses an FFT that only operates on powers of two).

For the above parameters, equation 46 predicts that we will get impulses at \( k = 0, 2, 4, 6, 8, 10, 12, \) and 14. Fig 5.4.7 shows, however, that only those spectra at \( k = 0, 2, 6, 10, \) and 14 have impulses at the proper location. Furthermore, it is seen that the magnitude of the impulses is no longer constant.

The reason for this is that the function \( F(k) \) that appears as the result of the first FFT is no longer constant with \( k \). The projection of the object on the axes is no longer an impulse but a rectangular function 4 pixels wide. Taking the FFT of the function below:

\[
f(x):
\begin{align*}
* & * & * & * \\
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16
\end{align*}
\]

yields a sinc function with zero crossings at \( k = 4, 8, \) and 12. This explains why there are no impulses at these values of \( k \). The nonuniform magnitude of the remaining impulses is also explained by the fact that these spectra are being weighted by \( F(k) \), a sinc that is a function of \( k \).
The next sequence examined contains an 8 x 8 uniform intensity box moving at 1 pixel per frame. The sequence length is 16 and the window size is 32 x 32 (the window size must be at least 24 x 24 to keep object from moving out of scene). The background is all zero. The resulting spectrum is in Fig. 5.4.8.

Equation 46 predicts impulses at \( k = 0, 2, 4, 6, 8, 10, \ldots, 30 \). Fig. 5.4.8 shows impulses at \( k = 2, 6, 10, 14, 18, 22, 26, \) and 30.

As in the case examined previously, the "missing" impulses are accounted for by the function \( F(k) \). By definition, \( F(k) \) is the spectrum of the projection \( f(x) \) of the image on an axis for frame 1. Thus, \( f(x) \) is a rectangular function 8 pixels wide whose left edge is the left edge of the 32 pixel wide window.

The FFT of \( f(x) \) yields \( F(k) \) which is a sinc function with zero crossings at \( k = 4, 8, 12, 16, 20, 24, \) and 28.

\( F(k) \) multiplies the exponential "space delay" function of \( k \). \( F(k) \) is time invariant and thus does not participate in the second FFT. The second FFT transforms the sequence of exponentials into the vector of impulses predicted by Equation 46. This vector is multiplied by \( F(k) \) so that wherever \( F(k) \) is zero, the resulting spectrum is zero. This accounts for the fact that the spectra for certain values of \( k \) are all zero.

Here again, the d.c. term in the k-f spectrum is so high that in order to bring out the detail of the other spectra, a log scale was used.

The next test was to verify that Equation 46 held for velocities other than one. Fig. 5.4.9 shows the spectra resulting from performing the MDT on an 8 frame sequence containing a single pixel object of intensity 200 moving at 2 pixels/frame against a zero background. Window size is 32 x 32. Inserting the above values into Equation 46 gives:

\[
\frac{(k \times 2 \times 8)}{32} = \text{an integer}
\]
This implies impulses at \( k = 0, 2, 4, 6, 8, 10, 12, \ldots \), 30. Fig 5.4.9 confirms that this is the result.

5.4.3 Case 3: \( m \times v > n \)

The last broad category to be examined is when the number of frames multiplied by the velocity is greater than the window size.

To analyze the result, the same image sequence used to generate Fig. 5.4.1 was used. The sequence contains a single pixel object moving at a velocity of 1 pixels/frame over a zero background. This time, however, the frames analyzed were increased to 16 and the window size held constant at 8 x 8 pixels. Consequently,

\[ m \times v = 2 \times n. \]

In taking the projection of the image, we generate the projection arrays \( f(x) \) and \( f(y) \), representing the row and column projection arrays respectively. The left edge of the window coincides with the starting position of the object, so that the projection array for frame 1 contains the object's projection in its leftmost element of the window. By the eighth frame, the projection of the object appears in the rightmost position of the window.

As only the part of the image within the window is projected onto the axis, the projection arrays for frames 9 through 16 are all zero - since the object has already moved out of this window. So for frames 1 through 8, \( f(x) \) is the projection of an impulse moving 1 position with each frame, and for frames 9 - 16, \( f(x) \) is zero. Taking the first FFT of the 16 \( f(x) \)'s yields:

- **Frame 1:** \( F(k) \)
- **Frame 2:** \( F(k) \times \exp \left( -j2 \times \pi \times k \times 1 \right) / 8 \)
- **Frame 3:** \( F(k) \times \exp \left( -j2 \times \pi \times k \times 2 \right) / 8 \)
- **Frame 4:** \( F(k) \times \exp \left( -j2 \times \pi \times k \times 3 \right) / 8 \)
- **Frame 5:** \( F(k) \times \exp \left( -j2 \times \pi \times k \times 4 \right) / 8 \)
- **Frame 6:** \( F(k) \times \exp \left( -j2 \times \pi \times k \times 5 \right) / 8 \)
- **Frame 7:** \( F(k) \times \exp \left( -j2 \times \pi \times k \times 6 \right) / 8 \)
- **Frame 8:** \( F(k) \times \exp \left( -j2 \times \pi \times k \times 7 \right) / 8 \)
The FFT of the projection of frame 1 yields $F(k)$, the Fourier transform of an impulse, which is a constant— invariant with $k$.

The FFT of the projection of frame 2 yields $F(k)$ delayed by the one pixel phase shift which appears as the multiplicative exponential. The FFT of the other frames' projections follow from that of frame 2.

In the above example, valid values for $k$ are 0 to 7, since the window is 8 pixels wide and there are thus 8 spatial frequencies possible. Substituting the range of values of $k$ into the above sequence yields $F(k)$ * an exponential with $k$ periods ($0 \leq k \leq 7$). In other words, the above time sequence of Fourier transformed projections will, for $k = 2$ (say), produce two periods of an exponential times a scaling factor of $F(k=2)$.

These two periods of the exponential appear only in the first half of the time sequence however. The latter half is zero. The result is that of the following rectangular function

```
* * * * * * * *
```

```
--------------------------------*---*---*---*---*---*---*---*
```

multiplied times an exponential with two periods in the first 8 points.

The rectangular function, in effect, samples the exponential. Alternatively, the exponential modulates the rectangular function. The result is an exponential pulse of width 8 pixels.
By the modulation theorem of Fourier transforms, we should get the spectrum of the rectangular function (a sinc with zero crossings at \( f = 2, 4, 6, 8, 10, 12, 14 \)) shifted to the frequency of the exponential. This is exactly what happens as is seen in Fig. 5.4.10.

It is important to note for this discrete situation that when \( k = 1 \), there is one period of the exponential in the 8 pixel window. But since this is only half the period over which the second FFT is being performed, the effective frequency of this exponential is two. Of course, the amplitude of the frequency component resulting from the FFT of this exponential is half what it would be if the exponential covered the entire area.

Thus, since the window size is half the period of the second FFT, the frequency translated sincs will appear at two times the frequency of the exponential that appears in the window.

For \( k = 0 \) we get a sinc function centered at \( f = 0 \). For \( k = 1 \), we get a sinc function centered at \( f = -2 \). For \( k = 2 \), we get a sinc function centered at \( f = -4 \) (or +4).

Fig 5.4.11 shows a similar case. Again \( m \times v > n \). Here, we examine the same image sequence as previously, but further reduce the size of the window to a 4 x 4 pixel area. Since the window is 4 pixels wide, only 4 values of the spatial frequency variable \( k \) exist (of course). Also, since the window is now 25% of the length of the time sequence (the number of points in the second FFT), the frequency of the exponential in the window will be multiplied by four to yield the actual frequency as seen by the second FFT.

So for \( k = 1 \), an exponential of frequency 1 appears in the 4 pixel window (it must necessarily be a rather coarsely sampled exponential to fit in 4 points). This is an exponential of frequency 4 to the second FFT, and so the sinc function which is the Fourier transform of the 4 pixel wide rectangular function, is translated to \( f = -4 \) (or +4).
The zero crossings on the translated sincs occur at \( f = 4, 8, \) and 12. This is where one would expect the zero crossings for a rectangular function like that pictured at the beginning of Section 5.4.2.2.

Thus, the characteristics of the output spectra were examined for a number of cases in which the window size, the number of frames, the object size, and background were varied. This study demonstrated the interrelationship among the above parameters and noted their effects on the ability to detect velocity. Velocity detection is dependent on the ability to generate detectable velocity impulses. Based on the above analysis, it is seen that as the scene complexity increases, the ability to generate clean impulsive spectra decreases and with it, the ability to detect velocity.

5.5 Reconstruction

5.5.1 MDT2

At first, the basic MDT was applied, without modification to the reconstruction problem. To be specific, we wanted to see what kind of information was preserved through the projection process and to see if this information could be used effectively in teleconferencing applications, where the ability to isolate and reconstruct the moving parts of an image is important. In this implementation, the MDT program could be viewed as a frontend process providing information to downstream processes as to which parts of the image require updating.

A set of orthogonal projections was generated. Each projection was Fourier transformed to get the spatial frequency content for each projection. This sequence of Fourier transformed projections was then Fourier transformed across time to yield a spectrum giving the time frequency content for each spatial frequency.
Next, the d.c. value of the time-frequency spectrum for each spatial frequency $k$, was set to zero. Two inverse Fourier transforms recreated the orthogonal, dc-filtered projections. An outer product of the projections for each image was used to generate the "reconstruction."

Of course, speaking in terms of image reconstruction is something of a fiction since the best one can do is reconstruct the projection arrays (minus d.c.) for each image. The image "reconstruction" one gets by performing the outer product between row and column projection arrays may or may not closely approximate the moving features in the original image sequence depending on the characteristics of the moving object and background.

In the next paragraphs, the problems with this approach to reconstruction are detailed. The problems fall into the categories of restoring object shape and intensity information, separating object generated spectral content from background generated spectral content, and correcting a troublesome d.c. shift in the reconstructed moving object.

Any reconstruction based on the orthogonal projections of the original image will only recreate the shape of the object in the limited case of a point object or a rectangular object.

Even for these objects, the object's intensity information is likely to be obscured by the projection process. For example, Fig. 5.5.1 shows a square object of uniform intensity $A$, with a diagonal band of intensity $B$ moving on a zero uniform background. The projection arrays contain adequate information to recreate the shape of the object, but the intensity information has been transformed by the projection to some uniform intensity $C$. This renders reconstruction of the intensity information impossible.

Additionally, it is clear that the shapes and intensities of irregularly shaped objects and of multiple moving objects will also tend to be obscured by the projection process involved in this reconstruction technique.
In the analysis of Section 5.3, it was shown that the more realistic the image sequence, the greater the intermixing of object generated and background generated spectral content. Consequently, d.c. filtering aimed at separating the spectra of the moving object from that of the background becomes less effective. This is because image sequences with noise and clutter have spectral content at other than $f=0$. Thus, d.c. filtering does not totally eliminate the background. This in turn makes reconstruction, where the moving object is to be separated from the background, more difficult.

Additionally, the above technique yielded poor results due to problems associated with the d.c. bias created by the d.c. filtering process. This d.c. bias results from the elimination of the object generated velocity peak whenever this peak occurs at $f=0$. In all cases examined, a velocity peak will occur at $f=0$ for $k=0$, and there may be other values of $k$ for which the peak occurs at $f=0$. For instance, Fig. 5.4.2 shows velocity peaks occurring at $f=0$ for both $k=0$ and $k=8$.

Eliminating these peaks makes the $k$ spectra for which they occur zero mean. As a consequence, when taking the inverse FFT, these values of $k$ may introduce negative elements to the reconstructed projection arrays.

For example, in Fig. 5.4.1, there is no background and so the peak at $f=0$ is entirely object-generated. Eliminating this peak makes the $k=0$ spectrum zero mean (in this case all values of the spectrum are zero). Taking the inverse FFT of this spectrum yields the $k=0$ values for the projection arrays for each frame. Thus, all the projection arrays will be zero mean and hence will contain enough negative values to compensate for any positive values. This is a problem as the original projection arrays did not have negative values.
In Fig. 5.4.13 the \( k=0 \) spectrum has values at other than \( f=0 \), though the d.c. peak is dominant. Eliminating the \( f=0,k=0 \) peak makes the \( k=0 \) spectrum zero mean. Performing the inverse FFT results in one positive value of \( k \) (the one corresponding to the first frame) and 7 negative values of \( k \) (the ones corresponding to frames 2 through 8). Thus, the first reconstructed projection array will have positive d.c. while the others will have a negative d.c. value.

Though this d.c. bias was a problem, it was found that by applying an ad hoc technique, reconstruction was simulated for certain classes of objects—small rectangular objects to be specific.

In examining the reconstructed projections it was noted that the projections consisted of positive, negative and zero values and that the positive values corresponded to where the object's projection should be. Thus, this heuristic involved zeroing out all nonpositive values in the orthogonal projection array before performing the outer product.

The resulting output was relatively close to the original image for small box objects, though even in these cases, the reconstructed objects faded in intensity towards the middle of the reconstruction sequence.

By using a more elaborate technique, it is sometimes possible to eliminate the d.c. projection components that were contributed by the background and leave those components contributed by the moving object. One can "track" the vector of impulses generated in the spectra such as Fig. 5.4.2 and determine where a velocity generated peak will cross the \( f=0 \) axis. The adjacent velocity peaks could then be used to predict (via linear interpolation) the value of the peak that crosses the \( f=0 \) axis. This interpolated value is then added back to the spectrum at \( f=0 \) after d.c. filtering has taken place.
The effect is that both background and object components are eliminated in the d.c. filtering and an approximation to the object component is added back later.

There are limitations to this approach. For spectra resembling Fig. 5.4.2, estimating the value of those peaks that cross the \( f=0 \) axis is relatively straightforward. If the spectra resemble Fig. 5.4.11, estimating the value of those peaks is much more difficult. In general, interpolating the values of these peaks without a priori knowledge of the spectrum of the moving object is viable only for simple objects moving in a constrained way (constant velocity for example).

So as has been shown, this reconstruction technique is inadequate. This inadequacy is primarily a consequence of the effect of the projection process on reconstruction. The d.c. shifting process is also a problem, but as will be shown, this problem in one form or another is present in all the reconstruction techniques.

5.5.2 MDT3

The next step was to implement a non-projection based reconstruction technique. The process in brief will first be described. The first implementation of this non-projection based reconstruction technique involved taking a full 2D FFT of the image window. For an \( n \times n \) window, this generates \( n^2 \) spatial frequency variables, (versus \( 2 \times n \) for the projection method).

Each spatial frequency is then FFT'ed across time. The d.c. value resulting from this transform is eliminated - thereby removing the time invariant content of each spatial frequency variable. The reconstruction was
obtained by taking two inverse FFT's - the first to return the transformed spatial frequency content to the time domain, and the second to return the spatial frequencies to the pixel domain.

The program implementing the above algorithm was called MDT3 and the details of this implementation follow.

MDT3 operated by windowing into part of an image (the window location was specified manually). The windowed part of the image was copied into a working array where a 2D Fourier transform was performed on it. This transform, of course, retrieves the spatial frequency content of the window and is analogous to the "space to inverse space x \rightarrow k" transform of MDT parlance. Then, the next image in the sequence was read in. Here again, the specified window area was moved into a slot in the working array, (the second slot), and a 2D Fourier transform was performed on this image window. This process was repeated for all the images in the sequence.

The next step was to perform the Fourier transform across time. Thus, for each spatial frequency \( F(k_x, k_y) \) in each image a Fourier transform was performed with the corresponding spatial frequencies in the other frames of the sequence. For instance, if a sequence of length eight was examined, then an eight point FFT would be performed on, say, the \( F(k_x=0, k_y=0) \) frequency value for all eight image windows.

The program was set up to optionally plot the results at this stage of the process. The plots consisted of 3D perspective plots of each frame in the now twice transformed image window sequence. These plots illustrate how the sequence length, window size, and object velocity interact with a given spatial frequency to produce varying effects in the temporal frequency spectra. As described before, spectra were strictly impulsive whenever \( m \times v = n \), where \( m \) is the number of frames in the sequence, \( v \) is the velocity, and \( n \) is the window size.
The information extracted from the plots, in the orthogonal projection version of the transform, is possibly useful from an analytic point of view because it reveals the conditions under which the object spectra will overlap the d.c. background spectra. Since the d.c. "frame" will be eliminated in order to get rid of the time invariant background, the degradation of the object will largely be a function of which of its spatial frequency components had significant d.c. values.

In order to eliminate the time invariant portions of the image sequence, the first frame in the sequence, which at this point contains the time invariant component for each spatial frequency variable, was set to zero. Of course, removing this d.c. frame, in addition to eliminating the stationary image entities, eliminates the d.c. component of any moving object in the window. The result of this will be discussed shortly.

To reconstruct the image sequence, a reverse Fourier transform was performed across the corresponding frequency elements in each frame. This brings us back from $G(k_x,k_y,f)$ space to $F(k_x,k_y,t)$ space. Next, a 2D reverse transform was performed on each image window to return us to $f(x,y)$ space. The image windows were then removed from their working array and placed, one each, into a sequence of 512 squared images for display. This program was run on a sequence created by overlaying one to several simple synthetic moving objects on a frame of the street scene noted previously.

No test was made of multiple moving objects in which one or more of the objects was occluded by a portion of the scene or by another moving object.

The images processed in this way contained all moving objects in their appropriate positions, but eliminating the d.c. had created a zero mean image with many negative pixels (appearing as bright intensities). To shift these
intensity values so that they would all be displayable, the absolute value of the most negative value in an image was added to each pixel. This resulted in the object being reconstructed along with its track. Thus, this nonprojection based reconstruction technique had the following advantages over the the previous reconstruction approach:

Object shape was preserved in the reconstruction. The object's relative intensity was preserved. Multiple moving objects could be reconstructed.

The presence of the object tracks is an artifact of the negative bias created by eliminating the d.c. portion of the images. This negative bias causes positions along the track of the moving objects that are not currently covered by the object itself to appear bright white upon being displayed. This is because these positions exit the final inverse Fourier transform as small negative numbers (which display as bright white).

While an object's relative intensity features were preserved, its absolute intensity could be shifted by the reconstruction process. Also, the resulting track, while possibly useful (or at least inoffensive) in some applications, would be a problem in teleconferencing applications. Consequently, several attempts were made to eliminate this track (leaving only the reconstructed objects), or at least to make the track less objectionable, i.e., make the boundary between the track and the object as clear as possible. These are discussed later. Next, a variant of the full reconstruction method is discussed that yielded the same results as previously, but involved far fewer computations and lent itself to a sometimes effective dc offset correction heuristic.

5.5.3 MDT4: An Improved Technique

In this technique, a single FFT is performed on single pixels across time. No initial 2D space to frequency-space transform is performed. Pixels
that are time invariant, or vary only slightly with time, will have their
energy content concentrated at d.c. Eliminating this d.c., then effectively
zero's out these pixels upon taking the inverse FFT. Thus, the only pixels
with significant energy levels are those that change with time — those pixels
through which the object passes and those subject to large scale noise and
clutter. Other pixels appear visually to be zero intensity, but have been
shown experimentally to have low level intensity in the range 0 to about 10.
Downstream processes must differentiate between these low level pixels and
those pixels that correspond to the object or its track. As for the implementa-
tion details, MDT3 was modified by eliminating the initial 2D transforma-
tions that precede the Fourier transform across time. In this new program,
called MDT4, the windowed portion of each frame in the image sequence was
placed into a working array. Then, a Fourier transform across time (over the
sequence of image frames) was performed on corresponding pixels in each frame.
This transform could be expressed as \( f(x,y,t) \rightarrow F(x,y,f) \).

Note that since we have not at any point performed a Fourier transform
with respect to a spatial variable, this transform does not really bear much
resemblance to the original motion detection transform (the name of the
program not withstanding).

We now have in the first frame of the transformed sequence, the dc
component (time invariant) for each pixel. So, this frame was set to zero in
order to eliminate the time invariant parts of the sequence.

The reconstruction process merely involved taking the inverse Fourier
transform across corresponding pixels in each frame.

Covered next is an explanation of object tracks and a compensation
procedure which effectively eliminates the track for the case where the moving
object is brighter than its background.
5.5.4 **Elimination of the Target Tracks**

As was noted, the first attempt at eliminating the negative d.c. bias that was added to the reconstruction as a result of the d.c. filtering involved computing the most negative value in each frame and adding the absolute value of this number to every nonzero element in the reconstructed frame.

This had the beneficial effect of causing most of the object track in each frame (the portion uncovered by the object itself) to go from a small negative number (displayed bright white) to a small positive number (displayed dark grey). The object itself was shifted upwards in intensity too, thereby increasing the contrast between the object and its track. This process did not, however eliminate the tracks.

Since we're taking the inverse Fourier transform on an array comprised of corresponding pixels in each frame (f(l,1) in frame 1; f(l,1) in frame 2, etc.), a unique d.c. bias correction factor should properly be applied to the output of each of these transforms, rather than adding one correction offset to each frame based on that frame's minimum value.

As an example, look at the time profile of a pixel at say x=1, y=1 over an eight frame sequence. In the first frame, the pixel will be covered by a object of intensity 200. In subsequent frames, the pixel value is zero (the background level).

Initially:

```
Intensity

↑

200

0 - ----x----x----x----x----x----x----x----x-> time

1 2 3 4 5 6 7 8
```
After FFT:

\[
\begin{array}{cccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\
\end{array}
\rightarrow \text{time}
\]

After d.c. filtering:

\[
\begin{array}{cccccccc}
0 & x & x & x & x & x & x & x \\
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\
\end{array}
\rightarrow \text{time}
\]

After inverse FFT:

\[
\begin{array}{cccccccc}
0 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\
-25 & x & x & x & x & x & x & x \\
\end{array}
\rightarrow \text{time}
\]

The values of -25 at pixel (1,1) for frames 2 through 8 are the track artifacts mentioned earlier. To eliminate, we add +25 to each element to yield again:

\[
\begin{array}{cccccccc}
0 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\
-25 & x & x & x & x & x & x & x \\
\end{array}
\rightarrow \text{time}
\]

Programmatically, one goes through a pixel's time/frequency sequence after d.c. filtering. The most negative value in this sequence is found and its absolute value is added to the pixel in question in each frame. If the minimum value is zero, this implies that this pixel was time invariant. We don't bother adding the absolute value of zero to each pixel for obvious reasons.

Note that the d.c. value after the first Fourier transform contains the d.c. content of both the variant and invariant parts of the image window.
being examined. Thus, this process is equivalent to adding the d.c. value of the time varying parts of the image back to the reconstruction.

This technique effectively eliminates object tracks for bright objects moving on a lower intensity background.

This d.c. bias correction process works even when the object is on a nonzero background. For example, take again an object with intensity 200 that covers the pixel in question during frame 1 after which we see only the background intensity, here equal 50.

Pixel time profile:

\[
\begin{array}{cccccccc}
250 \\
50 \\
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8
\end{array}
\]

Decompose into object:

\[
\begin{array}{cccccccc}
200 \\
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8
\end{array}
\]

and background:

\[
\begin{array}{cccccccc}
50 \\
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8
\end{array}
\]

FFT of object: (as shown before)

\[
\begin{array}{cccccccc}
25 \\
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8
\end{array}
\]
FFT of background:

\[
\begin{array}{cccccccc}
\hline
& 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\
\hline
50 & \cdots & \times & \cdots & \times & \cdots & \times & \cdots & \times \\
\hline
\end{array}
\rightarrow \text{time}
\]

Composite:

\[
\begin{array}{cccccccc}
\hline
& 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\
\hline
75 & \cdots & \text{25} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
\hline
\end{array}
\rightarrow \text{time}
\]

Composite after d.c. filtering:

\[
\begin{array}{cccccccc}
\hline
& 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\
\hline
x & \cdots & \text{25} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
\hline
\end{array}
\rightarrow \text{time}
\]

After inverse FFT:

\[
\begin{array}{cccccccc}
\hline
& 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\
\hline
175 & \cdots & 2 & \cdots & 3 & \cdots & 4 & \cdots & 5 & \cdots & 6 & \cdots & 7 & \cdots & 8 \\
\hline
& \cdots & \text{-25} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
\end{array}
\rightarrow \text{time}
\]

This is of course the same result as before.

A problem arises, however, when the object or a portion of the object is darker than the background it occludes. For example, consider a uniform background with intensity of 50. For the pixel we are considering, a moving object of intensity 10 occludes the background on frame 1. The pixel's time profile is:
This can be decomposed into background:

and equivalent object:

These Fourier transform respectively into:

and:

After eliminating the d.c. term and taking the inverse Fourier transform, we get:
Performing the same track elimination on the above as was done previously, results in setting the negative peak to zero and raising the intensity of the track.

This is clearly erroneous. If it is desirable to eliminate the track, then the case of a moving object darker than its background must somehow be handled differently than the case of a bright object on a dark background. Particularly, the two cases must be distinguished from one another. Thus, this technique does not eliminate the track when the moving object is darker than its background. The technique does not adversely alter the relative intensity of the dark object to its track, but neither is the absolute intensity of the dark object restored. A comprehensive procedure for eliminating the track under all conditions has not been found.

5.5.4 MDT4 Results

The output of MDT4 as applied to moving cars on a street scene will now be examined. Fig. 5.5.2 shows frames 1 and 8 of an image window sequence containing a white car turning a corner. Fig. 5.5.3 shows the output of MDT4 when no d.c. correction has been applied. Low intensities have gone negative due to the d.c. shift and thus are displayed bright white. Fig. 5.5.4 shows the reconstruction after application of the d.c. correction technique described in the previous section.

As can be seen, the combination of the MDT and the correction technique has rendered the object without its background. Though not apparent from the photographic illustrations, the background regions that appear to be black (zero intensity) have intensities that range from zero to about ten. This is a consequence of the fact that the background is not totally time invariant due to noise and possibly camera motion. Also visible are curved intensity
bands corresponding to the positions of the windows, tires, and shadow of the car. These bands result from the failure of the dc correction procedure to remove the d.c. bias for dark portions of the scene that move on a lighter background.

Fig. 5.5.5 shows a dark car from the same scene moving in a straight line. Fig. 5.5.6 shows the same scene processed by MDT4 but without d.c. correction. The reconstruction after d.c. correction is shown in Fig. 5.5.7. In this case, both the car and its path appear in the reconstruction. The track was reproduced because the dark car was moving on a lighter background. Though the relative intensity of the car to its track has been preserved, the absolute intensity of both is no longer what it was in the original sequence. This is as expected based on the analysis of the d.c. correction procedure in the preceding section.

As noted, the residual intensity level of background pixels is not zero whenever those pixels are time varying. In fact, if the degree of time variation is great enough, then the background can no longer be considered stationary and the reconstruction will fail. As an example, Fig. 5.5.8 shows the synthetic car-like object corrupted by very high levels of random noise. In Fig. 5.5.9, two frames from the reconstructed sequence illustrate that there has been no elimination of the background due to the variance of the background.

Another effect that presents problems is encountered when a moving object of relatively uniform intensity covers a given group of pixels throughout its movement. Such pixels, though part of the moving object, appear to be relatively time invariant from the point of view of the transform. As a result, these pixels are reduced to near zero intensity by the d.c. filtering reconstruction process. Fig. 5.5.10 shows a uniform intensity rectangular
object moving to the right over a sequence of eight frames, (frames one and eight are shown). The reconstruction of these two frames reveals that a gap was created corresponding to the pixels that remained covered by the moving rectangle throughout the eight frames examined by the MDT.

Because of the possibility of encountering this effect, this reconstruction technique could not be counted on to provide the full reconstruction of moving objects in all situations. The edges of the moving object are reconstructed in all cases. This is information that could possibly be useful in some applications.

5.6 Window Selection

As noted previously, the window to be examined by the MDT was selected manually by the user of the program rather than automatically by the program itself.

Window selection is an issue as one must have some a priori information about where to look for movement in a picture scene in order to hold down the computationally complexity to a reasonable level. Of course, what is reasonable depends on the application. Real-time teleconferencing applications will have much more severe constraints on both speed and memory requirements than would an off-line image processing application.

In a teleconferencing environment, restricting processing to small windows likely to contain motion is imperative for the full-reconstruction MDT variants. For an \( n \times n \) window and a sequence length of \( m \), MDT4 has \( O(n^2m\log m) \) time complexity. Windowing is required not only for reasons of time complexity but also for reasons of image storage requirements the full reconstruction versions of the MDT require that space enough for all pixels within the window for all frames be maintained throughout processing.
With the velocity detection version of the MDT, memory considerations have little relevance to the windowing issue. In this form of the MDT, only two projection arrays per image need be stored. As this MDT has $O(mn^2)$ projection operations and $O(mn \log n)$ FFT operations, windowing to hold down time complexity is still important. Also, as was noted previously, multiple moving objects within a window may produce spectra in which it is impossible to distinguish the velocity components for each of the moving objects. For this reason, in track acquisition applications, windows small enough to isolate moving objects would be preferred.

Having stated the case for windowing, there still remains the issue of automatic window selection. This topic was not explored in depth, but some scheme based on frame-to-frame pixel differencing (possibly thresholded) might prove to be a viable approach.

6.0 CONCLUSIONS

The Motion Detection Transform has been shown to be a robust technique for determining the presence of moving objects in an image window and for determining the velocity of those objects. It performs well for unresolved point objects as well as large objects of varying shapes, sizes and intensities, and it is unaffected by high levels of random noise. The principal difficulties for motion detection arise when the velocity of the moving object is not relatively constant, and when there are multiple moving objects within the image window.

The Motion Detection Transform as applied to the reconstruction problem is more of a qualified success. It does reproduce the moving objects within a scene with total faithfulness in a number of circumstances. However, the presence of many cases in which the reconstruction either fails to recreate
the entire moving object or introduces unwanted intensity shifts in a nonuniform manner renders the technique too unreliable to be used as a stand-alone method for isolating the moving objects in a scene. It is possible that the reconstruction implementation of the MDT could effectively complement other techniques in a teleconferencing environment.
FIG. 5.2.1 THE PROJECTION PROCESS

FIG. 5.2.2 THE SECOND SUMMATION
FIG. 5.3.1. (a)
FIG. 5.3.1. (b)
FIG. 5.3.2
FIG. 5.3.3 STREET SCENE BACKGROUND
FIG. 5.3.4 (a)
FIG. 5.3.4 (c)
FIG. 5.3.6

Frequency

$X$ Magnitude in dB

0 1 2 3 4 5 6 7

41.08 48.64 52.44 56.24 60.04 63.84 66.64 68.4
FIG. 5.3.8

Frequency

X Magnitude in dB

0 1 2 3 4 5 6 7

0 10 20 30 40 50 60 70
FIG. 5.3.9

X Magnitude in dB

frequency
FIG. 5.3.10
FIG. 5.3.12
FIG. 5.3.14

Frequency

X Magnitude in dB

-39.39
-45.22
-51.05
-56.88
-62.71
-68.54
-74.37
-80.10
-85.83
-91.56
-97.29

0 1 2 3 4 5 6 7

FIG. 5.3.14
FIG. 5.3.15 (g)
FIG. 5.3.15 (b)
Sequence: pntobj, 1 pel obj; vel = 1; frames = 8; window = 8 x 8

FIG. 5.4.1
Sequence: ptvv2obj; 1 pel obj; vel = 2; frames = 8; window = 16 x 16
Sequence: pntbg; 1 pel obj; v = 1; frames = 8; window = 8x8; bg = uniform

FIG. 5.4.3 (a)
FIG. 5.4.3 (b)
Sequence: 
put a; 1 pel obj.; v = 1.; frames = 8; window = B6B; bg = uniform

Freq-

Magnitude dB
0 17.14 34.98 51.57 88.17

B L 2 3 4 5 6 7 8
FIG. 5.4.4 (a)
Sequence: points; I act obj; v = I; frames = 8; window = 8x8; bg = neg

magnitude in dB
80
60
40
20
10
Sequence: pmts; 1 pel obj; v = 1; frames = 8; window = 8x8; bg = neg001

FIG. 5.4.4 (b)
FIG. 5.4.5

Sequence: protocol: T pel 0.2s; vel = T; tramee = 8; window = 16 × 16

Fig-a-t

30

0.31

magnitude

0.32

1.22

0.92

0.81

0.25

X-Bar

Z-Gear

0.88
FIG 5.4.7

Sequence: 49; y pel oil; vel = 1; frames = 8, window = 16; 16 × 16

freq t

magnitude in dB

10.08

0.98

0.45

0.98

1.17

2.37

14.99

3.15

2.95

4.78

9.45

14.99
Sequence: bigsq; 8 pel obj; vel = 1; frames = 16; window = 32 x 32

FIG. 5.4.8
Sequence: pntobj; 1 pel obj; vel = 1; frames = 16; window = 8 x 8;

FIG. 5.4.10
Sequence: pntobj; 1 pel obj; v=1; frames=16; window = 4x4; 0 bg
FIG. 5.4.11
FIG. 5.5.1 THE LOSS OF INTENSITY INFORMATION DUE TO THE PROJECTION PROCESS.
FIG. 5.5.2
FIG. 5.5.7
FIG. 5.10.10