10-2002

An IDL/ENVI Implementation of the FFT Based Algorithm for Automatic Image Registration

Hongjie Xie

Nigel Hicks

George R. Keller
The University of Texas at El Paso, keller@utep.edu

Haitao Huang

Vladik Kreinovich
The University of Texas at El Paso, vladik@utep.edu

Comments:
UTEP-CS-02-25.
Published in *Computers and Geosciences*, 2003, Vol. 29, No. 8, pp. 1045-1055.

An IDL/ENVI implementation of the FFT Based Algorithm for Automatic Image Registration

Hongjie Xie, Nigel Hicks, G. Randy Keller

Pan American Center for Earth and Environmental Studies (PACES) and

Department of Geological Sciences

University of Texas at El Paso, Texas 79968, USA

Haitao Huang, Vladik Kreinovich

Pan American Center for Earth and Environmental Studies (PACES) and

Department of Computer Sciences

University of Texas at El Paso, Texas 79968, USA

[1] Hongjie Xie xie@geo.utep.edu, fax: 915-747-5073, tel: 915-747-5599

[2] Nigel Hicks hicks@geo.utep.edu, fax: 915-747-5073, tel: 915-747-5599

[3] G. Randy Keller keller@geo.utep.edu, fax: 915-747-5073, tel: 915-747-5083

[4] Haitao Huang hhuang@utep.edu, fax: 915-747-5030, tel: 915-532-5008

[5] Vladik Kreinovich vladik@cs.utep.edu, fax: 915-747-5030, tel: 915-747-6951

Abstract

Georeferencing images is a laborious process so schemes for automating this process have been under investigation for some time. Among the most promising automatic registration algorithms are those based on the Fast Fourier Transform (FFT). The displacement between two given images can be determined by computing the ratio $\mathbf{F_1 * conj(F_2)/|F_1 * F_2|}$, and then applying the inverse Fourier transform. The result is an impulse-like function, which is approximately zero everywhere except at the displacement that is necessary to optimally register the images. Converting from rectangular coordinates to log-polar coordinates, shifts representing rotation and scaling can also determined to complete the georectification process. Our FFT-based algorithm has been successfully implemented in IDL (Interactive Data Language) and added as two user functions to an image processing software package - ENVI (ENvironment for Visualizing Images) interface. ENVI handles all pre-processing and post-processing work such as input, output, display, filter, analysis, and file management. To test this implementation, several dozen tests were conducted on both simulated and "real world" images. The results of these tests show advantages and limitations of this algorithm. In particular, our tests show that the accuracy of the resulting registration is quite good compared to current manual methods.

Key works: Automatic image registration; FFT algorithm; image processing; IDL/ENVI

1. Introduction

Image registration can be formally defined as the transformation of one image with respect to another so that the properties of any resolution element of the object being imaged is addressable by the same coordinate pair in either one of the images (Cideciyan et al., 1992).

Most of image registration approaches fall into local or global methods (Cideciyan et al., 1992). Local methods are referred to as rubber-sheeting or the control-points method. Global methods involve finding a single transformation imposed on the whole image and are also referred to as automatic registration methods. In all the image processing software packages evaluated during this study, registration was based on local methods that required manual selection of ground control points. The selection of these points is tedious work, requiring hours of effort by a skilled professional interpreter. Although several algorithms for automatic registration have been proposed and successfully tested (e.g., Reddy and Chatterji,1996), none of them is currently incorporated into any commercial image processing software package such as ENVI, ERDAS IMAGINE, PCI and ER Mapper. These algorithms were divided into the following classes by Reddy and Chatterji (1996): 1. algorithms that directly use image pixel values; 2. algorithms that operate in the frequency domain (e.g., the Fast Fourier Transform (FFT) based approach used here); 3. algorithms that use low-level features such as edges and corners; and 4. algorithms that use high-level features such as identified objects, or relations between features.

Scientists have investigated FFT-based approaches for image registration for many years. For example, Kuglin and Hines (1975) developed a method called phase correction by using certain properties of the Fourier transform. DeCastro and Morandi (1987) discovered a way to use the Fourier transform to determine rotation as well as shift. Cideciyan et al. (1992) determine the phase-correction (difference) function for each discrete rotation value and chose the parameter set resulting in the highest phase correction. Reddy and Chatterji (1996) improved on the algorithm of DeCastro and Morandi (1987) by greatly reducing the number of transformations needed. Additional theoretical foundations for this FFT-based algorithm of Reddy and Chatterji (1996) were provided by Sierra (2000). In this paper, we implement the

Reddy and Chatterji (1996) algorithm using IDL and incorporate it into the ENVI interface. A brief description of these results first appeared in Xie et al. (2000).

2. Description of the FFT algorithm

The FFT-based automatic registration method relies on the Fourier shift theorem, which guarantees that the phase of a specially defined "ratio" is equal to the phase difference between the images. It is known that if two images $\mathbf{I_1}$ and $\mathbf{I_2}$ differ only by a shift, $\mathbf{(x_0, y_0)}$, [i.e., $\mathbf{I_2(x, y) = I_1(x- x_0, y- y_0)}$], then their Fourier transforms are related by the formula:

$$F_2(\xi, \eta) = e^{-j*2\pi*(\xi \cdot x_0 + \eta \cdot y_0)} * F_1(\xi, \eta) \qquad \textbf{(1)}$$

The "ratio" of two images $\mathbf{I_1}$ and $\mathbf{I_2}$ is defined as:

$$R = \frac{F_1(\xi, \eta) * conj\ (F_2(\xi, \eta))}{abs\ (F_1(\xi, \eta)) * abs\ (F_2(\xi, \eta))} \qquad \textbf{(2)}$$

where ***conj*** is the complex conjugate, and ***abs*** is absolute value.

By taking the inverse Fourier transform of R, we see that the resulting function is approximately zero everywhere except for a small neighborhood around a single point. This single point is where the absolute value of the inverse Fourier transfer of R attains its maximum value. It can be shown that the location of this point is exactly the displacement $\mathbf{(x_0, y_0)}$ needed to optimally register the images (Reddy and Chatterji, 1996).

If the two images differ by shift, rotation and scaling, then converting *abs* $\mathbf{(F(\xi, \eta))}$ from rectangular coordinates **(x, y)** to log-polar coordinates $\mathbf{(\log(\rho), \theta)}$ (Fig. 1) makes it possible to represent both rotation and scaling as shifts. However, computing $\mathbf{(\log(\rho), \theta)}$ from the original rectangular grid leads to points that are not located exactly at points in the original grid. Thus, interpolation is needed to find a value of *abs* $\mathbf{(F(\xi, \eta))}$ on the desired grid. A

bilinear interpolation is used in this implementation. Let **(x, y)** be a point related to the desired grid point **(log (ρ), θ)**,

$$x = e^{\log(\rho)} * \cos(\theta) \qquad y = e^{\log(\rho)} * \sin(\theta) \tag{3}$$

To find the new value **M(x, y)** using this interpolation**,** take the intensities $\mathbf{M}_{jk}$, $\mathbf{M}_{j+1,\,k}$, $\mathbf{M}_{j,\,k+1}$, and $\mathbf{M}_{j+1,K+1}$ of four original grid points **(j, k), (j+1, k), (j, k+1),** and **(j+1, k+1)** surrounding **(x, y).** Then interpolate **M(x, y)** as follows:

$$\mathbf{M(x, y) = M_{jk} * (1-t) * (1-u) + M_{j+1,\,k} * t * (1-u) + M_{j,\,k+1} * (1-t) * u + M_{j+1,K+1} * t * u} \tag{4}$$

where **t** is a fractional part of **x,** and **u** is a fractional part of **y**.

The final algorithm for determining rotation, scaling, and shift is:

1. Apply FFT to images I_1 and I_2 → $\mathbf{F_1(\xi, \eta)}$ and $\mathbf{F_2(\xi, \eta)}$;
2. Compute the absolute values of $\mathbf{F_1(\xi, \eta)}$ and $\mathbf{F_2(\xi, \eta)}$;
3. Apply a high pass filter to the absolute values to remove low frequency noise;
4. Transform the resulting values from rectangular coordinates to log-polar coordinates;
5. Apply the FFT to log-polar images I_1 and I_2 → $\mathbf{Flp_1(\xi, \eta)}$ and $\mathbf{Flp_2(\xi, \eta)}$;
6. Compute the ratio $\mathbf{R_1}$ of $\mathbf{Flp_1(\xi, \eta)}$ and $\mathbf{Flp_2(\xi, \eta)}$ using equation **(2)**;
7. Compute the inverse FFT $\mathbf{IR_1}$ of the ratio $\mathbf{R_1}$;
8. Find the location $(\log(\rho_0), \theta_0)$ of the maximum of **abs**($\mathbf{IR_1}$) and obtain the values of scale ($\rho_0 = base^{\log(\rho_0)}$), and rotation angle ($\theta_0$);
9. Construct a new image, I_3, by applying reverse rotation and scaling to I_2 or I_1;
10. Apply FFT to images I_1 and I_3 (or I_2 and I_3) depending on whether I_1 or I_2 is chosen as the base image.
11. Compute the ratio $\mathbf{R_2}$ using equation (2);

12. Take inverse FFT $\mathbf{IR_2}$ of $\mathbf{R}_2$;

13. Obtain the values (x_0, y_0) of the shift from the location of the maximum of $\mathbf{abs}(\mathbf{IR_2})$.

The result of this process is the values of the scale, rotation and shift parameters needed to register the two images.

3. Main idea for implementing the algorithm using IDL/ENVI

The goal was to implement this algorithm into a commonly used image processing software package such as IDL/ENVI (Research Systems, Inc.). IDL (Interactive Data Language) is a powerful, array-based, structured programming language that offers integrated image processing and display capabilities with an easy-to-use GUI toolkit (IDL, 1999). ENVI (ENvironment for Visualizing Images), which is written in IDL, is one of the most widely used image processing and analysis packages; it is also actively used for integrated vector GIS analysis (ENVI, 1998). By employing ENVI's user function capability, algorithms written in IDL can be integrated into the ENVI menus (interface). ENVI provides a library of procedures and programming tools for user functions to handle input, output, plotting, reports, and file management. Thus, ENVI handles all of pre-processing, post-processing steps such as input, output, display, analysis, and file management after implementing a user function.

4. Implementation issues

One of the biggest advantages of using IDL and ENVI is that they have numeric built-in functions and procedures available, such as FFT, inverse FFT, rotation, bilinear and cubic interpolations. Thus, there was no need to write source code for these functions. Because of this, programs in IDL and ENVI appear very simple and straightforward. For example, to

obtain the two input images from the ENVI's Available Bands List (the list of opened images), only the following two statements are needed:

im1 = ENVI_GET_DATA(fid = state.im1fid, dims = state.im1dims, pos = state.im1pos)

im2 = ENVI_GET_DATA(fid = state.im2fid, dims = state.im2dims, pos = state.im2pos)

Here **im1** is the first (base) image and **im2** is the second image, which is rotated, scaled, and shifted image relative to the first image. ENVI_GET_DATA is an ENVI function that returns any image band that is already opened (i.e., it appears in the Available Bands List of ENVI). In this function, 'fid' specifies the unique ID (identification number) of a file that stores the image, 'dims' specifies the spatial dimensions of the image, and 'pos' specifies the band position (is a long integer with a value from zero to number-of-bands minus one) of the image.

The main part of the FFT based algorithm is implemented and described as follows. Some key techniques and ideas have been emphasized. Text following a ' ; ' are comments. Steps 1-13 are associated with the previous algorithm description:

Steps 1-2. Compute the forward FFT and take the absolute values of the FFT results:

fft_im1 = FFT(im1,-1) ; FFT is a IDL function, -1 means the forward FFT

fft_im2 = FFT(im2,-1)

absF_im1 = abs(fft_im1) ; abs is a IDL function for obtaining the absolute values

absF_im2 = abs(fft_im2)

Step 3. High-pass filter

Step 4. Transfer from rectangular to log-polar coordinates:

*DTheta = 1.0 * pi / RRows* ; the steps of the angle, RRows is the number of rows

b = 10 ^ (alog10(RCols) / RCols) ; b is the base for the log-polar conversion. In order to attain high accuracy (Reddy and Chatterji, 1996), we must require that the polar plane have the same number of rows as the rectangular plane.

for i = 0.0, RRows-1.0 do begin

*Theta = i * dTheta*

for j = 0.0, RCols-1.0 do begin ; RCols is the number of columns

r = b ^ j - 1 ; r is the radius corresponding to log-polar coordinate

*x = r * cos(Theta) + RCols / 2.0*

*y = r * sin(Theta) + RRows / 2.0*

Step 5. Apply FFT to log-polar images

Step 6. Compute the ratio R

Step 7. Compute the inverse FFT of the ratio R:

inv_R = FFT(R, 1) ; 1 means the inverse FFT

Step 8. Find the position with the maximum absolute value, and the values of scale and rotation angle.

maxn = MAX(abs(inv_R), position)

state.scale = b^(position MOD rows)

*state.angle = 180.0 * (position / rows) / cols*

Step 9. Construct a new image **im3** by doing reverse rotation and reverse scaling

im3 = ROT(im2, -state.angle, 1.0/state.scale, /CUBIC)

; ROT and CUBIC (cubic interpolation) are IDL functions

Steps 10-13. Similarly, compute the forward FFT of **im1** and **im3** and then obtain the IR by calculating the inverse FFT of the ratio R; finally, we obtain the shift.

maxn = MAX(IR, position)

state.IX = position MOD rows

state.IY = position / cols

After obtaining the values for the shift, the algorithm applies the shift to **im3** to obtain a new image **im3** (*Im3=temporary (shift(Im3, X, Y)*)), which is the final registered image. In the algorithm, one image must be chosen as the base image. It is assumed that the orientation of the base image is orientated and that the object of the process is to reference the second image. A positive rotation angle means that the image is rotated to the east relative to the base image (i.e. clockwise rotation), while a negative rotation angle means that the image is rotated to the west

(i.e. counterclockwise rotation). It is also assumed that the rotation angle is in the interval [–90, +90], because normally the two images have similar orientations. If for example one of the images is upside down, it can be rotated before we run the program. The angle computed in step 8 is between [0, 180]. IDL's rotation function operates in a clockwise (0 → 360) fashion, and rotation begins from the east. If the computed angle equals to zero, then no rotation is needed; if the angle computed is less than 90, the actual angle is the negative of the computed one; if the computed angle is larger than or equal to 90, then the actual rotation angle is 180–computed angle. So the computed angle (from the interval [0, 180]) can be transformed into the actual angle (in the interval [-90, 90]) with the following:

```
ang = state.angle
    if (ang eq 0.0) then begin
        widget_control, state.angtext, set_value = strtrim(string(state.angle),1)
            ; directly output the angle into the space next to the Rotation angle, see Fig. 2
        widget_control, state.scaltext, set_value = strtrim(string(state.scale),1)
            ; directly output the scale into the space next to the Scaling, see  Fig. 2
    endif else if (ang ge 90.0) then begin
        widget_control, state.angtext, set_value = strtrim(string(180.0-state.angle),1)
        widget_control, state.scaltext, set_value = strtrim(string(state.scale),1)
    endif else if (ang lt 90.0)  then begin
        widget_control, state.angtext, set_value = strtrim(string(-state.angle),1)
        widget_control, state.scaltext, set_value = strtrim(string(state.scale),1)
    endif
```

After finishing the rotation and scale (or only shifts if that option is employed), several considerations must taken into account. The sign convention is that image shifts to the east or south are positive while shifts the west or north are negative. Also, special consideration is

needed when the computed value (state.IX and state.IY) in step 13 above is bigger than half of the image columns or rows (Table 1). The following approach is used to deal with these situations:

```
X = state.IX
Y = state.IY
if (X gt 0.5*state.imagens) and (Y gt 0.5*state.imagenl) then begin
    X = X-state.imagens
    Y = Y-state.imagenl
    widget_control, state.xtext, set_value = strtrim(string(X),1)
        ; directly output the value of X into the space next to the word Column, see Fig. 2
    widget_control, state.ytext, set_value = strtrim(string(Y),1)
        ; directly output the value of Y into the space next to the word Row, see Fig. 2
endif else if (X gt 0.5*state.imagens) then begin
    X = X-state.imagens
    widget_control, state.xtext, set_value = strtrim(string(X),1)
    widget_control, state.ytext, set_value = strtrim(string(state.IY),1)
endif else if ( Y gt 0.5*state.imagenl) then begin
    Y = Y-state.imagenl
    widget_control, state.xtext, set_value = strtrim(string(state.IX),1)
    widget_control, state.ytext, set_value = strtrim(string(Y),1)
endif else begin
    widget_control, state.xtext, set_value = strtrim(string(state.IX),1)
    widget_control, state.ytext, set_value = strtrim(string(state.IY),1)
endelse
endif
```

As a result, two ENVI user functions were constructed: one for images with shift only, and one for shift, rotation and scaling. The one with shift only contains two forward FFTs and one inverse FFT; the one with shift, rotation and scale contains six forward FFTs and two inverse FFTs and requires more time and memory. These two functions are called Automatic Registration under the Register option in the ENVI main menu. If the images are only shifted, computer time can be saved by calling the user function *Image with only Shift*, because we do not need to do the rotation, and scale computation. Otherwise, call the user function *Image with shift, rotation and scaling* (Fig. 2).

5. Experimental results and analysis

To efficiently evaluate the two user functions, tests were performed on both simulated and "real world" images. In the test of simulated images, a new image was made by shifting an original (base) image by certain numbers of columns and rows, or by rotating, scaling and shifting an original image with pre-defined values of rotation angle, scale, and shift. Then the user functions were run to reconstruct the values of rotation, scale, and shift. This procedure provided a straightforward comparison between the computed values with the pre-defined values that is discussed in detail below (Table 2).

Figure 3A provides an example of two images, to the left is the original image, and to the right is the simulated image obtained from the original (left) image by a rotation of 19° to east, scaling 1.3 times, and shifting to the right (east) by 3 columns (pixels) and down (south) by 19 rows (pixels). These two images provide a test for the user function *Image with shift, rotation and scaling*. The results of applying this user function are shown in Figure 2 and, the computed results are almost exactly the same as the known values. The computation provides a

registered (warped) image by using the computed values of rotation, scale and shift. The registered image can be output either to a file or to memory. Figure 3B shows a comparison of the registered image with the original (base) image using the Link Display tool in ENVI. One can visually confirm that the two images match exactly.

Several dozen similar tests of this user function were performed on images of different sizes from 400 × 400 to 2000 × 2000 and are summarized in Table 2. We can see that overall, the relative error (columns 6, 7 and 8) is very small and generally decreases as the image size increases (column 1). The last column shows the reconstruction error in terms of number of pixels. The 400 × 400 image has the biggest error of 4 × 4 pixels. The 600 × 600 image has the smallest error of 0.4 × 0.4 pixels. An error of less than 1 × 1 pixels can be considered to be perfect match compared to the traditional image registration approach of manually selecting ground control points (GCPs).

For the 400 × 400 image, the algorithm runs in only one minute, and for the 2000 x 2000 image, ten minutes were required. By using the IDL's TEMPORARY function that releases some memory that is not needed in later steps, the running time can be reduced in half. Thus, compared to hours or days of manually selecting the ground control points, the new user functions are very time efficient.

In the tests using "real world" images, images of different types (TM, ETM+, SPOT, AIRSAR/TOPSAR, and IKONOS) and dates were employed. The main tool employed to compare and evaluate the registered (computed) accuracy was the Link Display tool in ENVI. This tool can link two images based on pixel coordinates or a real coordinate system, and zoom in/out to any extent that desired to see detail and evaluate the results.

Table 3 shows the attributes of the "real world" examples of TM images (July 12, 1997, July 19, 1988 and June 6, 1991) used from the El Paso, Texas region. As required by the algorithm, they have the same size of 600 × 600, but different coordinate systems (UTM, zone 13 or State Plane, Texas Central 4203). The image from July 12, 1997 was chosen as the base image. After applying the *Image with Shift, Rotation and Scaling* user function, the final registered images have the same UTM, zone 13 coordinate system as the base image (Table 3). The three images are shown in Figure 4. The images for July 12, 1997 and July 19, 1988 have the same coordinate system (UTM, zone 13). Thus, they do not require any rotation or scale, and the computation results for these values are 0 and 1, respectively. However, shifts of –174 (column) and – 81 (row) were determined. Thus for this case, numeric computation time could have been saved by using the *Image with only Shift* function. However, the base image and the June 6, 1991 image are in different coordinate systems, and the computation results derived values for rotation (13.2°), scale (1.0172), and shift (-55, -116). This is due to the fact that UTM involves a Transverse Mercator projection, while the State Plane involves a Lambert Conformal Conic projection. To check the accuracy of the registration, the Link Display tool in ENVI (Figs. 5 and 6) was used. One can visually confirm that the images match exactly. The error is about 1 pixel or less, because no visible offset could be seen. Thus, the registration accuracy using "real world" data is as good or better than for the simulated data.

Other types of "real world images" and images with different overlapping areas and scale factors were tested. Overall, the algorithm works very well. However, as should be expected, the algorithm is not without limitations:

1. The algorithm requires images of the same type acquired during the same season. For example, the algorithm can not be used to register a Landsat TM

image with a radar image, since they have different signatures for the same surface object.

2. The algorithm only works for two images of the exact same size. Especially for the rotation and scale computation, images also need to be square. This limitation is not severe because it is very easy for ENVI (and most other processing packages) to produce images that are of equal size and, if necessary, square before we run the user functions.
3. The algorithm requires images that have an overlapping area larger than 30%. This is usually true for adjacent satellite images, digital airphotos, or medical X-rays and CT. However, if the overlapping area is less than 30%, we still can apply the user functions by cutting a large image into pieces to make the overlapping area satisfy the requirement of larger than 30%.
4. The algorithm only works for images in which the scale change less than 1.8. Otherwise, the criteria of 30% overlapping area is not satisfied. For satellite images of same type, this is not a problem, because the pixel size of the images is the same, and the scaling is about one. A case in which the algorithm will not work is for the combination of SPOT panchromatic and Landsat TM because the scale change would be 30/10=3. However, the case of SPOT multi-spectral and Landsat would work because 30/20 = 1.5.
5. The algorithm will not work for non-linear geometric distortions due to sensor orientation and the affects of relief on geometric distortions, because these local geometric distortions can not be determined by the global algorithm, which assumes that all areas of the image have the same rotation, scale, and shift.

6. Conclusions

We have successfully implemented an FFT-based algorithm for image to image registration using IDL and added it to ENVI as two user functions. This approach has the advantage of letting ENVI take care of all the pre-processing and post-processing work such as input, output, display, filter, analysis, and file management. These new ENVI user functions are very useful for image registration, reducing hours of tedious work done manually, which is currently the most common way of registering images. The accuracy of the FFT algorithm for automatic registration images is quite good, and moreover, the overall accuracy increases as the image size increases. Of course, the algorithm can't solve all of the real-life registration problems. As a result, these user functions can be easily applied for 3 types of image-to-image (either geocoded or not) registration applications: 1. Using an image to register other nearby images acquired during the same season; 2. For mosaicking purposes, to tie one image to an adjacent one; 3. To construct limited mosaicks in which a small portion of an adjacent image is mosaicked with a base scene.

One way to increase the accuracy of FFT-based image registration is to use a more sophisticated (and more accurate) FFT-based image registration algorithm described in Gibson (1999) and Gibson et al. (2001). This algorithm allows for shifts and rotations that are a fraction of pixel. Another possible method of increasing registration accuracy is to use multiple bands (i.e., all Landsat bands, Srikrishnan et al. 2001; Araiza et al., 2002).

Acknowledgments

This work was supported in part by NASA under cooperative agreement NCC5-209 (PACES), and Future Aerospace Science and Technology Program (FAST) Center for

Structural Integrity of Aerospace Systems, effort sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant number F49620-00-1-0365. We appreciate the effort of the reviewers whose helpful comments improved this paper.

References

Araiza, R., Xie, H., Starks, S. A., and Kreinovich, V., 2002. Automatic referencing of multi-spectral images. In: Proceedings of 15th IEEE Southwest Symposium on Image Analysis and Interpretation, Santa Fe, USA, 21-25.

Cideciyan, A. V., Jacobson, S. G., Kemp, C. M., Knighton, R. W., and Nagel, J. H., 1992. Registration of high resolution images of the retina. SPIE Medical Imaging VI: Image Processing 1652, 310-322.

DeCastro, E., and Morandi, C., 1987. Registration of translated and rotated images using finite Fourier transforms, IEEE Transactions on Pattern Analysis and Machine Intelligence 95, 700-703.

ENVI Programmer's Guide, 1998. Research System, Inc., 930pp.

Gibson, S., 1999. An optimal FFT-based algorithm for mosaicking images, M. S. thesis, University of Texas at El Paso, 171pp.

Gibson, S., Kreinovich, V., Longpre, L., Penn, B., and Starks, S. A., 2001. Intelligent Mining in Image Databases, With Applications to Satellite Imaging and to Web Search. In: A. Kandel, H. Bunke, and M. Last (eds.), Data Mining and Computational Intelligence, Springer-Verlag, Berlin, 309-336.

Kuglin, C. D., and Hines, D. C., 1975, The phase correlation image alignment method, in: Proceedings of IEEE 1975 International Conference on Cybernetics and Society, New York, USA, 163-165.

IDL User's Guide, 1999, Research System, Inc., 1200 pp.

Reddy, B. S., Chatterji, B. N., 1996. An FFT-based technique for translation, rotation, and scale-invariant image registration. IEEE Transactions on Image Processing 5, 1266-1271.

Sierra, I., 2000. Geometric foundations for the uniqueness of the FFT-based image mosaicking, with the application to detecting hidden text in web images, M. S. thesis, University of Texas at El Paso, 131pp.

Srikrishnan, S., Araiza, R., Xie, H., Starks, S. A., and Kreinovich, V., 2001. Automatic referencing of satellite and radar images. In: Proceedings of the IEEE System, Man, and Cybernetic Conference, Tucson, USA, 2170-2175.

Xie, H., Hicks, N., Keller, G.R., Huang, H., and Kreinovich, V., 2000. Automatic image registration based on a FFT algorithm and IDL/ENVI. In: Proceedings of the ICORG-2000 International Conference on Remote Sensing and GIS/GPS, Hyderabad, India, I-397—I-402.

List of Figures:

Fig. 3B. Comparison of the warped (registered) image with the base image. Upper left: the base image. Upper right: warped image. Lower left and lower right: enlarged (4 X) images from the small squares inside the upper images.

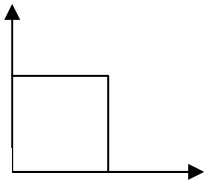
Fig. 4 Examples of TM images that have been tested. A. Base Image of band 7 for July 12, 1997; B. Image of band 7 for July 19, 1988 before input into the algorithm to calculate shift, rotation, and scale; C. Image of band 7 for June 26, 1991 before input into the algorithm

~~Fig. 7. Using the user function of *Image with Shift, Rotation and Scaling* for images of Fig. 7A and 7B. The computation results are no rotation and scale, while image of Fig. 7B shifts 174 pixels to west, and 81 pixels to north. The warped (registered) image output to the memory.~~

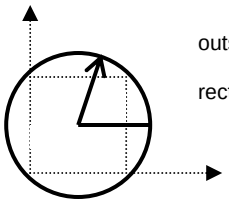
~~Fig. 8. Using the user function of *Image with Shift, Rotation and Scaling* for images of Fig. 7A and 7C. The computation results are 13.2° of rotation angle from north to east, 1.01072 of scale, image of Fig. 7C shifts 55 pixels to west, and 116 pixels to north. The warped (registered) image output to the memory.~~

Fig. 5. Warped (registered) image (right) for July 19, 1988 compared to the base image (left) for July 12, 1997 using the Link Display tool in ENVI. By comparing the shapes and structures of the two images, one can tell they exactly match even though they have different signatures and DN values.

Fig. 6. Warped (registered) image (right) of June 26, 1991 compared to the base image (left) for July 12, 1997 using the Link Display tool in ENVI. By comparing the shapes and structures of two images, one can tell they exactly match even though they have different signatures and DN values.

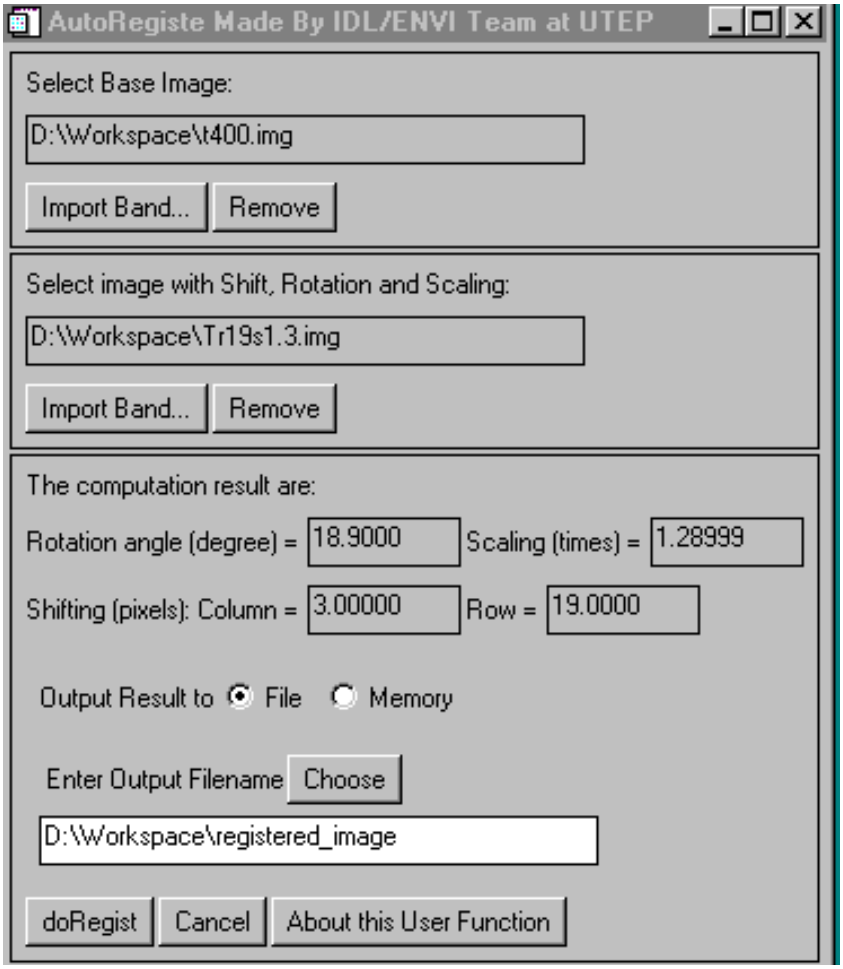


Rectangular



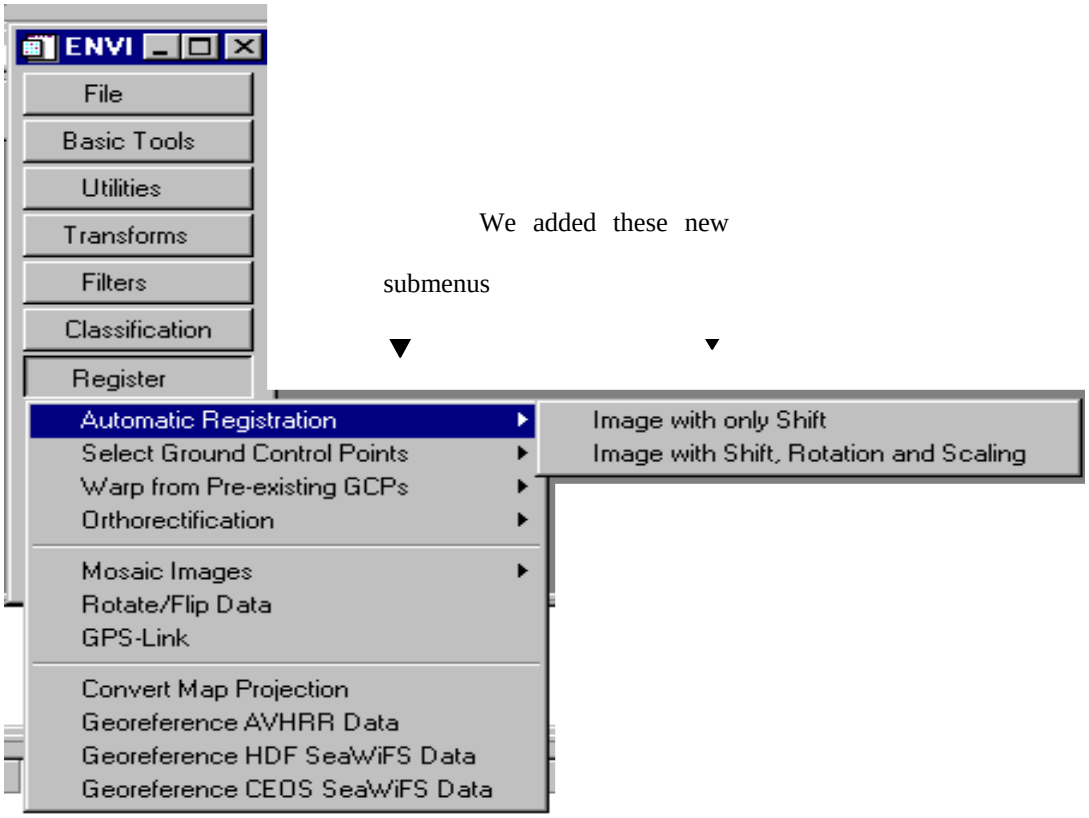
Area outside the rectangular but

Polar

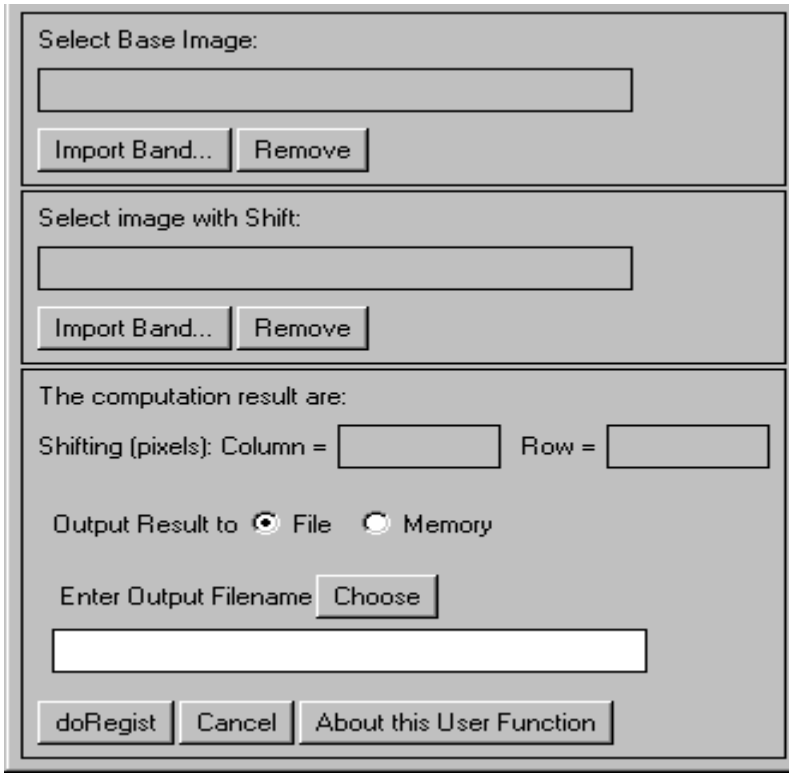


This should read “computation results”

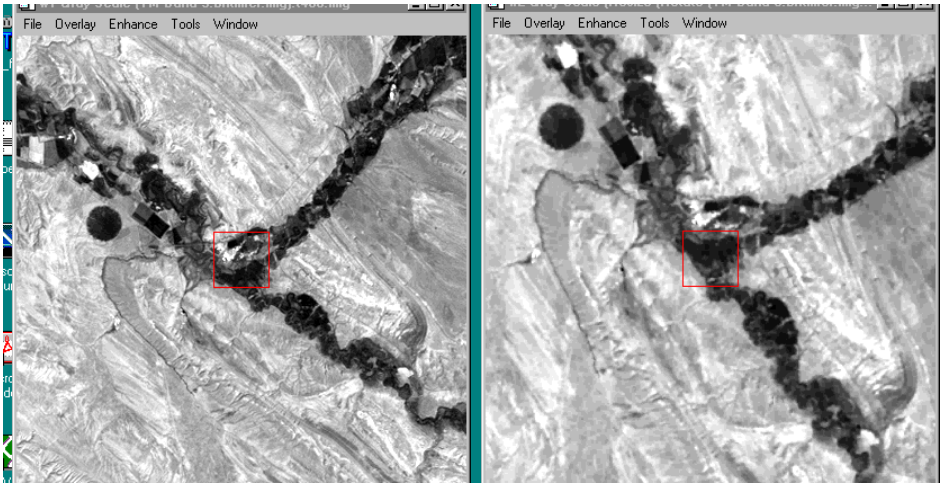
“Shifts (pixels)”



DELETE THIS FIGURE

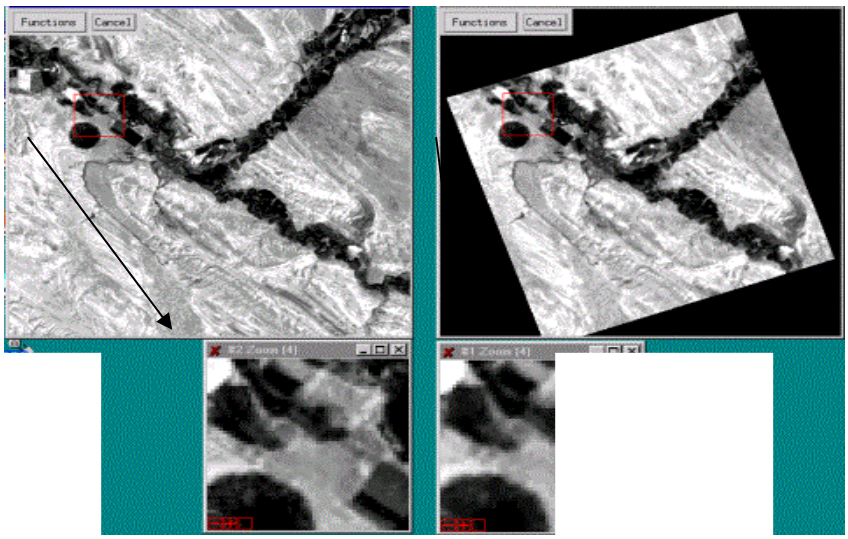


DELETE THIS FIGURE



ENLARGE THIS FIGURE

Explain in text better, this does not look referenced

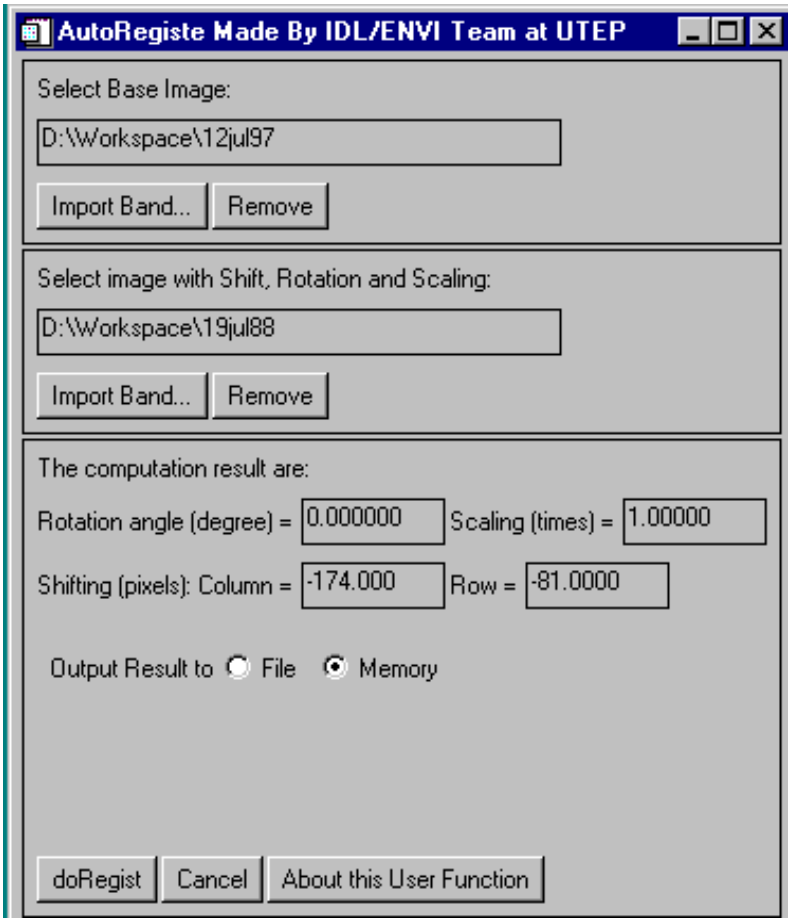


ENLARGE THIS FIGURE

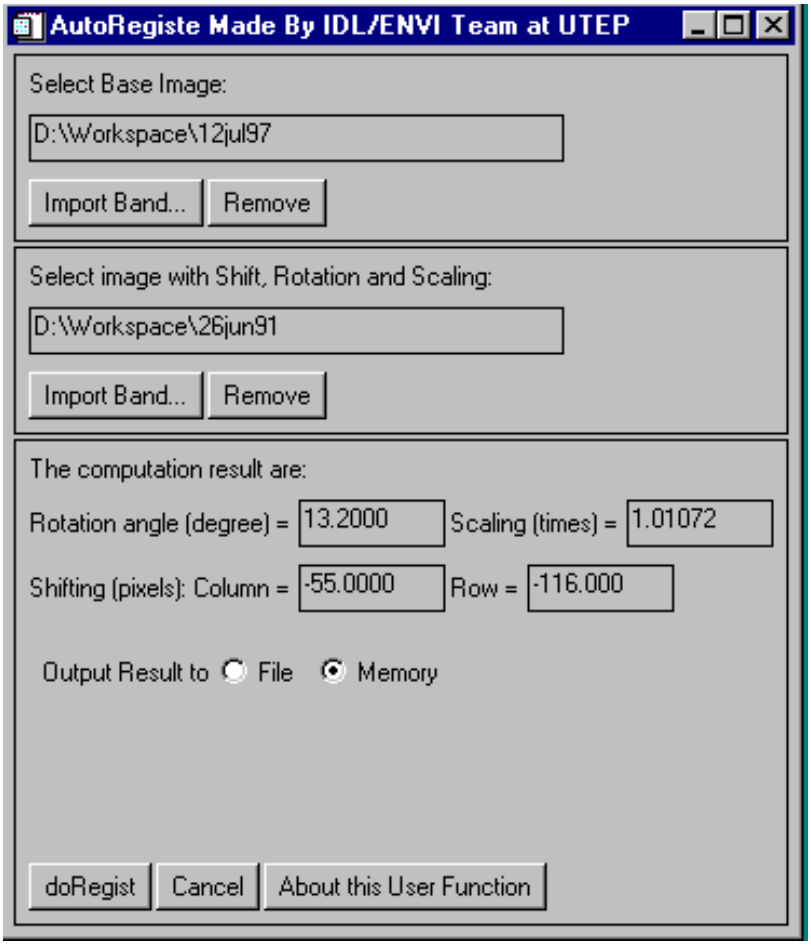
A B

C

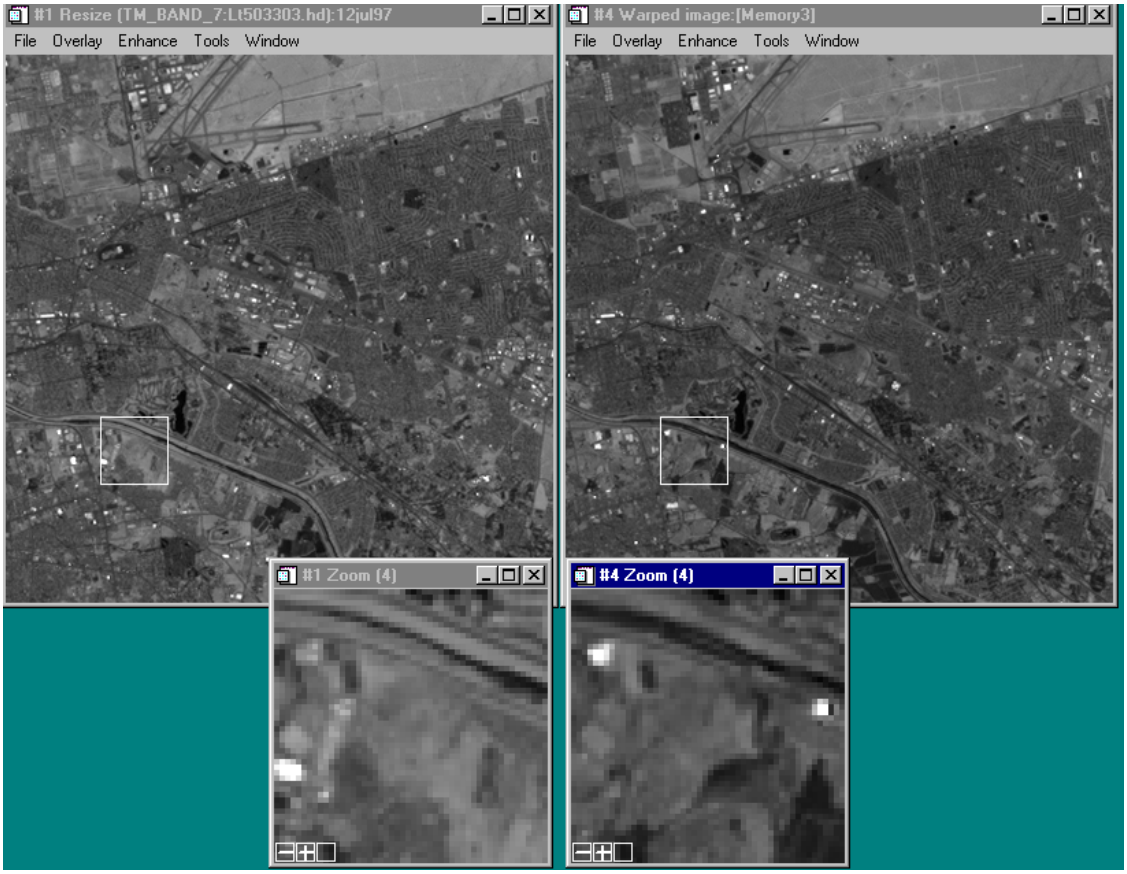
The contrast in these figures needs to be increased. They are too dark.


AutoRegiste Made By IDL/ENVI Team at UTEP
Select Base Image:
D:\Workspace\12jul97
Import Band...
Remove
Select image with Shift, Rotation and Scaling:
D:\Workspace\19jul88
Import Band...
Remove
The computation result are:
Rotation angle (degree) =
0.000000
Scaling (times) =
1.00000
Shifting (pixels): Column =
-174.000
Row =
-81.0000
Output Result to
File
Memory
doRegist
Cancel
About this User Function

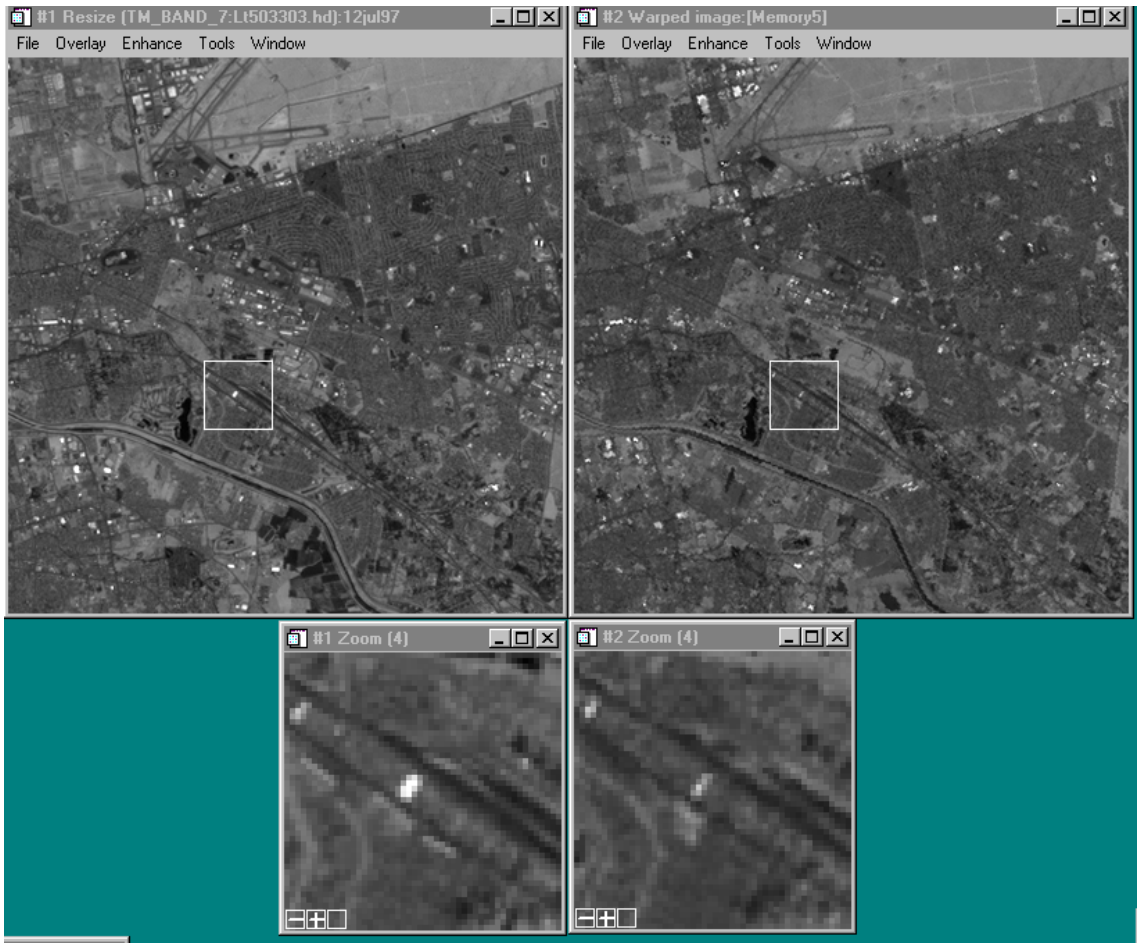
DELETE


AutoRegiste Made By IDL/ENVI Team at UTEP
Select Base Image:
D:\Workspace\12jul97
Import Band...
Remove
Select image with Shift, Rotation and Scaling:
D:\Workspace\26jun91
Import Band...
Remove
The computation result are:
Rotation angle (degree) =
13.2000
Scaling (times) =
1.01072
Shifting (pixels): Column =
-55.0000
Row =
-116.000
Output Result to
File
Memory
doRegist
Cancel
About this User Function

DELETE


#1 Resize (TM_BAND_7:Lt503303.hd):12jul97
File Overlay Enhance Tools Window
#4 Warped image:[Memory3]
File Overlay Enhance Tools Window
#1 Zoom (4)
#4 Zoom (4)

ENLARGE


#1 Resize (TM_BAND_7:Lt503303.hd):12jul97
File Overlay Enhance Tools Window
#2 Warped image:[Memory5]
File Overlay Enhance Tools Window
#1 Zoom (4)
#2 Zoom (4)

List of Tables:

Status of computed shift value	Output shift value	Image shift direction
state.IX< ½ columns	positive	to east
state.IX> ½ columns	negative (state.IX-columns)	to west
state.IY< ½ rows	positive	to south
state.IY> ½ rows	negative (state.IY-rows)	to north

Image size (pixel)	Actual angle	ctual cale	C omputed a ngle	C omputed s cale	rror in ngle	rror in cale	elative rror	E rror in p ixels
40 0 x 400	-17	.3	- 17.100	1 .2899	.1	.0101	.00588	4 x 4
50 0 x 500	-17	.1	- 16.920	1 .10454	.08	.0045	.00471	2 x 2
60 0 x 600	-17	.1	- 17.10	1 .10071	.1	.0007	.00588	0.4 x 0.4
65 0 x 650	-17	.1	- 16.8923	1 .10478	.1077	.0048	.00634	3 x 3
10 00 x 1000	-17	.1	- 16.92	1 .10154	.08	.0015	.00471	1.5 x 1.5
10 24 x 1024	10	.1	10.0195	1 .09940	.0195	.0006	.00195	0.6 x 0.6
10 24 x 1024	13	.1	13.0078	1 .09940	.0078	.0006	.00060	0.6 x 0.6
15 00 x 1500	16	.1	15.9600	1 .10242	.0400	.0024	.00250	3.6 x 3.6
16 00 x 1600	- 17	.1	- 16.9875	1 .10168	.0125	.0017	.00074	2.6 x 2.6
20 00 x 2000	- 13	.1	- 12.9600	1 .09967	.04	.0003	.00308	0.6 x 0.6

	Coordinate of original image	Size	Coordinate of registered image
July 12, 1997 (base image)	UTM, zone 13	600 × 600	
July 19, 1988 (to be registered)	UTM, zone 13	600 × 600	UTM, zone 13
June 26, 1991 (to be registered)	State Plane, TX Central 4203	600 × 600	UTM, zone 13