

AD-A106 070

ROYAL AIRCRAFT ESTABLISHMENT FARNBOROUGH (ENGLAND)

F/6 14/5

A TECHNIQUE FOR LINE EXTRACTION FROM LANDSAT MULTI-SPECTRAL SCA--ETC(U)

JAN 81 A H BENNY

RAE-TR-81010

DRIC-BR-78637

NL

UNCLASSIFIED

1 of 1
Page 0/0

END

DATE

FILED

11-81

DTIC

TR 81010

AD A106070

DTIC FILE COPY

BR78637

TR 81010

UNLIMITED



LEVEL II

①

ROYAL AIRCRAFT ESTABLISHMENT

*

Technical Report 81010

January 1981

DTIC
ELECTE
OCT 22 1981
S
E

**A TECHNIQUE FOR LINE EXTRACTION
FROM LANDSAT MULTI-SPECTRAL
SCANNER SATELLITE DATA WITH
SOME APPLICATIONS OF THE TECHNIQUE**

by

A.H. Benny

*

Procurement Executive, Ministry of Defence
Farnborough, Hants

UNLIMITED

81 10 9 006

(14) RAE-TR-84040

(19) BR-78637

UDC 629.195 : 531.7.084.2 : 681.3.056 : 526.8

(18) DRIZ

(12) 42/

ROYAL AIRCRAFT ESTABLISHMENT

(9) Technical Report 1010

Received for printing 29 January 1981

(6)

A TECHNIQUE FOR LINE EXTRACTION FROM LANDSAT MULTI-SPECTRAL SCANNER SATELLITE DATA
WITH SOME APPLICATIONS OF THE TECHNIQUE.

by

(10) A. H. Benny

SUMMARY

An automated technique is described, for extracting lines from Landsat multi-spectral scanner (MSS) images by means of 'density contour' threading of the data. The resulting lines can then be transformed to fit any map system - in particular the British National Grid - with the aid of selected ground control points. The application of these techniques to the interpretation of Landsat imagery is discussed.

Departmental Reference: Space 592

Copyright

©

Controller HMSO London
1981

320450

LIST OF CONTENTS

	<u>Page</u>
1 INTRODUCTION	3
2 LINE EXTRACTION AND TRANSFORMATION	4
2.1 Lines in map files	4
2.2 Lines in image files	5
2.3 Transformation of lines to a known map projection	6
3 EXAMPLES OF USE	7
3.1 Land-water boundaries	7
3.2 Sandbanks	8
3.3 Multi-spectral examination	9
3.4 Sea features	10
3.5 Resolution and accuracy	11
3.5.1 Line threading	11
3.5.2 Accuracy	11
4 DETAILS OF PROGRAM IM.CONTOUR	13
4.1 Line threading: outline of method	13
4.2 Detection of start of lines	14
4.3 Extraction of lines	15
4.3.1 Line routing through the cell	16
4.4 The main program	17
4.5 Nature of output map file produced by IM.CONTOUR	19
4.6 Subroutines	19
4.6.1 Subroutine CONTR	19
4.6.2 Subroutine CONTRB	20
4.6.3 Function subroutine IVAL	21
4.6.4 Subroutine IBITS	22
4.6.5 Subroutine NSTORX	23
5 DETAILS OF PROGRAM IM.CONTIM	24
5.1 Operation of the program IM.CONTIM	24
5.2 Construction of output file	24
5.3 Examples of the use of IM.CONTIM	25
6 CONCLUSIONS	25
References	26
Illustrations	
Report documentation page	

Figures 1-14

inside back cover

Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/ _____	
Availability Codes	
Dist	Avail. and/or Special
A	

1 INTRODUCTION

The surface of the earth is under continual observation from, amongst others, satellites of the Landsat series. The most widely used observation system carried by these satellites is the multi-spectral scanner, universally referred to as MSS and described in Ref 1.

Briefly, the Landsat MSS consists of a downward looking optical telescope, an oscillating mirror which scans through an angle of about 6° to either side of the vertical, and a detector system. The oscillations are at right angles to the direction of movement of the satellite so that, as the latter travels above the earth, a broad swath 185km wide is scanned. Light from the mirror is directed onto a number of photosensors through optical filters, so that four different spectral bands are sensed. The response signal is taken from each sensor at regular intervals of time and this, together with the geometry of the device, has the effect that the earth is scanned in picture elements (pixels) of about 60×80 m.

Signals from the Landsat MSS sensors are transmitted from the satellite and received at earth stations. These signals are recorded and can be processed and re-recorded on computer-compatible tapes (CCTs). Alternatively, the signals may be converted by suitable equipment into images somewhat resembling photographs, both black and white (for example Fig 1 of this Report) and in colour using the information from three spectral bands.

The CCTs and images are distributed as required to users who interpret them to obtain information of many different kinds.

Since the launch of Landsat 1 in 1972 there has been a copious flow of data, augmented subsequently by that from Landsat 2 and 3, and to be increased yet more in the future by Landsat D and other planned earth resources satellites. To deal with this flow, there has been a large scientific effort, concerned with the extraction of useful information. Due to the continuing large data input there is, as pointed out by Bernstein², a need for machine extraction of information.

One useful form of extracted information is boundary lines between distinguishable earth features. This paper describes a technique for the machine extraction of boundary lines from satellite images. Its first application is to the most conspicuous feature of Landsat images, the difference between water and land, *ie* the delineation of coastlines, etc. However, the technique is applicable to the delineation of any feature which may be distinguished from its surroundings, whether by the information conveyed in one spectral band, or by a simple or complex manipulation of the information in more than one band.

The information resulting from this process is in a form suitable for use by a display device, such as a graphics VDU or an X-Y plotter, and may be used, either directly or with transformation into a known cartographic projection, to produce maps of the relevant terrestrial features.

This Report is illustrated by examples of the application of the delineation process to features to be found in images of south-west Britain.

2 LINE EXTRACTION AND TRANSFORMATION

Landsat MSS data is brought into Space Department RAE in the form of computer-compatible magnetic tapes, which are processed to provide computer files, one for each of the four spectral bands. Each such file, often referred to as an image file or image, consists of a raster-scan sequence of picture element values (pixels). The pixel values are related to the intensity of the radiation received at the sensors of the Landsat MSS and represent the intensity returned from the corresponding portion of the earth's surface, possibly modified by atmospheric absorption and scatter.

The MSS information derives from four spectral bands:

<u>Band</u>	<u>Wavelength</u> <u>(microns)</u>	<u>Colour</u>
4	0.5-0.6	green
5	0.6-0.7	red
6	0.7-0.8	infra-red
7	0.8-1.1	infra-red

The pictures formed from the data in these four bands differ considerably in appearance, depending mainly upon the varying reflectance of the different features on the earth's surface in the above wavelength bands.

It is well known that, at the wavelength range of band 7, water surfaces display low reflectance, apart from sunglint under certain conditions, whereas most other surface features have a significantly higher reflectance. Hence a black and white image for band 7 shows water dark and land clearly lighter (Fig 1). Thick clouds mask both land and water, so this study, as with most Landsat interpretation, is confined to cloud-free areas.

The idea was conceived of threading a line between pixels having high and low radiance values and thus delineating the coastlines, islands, shorelines of lakes and so forth. If such a process is to be done by machine, two problems exist; first, how to define the radiance level at which the operation is to be performed, and second, how to construct the lines.

Once a suitable radiance level has been chosen, as described in section 3.1, the next process is to construct boundary lines at that level. Lines can be stored in various ways, and the method used for extracting lines from images will depend upon the nature of the line to be created. The next two sections describe two formats which may be used to store lines.

2.1 Lines in map files

Before this work was commenced, most satellite imagery in Space Department was stored as a raster scan of pixel values in what had become known as the image format. This format is not readily suitable for storing lines, although it may be capable of doing so with certain limitations, as described in section 2.2.

An alternative means of storing data was therefore devised. This is the so-called 'map format', which is described in detail in Ref 3. Briefly, lines are stored in the form of a series of X,Y coordinate-pairs, sufficient coordinates being present to define each line with the required amount of detail. Ref 3 also describes a set of computer programs for handling map files, in particular for transferring them to a known map projection and displaying them on a graphics visual display unit or on an X-Y plotter. Lines stored in map format do not suffer from any of the disadvantages enumerated below for a line stored in image format.

A program was developed to create boundary line from image files. Since the starting information is an image, the program is named in the 'IM.' series of programs⁴. The process of extraction used is similar to the process of creating contour lines from a matrix of height data, so that the program is named IM.CONTOUR. The operation it performs is in fact radiance (rather than height) contouring. This program is described in detail in section 4 and examples of products obtained with it are used to illustrate this Report. Fig 2, for example, is a plot of the map file resulting from the application of IM.CONTOUR to the image file shown in Fig 1.

2.2 Lines in image files

The images dealt with in this Report consist of a raster-scan of pixel values, that is to say the scene is divided into an integer number of locations in X and Y, and each of these locations has a radiance value assigned to it, in many cases in the range 0 to 255 arbitrary units.

To illustrate the following discussion, consider a simple scene in which all the pixels in the left hand half have low radiance values and all pixels in the right hand half have high values. The boundary between the two types of pixel is thus a vertical line down the centre of the scene. How may this line be represented?

In the context of image files, the most obvious way to represent a line is in the form of another image file. It is possible to create a new image file, having the same dimensions as the original, with all pixels assigned the value 255 (say), except for the central column, in which the pixels have the value 0. This scene, when displayed, would appear as a white rectangle with a vertical black line down the centre, at the required place. Such a type of image may be described as a 'line image'.

As part of the work described in this Report, a program IM.CONTIM has been written to create line images. This program functions both on simple scenes, such as the hypothetical example mentioned, and also on normal Landsat scenes. An example of its use is shown in Fig 3, which was obtained using a portion at the top right of the image shown in Fig 1.

The line image shown in Fig 3 may appear at first sight to provide a satisfactory representation of the coastline and other features. Closer study, however, reveals some drawbacks to line images.

It is inherent in the nature of an image file that all locations are integers. Several undesirable consequences arise from this:

(a) The boundary of a very small island or lake may not be drawn as a small closed loop but as a single pixel.

(b) Closely spaced lines will coalesce into one and their separate identities will be lost.

(c) Since the pseudo-line is restricted to integral locations, whereas usually the required place will lie between these locations, there will inevitably be a loss of accuracy.

(d) Possibly more important than any of the above is the fact that an image file does not allow spatial connection between pixel locations, whereas the essence of a line is its continuity: it starts at a specified location and continues to other locations until it reaches its end (which is its starting point in the case of a closed loop). It is possible to perform many types of manipulation on connected lines which cannot be done with a raster scan of disconnected points.

Fig 4 compares the product of program IM.CONTOUR (Fig 4a) with that of IM.CONTIM (Fig 4b), both having been applied to a small subimage containing the reservoir to be seen in the lower right of Fig 1. In this case, two density levels have been contoured. This figure demonstrates the points mentioned in (b) to (d) above. Point (a) may be noted by careful inspection of Fig 3.

This subsection may be concluded by noting that constructing lines in image files is not seen as a very useful technique, and most of this Report is concerned with the method of section 2.1.

2.3 Transformation of lines to a known map projection

The nature of the orbit of Landsat satellites, together with the rotation of the earth, results in MSS images which are not in any hitherto accepted cartographic projection. Indeed, Colvocoresses⁵ has suggested that the pseudo-projection of MSS imagery be named Space Oblique Mercator.

It was considered desirable to convert the lines extracted from images into a recognised map projection. Since initially work was being done on images of Britain, it was decided that conversion should be made to British National Grid. Details of this conversion are given by Williams⁶ and a program was designed by the author to transform lines obtained from satellite imagery into National Grid maps. This program, MAP.TRANS, is described elsewhere³.

Since this geometric transformation work has been described in detail elsewhere, it will not be repeated here. However, the transformation program is now in regular use within Space Department and results of its application are shown in this Report: for example, Fig 5 shows the result of transforming Fig 2 to National Grid, and superimposing a 10km grid. Fig 5 includes the whole area covered by Fig 1, and the rotation

(about 15°) may be seen. It is possible to select a portion of such a figure, so that no empty corners remain, and this has been done for other transformed data in this Report.

3 EXAMPLES OF USE

This section describes some applications of the line extraction process, to both single-band and multi-spectral images. It is not intended to be exhaustive, but merely mentions some examples of the work which has already been done in Space Department. It is hoped that many more applications will be found in the future.

3.1 Land-water boundaries

As mentioned earlier, the most conspicuous feature of many Landsat MSS images is that in band 7 water appears much darker than most other materials, *ie* it has a lower reflectance in that infra-red band. The delineation of land-water boundaries was therefore selected as the first application for the program IM.CONTOUR.

Operationally, IM.CONTOUR demands a radiance level at which to perform the line threading, so an appropriate value must somehow be selected. The selection can best be done by reference to the histogram of the image, *ie* the distribution of occurrences of the various pixel radiance values. Fig 6a shows the histogram of the band 7 image shown in Fig 1. (The gaps at fairly regular intervals in the histogram are artefacts due to the method of radiance correction applied at the receiving station.) The histogram is seen to be bimodal, *ie* to have two distinct peaks. A value D, which lies at about the lowest part of the dip between the peaks, may be selected by the user.

The radiance values appreciably less than D correspond mainly to water areas, whereas the values much larger than D are mainly due to various types of land surface. The relatively small number of values close to D are most commonly due to regions which are neither land nor water but are at the boundary between the two in regions of gently shelving shoreline, such as shallow beaches and mud banks. In such cases, a gradual change from land to water values is not surprising.

The areas under the two peaks are therefore proportional to the areas of water and land in the image under examination, so for different scenes the relative heights of the two peaks will vary greatly. For example, a scene having only a small proportion of water would have few pixels with values below D and a correspondingly low peak to the left of D.

Selection of a radiance level intermediate between that of land and water is therefore a relatively simple matter. Selection could be at a value which is half way between the two peaks, or at the lowest point of the valley between them. If the selection were to be made by an automatic process, it might be necessary to perform some form of filtering to smooth the histogram, particularly for the second case, the lowest part of the valley.

In the case of the image shown in Fig 1, the radiance level of 39 was selected from the band 7 histogram shown in Fig 6a. The program IM.CONTOUR was then run on the

image at that level and produced the map format file plotted in Fig 2. This plot shows clearly a number of land-water boundaries, including coastline, islands and edges of lakes, reservoirs and docks. Some other features have, however, been delineated, and these are discussed in section 3.3.

Fig 4a illustrates the application of IM.CONTOUR to a very small subimage, 50 columns \times 50 rows, *i.e.* approximately 3×4 km. This subimage includes a lake which may be seen towards the lower right of Fig 1. Two considerably different radiance levels, 20 and 60, have been employed and the resulting lines do not differ greatly, in fact by less than one pixel for much of the shoreline. In the lower right corner of the figure a road is known to bridge the lake and this has resulted in a pixel of intermediate value, so that one contour level shows a continuous body of water whereas the other level shows a separate lakelet.

Fig 4 has not been geometrically transformed to National Grid and is drawn as if the pixels were square instead of (effectively) about 60×80 m. It is therefore distorted and should not be directly compared with a map.

3.2 Sandbanks

The image shown in Fig 1 was acquired by Landsat 2 on 2 July 1977. Other images of the same part of Britain have been obtained, and one of particular interest occurred on 9 June 1976, when the satellite pass (Landsat 1) was at a time near to low tide. The region in question - the Severn Estuary and Bristol Channel - is one having a very wide tidal range, 14.4 m in 1976, between the highest and lowest tides of the year. The pass of 9 June 1976 occurred at 1 h before low tide for that day, the water level being about 2.7 m above datum (which is near the lowest tide level of that year). The band 7 histogram of a portion of the image, shown in Fig 6b, exhibits a much less clear dip than the histogram shown in Fig 6a, and this is due to the presence of pixels having appreciably higher radiance than water but lower than most types of land.

The band 7 image of 9 June 1976, was contoured at a number of different radiance levels and one of these is shown in Fig 7, together with a line selected to represent the coastline. These two levels are marked on Fig 6b as S and D respectively.

An enlarged portion of one region is shown in Fig 8, in this case transformed to the National Grid and with a 2km grid superimposed and labelled with the eastings and northings in kilometres. The land areas are, upper left, part of south Wales and lower right, south west England. A number of regions are delineated within the sea area, and these correspond in a general manner, though not exactly, with the low-water mark shown on the Ordnance Survey 1:50000 map of the region. It is to be expected that the features delineated in Fig 8 are of a non-water nature, since the penetration of water by light of the wavelength of band 7 is small. It is also expected that the lines will not correspond exactly with the low water mark shown on the map, as the latter was obtained at a different date, and movements of the sandbanks are known to occur in these waters.

Another Landsat 1 pass occurred at about 0.5 h before low tide on 1 October 1975, the water level being at 3.4 m above datum at that time. A portion of the resulting image was similarly processed to give Fig 9. This figure shows sandbanks in the same general locations as in Fig 8, but somewhat reduced in extent, due to the higher water level. In both Figs 8 and 9 some smaller features have been selectively omitted, so that the shape of the larger sandbanks can be more clearly seen.

Clearly Figs 8 and 9 depend on the exact radiance values used to obtain the lines, the selection of these levels being to some extent a matter of judgement. However, it seems likely that these figures do give a general indication of the extent of the sandbanks at the relevant times.

Figs 8 and 9 form a demonstration of the use of Landsat imagery to give a synoptic view of coastal waters at one time.

Much of the image used to prepare Fig 9 was obscured by clouds, but, fortunately, sufficient was clear to reveal the area of interest, and also to enable a number of ground control points to be obtained, for geometrical transformation purposes. (The upper left portion of Fig 9 is in fact affected by the presence of cloud.) Due to cloud cover, no other image has been obtained at such a low tidal condition as at the time of the satellite pass of 9 June 1976: the water is, in any case, below that level for only a small amount of the time.

The tidal figures mentioned in this section are for Avonmouth: the Bristol Channel has complex tidal movements, so other locations may differ appreciably.

3.3 Multi-spectral examination

Close examination of Fig 2 shows that it contains a large number of different items. Many of the line items can be identified from maps of the region as land-water boundaries of various types: coastlines, shores of small islands, edges of lakes, reservoirs and docks. Additionally, a number of lines cannot be identified as land-water boundaries but appear to represent entirely different types of feature.

Since Fig 2 was derived from band 7 only, there remains available the information in the other three spectral bands. (In fact there is evidence that the so-called specific dimensionality is only 2 for most terrestrial features in the spectral range covered by bands 4 to 7 inclusive, and this is supported to some extent by the relatively high correlation between bands 4 and 5 and bands 6 and 7: thus the four bands together are able to distinguish features little better than the use of two bands only.)

As an example of the use of all four bands, one of the small bodies of water in Fig 2 was selected for study. All pixels within that body were listed in the four bands and the spread of radiance values in all bands was determined. A subsidiary computer program was then written (IM.COMBINE) to select all pixels within the entire image whose values lay within the same spread of values for all four bands. These were the delineated by IM.CONTOUR and converted to British National Grid, the result being

plotted and shown in Fig 10. National Grid kilometre lines were added with the aid of another computer program, to assist in the location of the delineated items. It is noteworthy that every item delineated by this means is identifiable as an enclosed body of water (unlike Fig 2 where many of the small items are known to be non-water in nature). All but one of these bodies of water can be identified as fresh water - a reservoir or small lake - the remaining one, a dock, being of unknown salinity. The spectral signature of sea water is sufficiently different that no known sea-water body has been detected.

The example described was obtained by a simple combination of data based on limits set for each spectral band. It is possible to devise more complicated and subtle methods of combining the spectral information and these may enable fairly similar types of item to be separated.

3.4 Sea features

Study of coastal waters in all four spectral bands shows that there is considerable variation in the radiance from place to place. Use of IM.CONTOUR enables coastal or sea features to be delineated.

Before delineation of sea features is possible, it has been found necessary to perform some preprocessing of the images. This is necessary for several reasons:

(a) The Landsat MSS has six sensors for each spectral band, so the image is in fact scanned six lines at a time. These sensors are known to vary somewhat in relative sensitivity and also do not have exactly identical response over their spectral band. (A detailed discussion of these and other variations is given in Ref 7.) Although attempts are made by the receiving station to overcome these variations between sensors, it is often found that there remains a residual 6-line stripiness. This is often not detectable in land areas with their considerable radiance variation, but has a marked effect on areas such as sea which are only slowly changing in brightness.

(b) The signal returned from each sensor is converted from analogue to digital form within the satellite. This conversion gives rise to what is known as digitisation noise: for example, if a number of adjacent pixels all have values very close to each other and to the division between two digital values, only slight variations of value would cause pixels to fall into one or other of the two digital values. This is exaggerated at low digital numbers: pixels all in the range 1.49 to 1.51 are within little more than 1% of each other, yet can be digitised and transmitted from the satellite as either 1 or 2, i.e. having 100% difference.

A preprocessing program has been prepared which performs a statistical destriping calculation on the sea areas of an image only and which also applies some smoothing, for example by convolving with a 5×5 pixel 'unit patch'. Further the image may be cleaned up by programs which remove individual pixels or small groups of pixels which are only slightly different from their surroundings. After such treatments, it is then

possible to use IM.CONTOUR to obtain density contours of bands 4 to 7 of the coastal region. An example of such a map in band 4 for an area known as the Wash in eastern England is shown in Fig 11: this was prepared by R.H. Merson⁸. Such maps have been found to contain useful information when studied by coastal geomorphologists⁹.

3.5 Resolution and accuracy

This subsection discusses some aspects of the resolution and accuracy of the lines prepared by IM.CONTOUR and the maps provided by MAP.TRANS. It should be considered as somewhat speculative, as work is still continuing actively in this area in Space Department.

3.5.1 Line threading

The method of line threading used by IM.CONTOUR includes linear interpolation between pixels and this is likely to lead to quite a high order of accuracy, making good to some extent for the digitisation into pixels. This can be seen by considering a situation where the scene under observation consists of land (radiance say 100 units) and sea (radiance say 20 units), and locating a contour at the intermediate level $(100 + 20)/2$, i.e. 60. A scan across the land-water boundary will yield a string of values of 100 followed by values of 20, usually with one intermediate pixel whose value is based on a proportion of both land and sea and therefore has any value between 100 and 20. IM.CONTOUR will locate the coastline coordinate in a position dependent upon this intermediate value and will tend to locate the position to an accuracy very much better than one pixel.

Fig 12 illustrates the situation. Fig 12a shows the land-water transition, and Fig 12b shows the resulting scan of pixel values. If the transition occurs at a distance p pixels from a pixel boundary, a simple analysis shows that the resulting error in the contour line position (shown by the arrow in Fig 12b) is:

$$\text{Error} = (p(2p - 1))/(2(1 - p)) \text{ pixels}$$

where p can have values $0 < p < 0.5$. The maximum error is thus less than 0.09 pixels. In practice, the situation is complicated by the fact that pixels overlap in the scan direction, so the error may not be the value shown.

It may be noted that the image-file pseudo line extraction of IM.CONTIM cannot perform this adjustment, so the location of the boundary can be up to one pixel in error.

3.5.2 Accuracy

After lines have been extracted, they can be transformed to British National Grid by means of program MAP.TRANS. The accuracy of the resulting map will now be briefly considered.

The image provided by the satellite is subject to a number of distortions, several of which are listed by Ref 10. The CCTs normally include corrections for some of these distortions, in particular mirror speed non-linearity, panoramic distortion and earth rotation. The corrections are made by such means as adding pixels for the first two and displacing rows of pixels relative to each other for the third. Since such

additions and displacements must be made in whole numbers of pixels (usually one at a time) each such correction must have associated with it two adjacent errors of at least half a pixel. These corrections are mainly or entirely in the direction of the mirror scan, not in the direction of satellite travel.

Transformation is made from image to map coordinates with the aid of a transformation matrix. At present, the matrix used is only a first-order one, i.e. the data can be translated, sheared, rotated and scaled in x and y but only in a linear manner. The present transformation would therefore enable the earth rotation correction (a shear) to be made. The panoramic distortion correction, being non-linear, would need a second-order matrix.

There are thus several possible routes to a geometrically corrected and transformed product. The present method is to use CCTs which have been geometrically corrected by adjusting pixels and to transform them by a first-order matrix to a map projection. An alternative would be to use a partially or completely uncorrected image and apply a higher order transformation matrix to perform both some correction and transformation. The choice of method depends amongst other things on how the matrix is obtained.

At present, in Space Department, the transformation matrix is formed by obtaining a large number (50 to 80) of ground control points (gcps) for each satellite image and performing a least-squares calculation. Ground control points are locations which can be identified accurately both on the image and also as coordinates on the map grid, usually with the aid of maps: Ordnance Survey 1:50000 scale maps are suitable and an absolute accuracy of 20 m has been quoted for these. Since the gcps are usually spread fairly evenly over an image, the least-squares method of fitting results in the centre of the image having a very high accuracy (perhaps 20-30 m) based on all the gcps, whereas the edge of the image may have a more substantial error, such as 150 m. Adjoining images may in principle be used to improve the location of edge gcps, but this has not yet been done.

The half-pixel errors (30 m) introduced by the geometric correction are localised but are additional to the transformation errors. These correction errors can be reduced or eliminated by the use of raw data and a second or higher order transformation matrix, but such a matrix may have less absolute accuracy for a given number of gcps.

To summarise therefore, the line extraction process of IM.CONTOUR tends to produce line locations within the image with an accuracy considerably better than one pixel. The presently used methods of geometric transformation appear to give an absolute accuracy of about two pixels (say 150 m) in the corners of the image, but better in the centre. The method of geometric correction pre-applied to the image introduces local errors of up to two half pixels (60 m) in the mirror scan direction. It seems that further work, using an uncorrected image, a large number of gcps, and a second or higher order transformation matrix should be capable of giving an accuracy over the whole image of better than a half pixel.

4 DETAILS OF PROGRAM IM.CONTOUR

This section describes in detail the operation of the program IM.CONTOUR and in particular those portions devoted to the extraction of contour lines from an image, which is an orthogonal raster scan grid of radiance values.

Since the contour level may be set to any real number, an image is in general capable of yielding an infinitely large number of different contour lines. One or more contour levels may be provided to the program IM.CONTOUR, and these are then stored and processed one by one. In much of the following, the method of generating lines at one particular contour level is described. The method is then repeated for each of the levels required.

4.1 Line threading: outline of method

The line extraction process is in principle quite simple. Given a contour level, all possible lines must be constructed which thread through the grid such that all locations to one side have a higher or similar value and all locations to the other side have a lower value. Some of these lines may start and finish at the edges of the image, whilst others may form closed loops.

The subroutine CONTR performs the evaluation of the contour lines. The operation is divided conceptually into two parts - the detection of a suitable place to start a line and the actual threading of the line. The former operation is done by statements within CONTR itself, whereas the latter is performed by the routine CONTRB which is called by CONTR.

The image will in general yield several separate contour lines at any specified contour level, so some systematic method of searching is necessary to ensure that none is missed. The method of searching is based upon the observation that a line must run between any two adjacent pixels if one of them has a lower radiance value and the other a higher value than the selected level for the boundary line. If the selected level is the same as a pixel value, the line must run through that pixel, and to avoid missing this condition the test is altered to include the case in which one pixel has a lower value and the other a value greater than or equal to the selected level. This criterion is sufficient to determine whether or not a line must eventually run through or between the pixels of the pair.

The contouring operation starts therefore with a systematic search for, and recording of, all horizontal pairs of adjacent pixels in the interior of the image which satisfy the above criterion. Each adjacent pair of pixels of each scan line of the image is examined. In practice it is only necessary to select those pixel pairs which meet the test in one direction - namely the first pixel of the pair is lower and the second is the same or higher. As a consequence of the process of line threading, such pairs correspond to places where the line is being threaded in a direction down the image (*ie* from above the row to below it). Clearly any closed loop must contain at least one such pixel pair, so all loops can be detected by a search of such pairs. For a reason

which will be seen shortly, it is not necessary at this stage to examine the top and lowest scan line of the image. It is not necessary to repeat the operation for pairs formed from adjacent rows, that is, for vertical pairs between which the line is to be threaded from side to side. The downwards-only pair selection, together with the procedure for edge searching described below, guarantees that all lines will be detected.

If a pixel pair satisfies the above criterion, the fact must be temporarily recorded. The pair may be thought of as 'active', i.e. waiting to have a line threaded through it. If, however, it does not meet the requirement, no line need be threaded. Since the pixel pair can have one of two states - active or not - it is sufficient to use one binary bit of data to record this state.

Since the original image is an orthogonal grid of pixel values, the record of active/inactive pairs will consist of a slightly smaller (one column and two rows less) grid of bits. This grid can be quite large: for example a 3000×2000 pixel image would need almost 6M bits or almost 375000 16-bit words. This is inconveniently large to be retained in the fast memory of a computer, so the bits are stored as words into a temporary file created especially for the purpose. This is performed by a section of code within the main program. Since the information required concerning the activity of a pixel pair is stored as one bit of a word, it is of considerable programming convenience to be able to retrieve it by referring to its location with respect to image coordinates, and this is achieved by the subroutine IBITS. This subroutine allows the state of any bit to be determined and also allows bits to be cleared, i.e. set to the inactive state once they have been used. This prevents lines from being detected a second time.

4.2 Detection of start of lines

Contour lines fall into two types: open lines and closed loops. The former have ends which are at the edges of the image, whilst the latter do not in any meaningful sense have ends as they are continuous. Ends of open lines can be found by searching the edges of the image, this search being done in this program by starting at the top left hand corner and proceeding in a clockwise direction. Each adjacent pair of pixels is examined to see if the first pixel met has a lower value and the second pixel a value the same as or higher than the selected line value, H . If this test is satisfied, a line is considered to start somewhere between the two pixels and the subroutine CONTRB is used to extract the line coordinates, as described later. Note that if a pixel pair is found in which the first pixel has the value H or greater, and the second pixel is less than H , the pair is ignored, although a line must pass between them. Such a case is arbitrarily defined as the end of a line. This rule ensures that open lines are only detected once: otherwise they would be found twice, in opposite directions, and the duplicates would have to be found and removed.

Subroutine CONTR searches first in a rightward direction along the top edge of the image, then down the right hand side, leftwards along the bottom edge, and up the left hand side. By this means, all lines which start (and therefore must end) at the

perimeter of the image are detected and may in due course be extracted. Each time subroutine CONTR finds the start of a line, CONTRB is called to perform the threading of the line. During the threading operation the line normally proceeds into the interior of the image and frequently passes between a pair of pixels which are active. When this occurs, the appropriate bit, held in the temporary file, is changed to the inactive state to indicate that it is no longer necessary or indeed possible for another line having that contour level to pass between the pixels of the pair.

CONTR completes its search by conducting a raster scan for all active bits in the interior of the image, in this way finding any lines which start internally and which therefore must form loops. This raster scan proceeds from left to right along each row of bits held in the temporary file. When an active bit is encountered, CONTRB is called to thread the line. All active bits encountered along the loop are set inactive. By this means the whole image is systematically processed until eventually no bits remain active, at which stage all contour lines of the required level must have been located and evaluated.

4.3 Extraction of lines

Each time the subroutine CONTR detects the start of a line, it calls subroutine CONTRB to extract the line coordinates. These are x, y locations, and are expressed as real numbers, unlike the image coordinates which are integer values.

Two types of start-of-line have been shown to occur. In the case of lines starting from the edge of the image, the situation existing is that shown in Fig 13a: a pixel pair AB has been located with A lower than the selected level H and B the same or higher. The location of the line coordinate P1 is then calculated by linear interpolation. The region to the right of AB is known to be inside the image (from the method of selecting edge locations described above) and the region to the left of AB is outside. The line must therefore be followed to the right. In the case of an internal starting point, when the first active bit of a new loop is encountered CONTRB arranges to select the two pixels such that the lower value is at A and the higher at B (Fig 13a again) and again the line is followed to the right. (In fact, in the case of an active bit, the pixels are adjacent ones from the same row and the line is proceeding down the image, but the situation may be rotated anticlockwise by 90° to correspond with Fig 13a.) At this stage CONTRB can then proceed in the same manner for the location of subsequent line coordinates, for both open-ended and closed contour lines. As a consequence of these rules, all lines are evaluated in the same sense, that is with the lower values to the right of the line. Hence all lines moving around a relatively lower numbered (*ie* darker) portion of the image do so in a clockwise direction and lines travelling around a relatively brighter part of the image do so anticlockwise.

To determine subsequent points along the line, a square cell is constructed. Referring to Fig 13b, points A and B are the pixels shown in Fig 13a between which the line is known to start, and the line is to proceed to the right. Points C and D

represent the pixels adjacent to B and A and these four form a square cell through which the line must proceed and which it must eventually leave, through any side except AB.

4.3.1 Line routing through the cell

The detailed construction of the line through the cell is performed by subroutine CONTRB, the operation of which is indicated in Fig 14. A, B, C, D and P1 have the same meanings as in Fig 13. Because of the manner in which the starting pixel pair was found, it is known that the radiance value ZA of pixel A is lower than H, the line level and the radiance value ZB of pixel B is greater than or equal to H. P1 is located by linear interpolation of H between ZA and ZB. The radiance values of pixels C and D are ZC and ZD respectively. The centre of the square ABCD is called X and it has assigned to it the value ZX, which is the arithmetic mean of the four corner values, ZA, ZB, ZC and ZD.

The line is routed through the cell ABCD by a series of decisions based on the values ZA to ZD, ZX and H, this being illustrated in Fig 14. The numbers in circles in Fig 14 refer to label numbers in subroutine CONTRB, and the decision tests are placed within rhombus (lozenge shaped) boxes. The first test is whether ZX is less than H and the path taken is indicated in Fig 14. If ZX is less than H, a logical variable RIGHT is set TRUE (otherwise FALSE). A TRUE value for RIGHT implies that the line will have the point X on its right, whilst a FALSE value means that the line will have point X on its left. The line therefore cuts either line BX or line AX respectively, and the point P2 can be calculated accordingly, by linear interpolation between B and X or A and X. After crossing one or other of the diagonals, a decision must again be made concerning the routing of the line. The decision may result in the line leaving the cell via the upper or lower edge. Alternatively, the line may proceed to cut one of the other diagonals between B and X or between D and X, after which another routing decision is faced. Fig 14 shows all of the possible routes which a line may take, to leave the cell eventually by any of the sides other than the one by which it entered.

When the countour line has been threaded through a cell, the situation reached at the exit edge can be brought into correspondence with that shown in Fig 13a, that is, the line coordinate is at a new location P1 within a new side AB and travelling to the right. It is then possible to construct a new cell ABCD based on AB and the line can be threaded through that cell in the manner described above. The method of construction of the new cell depends upon which side of the old cell was reached. For example, if the line left via side BC the program steps from label 200 onwards are used, values of ZA, IA and JA being set up to define the new cell. Similarly, exit via CD or DA causes the program to go to label 210 or 220 with corresponding changes to other program values.

As each coordinate of the contour line is evaluated, it is stored in an output file by means of the subroutine NSTORX, which is described in detail later. The contour line is evaluated, cell by cell, until its end. The end of a line is detected in different ways, according to the type of line. An open line ends by reaching the edge of the image whilst travelling in an outward direction. A closed line eventually reaches

its starting location, which was originally an active pixel pair, but is now inactive. A closed line is therefore stopped when it reaches an inactive pixel pair.

Each line coordinate is calculated by linear interpolation between the two adjacent locations under consideration (AB, AX etc). It is doubtful whether a technique more sophisticated than this would be justified when account is taken of the manner in which the radiance values are provided by the Landsat MSS system.

4.4 The main program

This section describes some aspects of the operation of the program IM.CONTOUR, other than the method of contour determination described above.

The data used as input to the program is in the form of a raster scan image in the standard Space Department Image Format. The resulting density contour lines are produced as strings of x, y coordinates and these are written in Map Format³. Hence the program makes use of a number of standard subroutines for accessing and reading from images (eg OIR RL12 CLOSE1) and writing a new map file (eg NOPENO NSTORE NCLOS0). These subroutines require the user to answer appropriate questions, such as the names of the input image file and the output map file, together with other questions of a now standard nature. Additionally, the program opens a temporary file for storage of the data bits which indicate the state (active/inactive) of the pixel pairs. A small suite of subroutines was necessary, to open a temporary file, write to it, read from it, close it and, at the end of the program, delete it. This suite of programs is of general use and has subsequently been placed into a library for use by other programs.

When the input image file has been opened, the user is asked how many different density contour levels he wishes to be evaluated and the actual values of these levels. Up to 20 different levels can be stored and processed one by one. In practice, it is often convenient to evaluate one contour level at a time, so that the resulting lines may be stored in separate map files, one file for each contour level.

At this stage, the map format subroutine NOPENO asks for the name of the output file and PERMHD and UPDATE ask for some textual details for the header of the output file. The program is then able to supply a number of additional words to the output file header buffer, and write it to the disc. The four output map corner points or fiducials are then written to the output buffer by means of the subroutine NSTORE.

Next a comment level is requested. According to the number supplied, which must lie within the range 0 to 5, the program prints out statements to the VDU during the course of its operation. If the comment level 0 is provided, the user is only informed of the name of the temporary file in current use so that if the program should be interrupted for any reason that file can subsequently be inspected or deleted.

For other values of comment level, additional information concerning the progress of the program is output to the VDU. Output at each comment level listed below is additional to the output provided at lower levels.

Comment level 1: information on the temporary file:

START TEMP FILE FOR CONTOUR (level)
TEMP FILE PROCESSED FOR CONTOUR (level)
(Main program, formats 1012, 1013)

Comment level 2: progress as each side of the image is completed:

TOP EDGE DONE
RHS DONE
LOWER EDGE DONE
LHS DONE
(Subr CONTR, formats 1001-1004)

Comment level 3: information on every contour line evaluated:

CONTOUR HAS (N) POINTS
(Subr CONTRB, format 1001)

Comment level 4: some information on the progress of each contour line:

Every hundredth point of each line is output. Clearly this applies only to lines containing at least 100 points.
(Subr NSTORX, format 1001)

Comment level 5: the MK and coordinates are output for every point of each line.
(Subr NSTORX, format 1000)

The selected contour level(s) are then processed one by one. For each level it is first necessary to prepare the temporary file of active bits and then to process the image. The temporary file is formed by examining each adjacent pair of pixels of all but the first and last rows of the input image to see whether it is active (as described in section 4.1) and, if so, the corresponding bit is set to unity. This is done with the aid of the array IBIT which contains 16 values, set by a DATA statement, one for each bit position of a 16-bit integer word. Each of the 16 IBIT words has all except one bits zero. Thus IBIT(1) has the left hand or most significant bit set to one, IBIT(2) has the second bit one, and so on until IBIT(16) which has the rightmost or least significant bit set to one. Any chosen IBIT value thus provides a mask which can be used to set one bit of a word active, to test for the presence of, or to clear an active bit, by means of one or other of the various logical operations available.

When the temporary file is complete, the subroutine CONTR is called to perform the line threading operation and to output the coordinate values.

When the selected contour level(s) have been evaluated, the program asks whether further work is to be done: if the answer is YES the program returns to its start, otherwise it closes down.

The logical variables NEWIM and NEWLEV respectively are set TRUE at the start of each new image used and for each new density contour level being evaluated. Their chief purpose is to allow certain of the subroutines to perform some initialisation on their first calls. Once this has been done, the logical variables are set to FALSE.

4.5 Nature of output map file produced by IM.CONTOUR

The end product of operation of the program IM.CONTOUR is one or more data files in map format containing contour information in the form of strings of x, y coordinates. This information in some respects resembles a map, but it is in fact an imperfect one. It may later be converted to a conventional map by means of the program MAP.TRANS³.

To indicate the nature of the data, this pseudo-map has placed in its header the projection code 99, which indicates that it is a product derived directly and without transformation from an image. The header is also provided with values for the corners, CNR(a,b). Conventionally, the lower left hand corner (the origin for the output file) is given the value 0,0. The top right hand corner has a value numerically one less in x and y than the number of columns and rows of the input image. The reason for this is that the edges of the output map correspond to the four lines which pass through the centres of the edge pixels of the image, so that the separation of opposite edges of the map is one less than the number of columns or rows of the image. During contour evaluation, care is taken to ensure that the coordinates produced relate correctly to the origin and other corners as thus defined.

The header is also provided with some textual and other information supplied by user replies to questions from the subroutine PERMHD and material provided by subroutine UPDATE.

The output map format file has no immediate geographical meaning: it carries no scale and has no geographically significant corner points. To obtain a geographically useful map it is necessary to determine a transformation matrix and to use this to transform the data to an accepted cartographic projection. This is the subject of other RAE reports^{3,6} and will not be described in this section. However some results of using these transformations are discussed and illustrated elsewhere in this Report.

4.6 Subroutines

The program IM.CONTOUR makes use of a number of subroutines, other than library ones, and these are described in this section. Some of these subroutines have already been mentioned, but they are dealt with here in more detail.

4.6.1 Subroutine CONTR

This routine is used to find the locations of the starting points of all contour lines extracted from an image, for one given density contour level H.

When IM.CONTOUR is started, and at each subsequent restart for a new image, the logical variable NEWIM is set TRUE. On first calling CONTR, certain operations are performed which need not be repeated on subsequent calls. NEWIM is set FALSE by IVAL so that these initial operations are not repeated unnecessarily.

CONTR mainly consists of five loop operations, these being the searches for starting points of new contour lines along the four sides of the image and through its interior. During the first four of these loops, the logical variable OPCONT is held TRUE, which informs subroutine CONTRB that it is operating on an open ended contour. After the

completion of the edge searches, OPCONT is set FALSE so that CONTRB can alter its mode of operation suitably for closed lines.

Before each of the five searches, the variables I, J, IA and JA are given values. I and J correspond to the image column and row number respectively, whilst IA and JA are direction indicators.

For the four edge searches, adjacent pixels are examined to determine if a line starts (not ends) between them, as described in section 4.2. When this condition is met, subroutine CONTRB is called to evaluate the line.

For the search through the interior, the image is scanned raster-fashion, *ie* the temporary file is worked through sequentially until an active bit is found, when CONTRB is again called to evaluate the line.

4.6.2 Subroutine CONTRB

Subroutine CONTRB is designed to evaluate and store one contour line once its starting location has been found by CONTR. The start position and direction parameters I, J, IA and JA are passed to CONTRB. Since CONTR needs to retain these four values unaltered, CONTRB starts by making its own copies which it will alter leaving the input parameters unchanged. At its commencement, CONTRB sets the variable MK to 3, the start-of-line indicator for the output map file. Also NPTS, used to count the number of points in a contour line, is set to zero, for use by NSTORX.

Once the first point of the line has been calculated and stored, MK is set to 0 (interior point of a line). When the last point of the line is reached, MK is set to 1, which is the last-point-of-line value. The value of MK is used both to store the line correctly to the map file, via NSTORX and NSTORE, and also to control the flow of the subroutine CONTRB.

When CONTRB is set into operation, the situation shown in Fig 13a is established. The locations of points A and B within the image are determined from I, J, IA and JA, and the radiance values of A and B obtained by use of the function IVAL. The x and y coordinates of P1 are then calculated by linear interpolation. In the case of the first point of a line, the coordinates are then stored by means of subroutine NSTORX. CONTRB then moves to a section of code which is concerned with constructing a square cell to the right of AB and threading the contour line through this cell, storing coordinates as they are calculated on the way through, until it reaches one edge of the cell. This has been described in section 4.3.1 and is illustrated in Fig 14. A new cell is then constructed on the output edge of the old cell, using the current values of I, J, IA and JA. The manipulation of these variables according to the output edge of the old cell is performed by the section of code following labels 200, 210 or 220, and has the effect of rotating or translating the old cell to the new one.

For all points along a line, other than the first and last, a check is made by NSTORX to see whether the coordinate values are the same as the last values stored, in which case the repeat value is not stored. This prevents unnecessary duplication which may occur in certain circumstances. For example, consider the case where one pixel has a

radiance value higher than the specified contour level, and all the surrounding pixels have lower values. The resulting contour line is a closed loop containing nine coordinates, *ie* the starting value and eight more points at 45° intervals around the high value, the last one being identical to the first. If however the central pixel value were the same as (rather than higher than) the contour level, this nine-point line would have all coordinates identical, *ie* a loop of zero area, which should be a single point. By means of the check which avoids duplicate values, the line is reduced to a two-point line of zero length. It is a matter for discussion whether such items should be recognised by the program and stored as single points (map format code 4) rather than two-point lines (codes 3 and 1).

4.6.3 Function subroutine IVAL

It is usual for present day computers to have access to data both from what may be described as 'fast' memory and from other storage media. The computer on which IM.CONTOUR is run is no exception. By fast memory is meant what used to be referred to as 'core' (because it was usually formed from small magnetic toroids or cores), but what is now often bipolar or MOS transistor memory. The time required to read a data word from fast memory is typically $1\ \mu\text{s}$ or less (perhaps very much less). The amount of fast memory is usually limited: it may be as little as say 16K (1K = 1024 words) for a minicomputer and is seldom more than a few thousand K even for a large installation. The machine used for the present work has much less fast memory than is required to hold a single band of one Landsat MSS image. The image must therefore be read from its normal location on 'slow' memory, in this case a magnetic disc unit which can store large quantities of data. Discs rotate at a few thousand rev/min: a speed of 3000 rev/min corresponds to 50 turn/s, so that a wait of 20 ms might be necessary before a particular data word could be read. Thus, reading from a disc is at least four and possibly more orders of magnitude slower than reading from fast memory.

It is usual to store data on a disc in 'blocks', *ie* a number of consecutive values are located sequentially on one sector of the disc. In the case of the discs used for this work, 1024 data words are placed into one block and it is normal practice for all of these words to be read to a fast buffer store, irrespective of how many of these are to be transferred by the computer system to its fast memory under program control. Clearly, it will be much quicker to read several words which are all in one block together rather than to read each one separately.

During the evaluation of a contour line, it is necessary to know the values of the corners of successive cells. If these values were read one by one from the disc, the program would operate very slowly indeed. However, the whole image is far too large to be transferred in its entirety to fast memory. A compromise has therefore been made whereby a portion of the image is held in fast memory. The nature of this portion will next be described, but it suffices to say here that the method will always be at least twice as fast (and often very much faster than that) when compared with reading individual image values one by one from the disc.

Data is stored by function IVAL into an array ITAB which has 64×64 words available for data storage and 64×2 words used as pointers. IVAL is used to return the value of the pixel whose column and row numbers are supplied as parameters. The pixel value may

already be held within array ITAB, in which case it is extracted and returned. If, however, the value is not already present, it is brought, together with 63 neighbouring values centred on it, from disc into one row of the array and then supplied to the calling program. Thus ITAB becomes filled with portions of rows of the image in the general region in which the contour line is being threaded. If the line proceeds sideways across the image or retraces its vertical direction, image values will already be present in ITAB without need for further disc reading, so the operation takes place at the speed of fast memory access.

Some points may be noted about the operation of IVAL.

(a) In a situation where new rows are continually being written to an array, overwriting existing data, several choices are possible concerning which row should be overwritten. One method is to keep a record of the 'age' of each row and to overwrite the oldest, another is to note how frequently each line is used and to overwrite the least used. In the case of IVAL, each row is written into a predetermined location, the number of the row in ITAB being the modulus 64 of the image row number. This has the advantage that no search is necessary. The location is calculated and the 65th word of that row is examined to see whether it is the required image row number or not: if so the word can be read directly and if not the required portion of the image row is read from disc and written into the specified row of ITAB.

(b) The data held in ITAB will rarely be a 64×64 square subsection of the image. Each portion of an image row read in is centred on the current location of the contour line, which will wander about the image.

(c) It is possible to devise methods of operation of IVAL which operate more rapidly in some circumstances. For example, the image could be preprocessed into sub-images say 32 pixels square and these stored on disc. As the contour line is threaded through the image, the appropriate subimage could be read from disc. This would involve only one read of 1024 words, compared with the present method which needs 32 disc reads - one per row. The preprocessing would itself however be a fairly lengthy operation, so that there would only be an advantage if the image were to be extensively contoured, say several different contour levels, each of which spread over much of the image.

4.6.4 Subroutine IBITS

Subroutine IBITS has a conceptual resemblance to function IVAL: it addresses the same problem, that the amount of fast memory available is much less than the quantity of data to be accessed. In the case of IBITS, the data is in the form of individual bits (active/inactive) stored 16 to a word and, since the data is altered by the operation of the program (bits are made inactive), it is necessary to be able to write the altered data back as well as to read it. In the case of IBITS, the data is held on a temporary file and this, like normal image files, is stored on disc.

The parameters I and J refer to the image column and row of the lower numbered pixel of the pixel pair whose activity is being considered. The parameter IND controls the direction of operation of IBITS: IND = 1 (actually not equal to 2) causes the

requested bit to be returned in IBITV, whilst $IND = 2$ causes the bit to be cleared (*ie* set to zero). IBITS is called once in subroutine CONTR, with I and J equal to zero and IND equal to one. This has the effect of initialising the variables NW and DNW in IBITS for the first contour level of an image, and clearing the last column of IBITT at the start of each new contour level.

In the case of IBITS, the data is held in array IBITT, which is 256×16 (the 257th word of each row of IBITT holds the image line number). 256 words can hold 4096 bits, which is sufficient to contain a complete row of a Landsat MSS image (typically 3240 pixels). The size 256 was chosen as it is large enough to contain information on a whole image line and also because it is an exact submultiple (one quarter) of a data storage block of the computer system in use (1024 words). Consequently, any 256 word set of data stored in the temporary file should never bridge two data storage blocks, so it should always be possible to acquire 256 words in one disc read operation.

IBITS therefore holds 16 rows of the entire image, unlike ITAB which holds 64×64 data words. It may be argued that a square array of data would be better. It seems possible that, in practice, the slight advantage of a squarer array might be offset by the extra computational time needed to calculate and obtain the required portion of each partial row of bits. The general point can be made that investigation of the speed differences which can be achieved are likely to be laborious and time consuming, as the various alternative programs must be written and then tested under a variety of conditions. Also it appears that the speed of operation depends on the size and type of image being used and how many different contour levels are to be extracted. All things considered, the best course is probably to design a program such as this (which does not run in real time) with some regard to speed of operation, but not to devote great effort to relatively small further speed improvement. In a multi-user system, such as the present one, which also allows a program to be run as a 'phantom', there is no longer the necessity to obtain the ultimate in speed.

As with the main program, subroutine IBITS contains an array of 16 values (here named IBITR) which form masks for the 16 possible bit positions in a word. In subroutine IBITS, these masks are used either to test for the presence of an active bit or to clear an active bit to zero when it has been used and is no longer needed.

4.6.5 Subroutine NSTORX

Contour line coordinates are written to the output map file by means of the library routine NSTORE. However, some other operations often accompany an NSTORE call during line evaluation, so it is convenient to combine them into a subroutine NSTORX.

NSTORX starts therefore by examining the x, y coordinates to be stored, and if they are both the same as the previous value, and also provided that the point is an interior line point (not a start or end), the point is ignored. All other points are stored. Then, according to the comment level ICOM, information may be output to the VDU:

ICOM = 5 causes every coordinate to be printed out whereas

ICOM = 4 causes every hundredth coordinate of a line to be output.

NSTORX keeps a check on the number of coordinates in each contour line, incrementing the counter N by one for each coordinate stored. The number of points is output to the display device by subroutine CONTRB if the comment level is three or greater.

5 DETAILS OF PROGRAM IM.CONTIM

Program IM.CONTOUR creates lines in a map format file from an image file. As an alternative, program IM.CONTIM was designed to create an output file in image format from an input image file, the output being a representation of density contour lines at one or more contour level from the input file. The word 'representation' is used because an image file is inherently a matrix of points without interconnection, so a line cannot be stored in such an image file. It is however possible to form a simulation or representation of a line in an image file by setting all the pixel values to say zero and then setting a selection of pixels, which physically are arranged in the nearest locations to the places where the line should be, to a non zero value. If the resulting image is displayed in some suitable manner, with zeros as white (*eg* no mark on a sheet of paper) and non zero values as black (*eg* an asterisk or other mark on paper), this will superficially have the appearance of a line (Fig 3).

The disadvantages of representing lines in image files are discussed in section 2.2 of this Report. This section is concerned with a description of the program IM.CONTIM which creates such files.

5.1 Operation of the program IM.CONTIM

In operation, the user of the program is first asked to provide the name of the image file to be contoured, how many density contour levels are to be evaluated and the individual values of these levels. Then a line of text is requested for the output file, followed by a request for the name of that file. The computer then constructs the output file and closes it. The user is then given the opportunity to make more output files from the current input file, or to select another input file. If either of these offers is accepted, the appropriate earlier questions are posed and more files constructed; otherwise the program stops.

5.2 Construction of output file

The method of construction of an output file will be considered first for one contour level H only.

An output image is constructed, raster-scan fashion, pixel by pixel. The algorithm for determining each output pixel value is simple: the value is set to zero, then if the input value is less than H and either the preceding pixel in that row, or the following pixel, or the same pixel in the preceding or following row has a value equal to or greater than H, then the output pixel is set to 255. This algorithm must be modified for those pixels close to the edge of the image: in such cases there will be less than four suitable adjacent pixels and correspondingly less tests to be made.

If there is more than one contour level, the above algorithm is applied for each contour, except that the output pixel, once set to 255, is not returned to zero for subsequent tests.

Thus 'lines' can be considered as additive: each extra contour level can create extra lines, but cannot remove existing ones.

5.3 Examples of the use of IM.CONTIM

The program IM.CONTIM has been applied to a number of portions of Landsat MSS images. A convenient method of displaying the result is to plot the 'line' pixels as crosses on the X-Y plotter. In practice this has been done by converting the image file containing 'lines' into a map format file, by means of a program IM.MAP written for the purpose. Each line point of the image is represented by a point in the map file and these points can be plotted as crosses of any required size by means of the program MAP.HPLOT³. Fig 3 shows the result when the process was applied to a sub-image of size 280 columns by 300 rows. Fig 3 may be compared with Fig 8, which is the product of IM.CONTOUR applied to another sub-image of comparable area. The general impression given by the two figures is similar, but examination of Fig 3 shows that all fine detail is lost. Also, when two or more lines approach closely, their identities become lost.

Fig 4 shows examples of IM.CONTOUR (4a) and IM.CONTIM (4b) applied to a very small portion of a Landsat MSS image, a 50 column by 50 row sub-image including the Chew Valley lake containing Denny Island. This figure clearly shows the differences between the two methods of operation, and the difficulties which can arise with the interpretation of image file (4b).

6 CONCLUSIONS

This Report has described the details, and demonstrated the use of, a method of delineating land-water and other boundaries in Landsat MSS imagery. An alternative but less satisfactory method has also been described. Together with transformation to a known map system, this process enables a synoptic map of large areas (185km square) to be prepared, of regions such as coastal waters, which would otherwise be difficult and expensive to obtain.

REFERENCES

<u>No.</u>	<u>Author</u>	<u>Title, etc</u>
1	National Aeronautics and Space Administration	Landsat data users handbook. Goddard Space Flight Center, Document No.76SDS4258 (1976)
2	R. Bernstein	Digital image processing for remote sensing. IEEE Press (1978)
3	A.H. Benny	A cartographic computer data format and associated programs. RAE Technical Report 80037 (1980)
4	G.J. Davison	Image processing software (Issue 1). RAE Technical Memorandum Space 280 (1980)
5	A.P. Colvocoresses	Space oblique mercator. Photogrammetric Engineering, Vol 40, 931-926 (1974)
6	J.M. Williams	Geometric correction of satellite imagery. RAE Technical Report 79121 (1979)
7	P.N. Slater	A re-examination of the Landsat MSS. Photogrammetric Engineering, Vol 45, 1479-1485 (1979)
8	R.H. Merson	A note on masking and cleaning operations. RAE Internal Communication, Space Department (April 1980)
9	P. Mather	Private communication. University of Nottingham (1980)
10	European Space Agency	Format specifications for Landsat MSS system corrected computer compatible tapes produced at Fucino, Italy (1979)

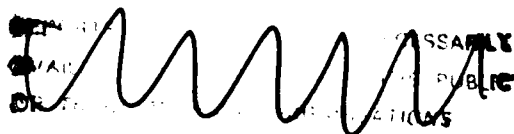


Fig 1



Fig 1 Portion of Landsat 2 image, path 219, row 24, 2 July 1977, showing the Severn Estuary in band 7

TR 81010

Fig 2

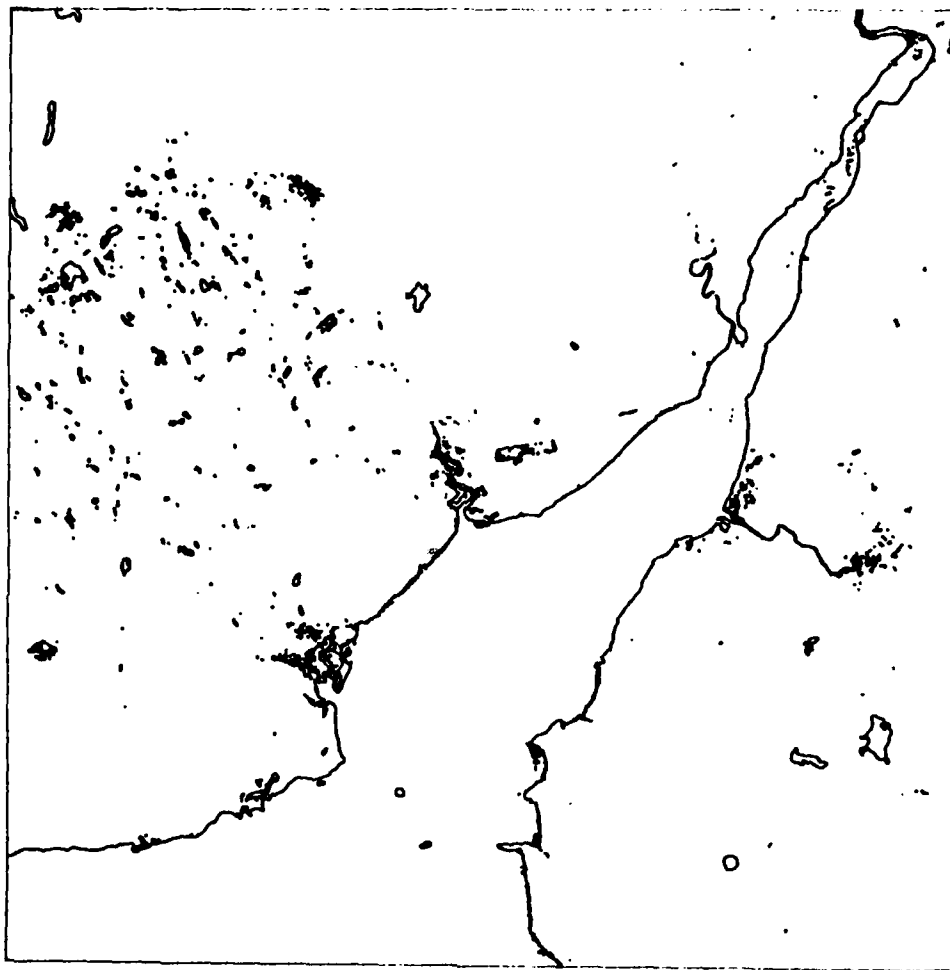


Fig 2 Land-water etc boundaries, extracted from Fig 1 using IM.CONTOUR

Fig 3

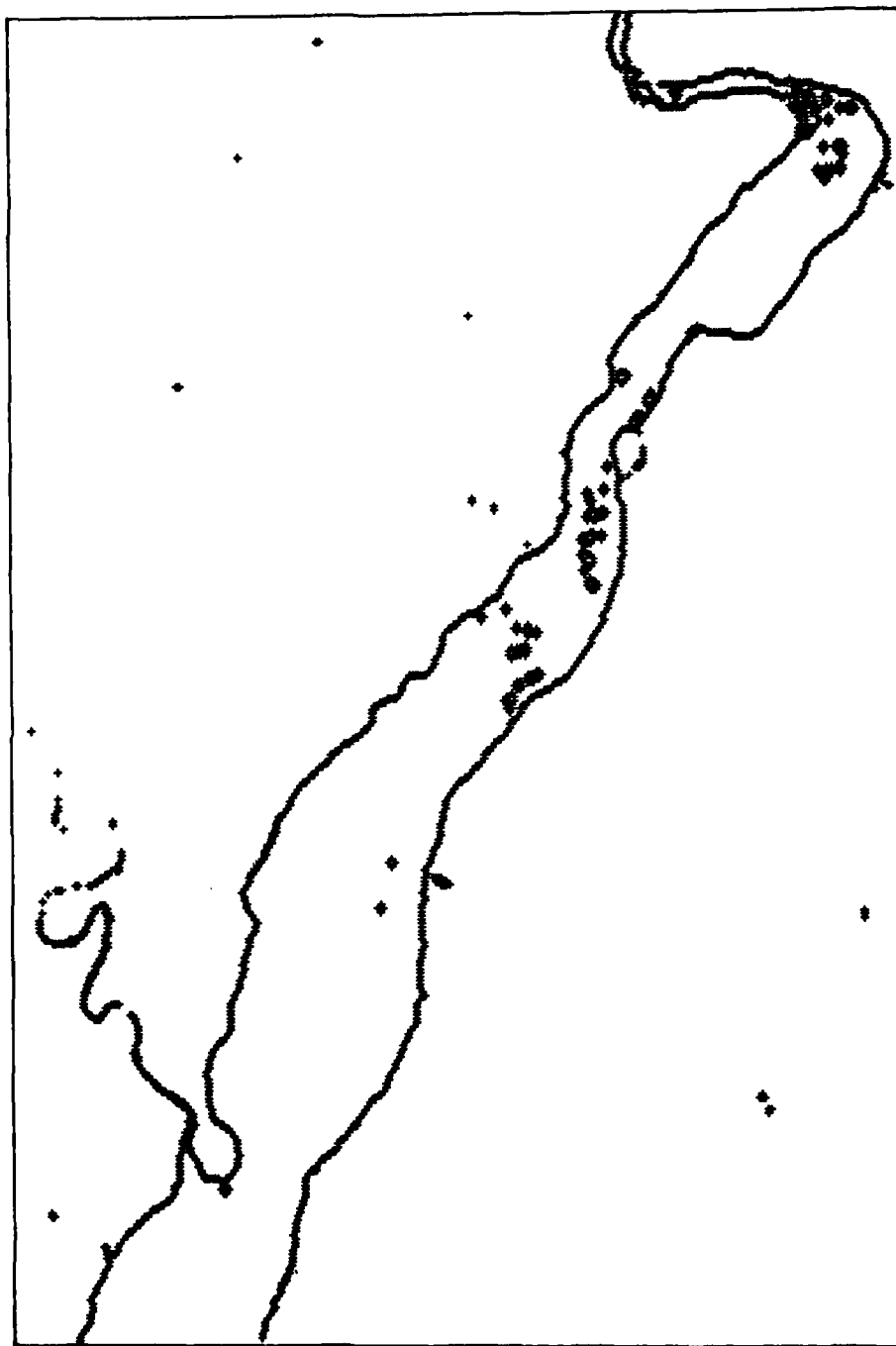
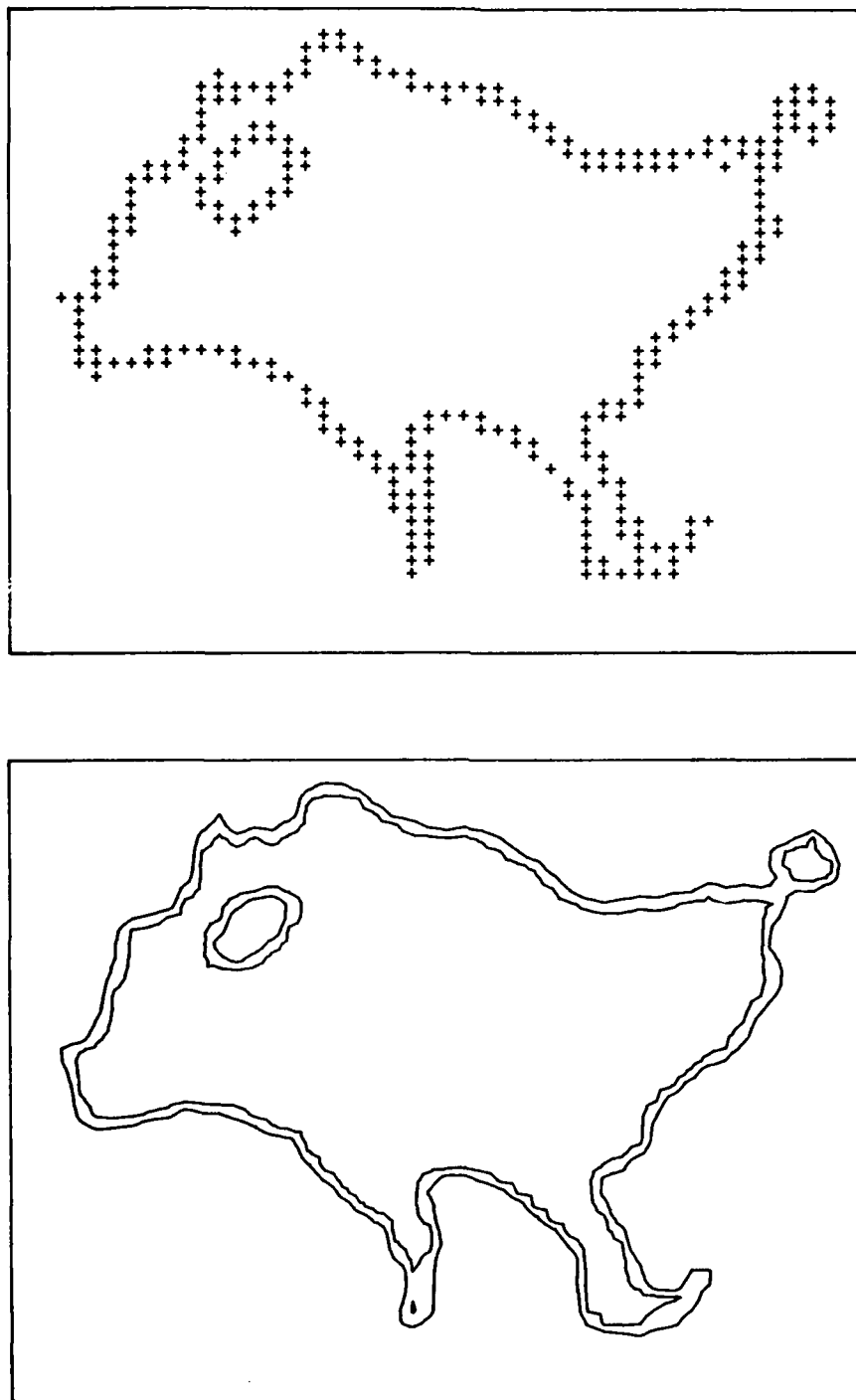


Fig 3 Land-water etc boundaries, extracted from a portion of Fig 1 using IM.CONTIM

Fig 4a&b



b

a

Fig 4a&b Delineation of a small portion of Fig 1 at two density levels, (a) using IM.CONTOUR (b) using IM.CONTIM. Note: these figures are not geometrically transformed

Fig 5

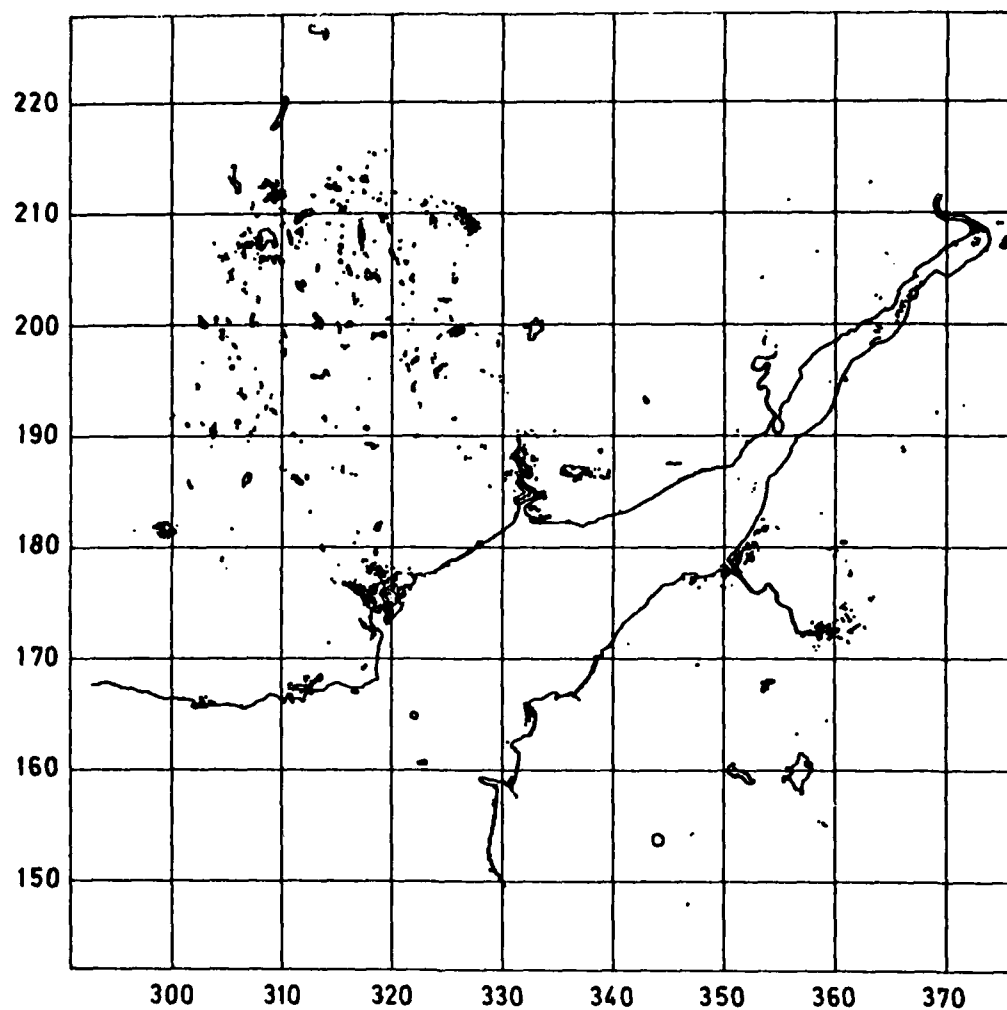
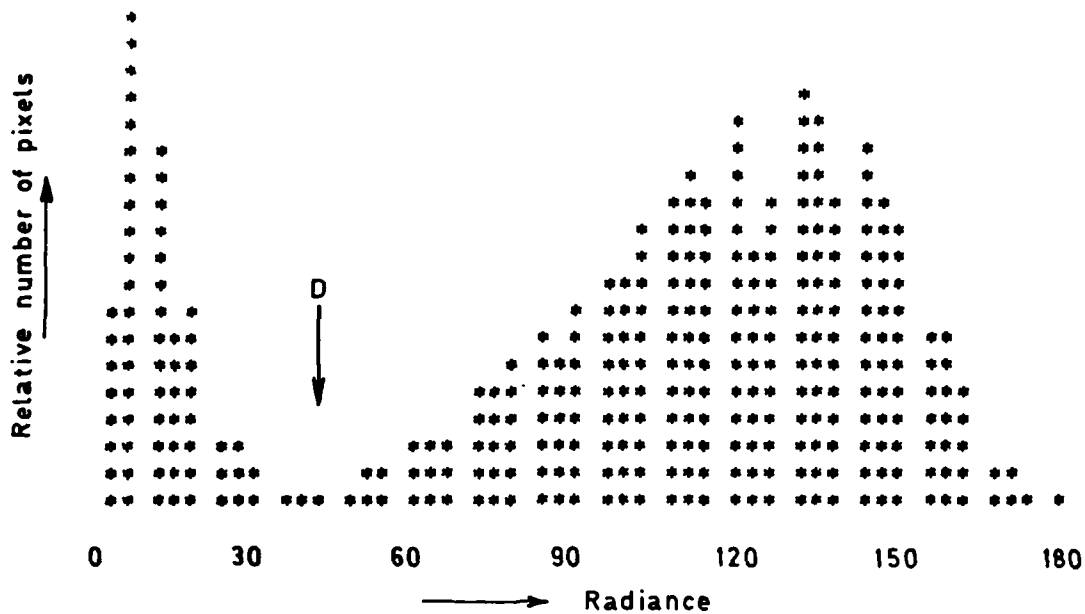
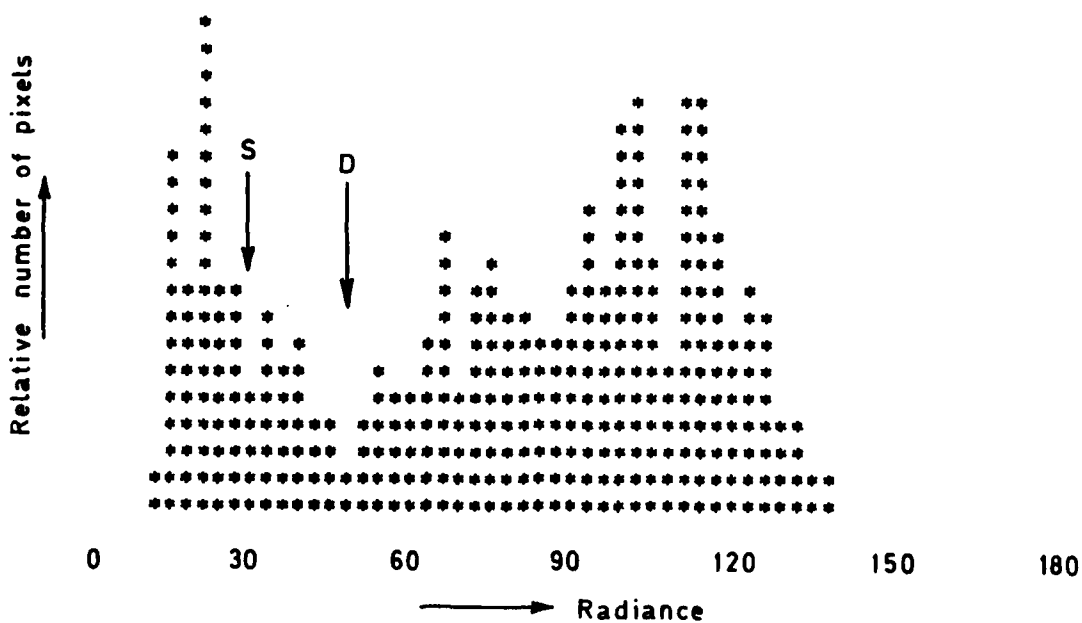


Fig 5 Geometrical transformation of Fig 1 to National Grid, with grid lines superimposed

Fig 6a&b



a



b

Fig 6a&b Histograms of pixel intensity distributions. (a) for portion of image shown in Fig 1. (b) for portion of image of 9 June 1976. For explanation, see text



Fig 7 IM.CONTOUR delineation at two intensity levels of a portion of Landsat 1 image of 9 June 1976

Fig 8

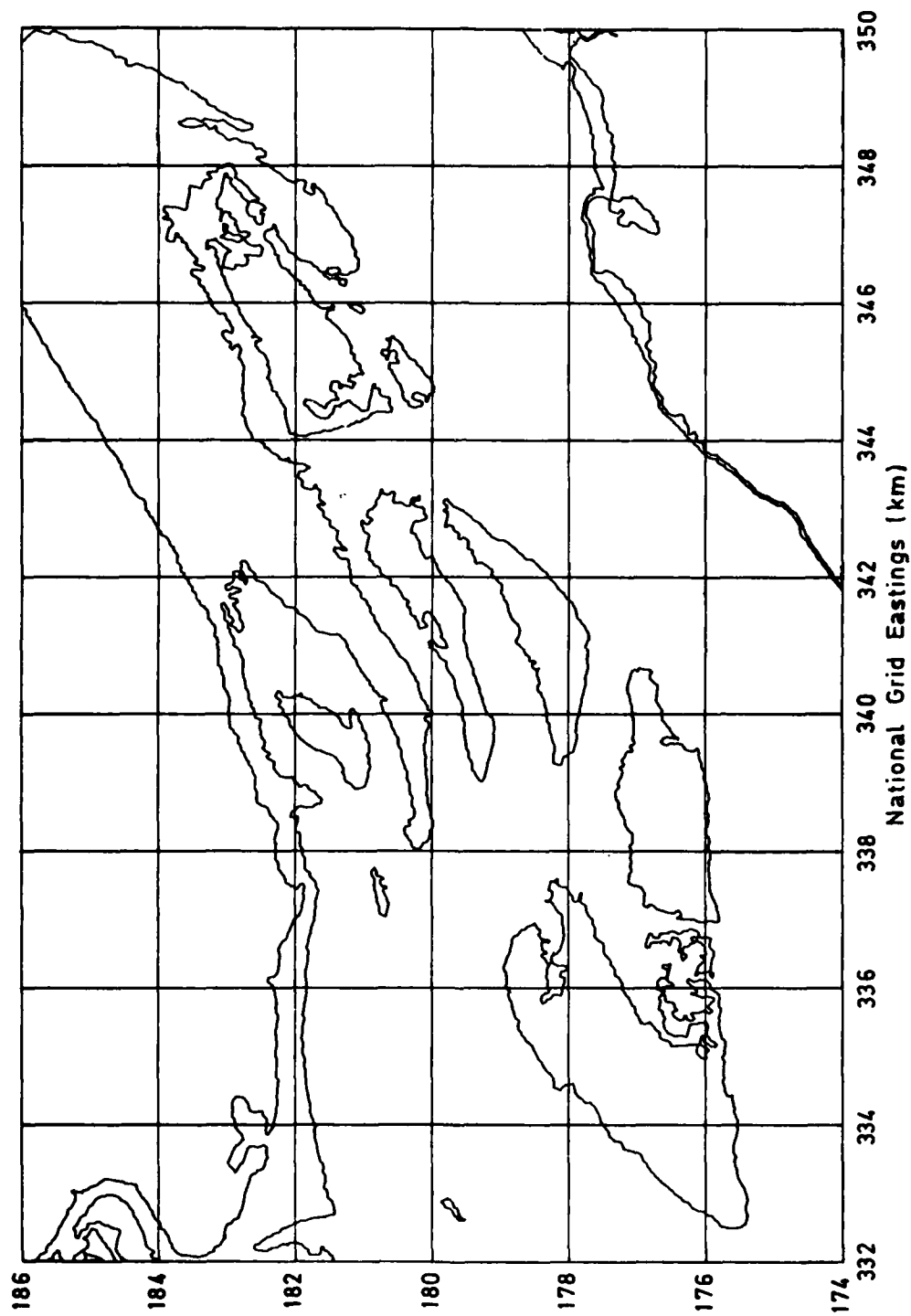


Fig 8 Delineation at two intensity levels of a small portion of Landsat 1 image of 9 June 1976: transformed to National Grid

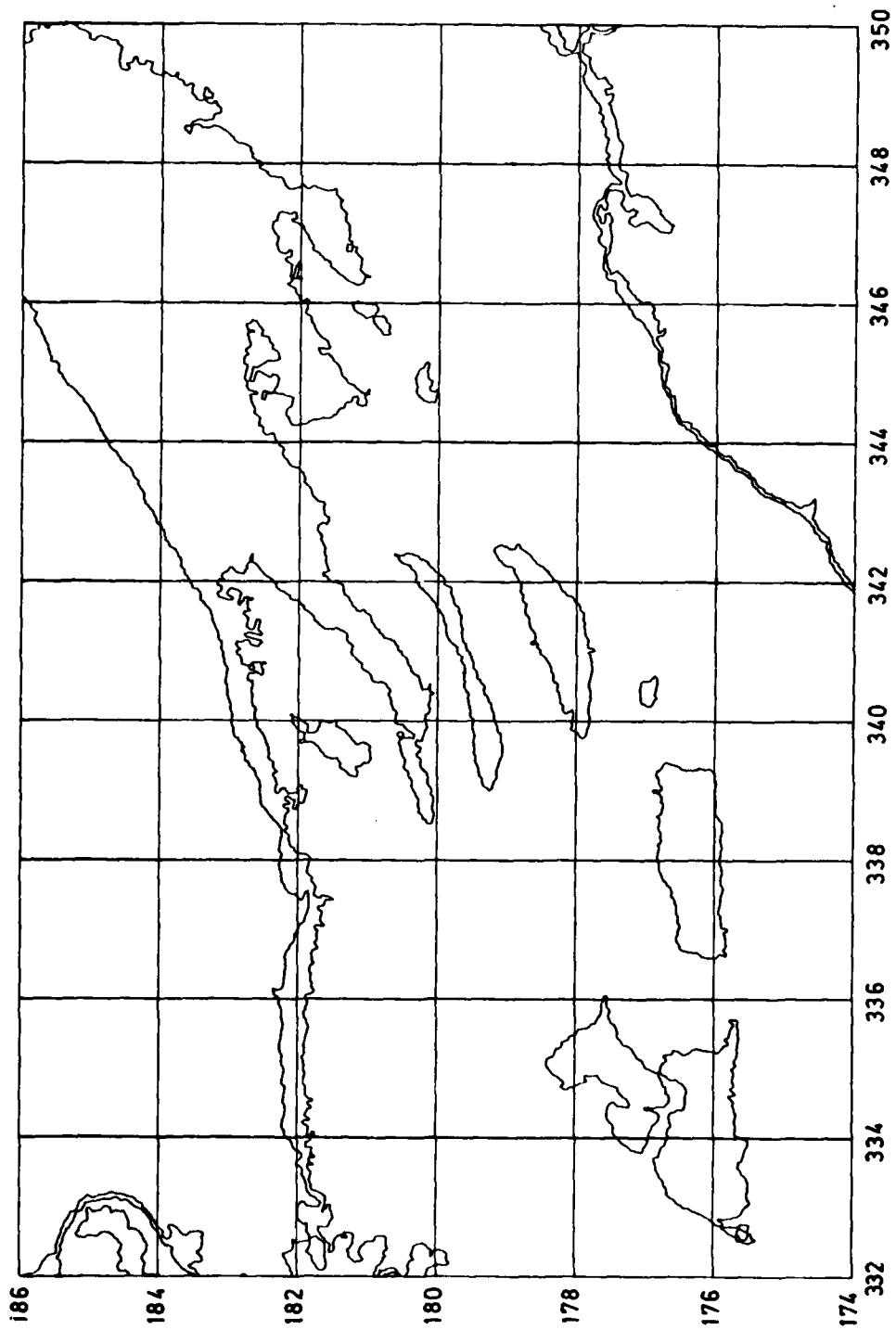


Fig 9 As Fig 8, but for image of 1 October 1975

Fig 10

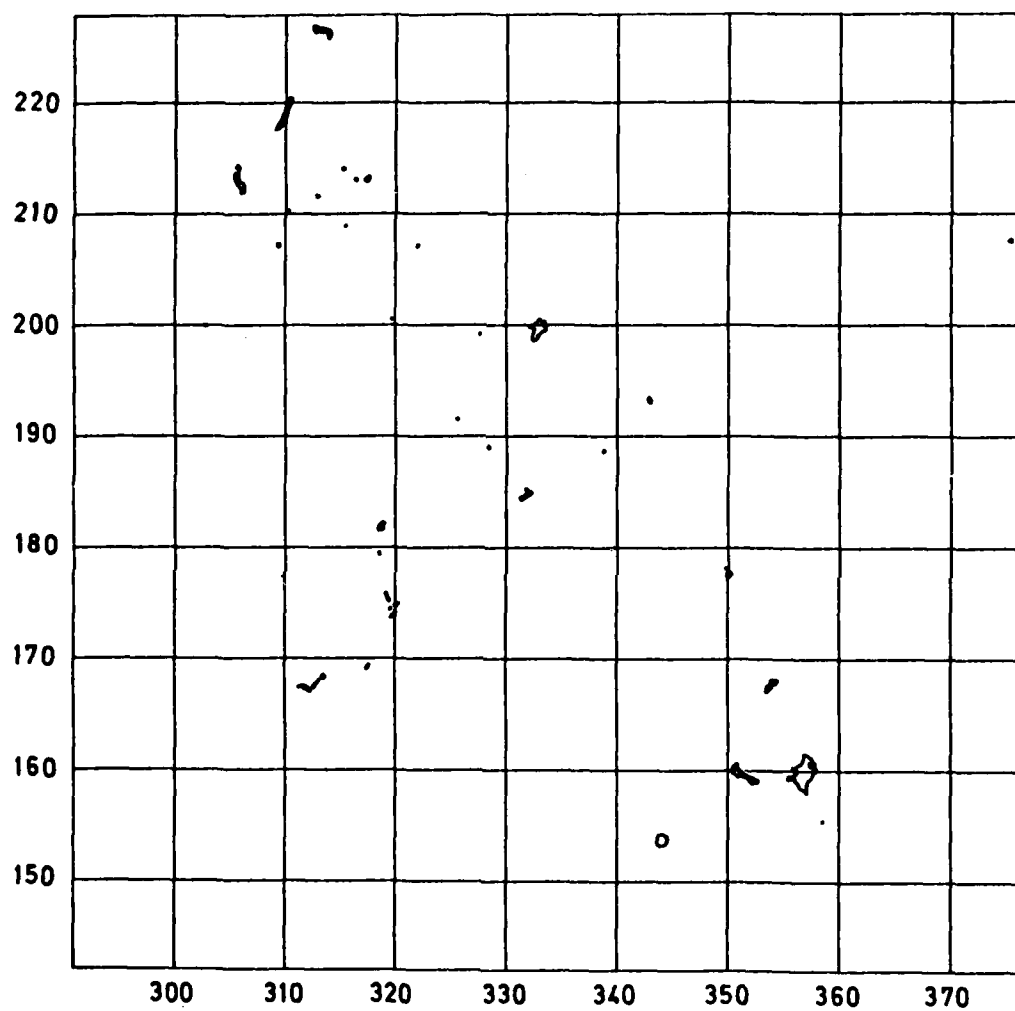


Fig 10 Delineation of areas selected from Fig 1 by all-band classification:
transformed to National Grid



Fig 11 Example of delineation by IM.CONTOUR at several intensity levels.
Portion of band 4 image of Landsat path 217, row 23, 2 July 1975:
transformed to National Grid

Fig 12a&b

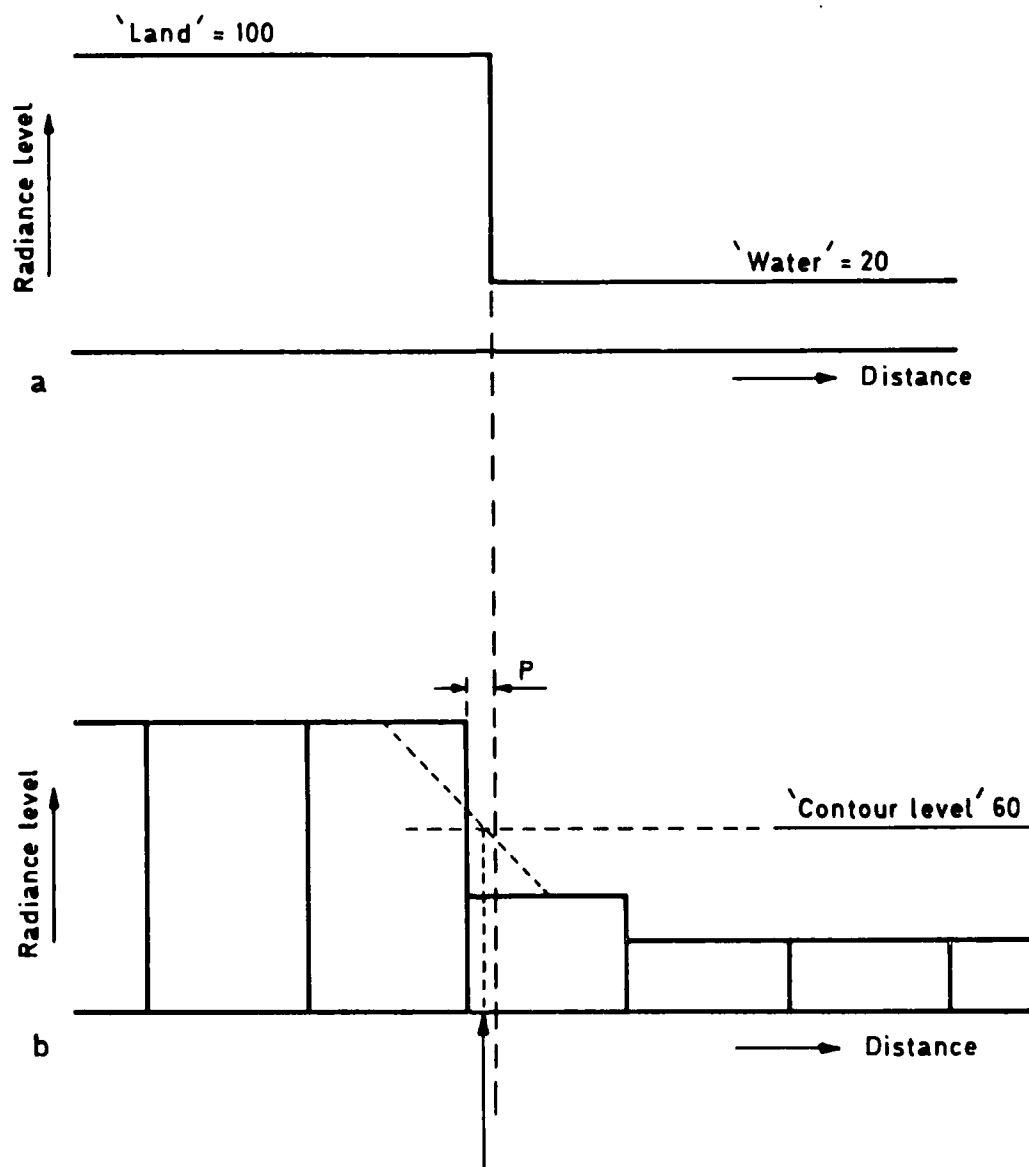
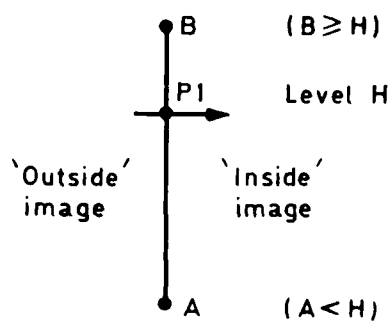
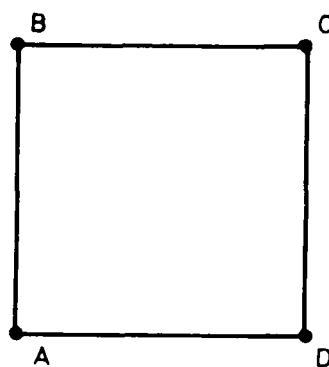


Fig 12a&b (a) idealised variation of radiance level versus distance along a scan, for a land to water transition. (b) pixel values resulting from 12a, with location of the boundary (arrowed) as determined by interpolation



a



b

Fig 13a&b (a) a pixel pair AB . (b) a cell of four pixels constructed on AB .
For details see text

Fig 14

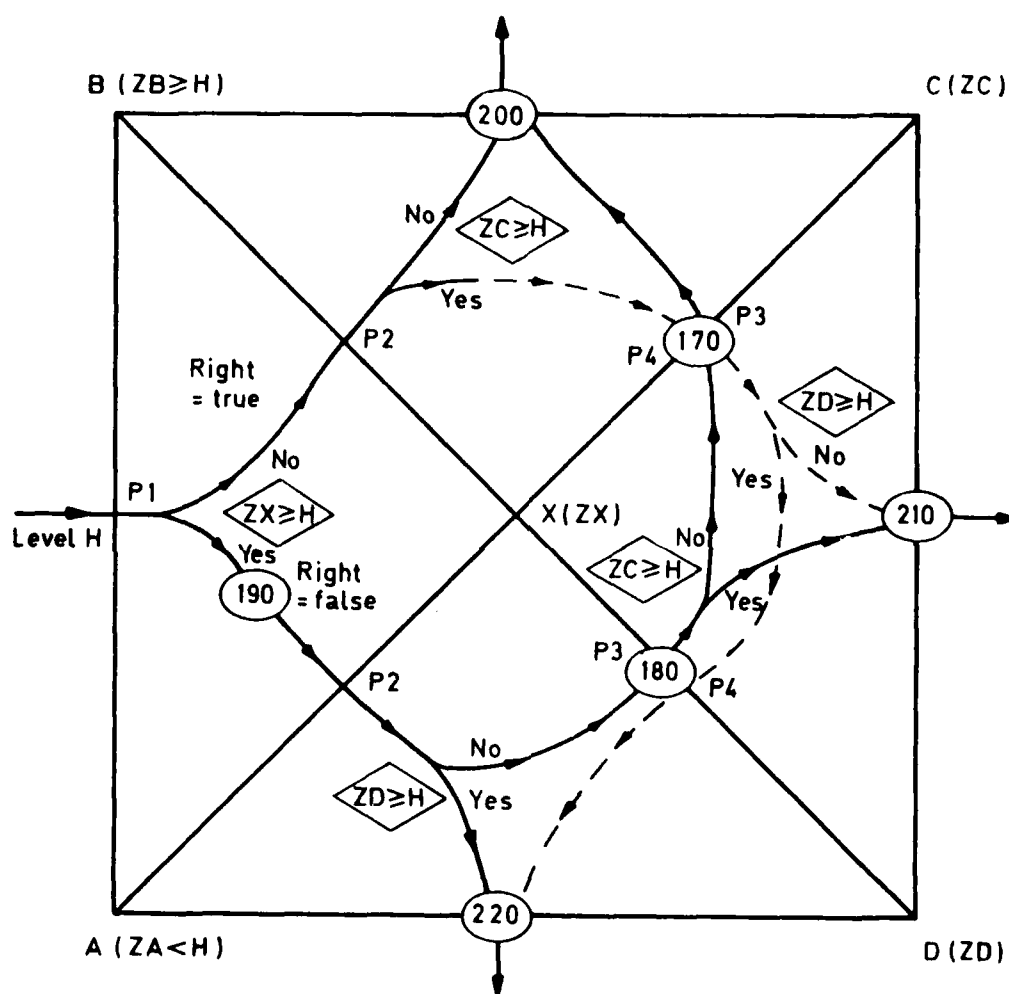


Fig 14 Routing a contour line through a cell of four pixels. For details see text

REPORT DOCUMENTATION PAGE

Overall security classification of this page

UNCLASSIFIED

As far as possible this page should contain only unclassified information. If it is necessary to enter classified information, the box above must be marked to indicate the classification, e.g. Restricted, Confidential or Secret.

1. DRIC Reference (to be added by DRIC)	2. Originator's Reference RAE TR 81010	3. Agency Reference N/A	4. Report Security Classification/Marking UNCLASSIFIED		
5. DRIC Code for Originator		6. Originator (Corporate Author) Name and Location Royal Aircraft Establishment, Farnborough, Hants, UK			
5a. Sponsoring Agency's Code N/A		6a. Sponsoring Agency (Contract Authority) Name and Location N/A			
7. Title A technique for line extraction from Landsat multi-spectral scanner satellite data with some applications of the technique					
7a. (For Translations) Title in Foreign Language					
7b. (For Conference Papers) Title, Place and Date of Conference					
8. Author 1. Surname, Initials Benny, A.H.	9a. Author 2	9b. Authors 3, 4	10. Date January 1981	Pages 40	Refs. 10
11. Contract Number N/A	12. Period N/A	13. Project	14. Other Reference Nos. Space 592		
15. Distribution statement (a) Controlled by - Head of Space Department, RAE (RAL) (b) Special limitations (if any) -					
16. Descriptors (Keywords) (Descriptors marked * are selected from TEST) Landsat. Imagery. Cartography.					
17. Abstract An automated technique is described, for extracting lines from Landsat multi-spectral scanner (MSS) images by means of 'density contour' threading of the data. The resulting lines can then be transformed to fit any map system - in particular the British National Grid - with the aid of selected ground control points. The application of these techniques to the interpretation of Landsat imagery is discussed.					

1/0155

