Complex filters and its applications

Sharat Chikkerur

December 12, 2006

Abstract

We present a detailed analysis along with matlab implementation of Bigun et al.'s "Recognition by Symmetry Derivatives and the Generalized Structure Tensor", [2]. In this paper, the authors present gradient domain complex valued filters that can be used for feature extraction, matching and pattern recognition. The filters, also called as symmetry derivatives possess several advantages such as rotation, scale and translation invariance properties. They also have important theoretical properties such as closure under convolution. They are useful in detecting linear, parabolic, triangular and several other rotational symmetries. We present a detailed analysis of these families of filters along with several applications.

1 Introduction

Symmetric patterns are used to perform image registration and alignment in several applications such as camera calibration, crash tests, alignment of fingerprints, photmetric stereo etc. Conventional methods to detect these patterns make use of a template that represents the ideal pattern. The patterns are located by performing an exhaustive search (using spatial or Fourier domain correlation) [3] over the image for the presence of the template. However, this method is not invariant to rotation and scaling. Detecting the pattern under rotation using this method would require the use of a separate template for each rotation and scale. In this report, we present Bigun et al. [2], work on symmetry detectors based on complex derivatives of Gaussian filters. The proposed filters have interesting properties such as

- The filters operate in the gradient domain instead of in the intensity domain and are therefore contrast invariant.

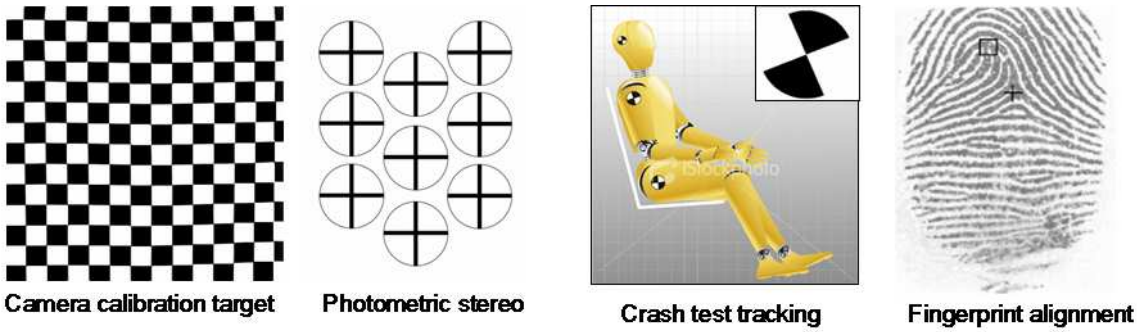

Figure 1: Several applications requiring accurate detection of symmetric pattern. (Left to right): Camera calibration, photometric stereo, crash test analysis, fingerprint alignment

- The filters are also rotation invariant. A single filter detects the pattern at all orientation. The orientation of the pattern can be determined from the phase of the filter response.
- They are closed under convolution. This means that higher order filters can be composed by successive convolution from lower order (but more efficient) filters.
- The filters are designed to detect linear symmetry in the cartesian co-ordinate system. However, by a simple co-ordinate transformation, they are capable of detecting n-fold rotational symmetries with the same computational complexity.
- They are insensitive to the actual pattern that exhibits linear symmetry.
- They can be efficiently implemented using separable convolutions.

2 Detecting Linear Symmetry: Structure Tensor

Let the image be represented as a two dimensional function $f(x, y)$. This image is symmetric if $f(x, y)$ can be written as $f = g(k^T r)$. Here k represents the direction vectors,perpendicular to which, the pixel values are constant. In other words, the figure is said to be linearly symmetric if the isophotes form a straight line. Figure

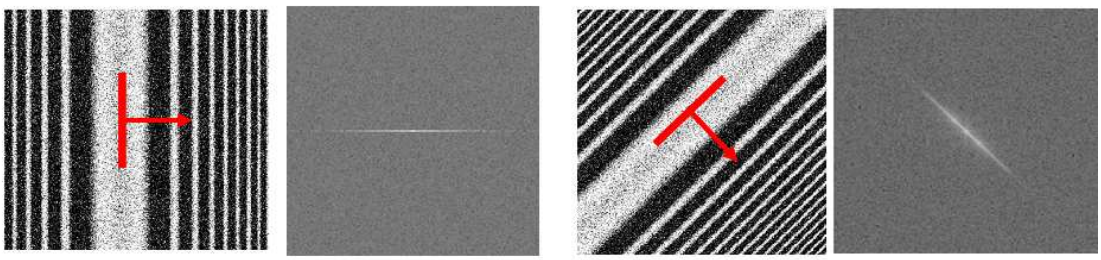

Figure 2: Examples of images exhibiting linear symmetry and their fourier transforms.

3 shows examples of images exhibiting linear symmetry and their isophotes. It can be seen that in each case, the power in the fourier spectrum is concentrated along a line. The degree of symmetry can be determined by how closely a line can be fit on the spectrum. The orientation of this line will give us the orientation of the symmetry axis. This problem is identical to finding the axis of minimum interia. The axes of interia can be obtained by considering the second order moment matrix of the Fourier spectrum. This is called the **structure tensor** [**?**, 1].

$$S(f) = \begin{pmatrix} \int\int \omega_x^2 |F|^2 & \int\int \omega_x \omega_y |F|^2 \\ \int\int \omega_x \omega_y |F|^2 & \int\int \omega_y^2 |F|^2 \end{pmatrix} \tag{1}$$

Fortunately, this can be computed directly in the gradient domain without resorting to computing the Fourier transform. Utilizing the fact that differentiation in time/space domain is equivalent to multiplying by the spectral frequency ($\frac{d}{dx} \rightarrow \omega \mathbf{F}(\omega)$), we can rewrite it as,

$$S(f) = \begin{pmatrix} \int\int (D_x f)^2 & \int\int (D_x f)(D_y f) \\ \int\int (D_x f)(D_y f) & \int\int (D_y f)^2 \end{pmatrix} \tag{2}$$

The eigen vectors corresponding to λ_{min} and λ_{max} represent the axis of least and maximum moment of inertia respectively. The value of the moment of inertia, as well as its orientation can be expressed in terms of particular second moments,

$$I_{20}(f) = \int\int ((D_x + iD_y)f)^2 dxdy \tag{3}$$

$$I_{11}(f) = \int\int |(D_x + iD_y)f|^2 dxdy \tag{4}$$

In order to extend linear symmetry to other forms, we define a new harmonic co-ordinate axes (has unit jacobian), ζ, η, such that straight lines in this co-ordinate

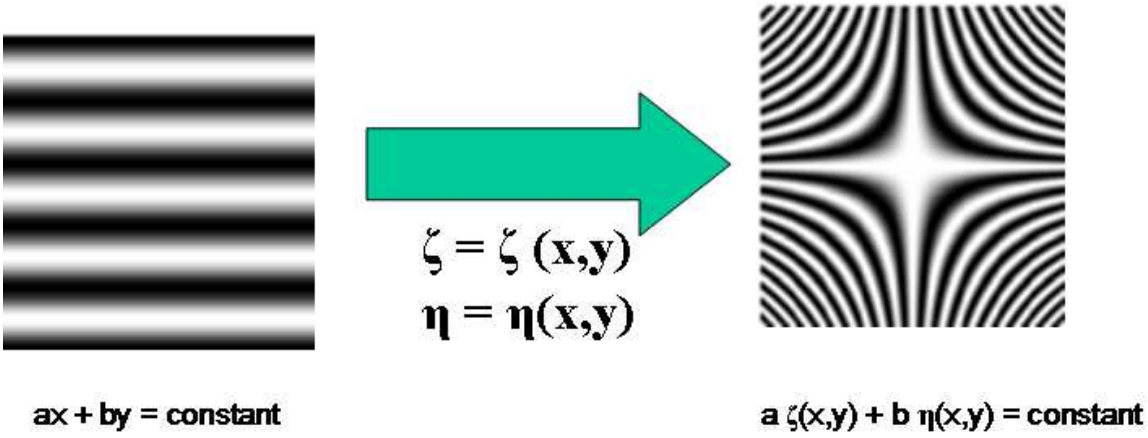

Figure 3: Shows as instance of co-ordinate transformation, where $g(z) = z^2$. This results in $\zeta = x^2 - y^2, \eta = 2xy$

system, correspond to curved lines in (x, y). In order to detect linear symmetry in (ζ, η), we rely upon the I_{20}, I_{11} in the new co-ordinates [**?**]. However, since (ζ, η) are chosen to be harmonic transformations with unit jacobian, the resulting expression after change of variables will be identical to equations 4. In general, the harmonic co-ordinates for the transformatin is generated using, For $n < 0$, the detectors are replaces by their complex conjugates. The filter Γ^{n,σ_2^2} detects linearly symmetric patterns in the harmonic co-ordinates ($a\zeta(x, y) + b\eta(x, y) = \text{constant}$)that are generated using

$$\zeta(x, y) = Re\{g(z)\} \quad (5)$$
$$\eta(x, y) = Im\{g(z)\} \quad (6)$$

2.1 Implementation: Symmetry Derivatives

The paper proves that linear symmetry in cartesian co-ordinates or in transformed harmonic co-ordinates related through the mapping function $g(z) = z^{n/2+1}$, can be detected using a parametric family of filters, Γ^{p,σ^2}. In order to understand these filters, we need to first understand symmetry derivatives. Symmetry derivatives and its conjugates of order n are defined as the differential operators

$$(D_x + iD_y)^n = \left(\frac{\partial}{\partial x} + i\frac{\partial}{\partial y}\right)^n \quad (7)$$

$$(D_x - iD_y)^n = \left(\frac{\partial}{\partial x} - i\frac{\partial}{\partial y}\right)^n \tag{8}$$

$$\tag{9}$$

The operator $\Gamma(p, \sigma^2)$, denotes the application of the symmetry derivatives to gaussian kernels of variance σ^2. In its original form, computing $\Gamma(p, \sigma^2)$, would require us to compute the complex derivative p times. This will lead to finite precision and sampling effects, when the filter size is small. However, it can be shown that for any polynomial of the symmetry derivative operator,

$$Q(D_x + iD_y)\Gamma^{0,\sigma^2} = Q\left(-\frac{1}{\sigma^2}(x + iy)\right)\Gamma^{0,\sigma^2}(x, y) \tag{10}$$

We use this generating function instead of repeated differentiation to implement the symmetry derivatives. The two second order moments I_{20}, I_{11} are obtained using,

$$I_{20} = C_n\Gamma^{n,\sigma_2^2} * (\Gamma^{1,\sigma_1^2)} * f_k)^2 \tag{11}$$

$$I_{11} = C_n|\Gamma^{n,\sigma_2^2}| * |\Gamma^{1,\sigma_1^2)} * f_k|^2 \tag{12}$$

Figure 4 shows a whole family of such curves. The columns represent the patterns generated for $n = -2, -1, 0, 1, 2$ respectively. The rows indicates the patterns generated for different values of the angular parameter θ, such that $\theta = \tan^{-1}(b/a)$. It is to be noted that while, changing the value of θ, rotates the patterns inplane, for $n = -2$, it changes the curvature.

3 Results

In this section, we present some results of applying the symmetry filters on images. Figure 5 shows the result of applying Γ^{2,σ^2} filter on a checkerboard image. Each junction in the checker board has 4-fold symmetry. The resulting filter response shows that the signal to noise ration obtained by using symmetry filters is very high. While the magnitude of the response, indicates the position of the landmark point, the phase angle of the response indicates the orientation of the pattern detected. It has to be noted that in case multiple target locations are present, the problem of detecting all of them becomes a classical detecting task, requiring us to select proper threshold.

Figure 7 shows the application of symmetry derivative filters to track the markers on a crash test dummy. We utilize a second order detector $\Gamma^{2,3}$, in order to detect the four fold symmetric pattern. It can be seen that the all three markers have been highlighted in the response. In particular, it can be seen that the pattern is scale invariant and the smaller marker at the top has been detected.

Figure 8 shows the application of triangular symmetry derivatives $Gamma^{1,15}$ to detect landmark points in the fingerprint. Most fingerprints have two land makr points. The core denotes the point of highest curvature and has parabolic symmetry. The delta indicates the inflection point exhibiting triangular symmetry. These landmark points are used to align two fingerprints. The orientation flow is also shown to highlight the fact that the symmetry derivative operates in gradient domain and not in the intensity domain. The figure also highlights the importance of choosing the size of the symmetry derivative filters. The first response map was generated using $\Gamma^{1,3}$ and the second using $\Gamma^{1,15}$. It can be seen that choosing smaller filter size generates a lot of false positives.

4 Conclusion

We presented a detailed analysis of symmetry derivatives of gaussians and its applications to detect symmetric patterns in images. We discussed and explained how the structure tensors can be used to detect linearly symmetric patterns in an image. We also discussed *generalized* structure tensor, that is capable of detecting more complex forms of symmetry using co-ordinate transformation. We presented implementation details accompanied with matlab code to implement these filters. We presented several applications of these filters such as location of camera calibration targets, tracking markers on crash test dummies and landmark location in fingerprints. The filters exhibit a wide range of rotatin and scale invariance. However, size selection and determining proper threshold still remains a problem.

References

[1] J. Bigun. Recognition of local symmetries in gray valued images by harmonic functions. In *International Conference on Pattern Recognition*, 1988.

[2] J. Bigun. *Vision with Direction.* Springer, 2006.

[3] J. Bigun, T. Bigun, and K. Nilsson. Recognition by symmetry derivatives and the generalized structure tensor. *IEEE PAMI*, 2004.

[4] J. Bigun and G. H. Granlund. Optimal orientation detection of linear symmetry. In *ICCV*, pages 433–438, 1987.

[5] Gonzalez, Woods, and Eddins. *Digital Image Processing.* Prentice Hall, 2004.

5 Appendix : Code

5.1 Displaying parametric family of filters

```
%-------------------------------------------------------------------------
%
%sharat@mit.edu
%
%-------------------------------------------------------------------------
clear all;
[x,y] = meshgrid(-16:0.1:16,-16:0.1:16);
z     = complex(x,y);
idx   = 1;
for n = -2:2
    if(n==-2)
         g_z = log(z+eps);
    else
         g_z = z.^(n/2+1)/(n/2+1);
    end;
    u  =  real(g_z);
    v  =  imag(g_z);

    %figure(1); axis tight; set(gca,'nextplot','replacechildren');
    for th = 0:pi/8:pi
        s  = cos(2^(-n)*(cos(th)*u+sin(th)*v));
        subplot(5,9,idx);imagesc(s);colormap('gray');axis off;
        idx= idx+1;
    end;
end;
```

5.2 Computing complex image gradient

```
%-------------------------------------------------------------
%
%
%sharat@mit.edu
%-------------------------------------------------------------
function g= complex_image_gradient(img,n,sz)
    if(nargin<3)
        n = 5;
```

```
        sz= 1;
    end;
    img     = im2double(img);
    img     = imfilter(img,fspecial('gaussian',n,sz));
    [px,py] = gradient(img);
    g       = complex(px,py);
%end function
```

5.3 Generating filter

```
%-------------------------------------------------------------------
%
%
%sharat@mit.edu
%-------------------------------------------------------------------
function g  = symmetry_derivative(n,sz)
    [x,y]   = meshgrid(-2*sz:0.1:2*sz,-2*sz:0.1:2*sz);
    g       = exp(-(x.^2+y.^2)/(2*sz*sz));
    z       = complex(x,y)*(-1/sz*sz);
    g       = g.*(z.^abs(n));
    g       = imresize(g,[4*sz 4*sz],'bilinear');
    if(n<0)
        g   = conj(g);
    end;
%end function
```

5.4 Computing filter responses

```
%-------------------------------------------------------------------
%
%
%sharat@mit.edu
%-------------------------------------------------------------------
function [i20,i11] = complex_response(g,s)
    i20 = imfilter(g.^2,s,'same','symmetric');
    i11 = imfilter(abs(g.^2),abs(s),'same','symmetric');
%end function
```

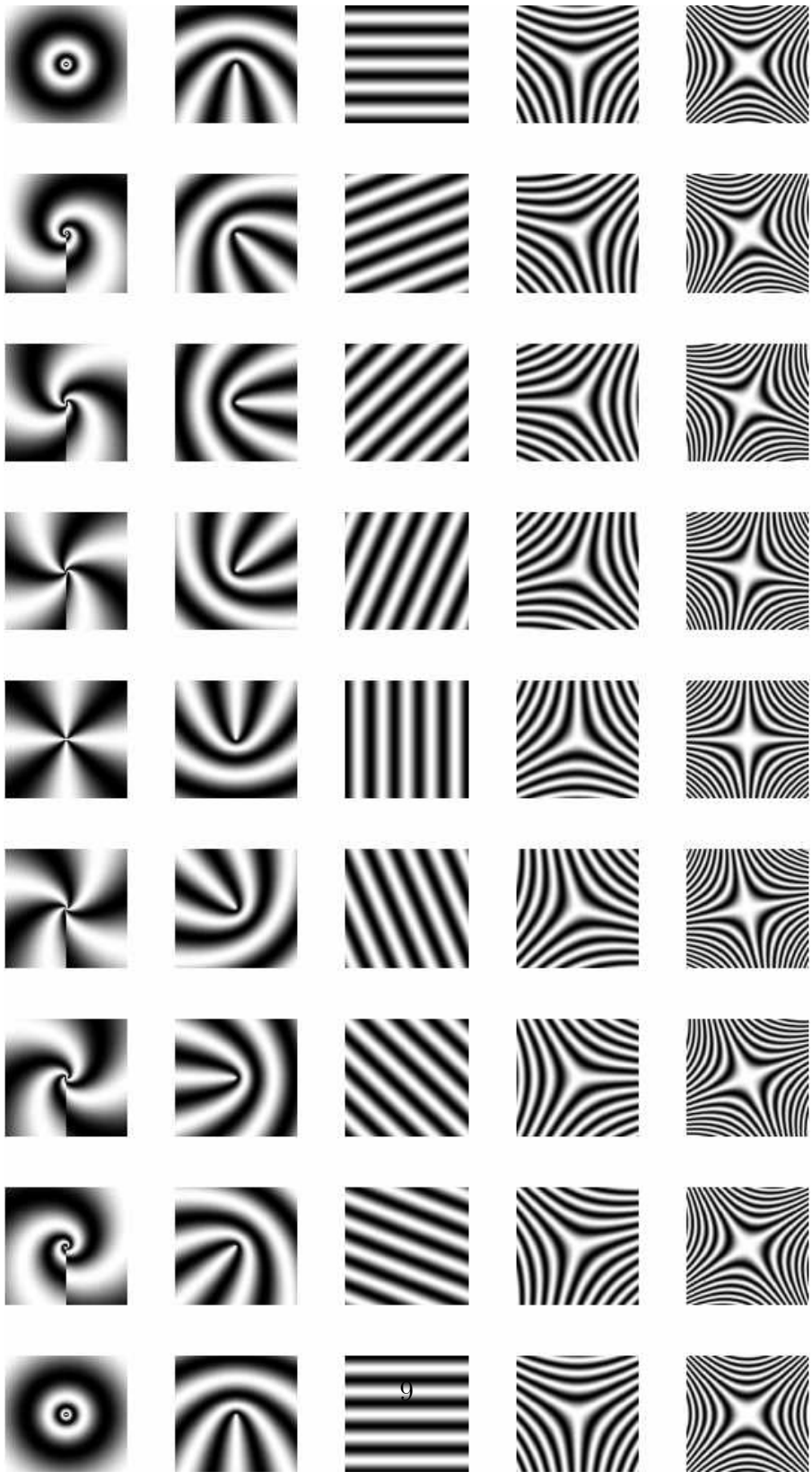

Figure 4:

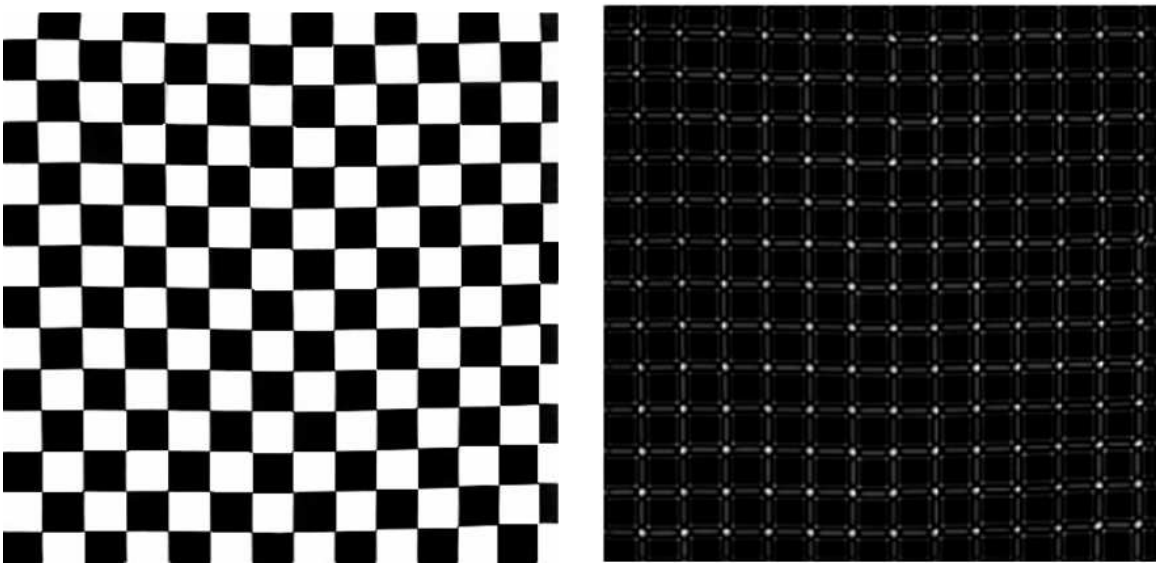

Figure 5: Results of applying second order symmetry derivative on checkerboard/camera calibration pattern.

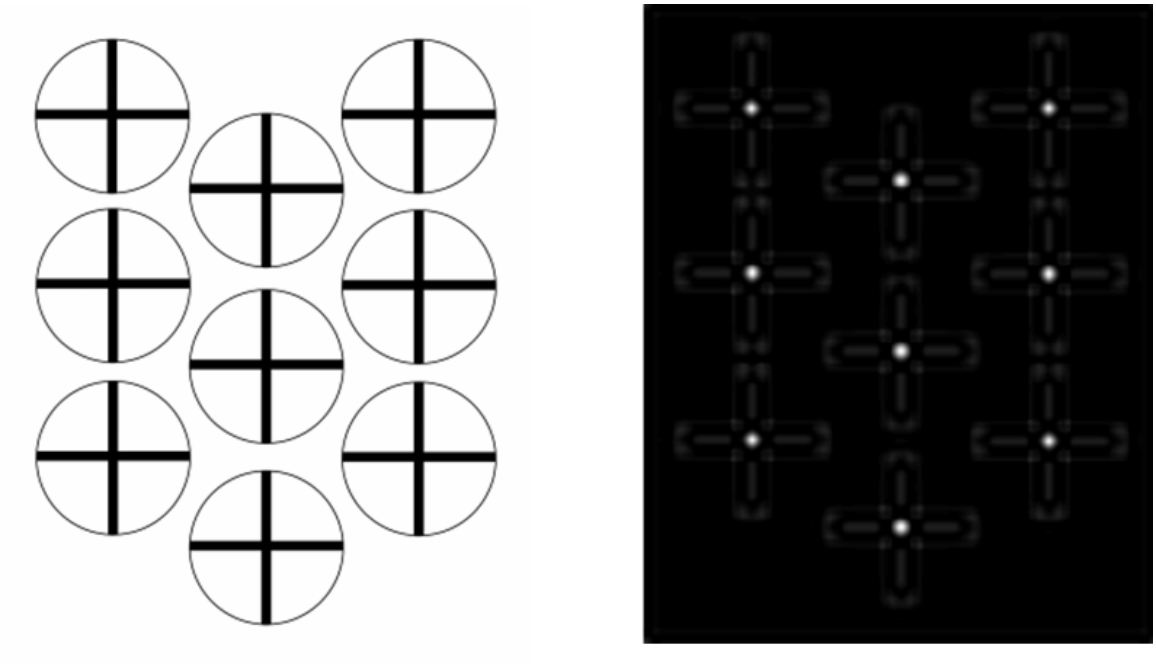

Figure 6: Results of applying second order symmetry derivatives on cross hair pattern

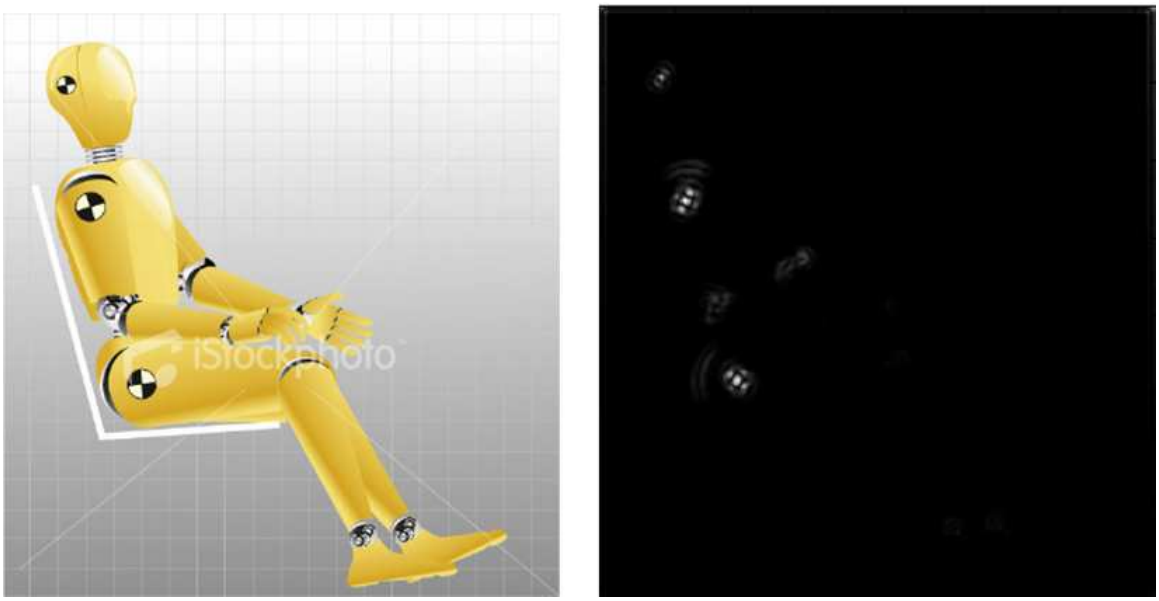

Figure 7: Application of symmetry derivative filters for tracking markers on a crash test dummy

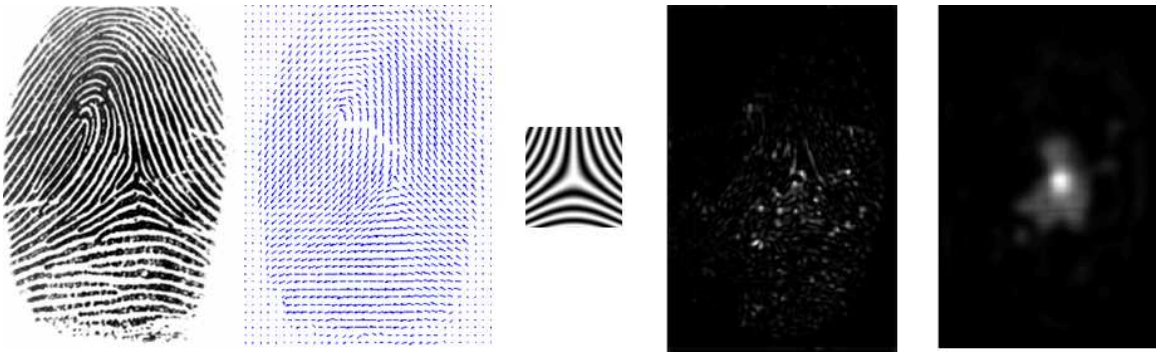

Figure 8: Application of third order symmetry derivative filters to detect land mark points in fingerprints