An Investigation into Sub-Surface Strain Measurement using X-Ray Radioscopy

by

Richard John Drew, B.Eng. (Hons), M.Sc.

Thesis

Submitted in partial fulfilment of the requirements for the award of

Doctor of Philosophy of the Loughborough University.

March 1999

Abstract

There are numerous techniques used to measure strain. Most are only capable of taking surface measurements. The penetrating nature of X-rays has been used to measure deformation, and thus strain, but only with radiographic images. Radioscopic techniques are faster and do not require film processing, but produce less detailed results than digitised radiographic images. The research covered by this thesis tested radioscopic images and showed them to be suitable for strain measurement.

The thesis includes details of the design and capabilities of the radioscopic equipment. Pin cushion distortion is a common feature of radioscopic images, and an automatic method of identifying, and correcting for the distortion was implemented.

A rubber sheet, seeded with tungsten, was found to be a suitable sample for strain measurement. The tungsten particles gave contrast to the radioscopic images, which was used to track the deformation of the material. If the seeds were restricted to known sections of an object with more complex shapes, strain measurements can be made for known regions within the object.

An investigation of pattern matching methods identified a new equation, which the author believes has not been previously used for pattern matching. Therefore, several of the methods were compared to find the most suitable for use with the radioscopic images. Tests were made both with radioscopic images, and with images with simulated deformations. The most accurate results from the two sets of tests used different pattern matching methods, which indicated inaccuracies with the simulation, and that the best choice of pattern matching method is dependent on the application.

Although the radioscopic images were of low contrast and suffered badly from noise, it was possible to measure both small and large (strains of the order of 20%) deformations of the target. The deformation measurement method found movements to the nearest pixel. Improvements in the accuracy of the method are possible by using fine interpolated grids, and by modelling the strains of the target.

Acknowledgements

This thesis would not have been possible without the help and guidance of my supervisor, David Kerr. Mention should also be made of Terry West and Steve Jeffs who assisted with setting up and installing the hardware. Barry Jones of the IPTME assisted by providing the rubber sample. The help of the department's mechanical and electrical workshop staff is also acknowledged for work on the project, in particular the construction of the camera mounts, the distortion phantom and the strain rig. I must also thank the members of the department (particularly Neil Brown, and the rest of the Mechatronics group) for countless examples of help and assistance. Finally I should mention my mother, and Simon Nicholson who were kind enough to try to comprehend and comment on the draft versions of this thesis.

Table of Contents

1 - Introduction

Strain measurement is an important technique for engineers. Applying a force to an object will cause stresses (the measure of force per unit area) and strains (the ratio of the change in length to the original length) within the object. The properties of a material can be determined by measuring the strain under test conditions. These properties can be used to model the behaviour of the material under loading. Using the model of the material's behaviour, the loading on an object can be found by measuring strains within the object, which can give warning of an impending failure.

There are many ways of measuring strain. Traditional techniques, for example strain gauges, have been supplemented by a range of optical methods. These include interferometry, X-ray diffraction, neutron diffraction and digital image correlation. It is sometimes possible to embed strain gauges within a test sample to obtain sub-surface measurements, but the size of the gauge can distort the properties of the object. With opaque materials, the optical methods are only suitable for measuring surface strains.

Depending on the loading and the structure of the object, there is potential for large localised stresses that could result in failure. The failure could be initiated anywhere within the object. This means it is desirable to be able to measure the strain at any point within the object.

Penetrating radiation can be used to measure strains provided that there is suitable contrast in the images of the test piece to allow features to be identified. Radiographic images have been used for sub-surface strain measurement with samples that have a natural structure to provide image contrast, and with samples where the contrast had to be added. Small seed particles of a material that is highly absorbent of the appropriate energy X-rays, have been used to add contrast to radioscopic images.

Radioscopic images are easier to acquire than radiographic images, as they have a video, rather than a photographic output. This means there is a potential for live images, and that it is easier to record a sequence of events. On the down side, radioscopic images are limited by the resolution of the image intensifier and the digitising process. This research investigates how to take advantage of the properties of radioscopic images for sub-surface strain measurement.

Three projects were undertaken in the first year of the research programme. The first was to investigate and improve the capabilities of the radioscopic equipment. A literature search was undertaken to determine possible measurement applications for the radioscopic equipment. Finally a technique was developed to measure and correct for the spatial distortion.

Later research investigated pattern matching methods and their suitability for radioscopic strain measurement. The aim of this research was to identify, accurately and reliably, the deformation of radioscopic targets with a minimum of user input. Deformation was measured with pattern matching methods. These methods take small regions from one image and find the equivalent region in a second image. If the two images are taken before and after the target is deformed, pattern matching can be used to measure the deformation of the target across the images. Strain measurements can be found from the rate of change of the deformation of the target.

For accurate identification of the deformation of the target, precise measurement of the movement of regions in the target is required. Radioscopic images suffer badly from spatial distortion caused by the image intensifier, and in some circumstances the X-ray source. As the distortion varies across the image, accurate measurement of the movement of the target requires the correction of the distortion.

The spatial distortion measurement technique was used to model the distortion using a radial model. The model was used to determine the correct location of points on the images, which improved the accuracy of the deformation measurement.

The major difficulty with the project was found to be the very low signal to noise ratio of the radioscopic images. This resulted in difficulty in tracking the movement of the target, and frequently produced erroneous locations for the best match. A number of pattern matching methods were identified in the course of the research. These were tested with the radioscopic images to find the most reliable deformation measurement method.

It is believed that no comparison of all the pattern matching methods has previously been published. The comparison of the reliability of the different methods will therefore be of interest for any application that uses pattern matching (or one dimensional signal matching).

A test rig was constructed to compare the deformation measurement methods. The rig could apply strains to a rubber sample. The rubber sample was seeded with tungsten particles to provide a contrast in the image. Although tungsten is one of the best potential seed materials for the low energy X-rays being used in the tests, the equipment was not able to provide images that used the full grey level range, due to the small size of the particles.

The choice of the rubber sample allowed the pattern matching methods to be tested over a wide range of strains, and to find limits to the usefulness of the methods. As the deformation method was optimised for use with low signal to noise ratios, it should be suitable for a wide range of applications. These would include examples that exhibit a small quantity of natural contrast in radioscopic images, and images where small seed particles are required to reduce the effect on the material's properties.

This thesis covers research into the implementation of radioscopic strain measurement. It shows the technique can be used to determine the deformation of a test object. Given a suitable application, with sufficient radioscopic contrast in the appropriate region of the target, the technique would be useful for strain measurement.

2 - Literature Review

In the course of this research, a number of subjects were identified which required a thorough investigation. The investigations included literature surveys to review the current state of research. This chapter gives details of the current state of research in three areas relevant to this thesis.

The first of the literature reviews was to determine suitable uses for the available radioscopic equipment within the remit of the department's activities, and is to be found in section 2.1. In addition to identifying the uses of X-rays, particularly in industry, this review includes a list of alternative equipment, and techniques for improving results. A number of topics were identified where the available equipment could make a contribution. Sub-surface strain measurement was considered the most suitable for further investigation, as the equipment available was considered suitable for its implementation, and strain measurement has not previously been attempted with radioscopic equipment.

To improve the accuracy of measurements in radioscopic images, spatial distortion needs to be corrected. Section 2.2 is a review of methods to correct image distortion. A number of papers were found on the correction of pin cushion distortion caused by image intensifiers. Details of the modelling and correction of distortion in cameras and other optical devices are covered in this section. The papers on image intensifier distortion provided some helpful distortion models, and suggested a number of different designs of phantom (an object of known shape used to measure the distortion). Unfortunately, none of the papers found gave details on automated methods of identifying features on the distortion measurement phantom, so chapter 4 includes details of new software written to identify the phantom details.

Section 2.3 of the literature reviews looks at pattern matching methods capable of measuring spatial deformations between images. These can be used to measure the strains applied to a radioscopic target. This review looks at numerous papers that use two dimensional pattern matching and lists the equations that have been used. Of the equations found, a number appear to be suitable for implementation with a modern computer, but no paper was found that compared the suitability of all the methods. Thus, chapters 6 and 7 include trials of the equations in order to find the most suitable.

2.1 - Radiological Non Destructive Testing

Since X-rays were discovered by Roentgen a hundred years ago, their properties have been used in various industrial measurement techniques. In this thesis, radiology is used as a term that encompasses both radiography, which uses photographic detection methods, and radioscopy, which uses an image intensifier to convert the radiological image to an image in the visible spectrum. The most common use of radiology in industry is to detect defects, particularly in welds or castings [1].

Traditional X-ray sources are radioactive materials that provide monochromatic X-rays (also referred to as Gamma rays), and X-ray tubes that provide a wide spectral range. Gamma ray sources are either point sources, or sometimes flat sheet sources placed in contact with the test object [2]. Alternative X-ray sources include flash X-ray tubes, Van de Graaff generators, betatrons, linacs, microtrons, synchrotrons and X-ray lasers.

An X-ray tube consists of a filament located in a cathode assembly. This accelerates electrons towards an anode, where most of the energy is converted into heat, and a small quantity is converted to photons in the X-ray energy band. The minimum wavelength (maximum electron energy, keV) of the emitted electrons (and photons) is determined by the voltage used in the tube. The lower energy photons are attenuated to a greater extent by air than higher energy photons. This effectively provides a lower limit to the X-ray energy.

The X-ray energy spectrum produced by an X-ray tube has characteristic peaks superimposed. These are dependent on the atomic structure of the anode material. For example, a tungsten anode produces peaks in the 57keV to 69keV region, which correspond to the energies required to move electrons between the element's orbits. A typical tube efficiency of less than 1% indicates that a major design requirement is to remove heat from the anode. The radiation intensity varies with the inverse square of the distance from the X-ray tube. Some X-ray tubes allow both the tube voltage and current to be varied independently, although these two variables are normally dependent [3].

Traditionally, radiological images were originally recorded using radiographic techniques on photographic paper. Fluorescent screens can be used to convert X-rays images to visible images. Image Intensifiers [4] use a fluorescent screen to provide a radioscopic image. The image is converted to electrons using a photocathode (the input phosphor). The electrons pass through a vacuum, where they are focused using electrodes, and targeted at the output phosphor screen. The output window produces an image several orders of magnitude brighter than the output of the fluorescent screen. As the output is in the visible spectrum, the image can be recorded with normal cameras (often a video camera is used to give a live image).

Variations in the basic radiological system, include linear and flying spot X-ray sources, which are used to reduce scatter noise. Alternative types of detectors include photo-stimulable phosphors, and electrostatic detectors [5]. It is possible to increase the resolution of radioscopic images by shifting the test piece by a known amount, and taking additional images [6]. However, the results will be affected by spatial distortion, and the accuracy of the measurement of the sample's displacement, which means higher resolution cameras should be used if available.

A number of methods are used to improve the quality of radiological images. Dye enhancement, where cracks and voids are filled with an X-ray opaque material to improve contrast, is popular with composite and ceramic materials [7]. The signal to noise ratio can be increased by eliminating scattered radiation [8, 9], for example by employing an oscillating lead grid called a Bucky grid between the specimen and the detector [4]. An intensifying screen [10] is a layer of fluorescent material placed next to the film to enhance the brightness of radiographic images, and to reduce the X-ray exposure.

Another image enhancement technique is to use filters to modify the X-ray spectrum. A lead sheet between the test piece and the detector is used in some circumstances to improve contrast, but obviously requires a longer exposure [11]. Filtration between the source and patient is popular in medicine, where a harder X-ray source will reduce the exposure to the patient. Materials tested include aluminium, yttrium [12], and niobium [13].

Radiographic images are dependent on the intensity of the radiation passing through the sample and interacting with the detector. The darkness of the detected image is affected by the composition of the sample, its thickness, and the energy spectrum of the source. Each element has an absorption coefficient that is dependent on the energy of the radiation. The variation in the absorption coefficients means it is easier to distinguish between components made from different elements at some X-ray energies than at other energies. This property can be used to highlight extra detail, by combining radiographic images taken at different energies into a colour image [14], and has potential for determining the elements in a sample [15].

The metric unit used to measure the radiation absorbed dose is the "gray", and measurement of the biological effect is the "sievert" [16]. The biological effects of radiation [17] have resulted in limits being set for working with radiation [18, 19, 20].

X-ray images have been subject to image processing by both optical and digital methods. Optical methods project the radiograph using a monochromatic light source. Using a series of lenses, Fourier transforms can be performed on the images by placing filters at the correct locations in the beam. This technique has been used to produce a high pass optical filter that was used to help to detect the edges of cracks in weld radiographs [21].

Digital image processing of X-ray images has for many years used digitised radiographs as the source material. More recent applications have been able to take advantage of digitally sampled radioscopic images [22]. Techniques in use included high pass filters to detect edges, Wiener filters to reduce noise and histogram functions to improve contrast [23, 24]. Combining radioscopy with suitable image processing techniques, it is possible to undertake automatic real time defect detection [25]. Digital image processing includes many other techniques, and has many applications other than X-ray imagery [26, 27, 28].

X-rays will either pass through, be absorbed, or be deflected by an object. Deflected (diffracted or scattered) radiation results in a degradation of radiographic images, but deflected radiation can be used to make other measurements. Bragg diffraction on highly polished silicon crystals has been used to magnify monochromatic X-rays [29]. Very precise cutting and alignment of the crystals is required to produce a magnification of the X-ray beam of between 20X and 50X. A new field for X-ray diffraction research is capillary optics [30], which can produce a very narrow (1-5μm) and intense X-ray source.

The diffraction of X-rays can be used to measure stress in some materials [31, 32, 33]. The technique relies on the loading causing a deformation of the lattice structure of the material. The lattice structure will diffract X-rays in a pattern dependent of the shape of the lattice. Surface stresses can be calculated from the diffraction pattern.

The technique requires a highly collimated beam of X-rays of a suitable wavelength and detectors placed to measure back scatter. As only surface stresses can be found, the surface has to be peeled away to find internal stresses. It is possible to correct mathematically for the removed material [34]. Recent papers have used the technique with ceramics [35] metal matrix composites [36] and plasma sprayed coatings [37].

The similar technique of neutron diffraction [38, 39] has also been used, particularly with metal matrix composite materials. Comparisons between the two methods [36, 40] come down in favour of the neutron method, as it is able to provide a more detailed result. However, there are technical problems in providing a source of energetic neutrons.

There are many examples of composite materials in the aerospace industry. Many types of failures occur in composites, and some can be detected using a wide range of non destructive testing techniques, including radiology [41]. The uses of radiography include checking for water ingress in honeycomb structures, the consistency and coverage of adhesives [42], the density of components [43], cracks [44] and fibre damage [45]. Radiographs of metal components normally use X-rays in the 80keV to 200keV range. Because of the lower atomic numbers of the elements used, composite materials tend to have a better contrast with lower power X-ray systems in the 5keV to 50keV range [10, 46].

Impacts can cause damage to laminated composite materials. This damage results in broken fibres and de-bonding that is not always visible from the surface. In order to detect the extent of the damage, the damaged area can be soaked in an X-ray opaque penetrant. This will fill the de-laminated areas, allowing the boundary of the damaged area to be found. If the penetrant is dried, it is possible to find the damage in each lamination by de-laminating the sample [45]. The visualisation of cracks in composite materials using radiography is also improved by using penetrants [44].

Radiographically opaque fibres have been used as markers in graphite epoxy prepreg tapes [47]. The marker fibre consisted of a 2.54μm tungsten core in a 0.1mm boron fibre and was detected using Kodak type M film at 25keV. The marker on the edge of each tape was used to check for errors in manufacture, and foreign object damage could be detected, provided some of the boron fibres are displaced or broken.

The darkness of radiographs has been used to determine the thickness and density of composite samples [43, 48]. In particular, the darkness of a radiograph has been used to find the resin content in a graphite fibre composite [49]. This technique took advantage of the different absorption profiles of the fibres and resin. The paper calculated the absorption coefficients at different X-ray voltages, over a range of fibre to resin ratios. Depending on the X-ray energy, the ratio of the absorption coefficients will increase or decrease, as resin content increases. At some energy levels, the resin will have the higher absorption coefficient, and at some energy levels the fibres have the greater absorption coefficient. The resin content is calculated by comparing with a step wedge.

Fibre composite materials can produce fringe patterns, produced by the interference between different plies [50]. The technique was used to measure the angles between two plies, to find the fibres per inch in the tape, and to measure filament wandering. The paper only mentioned examples with two layers, and it is thought the method would be unsuitable for use on composites with additional layers.

Some composite materials include easily identifiable components. For example, tyres often include reinforcing wires, which can be easily seen in radiographic images. Radiographs have been used to spot faults like missing or crossed wires [51]. Equipment has been developed to measure the deformation of a tyre under road conditions [52]. It consists of a 45keV X-ray source mounted within the wheel's hub, and radiographic film mounted flush with the road.

The steel reinforcing in reinforced concrete can be identified in radiological images. Radiography has been used to measure the corrosion of steel reinforcing bars within concrete [53]. The image density across reinforcing bars in areas with no corrosion, was found to have a sharper transition, than areas with corrosion. A method of quantifying the sharpness of the transitions was proposed and implemented. The technique is able to indicate likely areas of corrosion, but the method used to measure the corrosion appears to be dependent on the operator making consistent measurements.

Another use of radiography has been to measure internal strains [54]. This paper looked at microscopic strains in composite structures. Gold particles of 10-40μm in diameter were embedded within a graphite epoxy composite. A double exposure radiograph was taken of .he composite using a synchrotron source. One exposure was with, and one was without the sample under strain. By measuring the distance between the pairs of impressions from each gold particle, it was possible to measure the strain at that point in the composite. The paper gives a reference to microradiographs distinguishing objects of less than 0.0050μm.

Strains have also been measured in bones using radiography [55]. This used the natural texture in Trabecular bones to give contrast to the radiograph. The radiographs were taken for 55 seconds at 30keV and digitised with a Pulnix TM-745 video camera. Strains were measured using the normalised correlation equation (see section 2.3).

Microradiography has been used to detect many types of microscopic features [56, 57], and is capable of giving very fine detail. Methods of producing a suitably focused X-ray source have included optical focusing, pin hole sources, and electromagnetic focusing. The structure of ceramic materials has been examined using microradiograhy [58]. In this case, the focal spot was of around 100μm, which allowed the detection of defects down to 25μm, and of voids of 1% of the sample's thickness.

The properties of X-rays have often been used to visualise otherwise hidden details. Radioscopy has been used to measure the viscosity of opaque suspensions [59] by tracking the motion of a ball falling through the suspension. The technique used lead bricks to collimate the X-ray source. The image intensifier had a choice of 4" 6" and 9" images, and the camera was capable of up to 2000 frames per second, although only 60 to 200 frames per second were used.

Internal displacements within assemblies have been measured before and after thermal and dynamic tests [60]. The technique used a reference object placed in the same plane of the assembly to compensate for geometric distortion.

Radiographic markers have their uses in material processing. For example, information about the structure of paper has been determined by marking a proportion of the fibres used to make the paper, with a radiographic marker [45].

Tomography uses multiple radiographic images to map the shape of a sample. A simple version [61], used two radiographs to map the shape of voids and inclusions in welds. Laminography is the measure of a cross section of a sample. It can be achieved with an X-ray source positioned at an angle to the imaging plane [62]. The sample is placed in the X-ray path, and is rotated synchronously with the image plane. The result is that only one slice of the object is focused on the image plane throughout the exposure, which provides a blurred image of the slice. Whilst the paper described the use of hardware to rotate the image plane, software could be used in conjunction with a static detector.

Of the techniques listed in this section, some are more suitable than others for implementation with the radioscopic equipment available in the department. With the equipment's fixed configuration, surface strain measurement and other diffraction techniques would be unsuitable for implementation. The X-ray output of the equipment is more suited to imaging composite materials than metallic objects. Also methods that require a known, or a monochromatic X-ray source can also be discounted, whilst the geometry of the equipment would make viscosity measurement difficult.

The equipment could however be used to detect some defects, but the results would be specific to the problem, and would require a suitable sample that requires inspection. With a range of X-ray energies available, either colour radioscopy or an attempt to determine chemical composition are a possibility. However, the output of the equipment was found to be badly attenuated at lower X-ray energies, which would limit the implementation of these techniques. Laminography, and shape measurement are possible, given a suitable method of manipulating the test piece in the X-ray chamber. However, there would be a need to process large numbers of images, which would involve a heavy computational workload unsuitable for the 286 computer that operates the equipment.

Further investigations were made on the sub-surface deformation measurement, which did not requiring any alteration to the radioscopic equipment. The two papers discussing strain measurement both used radiographic techniques, which means radioscopic strain measurement is a relatively new field and ripe for further investigation. The inherently low contrast of the radioscopic images suggested seeded materials were required to provide contrast in a test sample, as used in the first paper [54]. It was proposed to use separate images for the deformed and un-deformed states, and use pattern matching to measure the deformation, as used in the second paper [55].

The research requires a suitable sample that can withstand measurable deformations. The sample would have to be thin enough to allow the penetration of the low energy X-rays and would need to provide contrast to radioscopic images that could be used to measure deformation. A rig would be required to apply strains, and a method was needed to measure the displacements used to calculate the strain. The solutions to these requirements are covered in the rest of this thesis.

2.2 - Distortion Correction

Strain measurement using the above technique requires accurate measurement of the location of features on the radioscopic images. The distortion of radioscopic images means the magnitude of a viewed displacement is dependent on the location of the feature. To improve the accuracy of radioscopic strain measurement, the distortion needs to be corrected [63]. Image intensifiers are the major source of radioscopic distortion, although with three dimensional objects, the distance to the X-ray source is also an issue.

Most of the spatial distortion of the radioscopic system is pin cushion distortion, where the distance between neighbouring pixels increases towards the edge of the image intensifier. Pin cushion distortion is caused principally by the curvature of the photocathode, although the electron optics also have an effect. There is also some “S” distortion [64] caused by the earth’s magnetic field interacting with the electron beam in the image intensifier.

Most methods of removing spatial distortion identify the locations of features in the distorted image that have a known location in the real world. The features are used to produce a model that can be used to find an undistorted version of the image. The features are frequently arranged in a grid pattern. The location of a number of nodes in the grid can be used for local area mapping. Alternatively, a model covering all, or part of the image, can be calculated. This is often done by minimising an error function. The global nature of pincushion distortion makes the distortion in radioscopic images suitable for correction with a model that covers the whole image. Targets used to measure distortion, known as phantoms, have been used with square, triangular and hexagonal grids.

The pincushion distortion (and intensity variation) has been measured as a function of the distance from the centre of the image intensifier [65]. This was extended [63] to compare the effectiveness of a number of global models used to correct the distortion. Models compared were the one parameter model:

$$r' = r + Dr^3$$

(2.1

the two parameter model:

$$r' = r + D_t r^3 + E_t r^5$$

(2.2

and the spherical model:

$$r' = r\frac{(1 + z / s)}{(1 - z / R_0)}$$

(2.3

where:

$$z = R_0 \frac{\left(1 - \left[1 - (r / R_0)^2 - 2(r^2 / R_0 s)\right]^{1/2} - r^2 / R_0 s\right)}{1 + (r / s)^2}$$

R₀ is the radius of curvature of the surface of the photo cathode, **r'** is the measured distance of the object from the centre of the distortion, **r** is the distance without distortion, **s** is the source to image distance, **D** and **E** are parameters. The models were tested on four designs of image intensifiers, using a commercial grid phantom to create the images. The phantom consisted of 2mm diameter holes drilled, 4mm centre to centre, in a hexagonal pattern to .001" accuracy. The errors between the models and the experimental results were compared. The results showed the two parameter model was the most accurate.

A twelve parameter global model has been used for distortion correction [66]. The parameter values were found by minimising the root mean square of the difference between the true location of known calibration points, and the location measured with the radioscopic equipment. In this paper, the phantom was made of lengths of wire mounted in a square grid pattern in a sheet of Perspex. The model used was:

$$X_P = a_1 + a_2 x_p + a_3 y_p + a_4 x_p^2 + a_5 x_p y_p + a_6 y_p^2$$

(2.4a

and

$$Y_P = b_1 + b_2 x_p + b_3 y_p + b_4 x_p^2 + b_5 x_p y_p + b_6 y_p^2$$

(2.4b

where $\mathbf{X_p}$ and $\mathbf{Y_p}$ are the true location of the object, **x** and **y** are the location of the object on the radioscopic image, $\mathbf{a_p}$ and $\mathbf{b_p}$ are the parameters of the model.

The “S” distortion is a cause of concern with mobile equipment, or equipment where the image intensifier is moved. As “S” distortion is dependent on the local magnetic field, moving the image intensifier changes the distortion. A model has been implemented for correction of tomographic images, which suffer from variations in the distortion as the image intensifier is rotated [67]. The model takes account of the orientation of the equipment. It is noted the technique also includes additional shielding of the image intensifier.

An alternative to measuring the distortion using an image of a phantom, is to calculate the distortion geometrically. A geometric distortion model has been calculated which includes the distortion caused by using an X-ray point source [68]. This model assumes the image intensifier has a spherical input phosphor screen, which for most image intensifiers, is a simplification. The authors also corrects for shading using a geometric model.

The shape of the input phosphor screen has been measured with a phantom made of a block of aluminium with orthogonal grooves cut into it [69]. The radioscopic image of the phantom was projected onto the image intensifier, and images were recorded of the input phosphor, taken at different angles. The intersections of the lines were identified manually and used to determine the shape of the phosphor.

Another design of radioscopic phantom that has been used for distortion measurement is a rectilinear grid of 2mm ball bearings mounted 20mm apart [70]. The grid was positioned at 45° to the axis of the camera to prevent potential difficulties with zero and infinity slopes. The grid was found by identifying a row of nodes towards the centre of the image, and using it to define an ideal grid. The imaged grid was matched to the ideal grid by using the nearest neighbour method. At the edges, the grid was extrapolated to square the grid. The grid was divided into triangles, and local mapping with sub-pixel interpolation was used to produce a corrected image. The authors also considered dc bias, the non linearity of the camera, veiling glare and shading. Shading was compensated by dividing the image pixel by pixel, by the average of four images taken with no object in the field. Four images were used to reduce noise.

A phantom made of a grid of ball bearings was used in a "high speed" distortion correction technique [71]. This method had an automatic grid detection process capable of detecting grids of varying size. Unfortunately the paper does not include any details of how the centres of the ball bearings were detected. The method mapped rectangular areas to the corrected image, using a fast array processor. Modern computers are now able to undertake the mapping process at a higher speed, making an array processor unnecessary in the majority of situations.

Radioscopic images are not the only ones to suffer from distortion. Video cameras can cause spatial distortion [72, 73]. Some images require correction, particularly with images used for aerial and satellite mapping [74, 75]. Other items, for instance windscreens [76] are also sources of distortion.

The television images from the spacecraft Mariner 9, were an early example of images that had distortion removed by computer [75]. The videcon cameras were fitted with reseau marks; a series of metallic squares located on the surface of the photosensitive surfaces of the cameras, in a triangular grid pattern. As the locations of these marks were known precisely, and they showed up on the camera's images, they could be used to remove distortion variations between individual frames. Reference was made to an algorithm to locate the reseau marks automatically. A linear two dimensional transform was used to map pixel locations from the original to the corrected image. The pixel values were found using bilinear interpolation. The Mariner 9 images were also processed to reduce residual images and non-linearities in the camera's response.

A more recent paper looked at the geometric distortion produced by a range of video cameras [72]. An image was taken of a grid of dots. The image was thresholded to identify the dots and connectivity analysis was used to find the centres. The grid nodes were used to find a locally affine model of the distortion. The accuracy of the model was improved by measuring the exact location of each dot, rather than assuming a perfect grid. The paper mentions one camera whose properties changed with time as the camera warmed up.

Global models have been used to determine camera distortion with a square grid of lines [73]. Intersections of the lines were found manually with the help of a 20 by 20 mask. Software computed the intersection to sub-pixel accuracy, by identifying the lines on either side of the intersection. The method was dependent on the grid being square to the camera, and assumed a certain line width. The grid locations were used to find the values of the coefficients of the model by the least squares method. Tests were made with different numbers of parameters in the model (equation 2.4). It was found that the model with only the first five coefficients in each equation was marginally better than the model with the sixth parameter included.

When mapping a corrected image from the original image, the location in the original image corresponding to the location of a pixel in the corrected image, rarely corresponds to the location of a pixel. There are a number of techniques to find the grey level of the image between pixels [74]. The nearest neighbour method takes the value from the nearest pixel, whilst the bilinear interpolation method uses the four nearest pixels. The cubic convolution method uses 16 pixels and a cubic spline approximation of the **(sin x)/x** curve. Other methods of sub pixel interpolation are also available [77].

After reviewing the literature, a measurement-based distortion correction method was considered preferable to one based on the geometry of the image intensifier, as it would correct for distortion introduced at any stage of the imaging process. For simplicity, an aluminium phantom with holes drilled in a square pattern was chosen. This was considered the easiest type of phantom to design and construct. Holes were chosen in preference to grooves, as the centres were expected to be easier to detect. It was noted that a grid at 45° to the camera axis made it easier to measure the slope of the grid lines, which is potentially an advantage depending on how the grid was detected. However this was discounted as it involved extrapolating a large number of grid nodes to complete the grid.

The literature search failed to find much information on how to automatically identify the structure of the holes in the phantom for equipment fitted with a zoom lens. The most helpful pointer was the mention in one paper of the use of thresholds and connectivity analysis to find the shape of the holes. This meant most of the software written to calibrate the phantom images was totally new. Because of the nature of the distortion, most of the papers dealing with modelling image intensifier spatial distortion used radial models. It was decided to use the radial model, but to expand the model to test the effect of adding additional parameters to the equation.

2.3 - Pattern Matching Methods for Measuring Displacements

Strains within an object can be measured by tracking the movement of features of the object, as forces are applied to the object. With digitised images, this can be undertaken with pattern matching methods. Pattern matching displacement measurement relies on matching part of one image with potential locations in a second image. The displacement is used to determine the location for the best match between the two images. The matching of digital images requires large amounts of computer processing power, which explains why the technique has only recently become economically viable.

A number of equations can be used to measure the match between two images. The majority of these have been listed in a review paper on image registration [78]. The metrics covered in this paper are:

1. Normalised Cross Correlation [28]
2. Correlation Coefficient [79]
3. Statistical correlation and matching filters [80]
4. Phase Correlation [81]
5. Sum of absolute difference of intensity [82]
6. Sum of absolute difference of contours
7. Contour/surface differences
8. Number of sign changes in pointwise intensity
9. Higher level metrics: structure matching: tree and graph distances
10. symmetric matching: automata.

A number of these methods are reviewed in this section, including the normalised cross correlation method, the correlation coefficient, statistical correlation, and the sum of absolute difference of intensity method. The other methods were not considered to be suitable for matching the speckled images in this application. Some parts of the equations listed in this section are suitable for implementation in the frequency plane; others are restricted to the spatial plane only. The choice of method of implementation will affect the speed of processing, but will not alter the final result. An investigation into the preferred implementation method is to be found in chapter 6.

When first investigating the measurement of the displacement between images, the variation of the mean level of a window was found to make cross correlation unsuitable. Normalised cross correlation compensates for variations in the mean level across images by restricting the range of the result. The first attempt at implementing the normalised cross correlation equation was delayed, as a number of papers were found to have errors in the equation (see equations 2.20 and 2.21). Before finding a paper with the correct normalised correlation equation, a new normalised cross correlation equation (2.17) had been derived.

As additional papers were examined, no references to this new equation were found in the field of image processing. A thorough search for pattern matching equations was made to see how novel the equation was, and eventually a passing reference to the new equation was found in a paper on signal processing [110]. This section is a literature review of the pattern matching equations found whilst searching for previous uses of the new normalised equation.

Most pattern matching methods use either the difference between two images, where the best match is taken as the location with the least difference, or multiplicative methods, where the best match produces the largest result.

One of the earliest uses of digital image correlation was to align aerial photos, used for agricultural surveys, that were taken from the same location, but at different optical wavelengths [83]. This paper used a variation on the normalised correlation equation:

$$C_{m,n}(x,y)=\frac{\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)-\overline{A}(m,n)\right)\left(B(x+i,y+j)-\overline{B}(x,y)\right)}{\sqrt{\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)-\overline{A}(m,n)\right)^{2}\sum_{i,j=0}^{k-1,l-1}\left(B(x+i,y+j)-\overline{B}(x.y)\right)^{2}}}$$

(2.5

where the general expression **A(a, b)** is the grey level value of the pixel at position **(a, b)** in image **A** (size **M, N**), and **B(a, b)** is the grey level value of the pixel at **(a, b)** in image **B** (also of size **M, N**). The variables **(k, l)** represent the size of the correlation windows (see figure 2.1). $\mathbf{C_{m,n}(x, y)}$ is the correlation value between the window in **A** with an origin at **(m, n)** and the window in **B** with an origin at **(x, y)**. The means of **A** and **B** are taken over the correlation windows:

$$\overline{A}(m,n)=\frac{1}{k\times l}\sum_{i,j=0}^{k-1,l-1}A(m+i,n+j)$$

and

$$\overline{B}(x,y)=\frac{1}{k\times l}\sum_{i,j=0}^{k-1,l-1}B(x+i,y+j)$$

The best correlation between the window in **A** at **(m, n)** and **B**, is the location in **B** with the largest value of $\mathbf{C_{m,n}(x, y)}$. The Cauchy-Schwartz inequality [28] means this function has a result in the range $\mathbf{-1 \le C_{m,n}(x, y) \le 1}$ with matching images (or multiples of matching images) giving a result of 1, and negative images, and multiples of negative pairs of images giving a result of -1. The paper found that correlating the gradient between neighbouring pixels provided the best results for aligning images taken at different wavelengths.

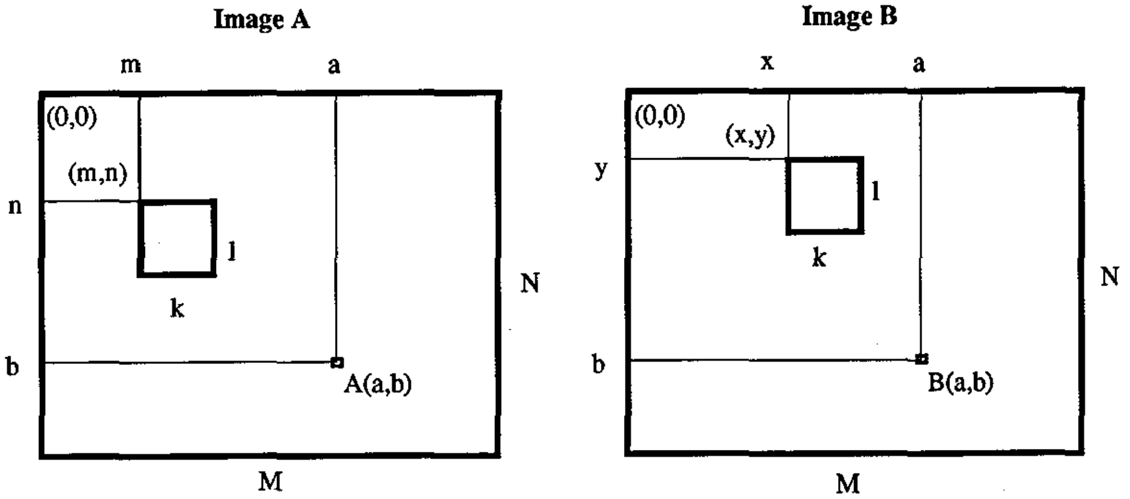


Figure 2.1 - Pattern Matching Windows in the Two Images.

A few years later, sequentially similarity detection algorithms (SSDAs) were proposed [82]. This method cuts down the processing overhead by reducing the number of calculations required. Measures of similarity, suitable for implementation with SSDAs, include the sum of difference equation that was expressed in this paper as either equation 2.6 or, when corrected for variations in the mean, as equation 2.7.

$$C_{m,n}(x,y)=\sum_{i,j=0}^{k-1,l-1}\left|A(m+i,n+j)-B(x+i,y+j)\right|$$

(2.6

$$C_{m,n}(x,y)=\sum_{i,j=0}^{k-1,l-1}\left|A(m+i,n+j)-\overline{A}-B(x+i,y+j)+\overline{B}\right|$$

(2.7

where

$$\overline{A}=\frac{1}{M\times N}\sum_{i,j=0}^{M-1,N-1}A(i,j)\qquad\text{and}\qquad\overline{B}=\frac{1}{M\times N}\sum_{i,j=0}^{M-1,N-1}B(i,j)$$

A running total was kept as each pair of grey levels was compared. If the total exceeded a threshold, the calculation was stopped. The best match was found at the location of the smallest final total, or the location where the largest proportion of the calculations were performed before the threshold was exceeded. The paper compared the processing overheads for the SSDAs with correlation techniques implemented with Fast Fourier Transforms (FFTs) [84], and showed that for the early computer architectures, the SSDAs were considerably faster. Both the cross correlation equation (2.8), and the normalised cross correlation equation (2.9) were mentioned in the comparisons.

$$C_{m,n}(x,y)=\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)\right)\left(B(x+i,y+j)\right) \qquad (2.8$$

$$C_{m,n}(x,y)=\frac{\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)\right)\left(B(x+i,y+j)\right)}{\sqrt{\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)\right)^2\sum_{i,j=0}^{k-1,l-1}\left(B(x+i,y+j)\right)^2}} \qquad (2.9$$

The paper noted that equation 2.8 was effective for pattern matching when the intensity was consistent across the image. Otherwise the best match was liable to occur in the region with the highest grey levels. Despite this problem, the equation is sometimes used for auto-correlation in Particle Image Velocimetry [85]. Equation 2.9 has a result limited to $\mathbf{0 \leq C_{m,n}(x, y) \leq 1}$ provided the images are all positive. It is a much simpler equation to implement than the version of normalised cross correlation in equation 2.5. In addition to matching identical images, this equation also matches images with a difference in the image gain. Thus, the signal $\mathbf{x_{i,j}}$ will match $\mathbf{ax_{i,j}}$ for all positive integer values of **a**.

A book on pattern classification [86] published in 1973 listed some more pattern matching equations:

sum of square of differences:

$$C_{m,n}(x,y)=\left\{\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)-B(x+i,y+j)\right)^2\right\}^{1/2}$$

(2.10

maximum difference:

$$C_{m,n}(x,y)=\max\left|A(m+i,n+j)-B(x+i,y+j)\right|$$

(2.11

and normalised cross correlation:

$$C_{m,n}(x,y)=\frac{\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)\right)\left(B(x+i,y+j)\right)}{\sqrt{\sum_{i,j=0}^{k-1,l-1}\left(B(x+i,y+j)\right)^2}}$$

(2.12

Equation 2.10 is more commonly expressed in the form in equation 2.13, which provides the same location for the best match.

$$C_{m,n}(x,y)=\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)-B(x+i,y+j)\right)^2$$

(2.13

Equation 2.11 is similar to equation 2.6 in its computational requirements, but as it takes the maximum value, the result is dependent on the grey levels at only one location, and thus is liable to effects of spurious noise. No references to the use of this equation have been found. Unlike cross correlation (equation 2.8), normalised cross correlation (equation 2.12) corrects for intensity variations by limiting the maximum value of the cross correlation. This version of the normalised cross correlation limits the result to the range:

$$0 \le C_{\mathrm{m,n}}(x,y) \le \left\{ \sum_{i,j=0}^{k-1,l-1} A(m+i,n+j)^2 \right\}^{1/2}$$

The normalised cross correlation equation is used as the matching method for statistical correlation [80, 87]. This technique pre-processes the images with a filter to maximise the correlation peak. This is claimed to improve the accuracy of the pattern matching.

A paper was published in 1978 [79] comparing the accuracy of pattern matching equations. The algorithms considered were a version of the normalised cross correlation equation (2.14, described as correlation coefficient), the cross correlation equation (2.8, described as the correlation function) and the sum of absolute differences (equation 2.6).

$$C_{m,n}(x,y) = \frac{\sum_{i,j=0}^{k-1,l-1} \left(A(m+i,n+j)-\overline{A}\right)\left(B(x+i,y+j)-\overline{B}\right)}{\sqrt{\sum_{i,j=0}^{k-1,l-1} \left(A(m+i,n+j)-\overline{A}\right)^2 \sum_{i,j=0}^{k-1,l-1} \left(B(x+i,y+j)-\overline{B}\right)^2}}$$

(2.14

The metric used was the percentage of images that were correctly matched. Of the three methods, the normalised cross correlation equation was found to be the most accurate, and the correlation function was the least accurate. The results indicated the need to normalise the cross correlation equation. Further improvements were noted when correlating the gradients of the images, particularly when comparing images taken at different spectral bands. Using equation 2.14 for cross correlation has the advantage over the equation 2.5 in that the mean values only need to be calculated once. Both images, can have the mean deleted globally, which allows the use of the normalised correlation equation (2.9). Equation 2.14 produces a result in the range **-1á•áC$_{m,n}$(x, y)á•á1**, with a result of -1 indicating the two signals are exact negatives of each other.

The same year, a theoretical investigation into the ideal window size was published [88]. Working with the cross correlation equation (2.8), it showed that the ideal window size decreased as the distortion between the two images increased. It also mentioned rectangular correlation windows for situations where the distortion is greater on one axis than the other. The results were calculated from the peak to side-lobe ratio and the probability of false acquisition. The peak to side-lobe ratio is a ratio between the peak correlation value and the value of background correlation. A later paper [89] showed that the ideal window size was the width of the auto-correlation function divided by the relative magnification or rotation.

The sum of squares equation (2.9), sum of absolute differences equation (2.6) and the sum of square of differences (2.13) were found in a review on pattern matching for dynamic scene analysis [90]. The paper investigated the advantages of correlating gradient images, and mentions the use of course-fine template matching. This technique reduced the computation requirements by first finding a match in a course image, and refining the location with more detailed images.

A comparison of the performance of different equations was made with overhead aerial images taken at different angles [91]. The difference equations (similar to 2.6 and 2.14) were found to perform better than the multiplicative equations (2.9 and 2.13). The results confirmed there is an ideal window size for a given quantity of distortion.

A research group at the University of South Carolina is credited with the development of stress analysis using digital correlation. Their initial paper [92] on the subject used laser illumination and the cross correlation function (equation 2.8). A white light version was soon developed [93, 94, 95]. These used a target lightly speckled with a spray of white paint. An image of the stressed target is corrected for applied stress and compared with the image taken before distortion. The authors used the sum of squares of difference equation (2.13) and bilinear interpolation between the pixels together with a course fine method to obtain a more accurate match. The four strain components were found as functions of the distortion correction mapping.

The group subsequently made a move to the normalised cross correlation equation (2.9) in a paper that also introduced polynomial interpolation [96], although a later paper used the previous difference equation for stress measurements at crack tips [97]. A modification to the normalised correlation equation, (2.15) which required a search for the minimum rather than a maximum value, was later introduced [98].

$$C_{m,n}(x,y)=1-\frac{\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)\right)\left(B(x+i,y+j)\right)}{\sqrt{\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)\right)^{2}\sum_{i,j=0}^{k-1,l-1}\left(B(x+i,y+j)\right)^{2}}}$$

(2.15

This paper used a Newton Raphson method to speed up the calculation of the matching process in the fine grid. However, the strain measurements produced by this technique were found to be variable. The method was investigated to identify the limits of its accuracy [99]. Higher order interpolation methods improved the results, as did the use of a 12 bit A/D converter, and using images sampled at a higher frequency. The use of bilinear spline interpolation was found to produce more reliable strain measurements [100]. Another paper [101] investigated a number of noise filters to test their suitability with digital image correlation.

Equation 2.15 has been used by the group to measure strains in sheets of paper [102], evaluate damage to glass fibre reinforced composites [103], measurements at high temperatures [104] and microscopic deformations [105]. The University of South Carolina's method of measuring strains has been widely used elsewhere [55, 106, 111, 117].

The cross correlation equation 2.8 has been implemented in hardware with a line detector for a rotation detector [107]. A similar device with a one dimensional input used the absolute difference equation (2.6) [108, 109]. Another paper investigating the correlation of two, one dimensional signals (**x** and **y**), using hardware [110] introduced the normalised correlation equation 2.16.

$$\tilde{r}_{xy}=\tanh\left[\frac{1}{2}\left(\ln\hat{\sigma}_{x+y}^{2}-\ln\hat{\sigma}_{x-y}^{2}\right)\right]$$

(2.16

where

$$\hat{\sigma}^2_{x+y} = \frac{1}{N}\sum_{i=1}^{N}(x_i + y_i)^2 \qquad \text{and} \qquad \hat{\sigma}^2_{x-y} = \frac{1}{N}\sum_{i=1}^{N}(x_i - y_i)^2$$

Equation 2.16 is expressed as logs to ease hardware implementation. This equation is derived from another method of normalising cross correlation, referred to in this thesis as the "new normalisation" cross correlation equation (2.17). No other references to these equations have been found. The result of equation 2.17 is in the range **$0 \leq C_{m,n}(x, y) \leq 1$.** This equation only produces a perfect result if the two signals are identical.

$$C_{m,n}(x,y) = \frac{2 \times \sum_{i,j=0}^{k-1,l-1}\big(A(m+i,n+j)\big)\big(B(x+i,y+j)\big)}{\sum_{i,j=0}^{k-1,l-1}\big(A(m+i,n+j)\big)^2 + \sum_{i,j=0}^{k-1,l-1}\big(B(x+i,y+j)\big)^2}$$

(2.17

Equation 2.18 is a modified version of equation 2.13 used by a group at California Institute of Technology who have measured deformations in a cylinder [111], and microscopic deformations [112]. The modification is a scaling factor, which will not change the location of best match.

$$C_{m,n}(x,y) = \frac{\sum_{i,j=0}^{k-1,l-1}\big(A(m+i,n+j) - B(x+i,y+j)\big)^2}{\sum_{i,j=0}^{k-1,l-1} A(m+i,n+j)^2}$$

(2.18

The normalised cross correlation equation (2.9) is a popular equation, that has been used with radiographic images [55], for terrain mapping [113], and to track the movement of glaciers [114]. The radiographic images were of bone, which provided sufficient contrast to allow measurements to be made without the use of markers. The glacier paper used reverse correlation to check that the correct location had been found. Reverse correlation finds the best match from the first image to the second, and checks it by seeing if this location matches back to the original point in the first image.

The normalised cross correlation equation with mean correction (2.14) is also popular. It has been used for component identification [115], where the degree of match is also important, as it is necessary to distinguish between different types of component. A paper on crack tip strains [116] included an alternative version of the normalised cross correlation equation (2.19), referred to as the "fully normalised" cross correlation method in this thesis:

$$C_{m,n}(x,y)=\frac{N.M\sum_{i,j=0}^{k-1,l-1}A(m+i,n+j)B(x+i,y+j)-\sum_{i,j=0}^{k-1,l-1}A(m+i,n+j)\sum_{i,j=0}^{k-1,l-1}B(x+i,y+j)}{\sqrt{\left[N.M\sum_{i,j=0}^{k-1,l-1}A(m+i,n+j)^2-\left\{\sum_{i,j=0}^{k-1,l-1}A(m+i,n+j)\right\}^2\right]}}$$

$$.\frac{1}{\sqrt{\left[N.M\sum_{i,j=0}^{k-1,l-1}B(x+i,y+j)^2-\left\{\sum_{i,j=0}^{k-1,l-1}B(x+i,y+j)\right\}^2\right]}}$$

(2.19

The fully normalised equation is noted to match images with different gains and offsets. Thus the signal $\mathbf{x}_{i,j}$ would match $\mathbf{a}\mathbf{x}_{i,j}+\mathbf{b}$ for all positive values of **a**, and all values of **b**. The equation has been used to measure concrete deformations [117]. It was also tested for use in pattern inspection [118], although the absolute difference equation (2.6) was considered more suitable.

A paper on digital particle image velocimetry [119] included the equation 2.20 for normalised cross correlation. This paper states the equation produces a result close to 1 when the images match. Unfortunately the equation in this paper is wrong, and is thought to be a misprint, of equation 2.9. The error is easy to show by applying small images (of say 2 x 1 pixels) to the equation. This error has been repeated in another paper [120].

$$C_{m,n}(x,y)=\frac{\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)\right)\left(B(x+i,y+j)\right)}{\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)\right)\sum_{i,j=0}^{k-1,l-1}\left(B(x+i,y+j)\right)}$$

(2.20

Nor it is the only misprint of the cross correlation equation found. A paper on measuring the shrinkage in powder compacts [121] expressed the normalised cross correlation as equation 2.21, which also has an error in the denominator:

$$C_{m,n}(x,y)=\frac{\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)-\overline{A}\right)\left(B(x+i,y+j)-\overline{B}\right)}{\sqrt{\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)-\overline{A}\right)\sum_{i,j=0}^{k-1,l-1}\left(B(x+i,y+j)-\overline{B}\right)}}$$

(2.21

Not all visual strain measurement methods use difference or multiplication equations. Some techniques directly measure the movements of identifiable features. Movements have been measured using microradiographic images [54]. A graphite epoxy composite was made with imbedded 10μm Gold markers. This was illuminated by a highly collimated synchrotron source. Strains were measured by comparing the positions of individual markers before and after loading. A similar method has been used to track markers glued to a piece of rubber to measure its long term degradation [122].

A wide range of pattern matching equations were found in this survey. A number of the techniques also pre-processed the images to improve the results. There is no reason why any of the pre-processing techniques could not be implemented with any of the pattern matching equations. The few papers that compare a number of different methods give no clear picture of a preferred method. This suggests the choice of equation and pre-processing technique is dependent on the application. Chapters 6 and 7 of this thesis therefore investigate the most suitable of the methods for radioscopic strain measurement. These chapters look at the speed of processing, the reliability and the accuracy of the different methods.

3 - The Radioscopic Equipment

Loughborough University's Mechanical Engineering department purchased a commercial radioscopic X-ray system in 1994 to enhance the department's capability of researching material properties. This chapter describes the X-ray source and the image acquisition components. The results of the tests on the equipment are included in the chapter, both to confirm the specification of the equipment, and to determine the system's capabilities and practical limitations.

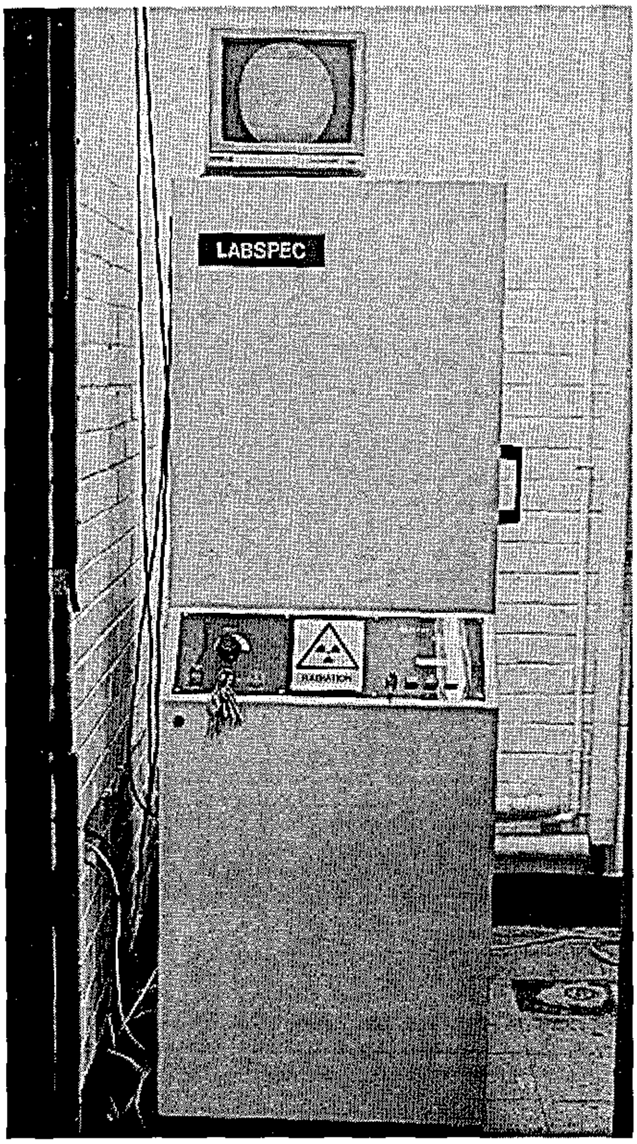


Figure 3.1 - The X-Ray Equipment.

3.1.1 - X-Ray Equipment

The X-ray unit (figure 3.1) was acquired after an objective review, that was the basis of a final year project in 1993 [123]. It is a Labspec 250 system supplied by Graseby Security of Wellingborough (Appendix I.1). It consists of an X-ray generator, a sample chamber and an image intensifier, and is about the size of a large filing cabinet. The X-ray energy source can be varied between 50 and 90keV and is rated at 4mA. The X-ray 240V half wave generator is mounted above the sample chamber, with the image intensifier below (figure 3.2). The dimensions of the target chamber are 424mm high x 560mm deep x 440mm wide.

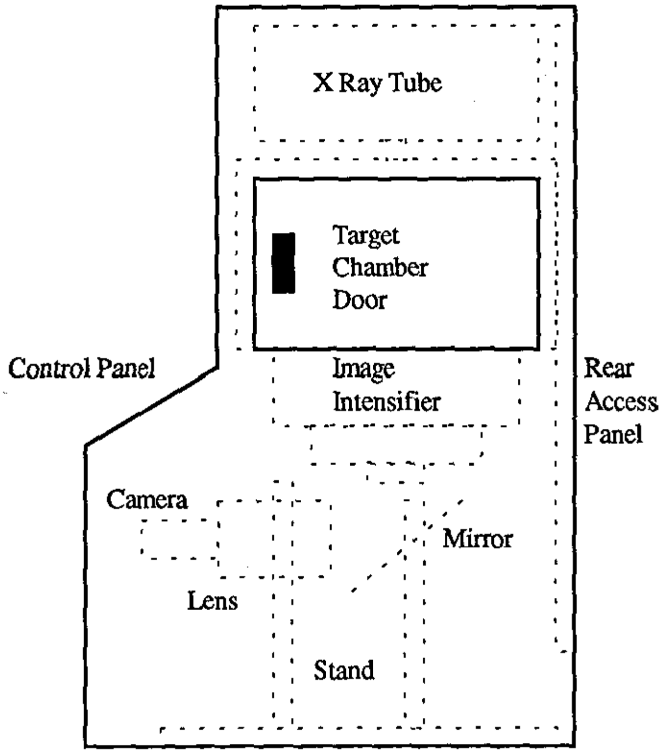


Figure 3.2 - General Layout of the X-Ray Equipment.

The image intensifier is a Thomson Tubes Electroniques TH 8428 HP (Appendix I.2). It has three zoom settings: 215mm, 160mm and 120mm. The nominal entrance field is 225mm diameter, and the output image is 20mm diameter, with a peak wave length of 520 to 540nm. The maximum resolution is between 64 and 42 line pairs per cm, depending on zoom setting and the distance from the centre of the image. The output is brightest at the centre of the image. The reduction in brightness at a location 90% of distance to the edge of the target area, is 80% for 215mm zoom setting, rising to 95% for the 120mm zoom setting. At the same location, the distortion in the size of a 10mm object, is given as 15% for the 215mm zoom setting, falling to 3% for the 120mm setting.

On the control panel, the user can turn on the X-ray generator for between one and ten minutes, in steps of one minute. A key switch is used to select between the three image intensifier zoom sizes, and a rotary control is used to change the X-ray voltage. Between the maximum and minimum X-ray voltages there are nine marks on this control. There are interlocks to prevent operation with the sample chamber door open, and to prevent operation without the operator inserting a key. The final year project included installing the CCD camera, a 50mm lens, and connecting the system to an image grabber board. The imaging system has since been developed further.

3.1.2 - Optical System.

The camera is a Pulnix TM765E high resolution CCD shutter camera with edge enhancement (Appendix I.3). This is a 625 line 50Hz CCIR camera with 2:1 interlacing. It is a 2/3" format camera with 11μm cells. The 768 horizontal pixels, and 581 vertical pixels can detect illumination down to .5 lux. The camera has a number of shutter speeds, and can also integrate over several frames. Both these features have been implemented as part of this thesis.

The camera is connected so that the shutter speed setting is selected from a binary coded decimal (BCD) switch. The BCD switch selects relevant pins in the 6 pin connector to the camera. This circuit is powered from the camera's 12V pin 4, which is regulated down to 5V by a Zenner diode. The BCD switch powers inputs d_0, d_1 and d_2 (pins 1, 5 and 6) as appropriate. Normal operation is with the switch set to 0 (1/60 second shutter speed). Table 3.1 shows the other settings.

A simple 555 timer circuit was initially built for the integration control system, but this was not linked to the grabber board, which made it difficult to acquire the integrated image. An input/output board was configured so that the integration pin (pin 11 on the 12 pin connector) is directly controlled from the computer. Software (see section 3.2) was written to interface with the board, so it would hold the integration pin low for the appropriate number of frames, before grabbing the next image.

Table 3.1 - Shutter Speeds of the TM765E Camera

Switch Setting	Shutter Speed (Seconds)	Pin d_0	Pin d_1	Pin d_2
0	1/60	0V	0V	0V
1	1/125	5V	0V	0V
2	1/250	0V	5V	0V
3	1/500	5V	5V	0V
4	1/1000	0V	0V	5V
5	1/2000	5V	0V	5V
6	1/4000	0V	5V	5V
7	1/10000	5V	5V	5V
8	1/60	0V	0V	0V
9	1/125	5V	0V	0V

The camera was originally fitted with a 50mm lens, and was located directly below the image intensifier. This was unsatisfactory as the minimum focusing distance of the lens prevented the radioscopic image taking up more than a half of the camera's field of view. A zoom lens was considered to be more suitable for this application as it would allow images of the whole of the radioscopic window, and would permit more detailed images of the centre of the window. It was calculated that with the camera in its present position, a lens of 87mm focal length would give a field of view matching the full radioscopic image. A lens with a focal length of 200mm would see the image intensifier at maximum resolution. However, the distance between the camera and the image intensifier was only 265mm, making it unlikely a 200mm lens would fit in, let alone focus on the image intensifier.

A new stand was designed for the X-ray cabinet (figure 3.3) which allowed a greater distance between the camera and the image intensifier. The new stand included a mirror mounted at 45°, which reflects the images to the horizontally positioned camera, located below the X-ray control board. The lens is screwed to the stand, and the camera is attached to the lens. The new arrangement gives a distance of around 500mm between the camera and the image intensifier. This equates to a focal length of between 124mm to 284mm. A motorised zoom lens is necessary as the camera operates within a sealed cabinet.

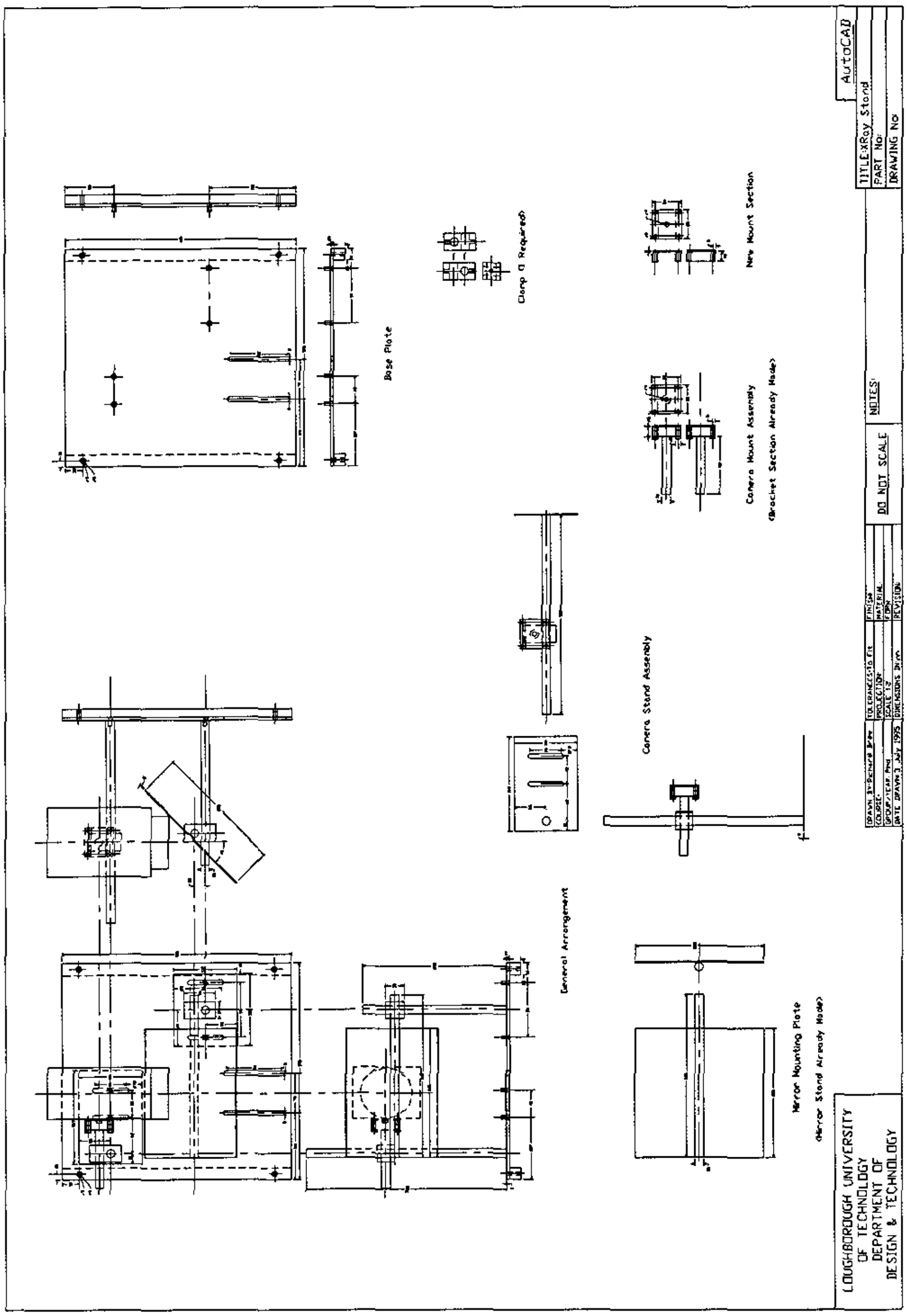


Figure 3.3 - Camera Stand Design.

Following a search, the most suitable lens was identified as the Tecsec TLZMA16160 motorised zoom lens. The lens has a focal length range of 16-160mm, which was increased to 32 to 320mm by fitting a 2:1 range extension. The aperture range is from f1.8/2.0 to f22. It has a 1" C mount capable of taking 67mm filters. The lens required a number 3 diopter to focus on the image intensifier. The camera is oriented so items nearest the chamber door appear at the bottom of the image.

Whilst it is possible to adjust the locations of both the camera and the mirror on the new stand, accurate alignment is difficult. A plumb line, 45° set square and a spirit level were used to get the best alignment. First the mirror is fixed at 45° degrees using the set square and spirit level. Next, it is accurately positioned underneath the image intensifier using the plumb line. The lens is lined up with the image, using the spirit level to check that it is horizontal. Rotational adjustments to the image can be made in the 1" C mount. Care has to be taken when turning the camera with respect to the lens as it is difficult to reach, and the lens was found to twist on the mount.

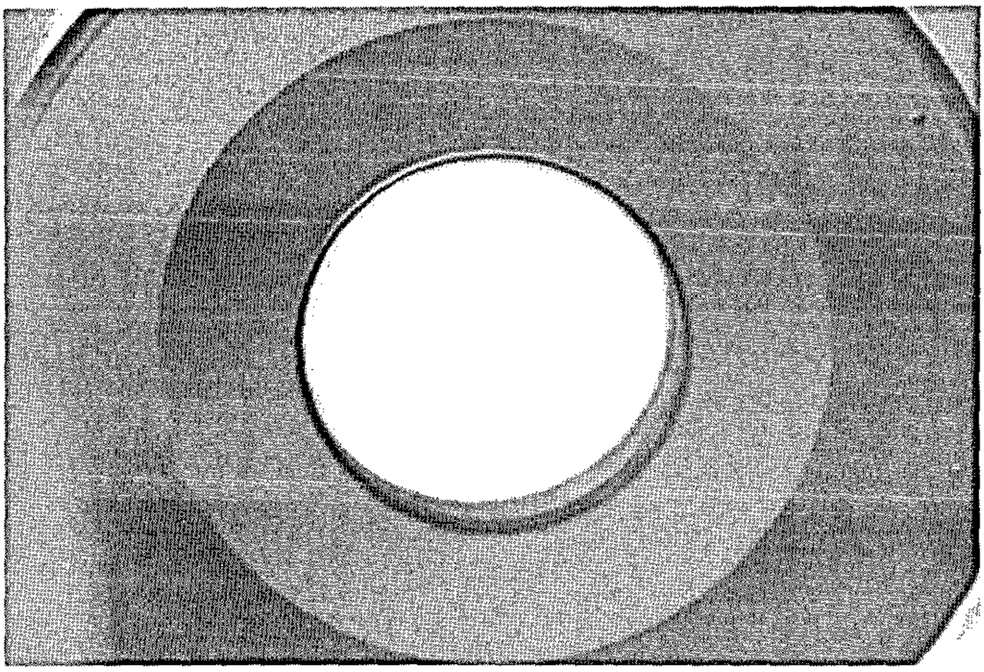


Figure 3.4 - Image Intensifier as seen by Camera when Illuminated by Normal Light Source.

It is not possible to adjust the optics whilst the image intensifier is switched on, as there is an interlock on the access hatch. Optical alignment of the camera was performed using a portable light source to illuminate the image intensifier output window (figure 3.4). Although this proves that the camera is aimed at the image intensifier, it is not possible to confirm that all of the components are fully aligned. The whole operation has to be performed in situ, in a very confined space (figure 3.5). The stands were made with only locking screws to keep the components in place. A stand with fine adjustment to the positions would make alignment easier.

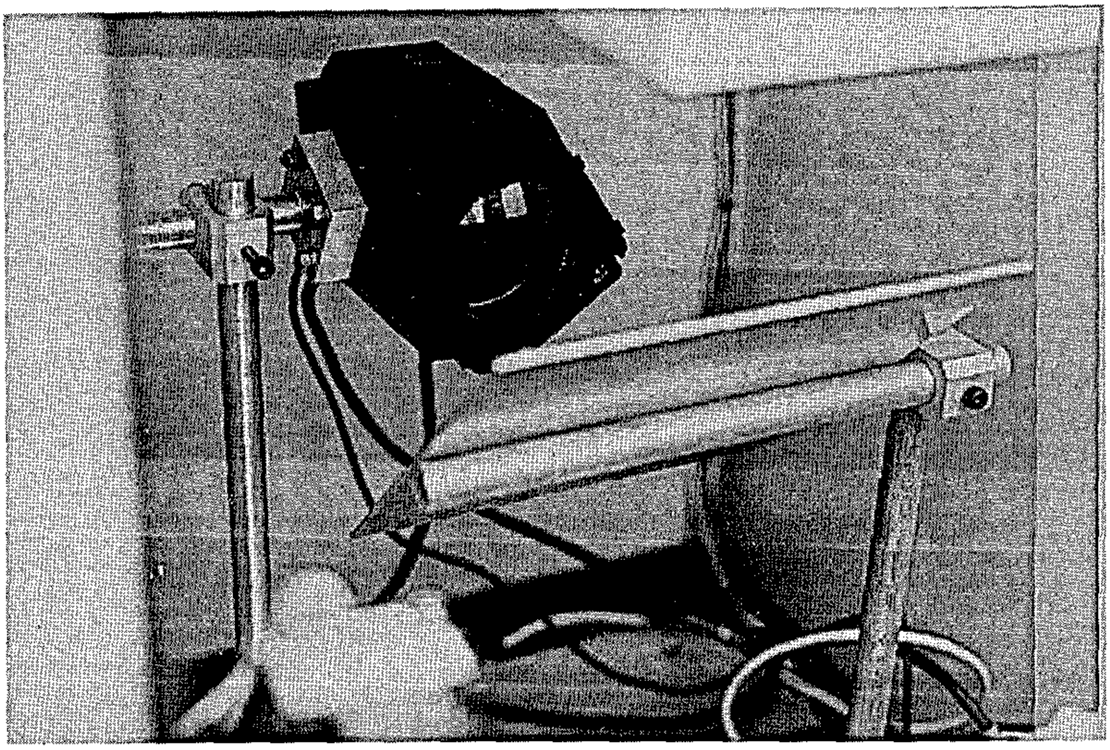


Figure 3.5 - Camera and Stand as Installed in X-Ray Equipment.

The original mirror was a rear surface reflective mirror. It was noted the radioscopic images included repeated overlapping images (figure 3.6). Tests showed that by moving the camera to an axis nearer the vertical, the distance between the repeat images was reduced. Thus it was deduced that these images were created by internal reflection within the optical system. A new front surface mirror was fitted, eradicating the problem.

The image from the camera is displayed on a monitor situated on top of the cabinet, and is also fed to the controlling computer. The lens is controlled from a box positioned next to the controlling computer. This has three switches, of the two way rocker, centre off type. The switches control the zoom position, focus and aperture. The box requires its own 12V power supply. There is no feedback from the zoom lens, so it is not possible to directly measure the zoom settings. A second box positioned next to the computer holds the shutter BCD switch, and the wiring for the integration control.

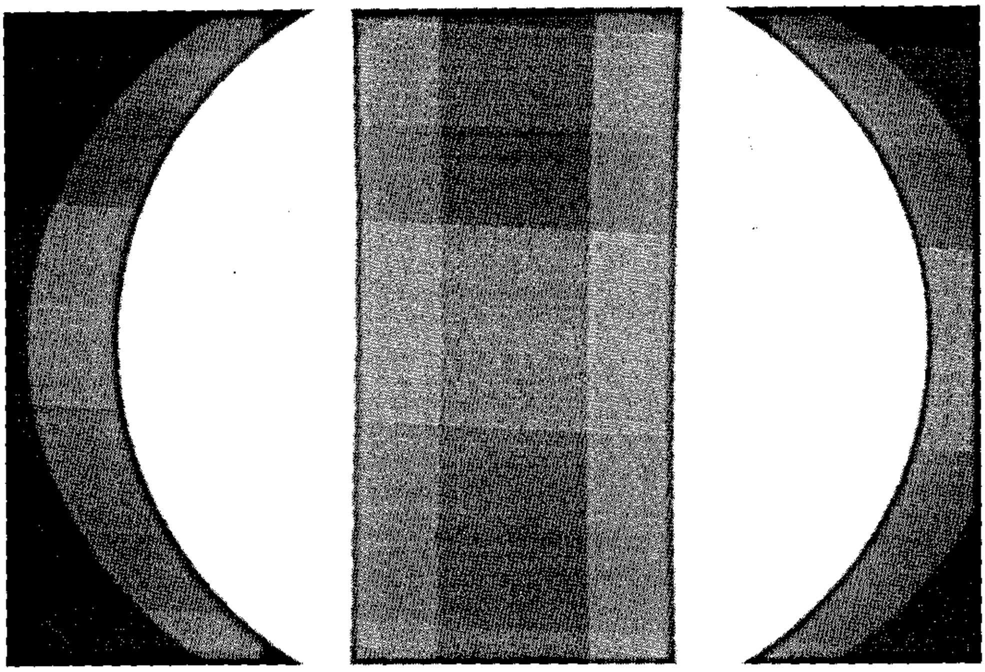


Figure 3.6 - Radioscopic Image of Ceramic Sample, Showing Overlapping Images caused by Original Mirror.

3.1.3 - Image Acquisition

The radioscopic images are acquired using an image grabbing board fitted in a Victor V286A computer that has 1M of RAM. The computer has a 5.25" **A** drive and an external 3.5" **B** drive. The computer has a second graphics monitor, which is used to display the radioscopic images. The machine's operating system is MS DOS V4.01a.

The computer was originally fitted with an Imaging Technology VS-100 frame grabber board. The board was able to process CCIR images of up to 512 lines of 768 pixels, and could be programmed by software to alter the signal gain between 0 and 4, in 256 steps. The offset could be programmed between 0 and -1 in 256 steps. The VS-100 board sampled square pixels.

The camera's image integration facility was implemented from the computer via an I/O board. A cable from the shutter speed controller box gives the computer access to the camera's integration pin. The computer holds the pin high whilst it counts the appropriate number of camera frames, and then grabs the next image. Integration is done in multiples of a twenty-fifth of a second.

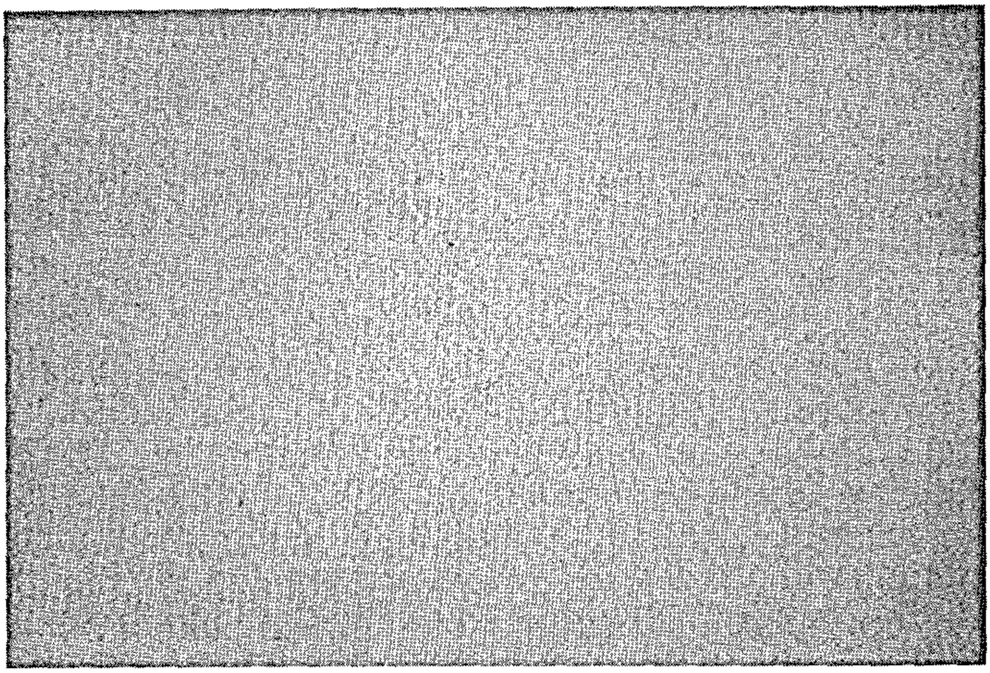


Figure 3.7 - Example of Blank Radioscopic Image from the VS-100 Board.

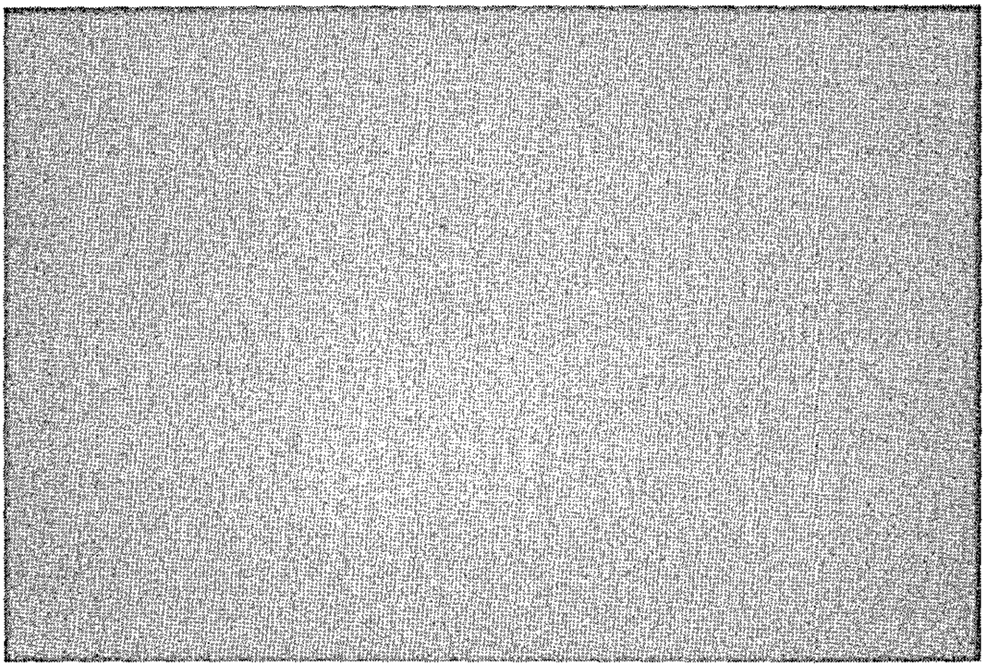


Figure 3.8 - Example of Blank Radioscopic Image from the FG-100 Board.

Large quantities of periodic noise were found to affect the equipment (figure 3.7). A process of elimination eventually proved that the source was the image grabbing board. An FG-100 board replaced the VS-100 board and was found to produce considerably less noise (figure 3.8). The replacement board does not have on-board gain and offset. The new board produces a 512 by 512 rectangular pixel image over the same area previously covered by a 768 by 512 square pixel image. The rectangular pixels (aspect ratio of 4:3) have to be accounted for when measuring distances. Unless otherwise stated, all images were taken with the FG-100 board.

3.2 - Image Acquisition Software

The computer has Image Pro II V2.0 software installed. This can be used to retrieve and analyse the images, although due to the limitations in the size of the images produced by this software (512 x 480 maximum), most images were retrieved using software written in Microsoft C version 5.0 using the ITEX100 toolbox. The images are transferred by disk to other, more powerful computers for processing.

The "C" capture software (appendix II.1) offers a menu of its features. The first option is to retrieve an image previously saved to disk. If selected, the user is asked for the name of the file, which is then loaded and displayed on the monitor. The second option is to save the image displayed on the monitor to a file, for which the user is prompted for a name. Next in the program comes the option of snapping a new image, followed by the option to clear the image display. The integration option is fifth. The program asks for the number of frames over which the image should be integrated, then holds the integration pin high for the required length of time, before grabbing the next image.

Sixth is the exit to DOS option. This is followed by an option to measure the maximum and minimum grey levels in the displayed image. This option takes about twenty seconds to run and is used to check for clipping. Finally, the gain and offset option was only valid with the VS-100 board. This option showed the present gain and offset values, and asks the user to input new values, which were downloaded to the grabber board.

3.3 - Testing the Radioscopic Equipment

Tests were made with the radioscopic equipment to identify its imaging capabilities and limitations. Where possible, each component in the system was tested separately, and as part of the whole system. Section 3.3.1 covers the X-ray source, which was tested to measure the distribution of the X-ray output. Section 3.3.2 investigates how to focus the equipment, and section 3.3.3 looks at the range of vision available from the zoom settings. These tests looked at both the lens zoom settings, and the image intensifier zoom settings, and compared the resolution of the images with the maximum resolution of the image intensifier.

The field of view tests were noticeably affected by spatial distortion. Section 3.3.4 investigates the distortion from the camera, and from the system as a whole. It was noted that the radioscopic images suffered from considerably more distortion than conventional images taken with the same camera and lens. The measured distortion was compared with the values on the image intensifier data sheet, and the distortion expected from the X-ray point source. The brightness of the image across the target is compared with the data sheet in section 3.3.5, together with an investigation of the effect of the zoom settings on the brightness of the image.

The aperture, shutter speed and integration time were investigated to see how they could be adjusted to maximise image contrast. The aperture is covered in section 3.3.6, with the aim of finding the maximum reasonable aperture before the image focus is degraded. Unfortunately it is not possible to measure the aperture setting directly, but by altering the aperture so that the light intensity is consistent, it can be measured as a function of shutter speed.

The investigation of the integration function and shutter speeds follows in section 3.3.7. Longer integration times allow smaller apertures, which result in better focus, and lower X-ray power settings. However, long integration times can result in a blurred image due to vibrations of the cabinet, and above a certain integration time, are impractical due to camera noise. Finally the X-ray power setting is investigated in section 3.3.8. The output brightness was found to be significantly reduced with lower X-ray settings, but depending on the target, lower power can produce better image contrast.

3.3.1 - X-Ray Source

Standard photographic paper was used to test the distribution of the X-rays, and as a comparison for the radioscopic images produced using the image intensifier. Fortunately the X-ray machine is located in a former darkroom, which made it easy to place sheets of photographic paper in the chamber for exposure. Four sheets of Ilford A3 multigrade IV RC deluxe paper were located against the left edge of the chamber and were exposed, with both the copper Image Quality Indicators (IQIs, see section 3.3.2) placed on the paper near to the centre, for 5 seconds, 20 seconds, one minute and 4 minutes respectively.

The photographic sheets were subjected to the normal development process. The sheets with the shorter exposure times were noted to be under exposed, indicating an exposure of at least 4 minutes is required, with thicker targets requiring even longer. All the wires on the IQIs were visible for all four images although the smaller wires on the five second exposure were indistinct. The four minute exposure was noted to have the best contrast.

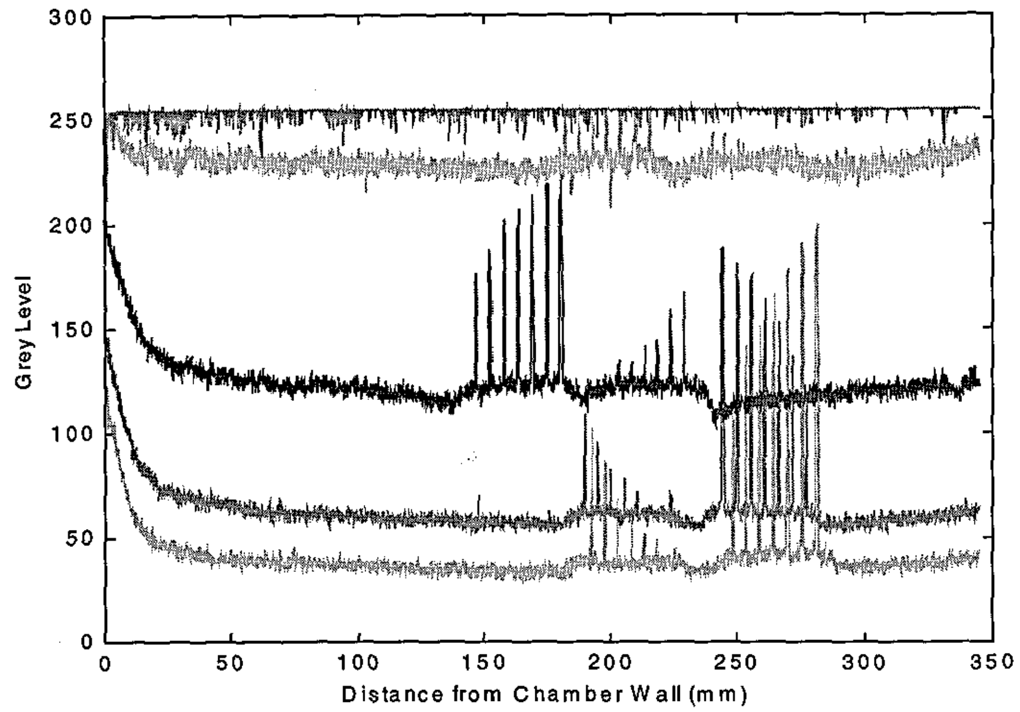


Figure 3.9 - Plots of Grey Levels Across Radiographic Images. Top plot is scan of sheet of white paper, the others have exposure times of 5, 20, 60, and 480 seconds. Note plots include cross sections of IQIs. Centre of chamber at 220mm from wall.

A scanner was used to scan a strip across each of the photographs. The scans were taken at 254 dpi (10 pixels per mm), and were all taken with the same contrast and brightness settings. A fifth image was also taken of a blank piece of paper for comparative purposes. Figure 3.9 shows a plot of the cross sectional brightness through each of the five images. The scan of the blank piece of paper indicates the scanner introduces some noise (although it is likely there are variations in the colour of the paper), and that none of the radioscopic images suffer from clipping. The target area is between 105mm and 330mm from the edge of the chamber, with the centre of the target at 217.5mm. The IQIs are identified on the plots as a slight increase in brightness over the area of the plastic covering, and as a significantly brighter section for each of the wires.

It can be seen that close to the chamber walls the image brightness increases significantly, but over the target region, ignoring the IQIs, there is very little brightness variation. The high scan rate is necessary to detect the finer wires in the IQIs, but results in the noise in the scanned image being very prominent.

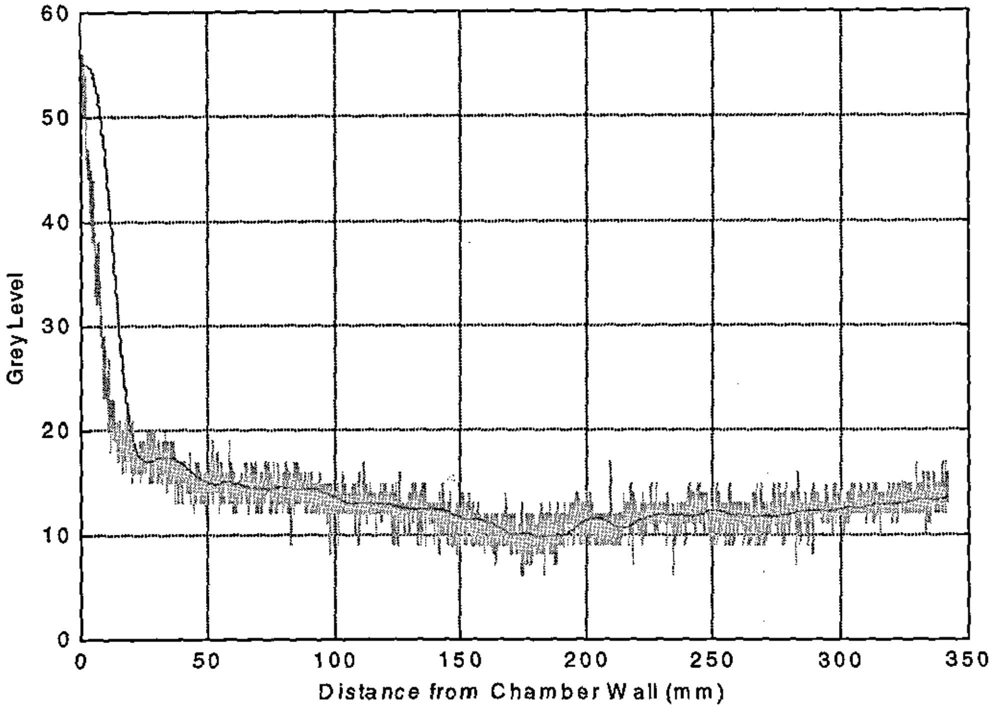


Figure 3.10 - Plot of Grey Levels Across Radiographic Image and Filtered Plot. Centre of chamber at 220mm from wall.

A second 4 minute exposure was taken without the IQIs. A scan was taken of a strip of this image. This scan is shown in figure 3.10, together with the same scan after being passed through a 3rd order Butterworth low pass filter to remove the noise. Whilst the filtered plot is inaccurate close to the side of the chamber, it is accurate over the target area. The standard deviation between the original and filtered plots over the target area was measured to be 1.3 grey levels, indicating a good match between the original and filtered images. A reduction of three grey levels is noted in the filtered image, between the centre and the edge of the target area. But without accurately testing the exposure characteristics of the photographic paper, it is not possible to calculate the relative exposure levels across the target area.

However, if the X-ray source is assumed to radiate evenly across the image, the difference in path lengths can be used to calculate the relative exposure levels. The X-ray source is located 515mm directly above the centre of the floor of the sample chamber. The path length to the edge of the target area is 527mm, an increase of 2.3%. This means that assuming the exposure is even, the edge of the target area is getting 0.977^2 = 95.5% of the X-ray energy the centre of the target region receives.

3.3.2 - Focus

The focus setting of the lens needs to be adjusted to suit each zoom position. The Focus setting has to be assessed manually by the operator. In order to make focusing the camera lens easier, six Image Quality Indicators (IQIs) were purchased from G. J. Wogan of Wokingham (figure 3.11). IQIs are designed to measure the quality of radiographs, particularly in conjunction with radiography of welds. Wire [124] and step/hole [125] types are the most common types of IQIs. The six IQIs are of the wire types W6 and W13, in copper, aluminium and iron. These IQIs are designed to help measure the resolution of X-ray equipment by seeing how thin a piece of wire can be visualised. The W6 type consist of seven wires of 0.25mm to 1mm diameter. Type W13 has seven wires of 0.05mm to 0.2mm diameter.

Radioscopic images were taken of the six IQIs. An average cross section plot (to reduce noise) of the radioscopic image through the W6 IQIs is shown in figure 3.12, whilst figure 3.13 shows an average cross section through the W13 IQIs. From the cross sections, the copper is noted to be the most absorbent of the three materials tested (at full X-ray power), whilst aluminium is the worst, with only the larger diameter wires detectable. The aluminium IQIs are comparatively transparent at the energies of interest. It is noted that aluminium is the material used for the base of the target chamber, thus attenuating the characteristic absorbency of this element. In addition to showing the wires in the IQIs, the edge of the plastic casing is also visible, particularly in figure 3.13.

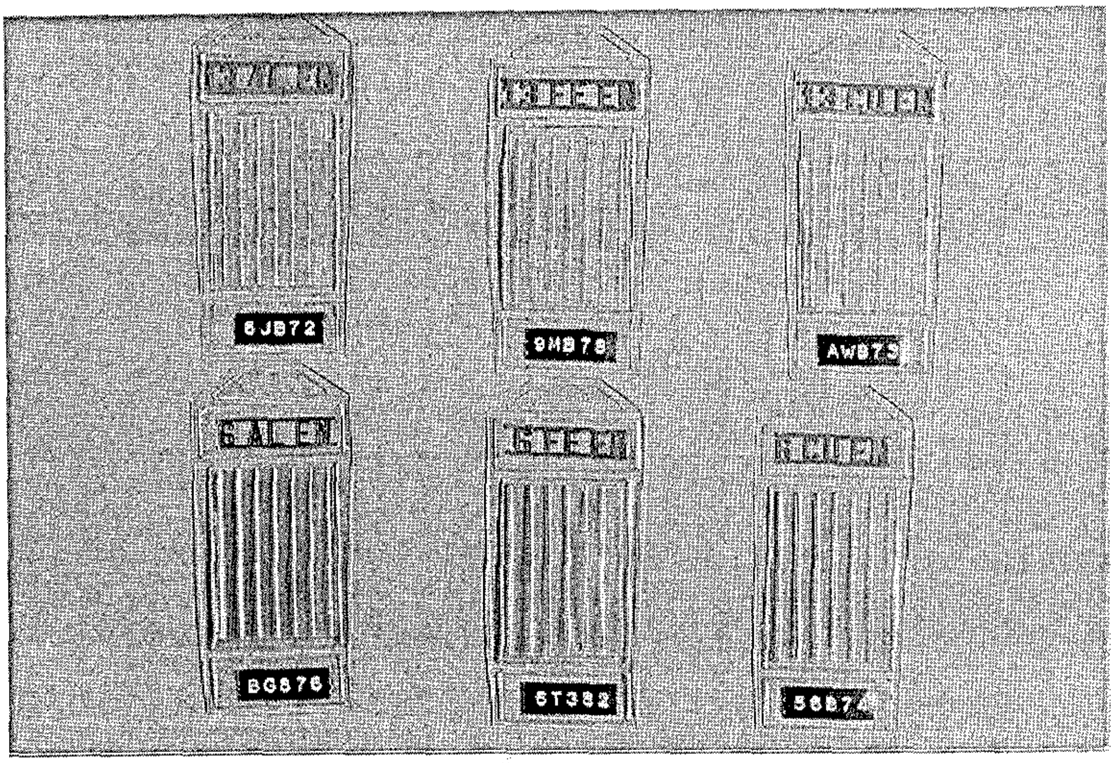


Figure 3.11 - Image Quality Indicators.

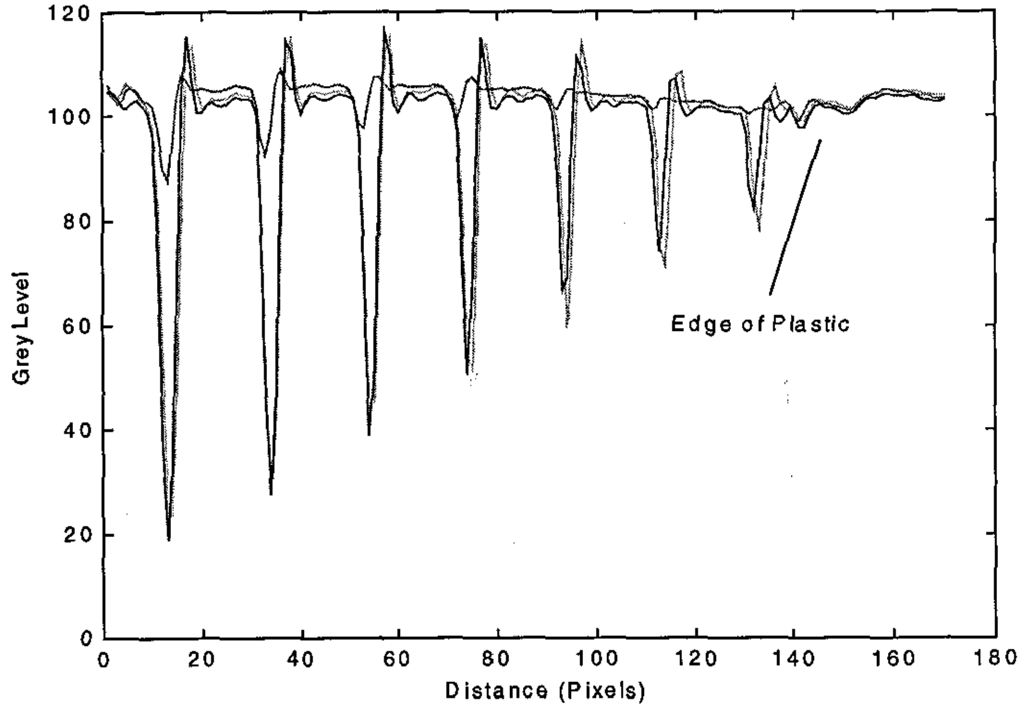


Figure 3.12 - Average Grey Levels of Cross-Sections through Radioscopic Images of the W6 IQIs. Cu: — , Al: — , Fe: — .

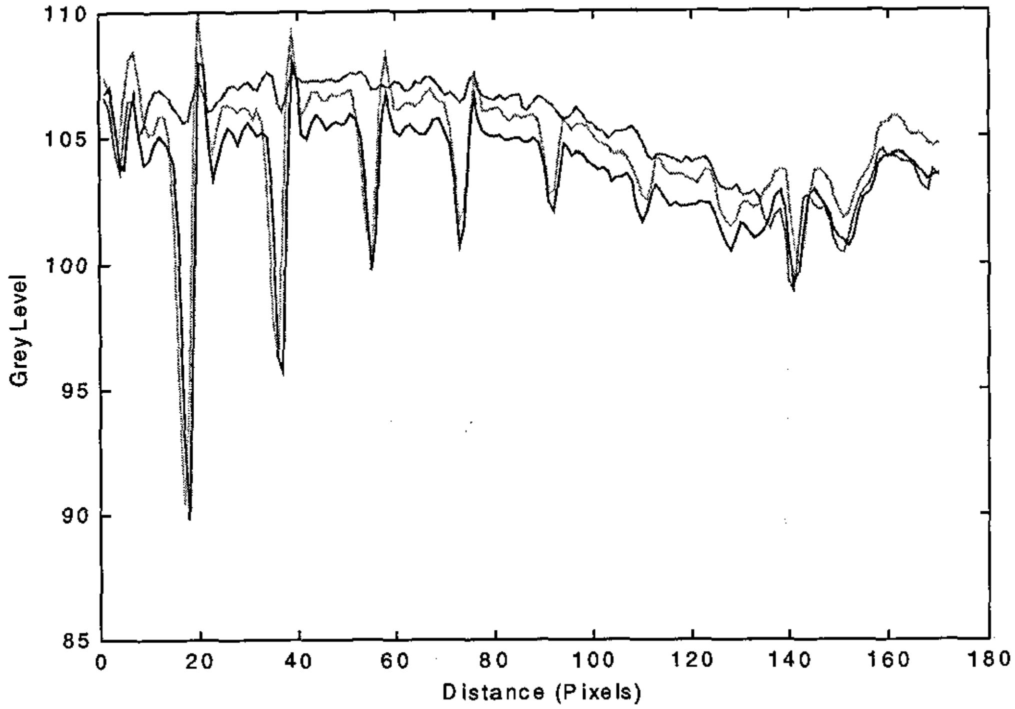


Figure 3.13 - Average Grey Levels of Cross-Sections through Radioscopic Images of the W13 IQIs. Cu: — , Al: — , Fe: — .

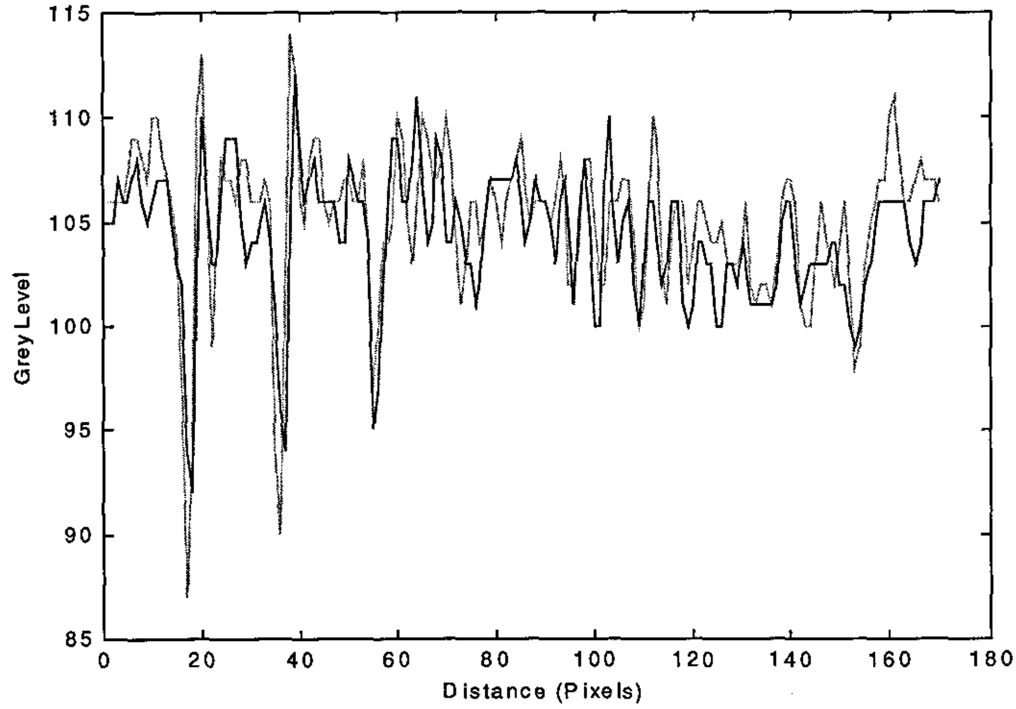


Figure 3.14 - Grey Level Cross Sections through Radioscopic Images of the Cu13 and Fe13 IQIs. Cu: — , Fe: — .

It is noted that the IQIs are not precisely lined up with each other, and that there is a definite "inverse shadow" to the right of each wire. This feature has been noted on all the radioscopic images, and appears as a bright line to the right of any dark objects. The feature is also noted on images taken with the camera and lens when they are out of the radioscopic equipment (figure 3.22), which means it is caused by these components. Inverse shadow is a feature that could be corrected.

As the cross sectional plots are of the average grey levels over a number of lines, the result is dependent on the IQI wires being square to the direction over which they are being averaged. Figure 3.14 is a plot of the grey levels on a single line across the copper and iron W13 IQIs. This plot suffers from a higher level of noise when compared to the average plots, which makes it impossible to distinguish the smaller wires. However it shows that the copper IQI is considerably more absorbent than the iron IQI. As the averaged plot shows the two IQIs to be very similar, this indicates that the iron IQI is slightly squarer to the direction of averaging than the copper IQI.

The IQIs are used to ensure that the lens is properly focused each time the zoom setting is adjusted. In most cases, the copper IQIs are used as they have the best absorbency characteristics. The W13 IQI is preferred as it allows better definition. With each new zoom lens position, the IQI is placed in the test chamber, and a live video image is taken. The aperture is adjusted to make the IQI clearly visible. Finally the focus control is adjusted until as many of the wires in the IQI are as clearly visible as possible. With the camera focused, radioscopic images can be taken.

3.3.3 - Range of Vision

An investigation was made into the field of view with the different image intensifier zoom settings and the full range of zoom lens settings. Images were taken for each of the image intensifier zoom settings, with the lens zoom adjusted to include the whole of the output window and with the lens zoomed to maximum. The radioscopic images were of the distortion measurement grid described in section 4.1. Table 3.2 gives details of the zoom settings, and of the measured field of view in each image.

The first three images show that the nominal image intensifier zoom setting sizes are larger than the measured sizes. The second three images show the maximum resolution for each image intensifier zoom setting. From this is calculated the average number of pixels per cm across the image, in each direction. The maximum resolution of the image intensifier (appendix II.2) is between 42 and 64 line pairs per cm. The resolution is different for each zoom setting, and increases with distance from the centre of the image. At least two pixels are required to distinguish each line pair.

Table 3.2 - Range of Vision

Figure	Image Intensifier Setting	Lens Zoom	Approximate Width	Average Pixels/cm (Horizontal)	Approximate Height Holes x Spacing	Average Pixels/cm (Vertical)
3.15	215mm	middle	33x6 = 198mm	-	33x6 = 198mm	-
3.16	160mm	middle	25x6 = 150mm	-	25x6 = 150mm	-
3.17	120mm	middle	38x6 = 114mm	-	19x6 = 114mm	-
3.18	215mm	maximum	26x3 = 77mm	66	19x3 = 57mm	90
3.19	160mm	maximum	19x3 = 55mm	93	14x3 = 40mm	128
3.20	120mm	maximum	14x3 = 42mm	122	10x3 = 30mm	171

Comparing with the data sheet (appendix I.2) and ignoring the image distortion, the lens has insufficient zoom to fully match the resolution across the width of the image for the 215mm setting, but is sufficient to give full resolution over all but the centre in the vertical direction. With the 160mm setting, the camera has sufficient resolution to measure all line pairs in the vertical direction, but insufficient resolution in the horizontal direction. For the 120mm setting, the camera has a higher resolution than the image intensifier in both directions.

When the computer was fitted with the original VS-100 board, the horizontal resolution was the same as the vertical resolution, making the maximum resolution of the camera close to, or better than the maximum resolution of the image intensifier for all three zoom settings.

An image was taken with the camera lens fully zoomed out (figure 3.21). This shows that the minimum zoom position results in an image of the target of 145 pixels horizontally by 207 pixels vertically.

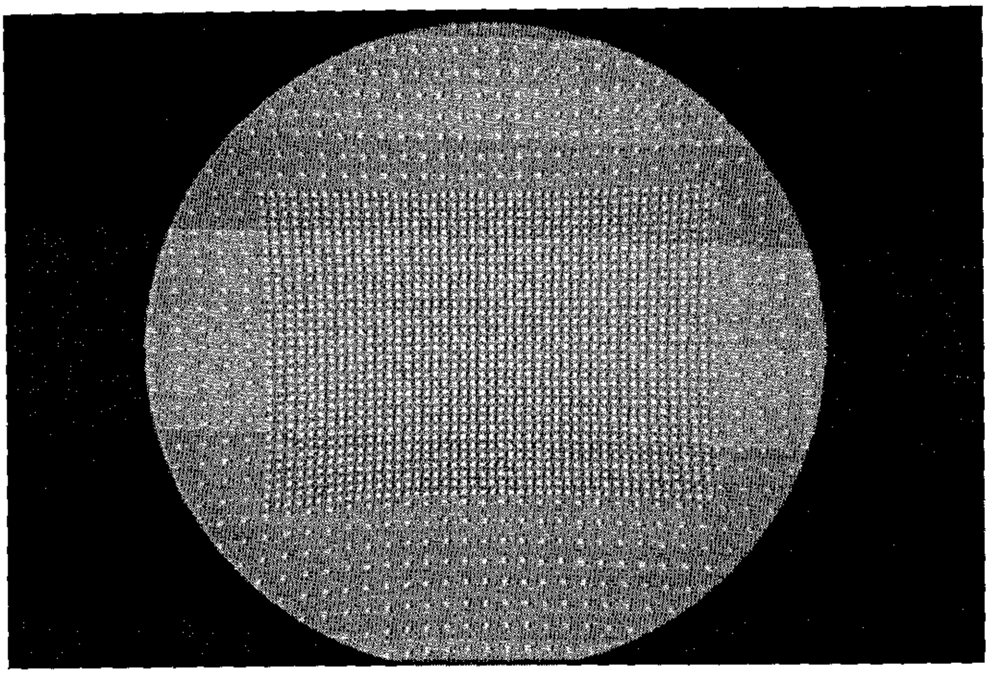


Figure 3.15 - Radioscopic Image of Distortion Phantom, 215mm Image Intensifier Setting.

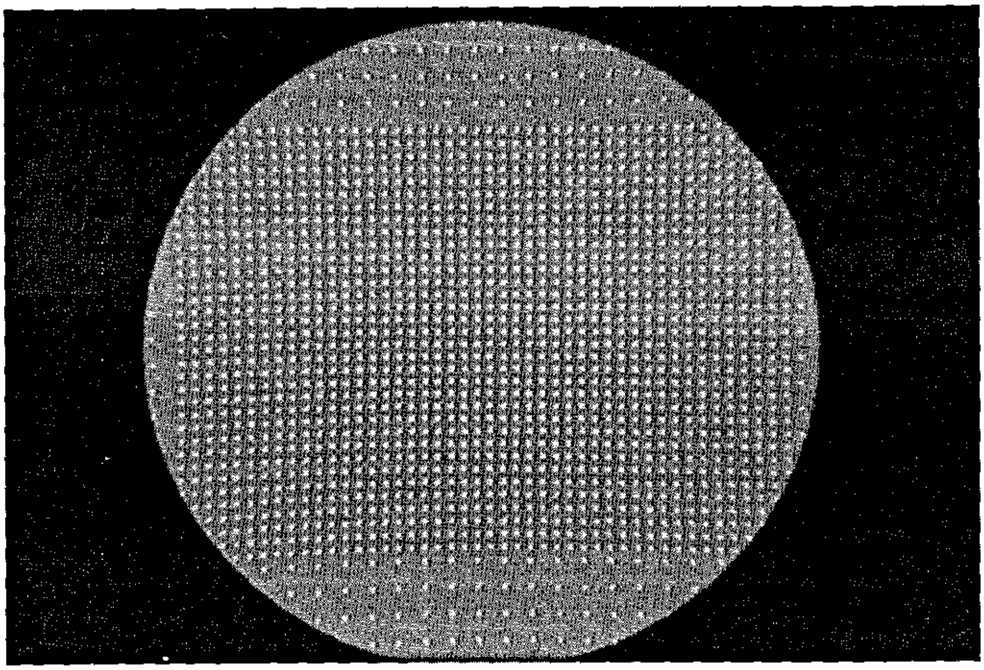


Figure 3.16 - Radioscopic Image of Distortion Phantom, 160mm Image Intensifier Setting.

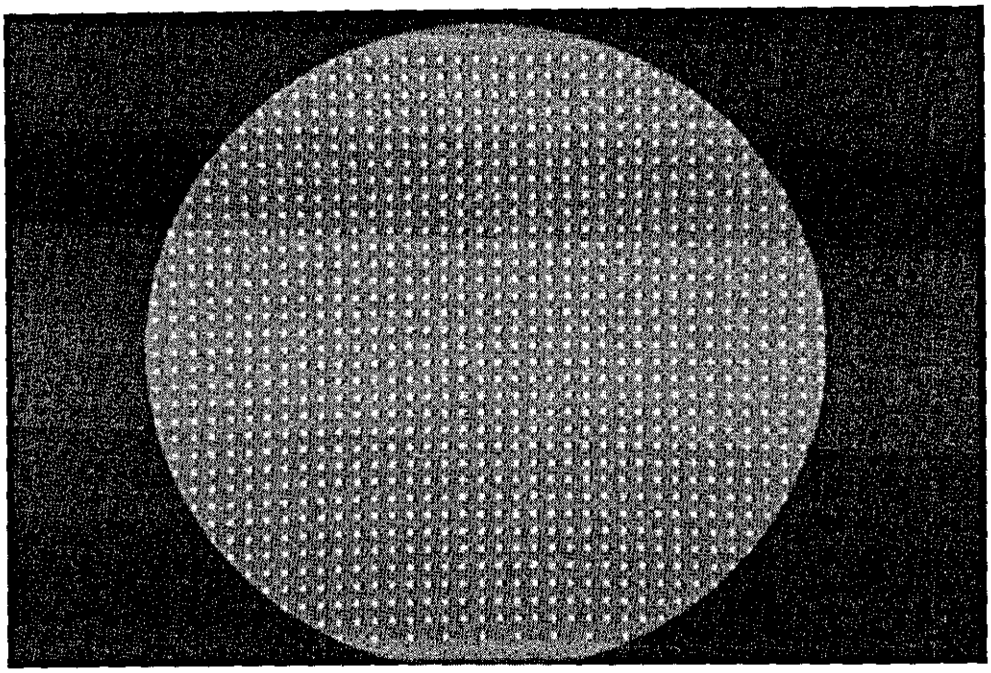


Figure 3.17 - Radioscopic Image of Distortion Phantom, 120mm Image Intensifier Setting.

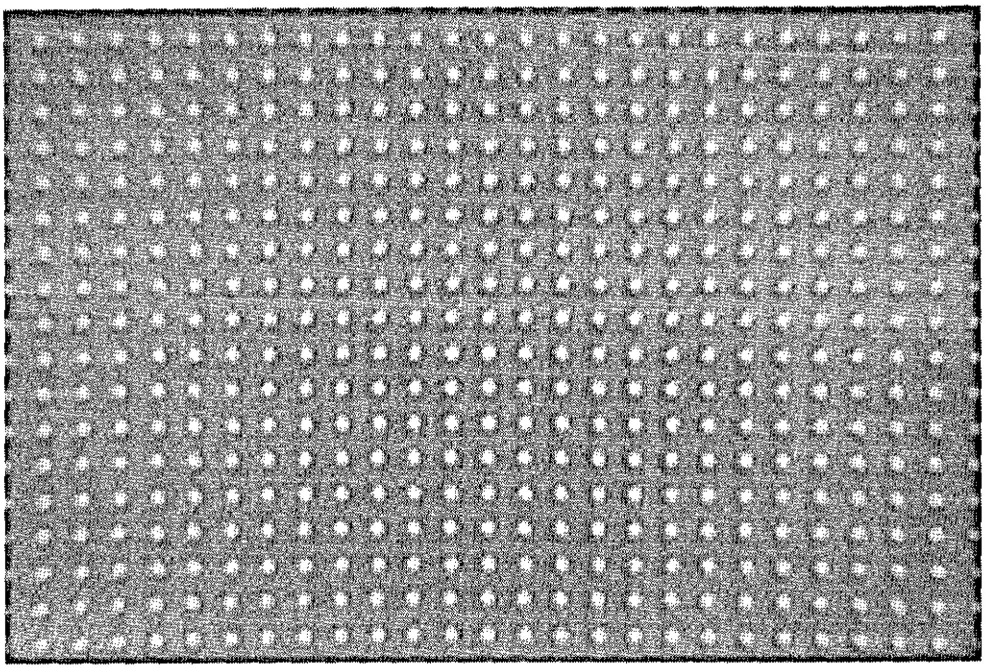


Figure 3.18 - Radioscopic Image of Distortion Phantom, 215mm Image Intensifier Setting, with Maximum Lens Zoom.

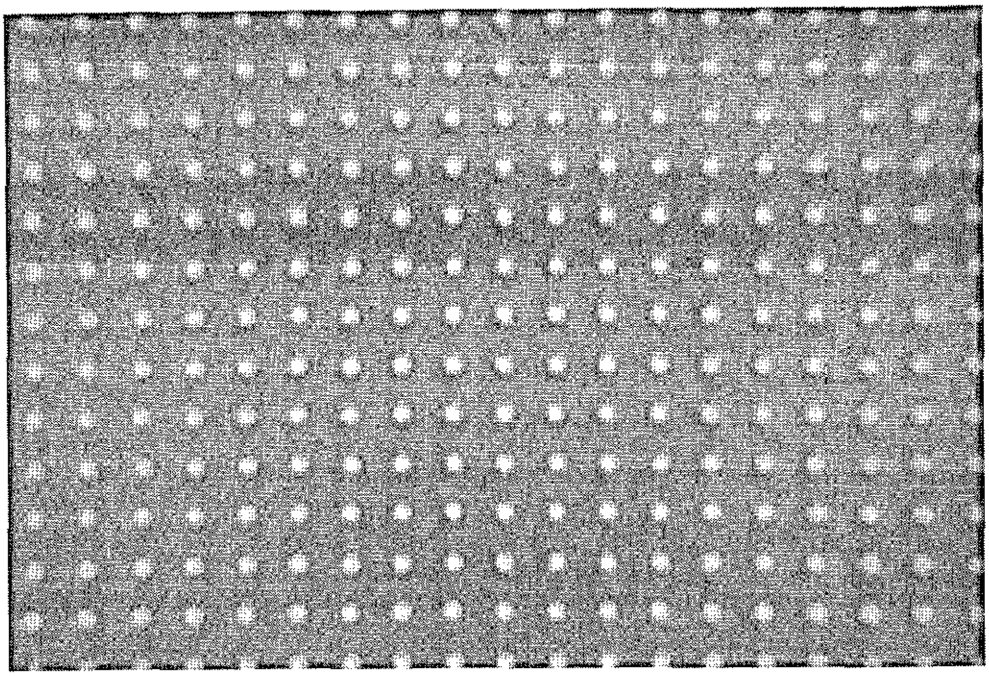


Figure 3.19 - Radioscopic Image of Distortion Phantom, 160mm Image Intensifier Setting, with Maximum Lens Zoom.

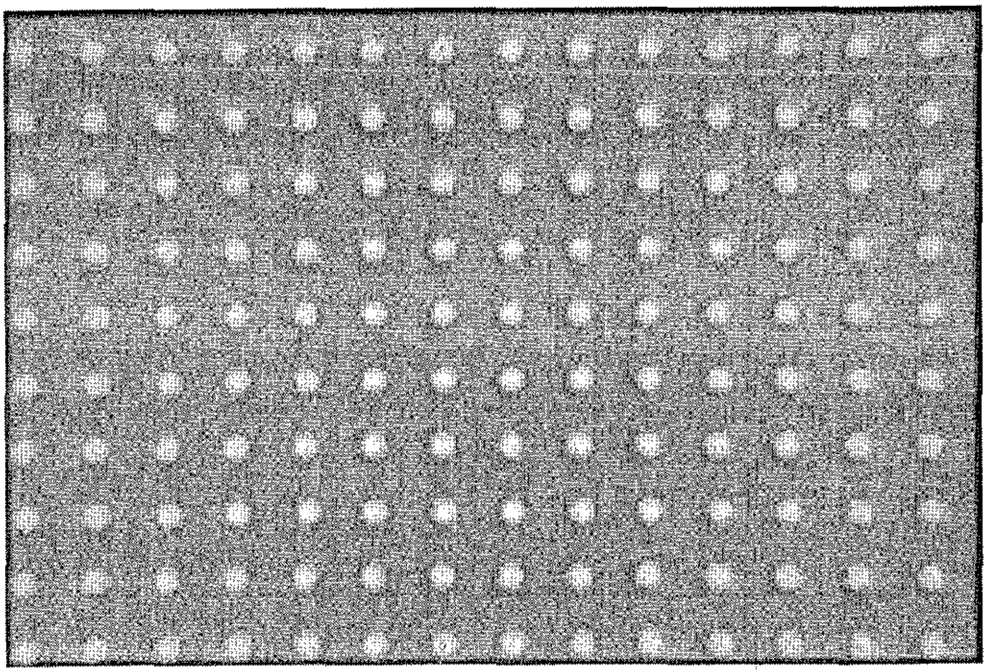


Figure 3.20 - Radioscopic Image of Distortion Phantom, 120mm Image Intensifier Setting, with Maximum Lens Zoom.

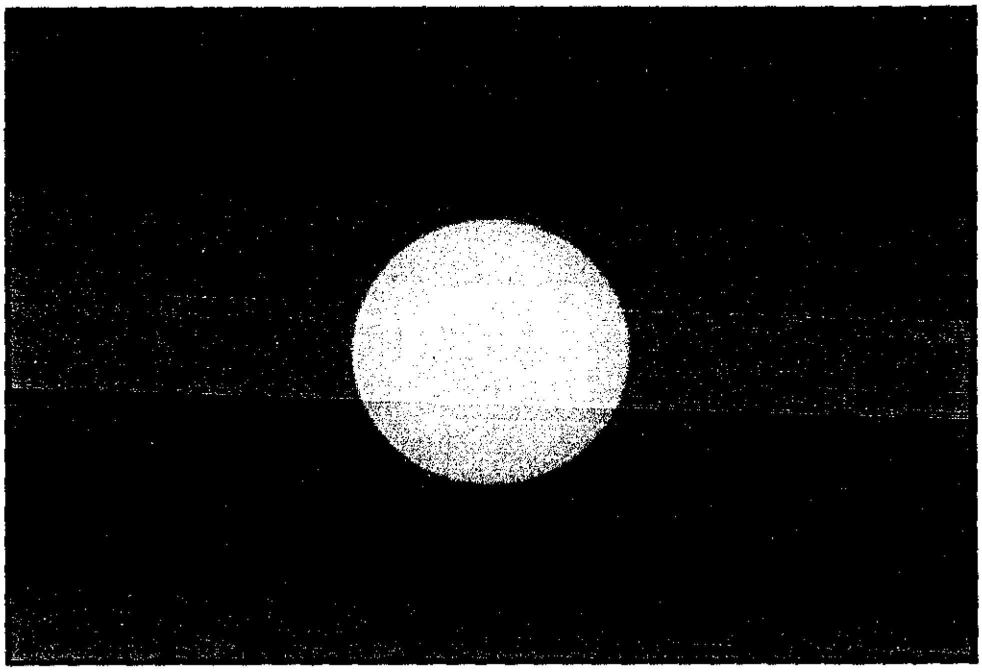


Figure 3.21 - Radioscopic Image of Distortion Phantom, 215mm Image Intensifier Setting, with Minimum Lens Zoom.

3.3.4 - Distortion

The camera and lens were tested for distortion before they were fitted in the X-ray cabinet. A grid of lines and circles was produced using a laser printer as an image target. Figure 3.22 shows an image taken using the camera with the lens fully zoomed out, and figure 3.23 is with the lens fully zoomed in Both images were taken with the VS-100 board and the Image Pro software. The camera was aimed at the centre of the target, but the software only saves the top left 512 x 480 pixels from the 768 x 512 image produced by the camera. Figures 3.22 and 3.23 display minimal distortion across the target when compared to the distortion in the radioscopic images (for example figure 3.15).

The radioscopic images of the grids show signs of pin-cushion distortion (in particular figure 3.15). The distortion across the image was measured using the software described in section 4.2. The software found the location of each hole in the grid. In figures 3.24 to 3.26 the distance between adjacent holes is plotted against the average distance from the centre of the image for the three images in figures 3.18 to 3.20 respectively. The figures show a least squares polynomial fit to each set of data. The 120mm zoom setting was best represented by a second order polynomial fit, whilst the 160mm and the 215mm settings used a fourth order equation.

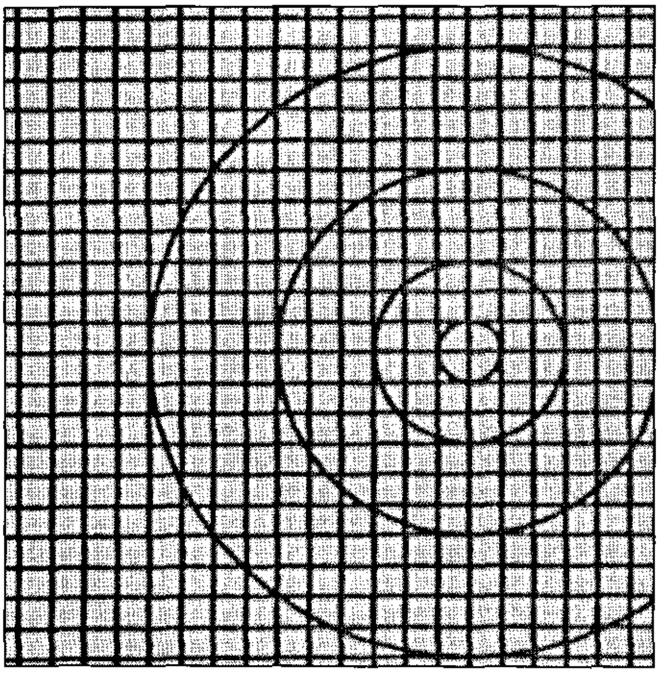


Figure 3.22 - Printed Distortion Test Image, Minimum Lens Zoom. Image cropped on left hand side by acquisition software.

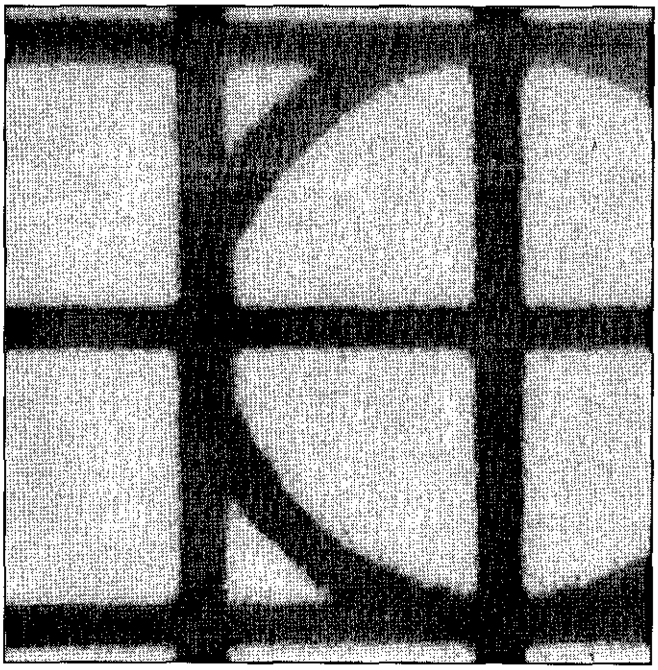


Figure 3.23 - Printed Distortion Test Image, Maximum Lens Zoom. Image cropped on left hand side by acquisition software.

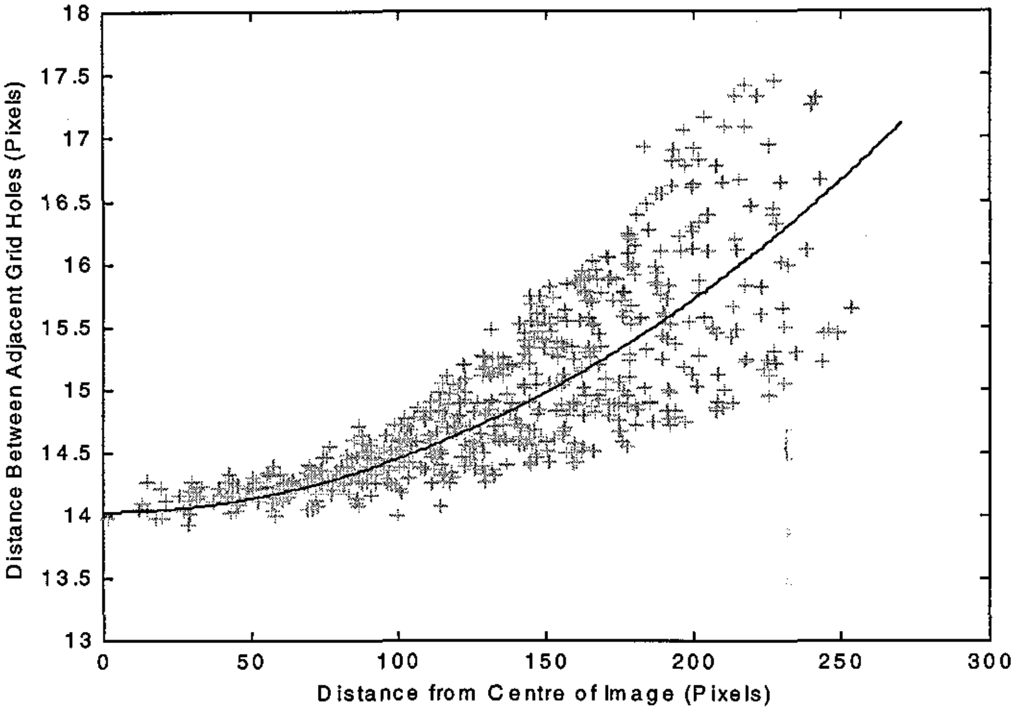


Figure 3.24 - Plot of Distance Between Holes in Radioscopic Images Taken with the 215mm Image Intensifier Setting.

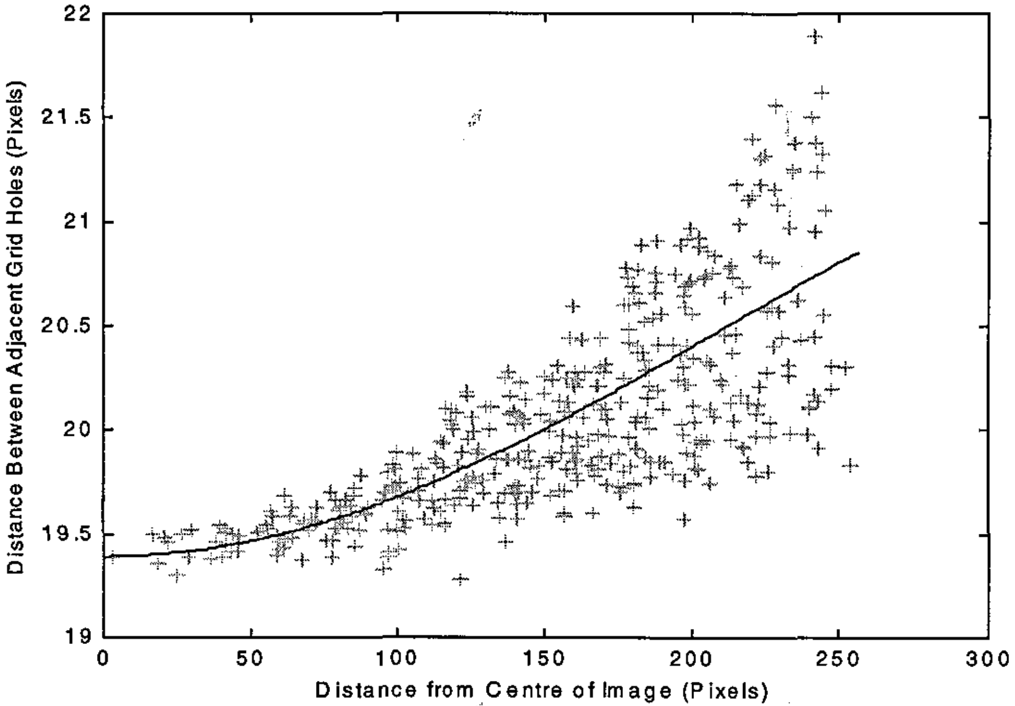


Figure 3.25 - Plot of Distance Between Holes in Radioscopic Images Taken with the 160mm Image Intensifier Setting.

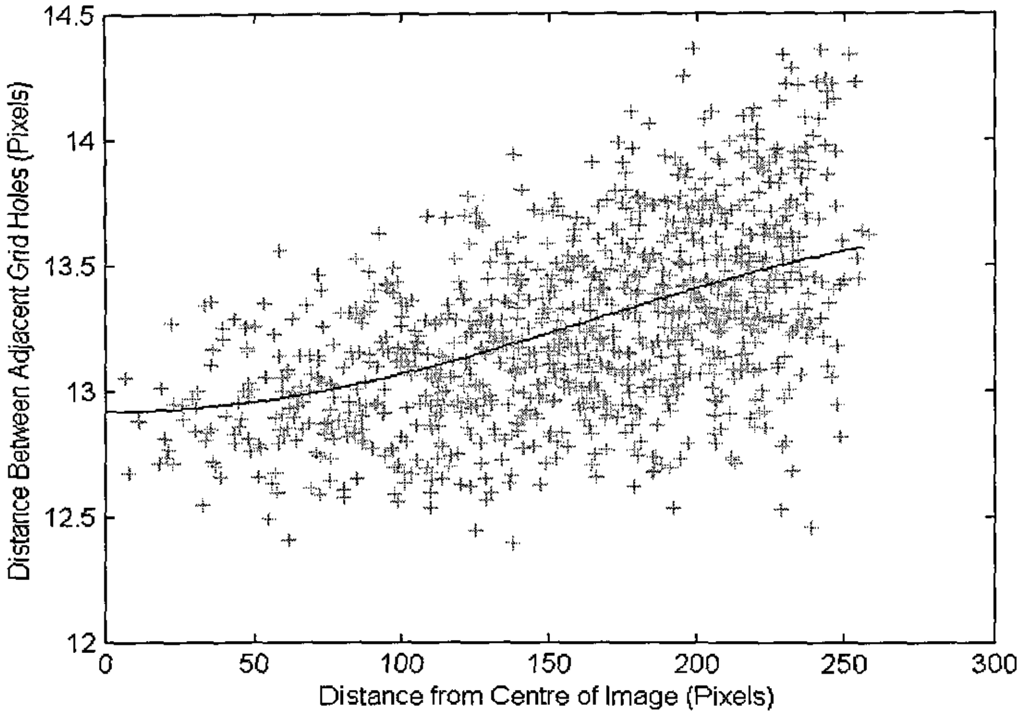


Figure 3.26 - Plot of Distance Between Holes in Radioscopic Images Taken with the 120mm Image Intensifier Setting.

Using the polynomial fits, the increase in line lengths at 90% of the radius of the target area is 4% for the 120mm zoom setting, 6% for the 160mm zoom setting, and 18% for the 215mm zoom setting. For comparison, the specification gives values of 3%, 6% and 15% respectively. For accurate measurement of displacements in radioscopic images, the distortion needs to be corrected. This is covered in chapter 4.

3.3.5 - Brightness Across the Image

The image intensifier specification states that the brightness at 90% of the radius is 80% for the 215mm zoom setting, rising to 95% for the 120mm zoom. This is measured as a percentage of the brightness at the centre of the image. These figures were checked with a number of aperture settings. The lens zoom was adjusted so that the left hand edge of the image intensifier was visible. Images were taken with different aperture settings for each of the three image intensifier zoom settings. The target was the Cu W13 IQI place at the right hand side of the image. Plots of a cross-section from each of these images are in figures 3.27 to 3.29. These plots are shown as a percentage of the average brightness over the centre 15% of the image, and have been passed through a third order Butterworth filter to remove noise (cut off for wavelengths over 25 pixels).

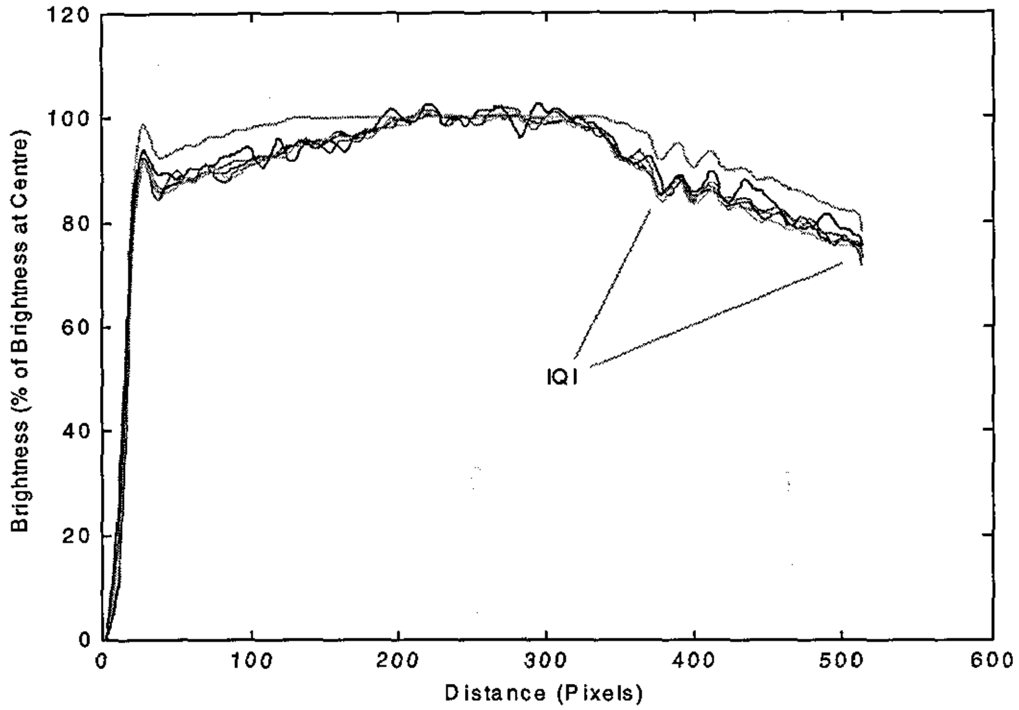


Figure 3.27 - Grey Level Cross Section of Brightness of Radioscopic Images. Taken with Varying Apertures, and the 120mm Image Intensifier Setting. Plots are low pass filtered. The images include the edge of the target area at the left hand side, and an IQI at the right hand side.

Examining the results, it is noted that each zoom setting has one plot with the brightest region "clipped", which means the average brightness over the centre region is lower than it should be, and the rest of the image is comparatively brighter. Ignoring the clipped images, each zoom setting has a consistent brightness profile across the target area, irrespective of aperture setting.

Using the filtered results, the brightness of the target with the 215mm zoom setting at 90% of the radius of the target is 70% of the brightness at the centre, the brightness with the 160mm setting is 85%, and with the 120mm setting it is 87%. The figures are worse than the figures quoted for the image intensifier. This discrepancy can be partly explained by the variation in X-ray power across the target, as described in section 3.3.1.

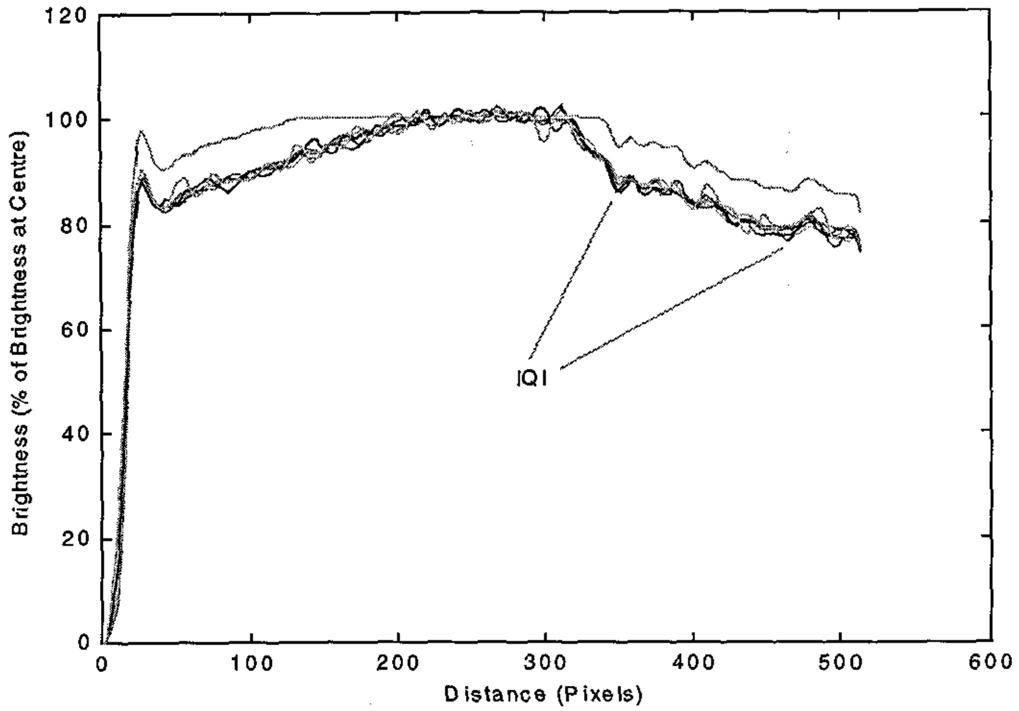


Figure 3.28 - Grey Level Cross Section of Brightness of Radioscopic Images. Taken with Varying Apertures, and the 160mm Image Intensifier Setting. Plots are low pass filtered. The images include the edge of the target area at the left hand side, and an IQI at the right hand side.

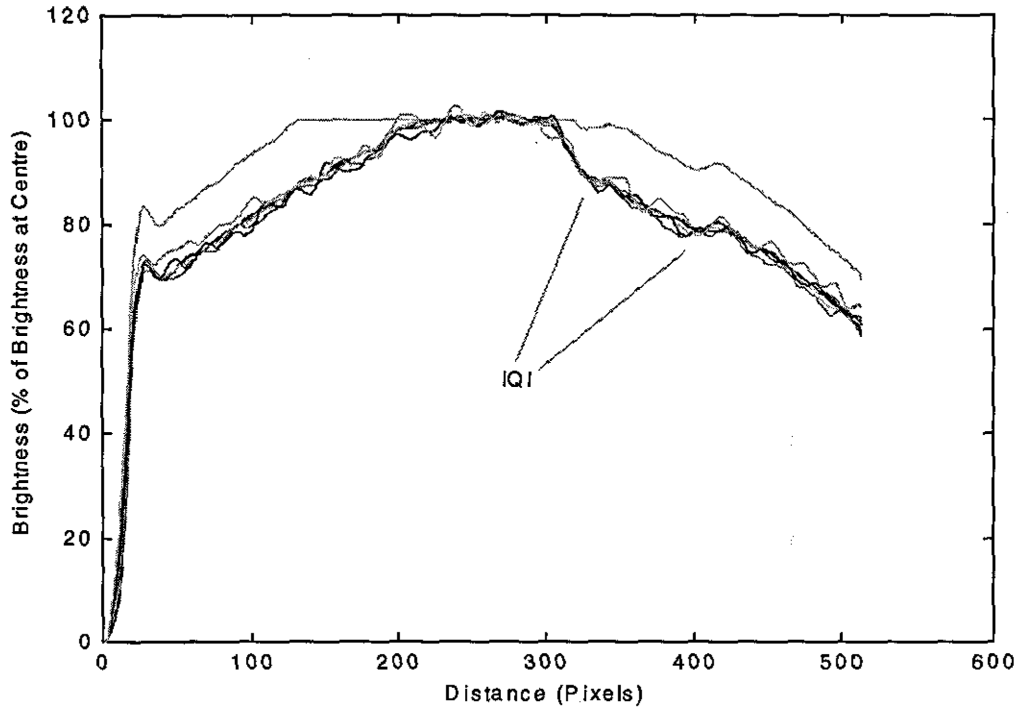


Figure 3.29 - Grey Level Cross Section of Brightness of Radioscopic Images. Taken with Varying Apertures, and the 215mm Image Intensifier Setting. Plots are low pass filtered. The images include the edge of the target area at the left hand side, and an IQI at the right hand side.

The results also indicate the location of the IQI, which appears as a slightly darker section to the right of the images. On the 215mm zoom setting the whole of the IQI is visible between the 300 and 400 pixel marks. The 160mm setting just gets the whole width of the IQI in the region of 350 to 500 pixels. With the 120mm zoom setting the wires are clearly visible, but the right hand side of the IQI is off the target area. The inverse shadow is particularly noticeable on the left hand side of these images.

Three images were taken with the same aperture to compare the brightness of the image intensifier zoom settings. Figure 3.30 shows cross sectional plots of these images, without filtering, which makes the IQIs more visible. Comparing the plots at the centre of the target area, the 215mm zoom setting is twice as bright as the 160mm zoom setting, and about 3.5 times brighter than the 120mm setting.

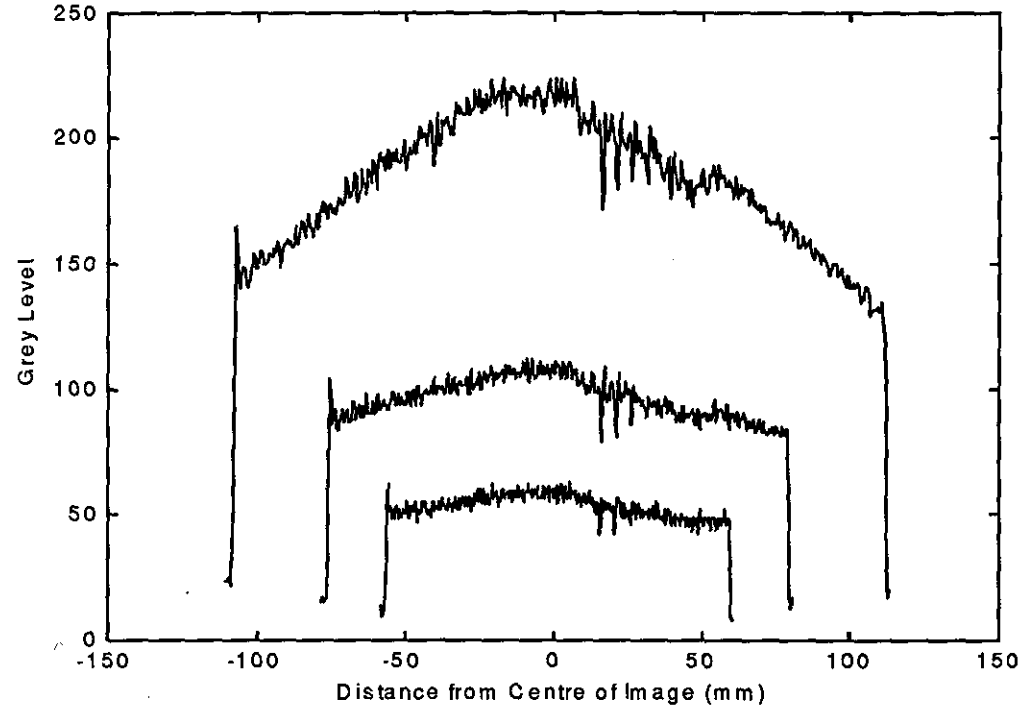


Figure 3.30 - Grey Level Cross Section of Brightness of Radioscopic Images taken with all Three Zoom Settings. Images taken with constant lens settings, and with an IQI on right hand side.

3.3.6 - Aperture

The lens does not have a closed loop control for the aperture, which makes it impossible to accurately repeat aperture settings. However, for a given radioscopic target, it is possible to estimate the aperture by finding the shutter speed or integration time required to produce a consistently bright image. This feature was used in an experiment to see how the aperture affects the sharpness of the radioscopic images.

The experiment was undertaken with the rubber-tungsten sample (see section 5.3) and the maximum X-ray energy setting. The aperture was closed to the smallest setting (F22 according to the lens specification) and an integrated radioscopic image was taken. The shutter time was chosen so that the maximum pixel intensity of the output was as close to the maximum intensity as possible. This first radioscopic image was taken over 14 frames and has a maximum pixel intensity of 244. Eleven further radioscopic images were taken, with the shutter open for between one and twelve frames, and in each case with an aperture chosen so that the maximum pixel intensity was in the range of 241 to 245.

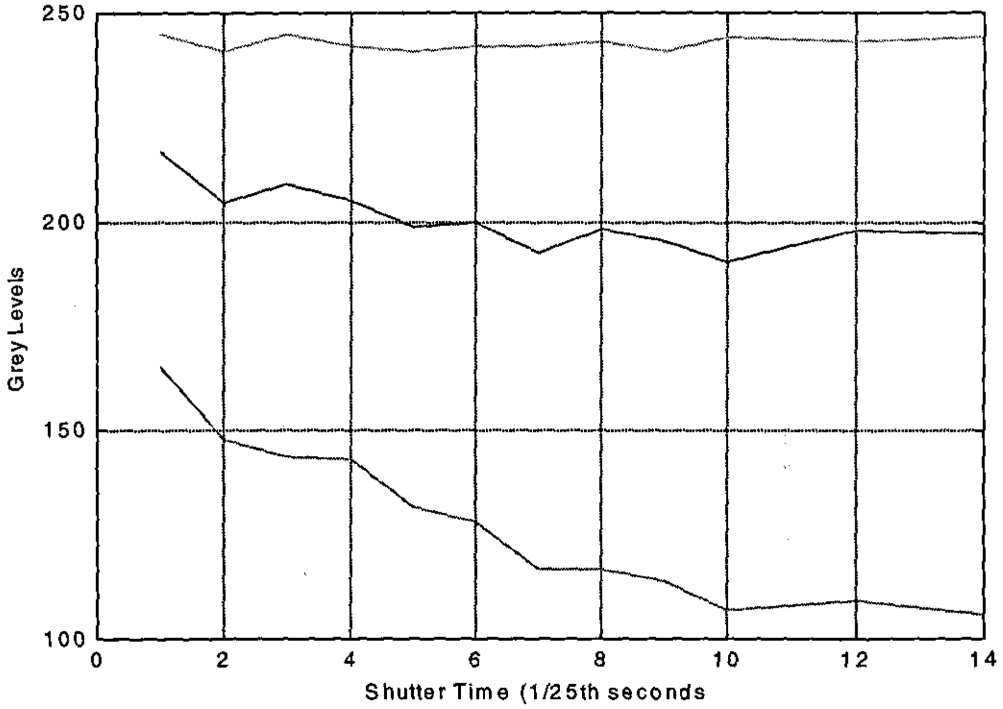


Figure 3.31 - Grey Levels for Images Taken with Different Shutter Times. The apertures are adjusted to keep the maximum grey level constant. Maximum: —, Minimum: —, Mean: —.

Figure 3.31 plots the maximum, mean and minimum grey levels of the twelve images. It can be seen that the images taken with longer shutter times have a greater contrast. There is some reduction in the mean level of brightness as the shutter time increases. Maintaining a constant mean level of brightness is likely to result in a more consistent relationship between the aperture setting and the shutter time. This measure was not used as the aperture had to be found by trial and error, and the 286 computer took about twenty seconds each time to find just the maximum and minimum intensity levels, without also calculating the mean. Using the maximum brightness measure had the advantage of ensuring the images were not clipped.

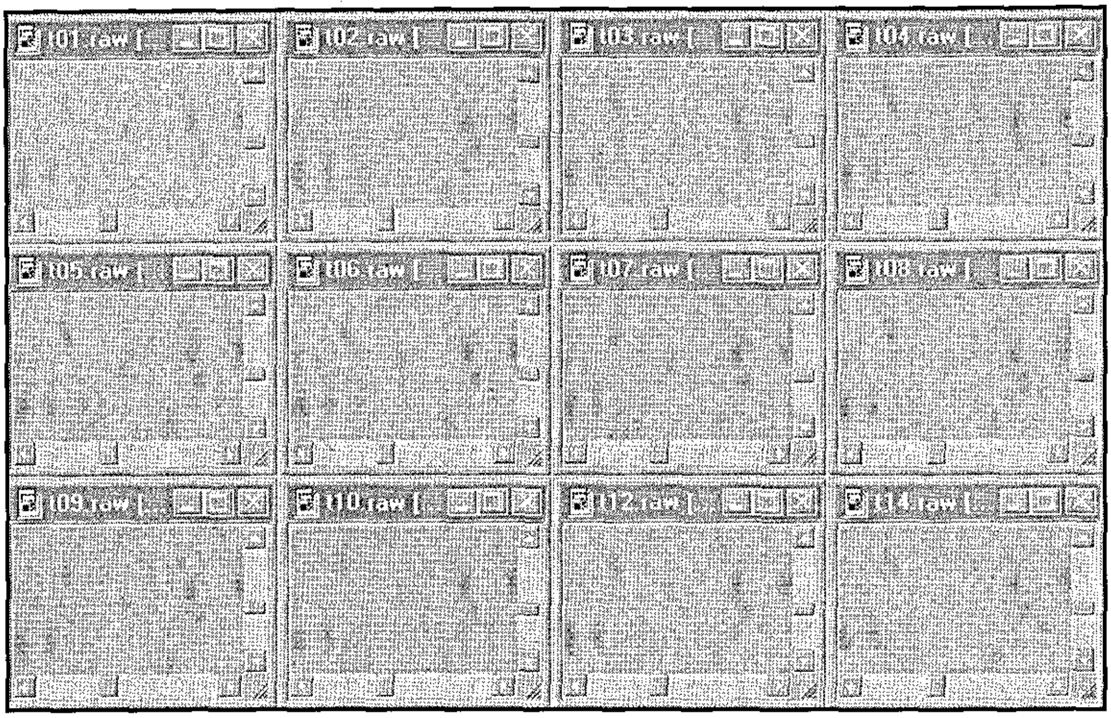


Figure 3.32 - Examples of Images Taken with Different Shutter Times. The number in the title in each section is the number of frames the image was integrated over. The apertures were adjusted to maximise the maximum grey level.

The larger the aperture F number, the smaller the lens aperture. Doubling the radius of the aperture halves the F number, and increases the brightness fourfold. Assuming all the images were produced by the same amount of light, and the output to the image detector was proportional to the amount of light received, halving the shutter time (reducing by one stop) was equivalent to reducing the F number by $1/\sqrt{2}$ (increasing by one stop). This meant that a shutter speed of seven frames was equivalent to an aperture of F16, whilst a F11 was approximately equal to a shutter speed of 3.5 frames. A shutter speed of one frame required an aperture of approximately F5.8.

Figure 3.32 shows the same portion of the twelve radioscopic images. The number in the title of each image indicates the number of frames for which the shutter was open. The most noticeable feature of these images is the horizontal noise on the images acquired with the integration option. This noise is not evident on the image taken over one frame. Close examination shows that every other line is slightly darker, indicating that the odd and even fields are poorly matched. This noise was not evident on the images taken with the VS-100 board, which indicates it is caused by the FG-100 board, or the software modifications needed to operate the FG-100 board. It could be caused by grabbing one more even field than odd fields, but this would result in the noise being more pronounced in the images taken with smaller shutter times, whereas the images with longer shutter times display, if anything, slightly more noise. It is therefore concluded that the FG-100 board has a fault that only affects integrated images.

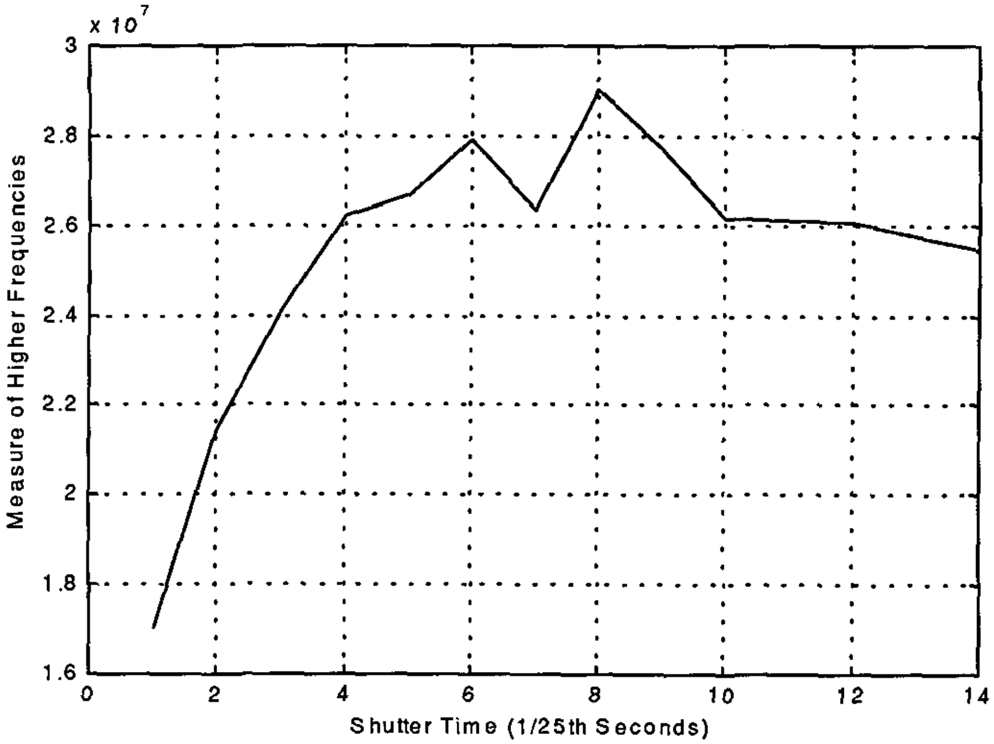


Figure 3.33 - Measure of High Frequency Detail in Integrated Radioscopic Images. The measure of high frequencies is the sum of the PSD for frequencies with wavelengths less than 10 pixels.

However it is still possible to compare the sharpness of the images. The images with larger apertures are slightly more blurred than those with smaller apertures. Visually, the 8 frame image looks to have the best results, but there is very little to choice between any of the images taken over more than five frames. The sharper images can be identified by more pronounced changes in the grey scale between neighbouring pixels. This means the images have more detail in the higher frequencies. This can be measured by taking the power spectral density (PSD) over every other line in the image. By restricting the samples to the even lines, the odd/even frame acquisition noise can be neglected. The sum of the PSD for frequencies with wavelengths of less than 10 pixels are compared in figure 3.33. It can be seen that the 6 frame and 8 frame images have the best results, with a definite drop off in the performance for the images taken over less than 4 frames (which approximately equates to an aperture of F12).

Smaller apertures are expected to result in sharper images for a fixed target. An aperture of less than F11 would appear to be necessary for the sharpest image, but other factors would have affected the final result. Smaller apertures require longer integration times, and are therefore more susceptible to camera shake. Camera shake is often noticeable in the live radioscopic images, particularly soon after the equipment has been switched on (which requires a button to be pressed), or when one of the lens motors is activated. The loss of contrast in the images with longer integration times is probably caused by vibrations in the camera stand assemblies, and is best avoided by allowing the equipment to settle before taking an image.

3.3.7 - Shutter Speed and Integration Time

As shown in section 3.3.6, smaller apertures result in better focused images. The camera's integration feature allows smaller apertures whilst maintaining the image brightness. Obviously keeping the shutter open would produce blurred results if the target is moving, but no moving targets have been investigated. It is noted that longer shutter times resulted in the images suffering from additive noise (figure 3.34). The noise can be seen in images with shutter speeds as low as 2 seconds (figure 3.35), but is not really noticeable until the shutter is open for 4 seconds (figure 3.36). The locations of the areas with noise problems are constant, and the intensity increases with shutter time.

A series of images were taken with the X-ray source switched off. The locations of the additive noise were noted to be consistent across the images, and increased in intensity as the shutter time got longer. The standard deviation of these images is shown against shutter time in figure

3.37. The mean brightness value of these images (background noise) is around 4.5 grey levels, and it is noted that they suffer from a small quantity of the periodic noise that caused serious problems with the VS-100 board (figure 3.38).

Radioscopic images were taken of the rubber-tungsten sample to see how the shutter time affects the image brightness. Figure 3.39 shows the maximum, mean and minimum grey levels of images taken with varying shutter times, and with four different X-ray power settings. It can be seen that the brightness of the image is proportional to the shutter time within the limits of the grey scale range.

The best contrast between the maximum and minimum grey levels is when the shutter is set so that maximum value is as close as possible to the grey level limit of 256. Detail may be lost by clipping when the maximum grey level is 256, as this could indicate that some of the image is being over exposed.

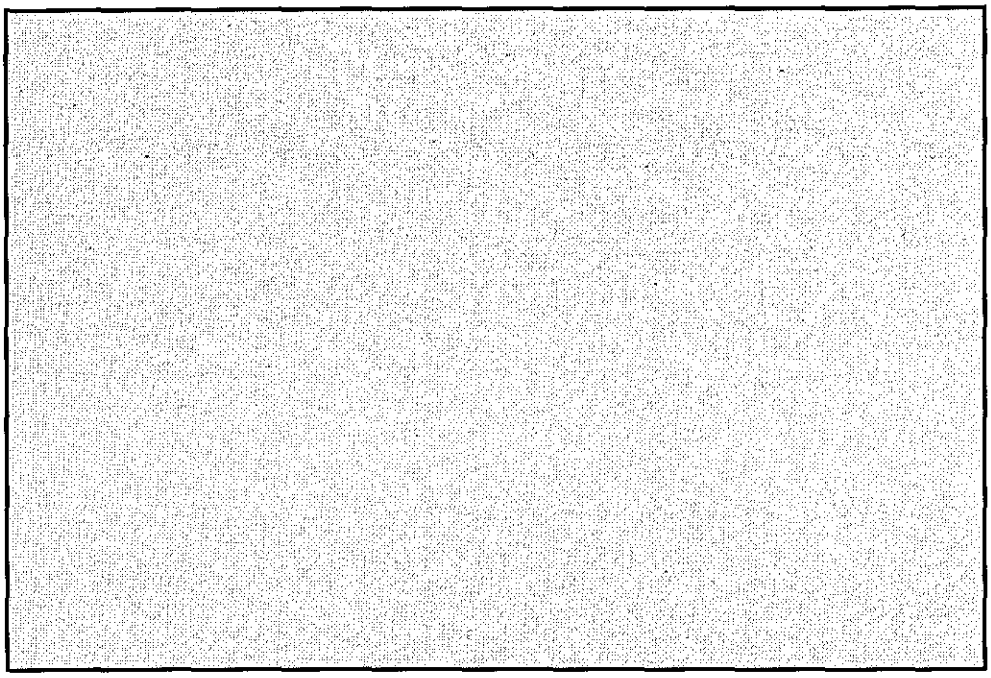


Figure 3.34 - Example of Integrated Image Suffering from Additive Noise. Image taken with X-ray source and image intensifier switched off. Grey levels have been inverted.

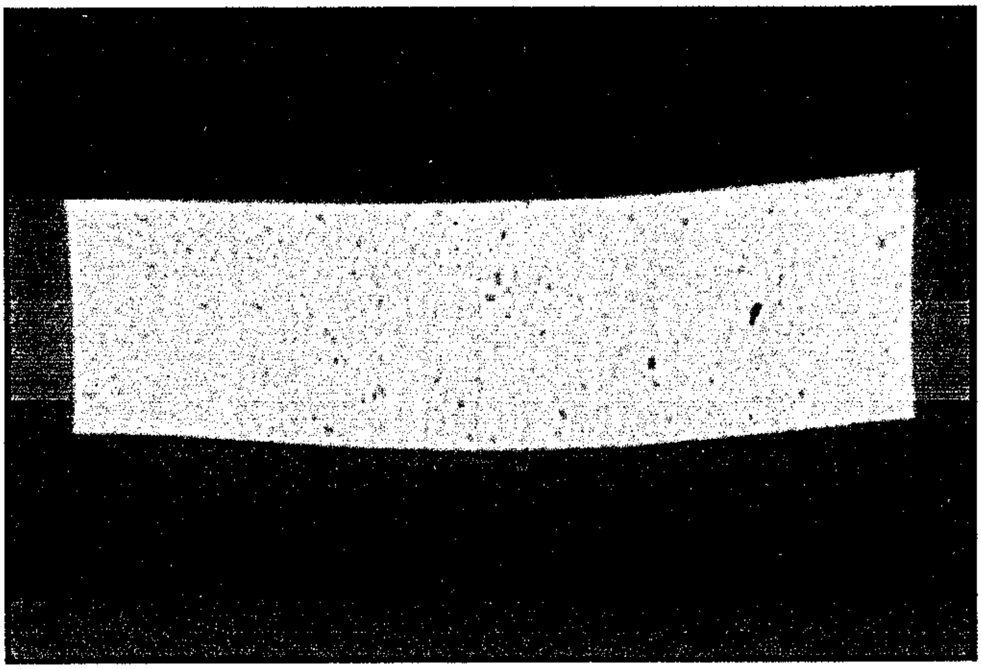


Figure 3.35 - Radioscopic Image of Ceramic Sample with Visible Additive Noise. Taken with a Shutter Time of 100 Frames (4 seconds). Sample is rectangular, with four raised sections. The image contains darker regions caused by more absorbent elements in the ceramic.

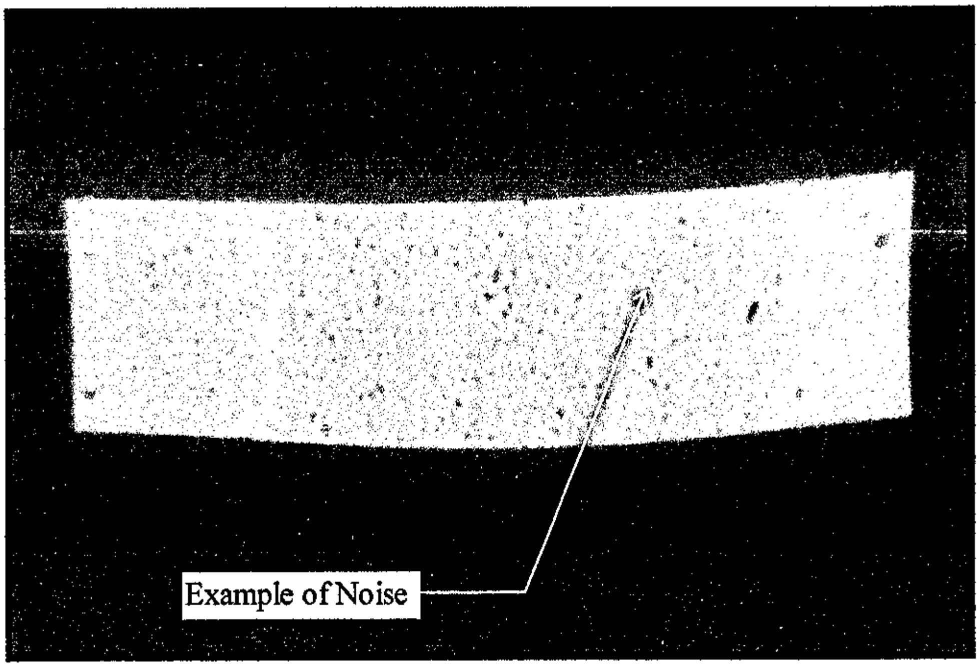


Figure 3.36 - Radioscopic Image of Ceramic Sample with Obvious Additive Noise. Taken with a Shutter Time of 200 Frames (8 seconds). Note the locations of pixels suffering from noise are in same location as those in figure 3.34.

Section 3.3.6 also tests a variety of shutter speeds, and notes that the integration function resulted in differences in the odd and even fields. However, grabbing a single frame does not suffer from this noise. Because of the dim image produced by the image intensifier, the shorter shutter times selected with the BCD switch are of little use when taking radioscopic images.

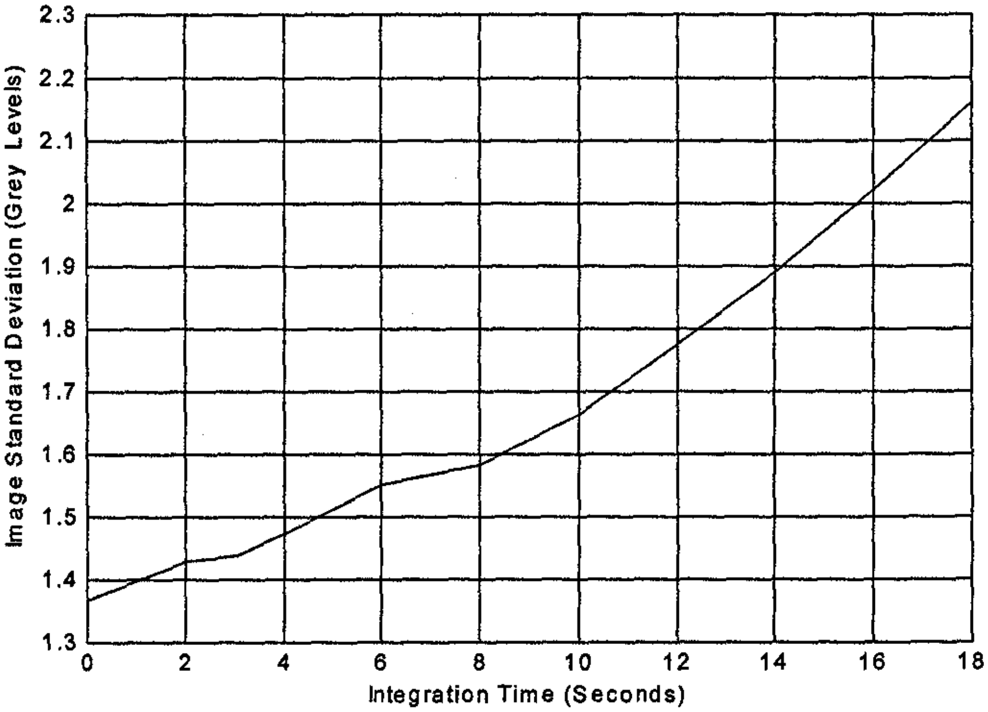


Figure 3.37 - Measure of Additive Noise. Images were taken with X-rays switched off. Measure is the standard deviation (STD) of the grey levels of the images.

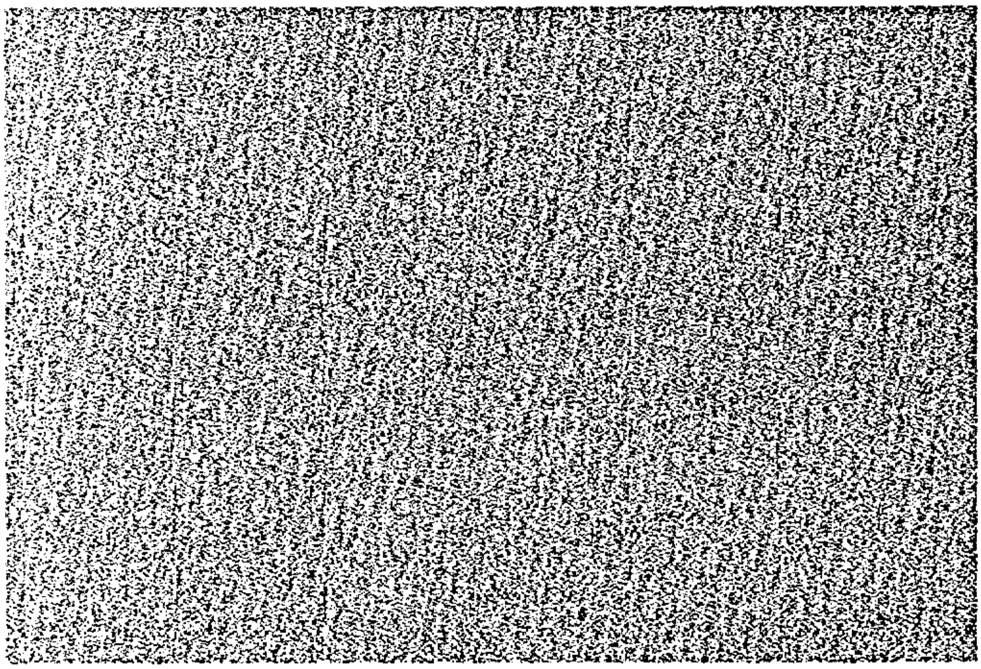


Figure 3.38 - Radioscopic Image Displaying Periodic Noise. Image taken with FG-100 board, over 500 frames (20 seconds) and the histogram was equalised.

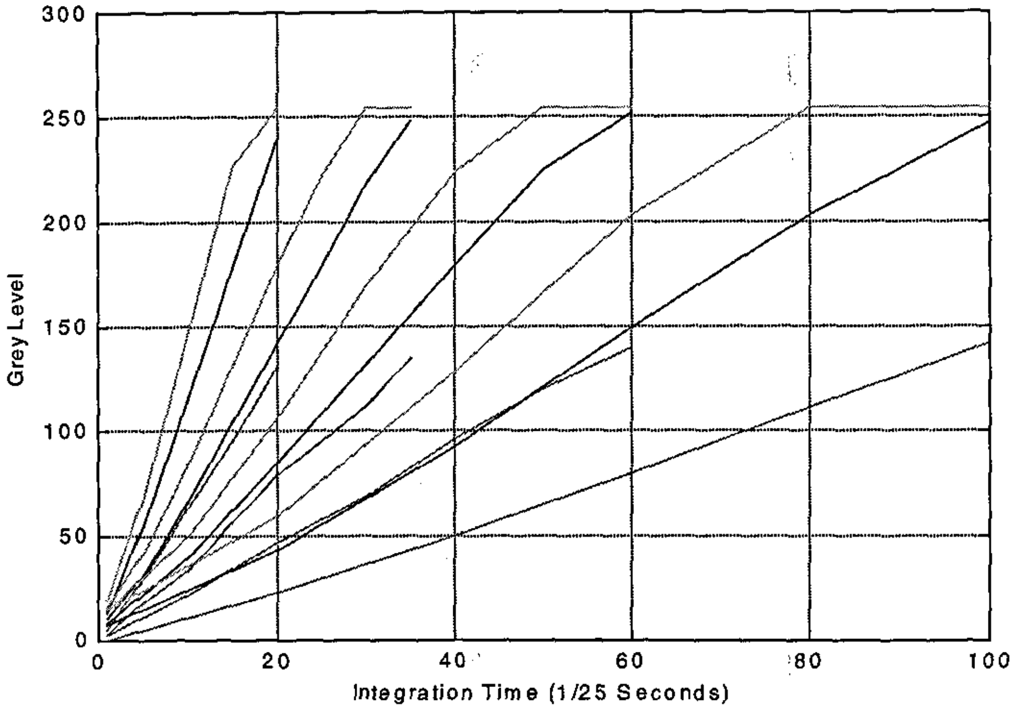


Figure 3.39 - Brightness of Integrated Radioscopic Images. Four sets of images, each with a different aperture. Maximum: — , Minimum: — , Mean: — .

3.3.8 - X-Ray Energy

The X-ray machine is able to provide a maximum X-ray energy of between 50keV and 90keV. This is controlled by a dial that has nine marks. In this thesis, these settings are referred to as setting -1 (off), setting 0 (minimum), setting 1 (the first mark, just a little bit more than minimum) to setting 9 (the last mark, almost maximum), and setting 9.2 (maximum, which is when the dial is at its stop, just beyond setting 9). The first test on the X-ray setting was to see how it affects the brightness of the output image with a constant aperture and shutter speed. Images were taken of an IQI with the lens adjusted so whole target area was visible, and with the 215mm zoom setting. Figure 3.40 plots the grey level range against X-ray setting. These results show that the full power image is about ten times brighter than setting 5, and that the images below setting 3 are little brighter than the background noise level.

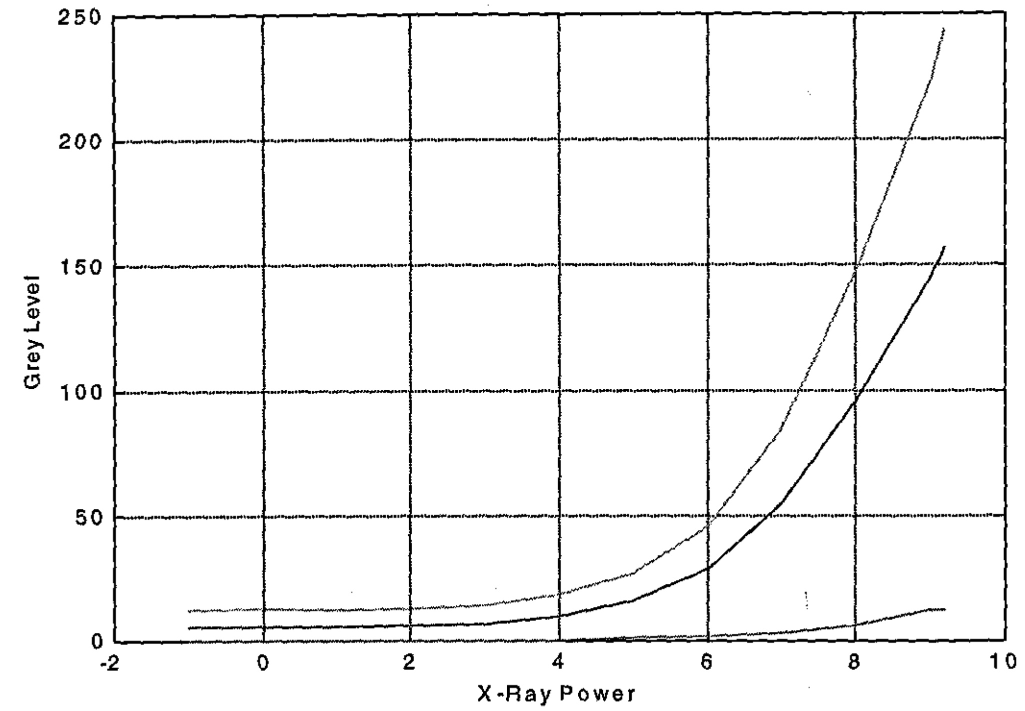


Figure 3.40 - Brightness of Images Against X-Ray Power. Images taken with constant aperture. Maximum: — , Minimum: — , Mean: — .

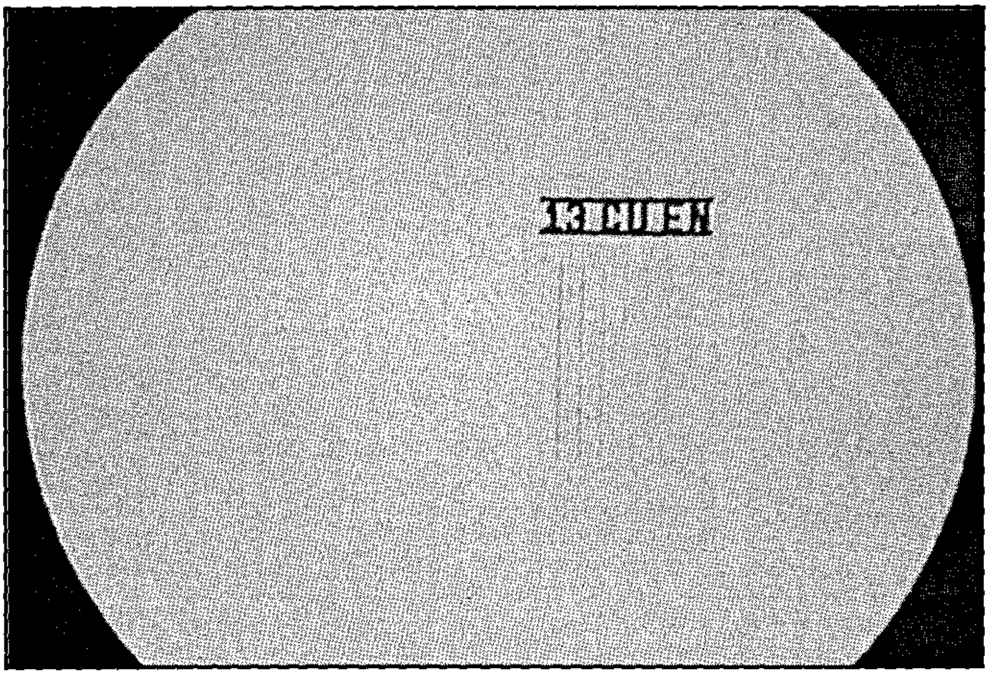


Figure 3.41 - Radioscopic Image of IQI at full X-Ray Power. Image taken over one frame.

Whilst the lower power settings result in dimmer images, they can be used to pick up details not visible at the higher settings. Figure 3.41 is a radioscopic image of the IQI with full power and a 1 frame shutter time. Figure 3.42 is the same image with a power setting of 2, and a shutter time of 12 seconds. The lower power setting detects the plastic IQI cover far better than the full power image. It is also noted that the lower power image has a greater variation of brightness across the image, and suffers from additive noise.

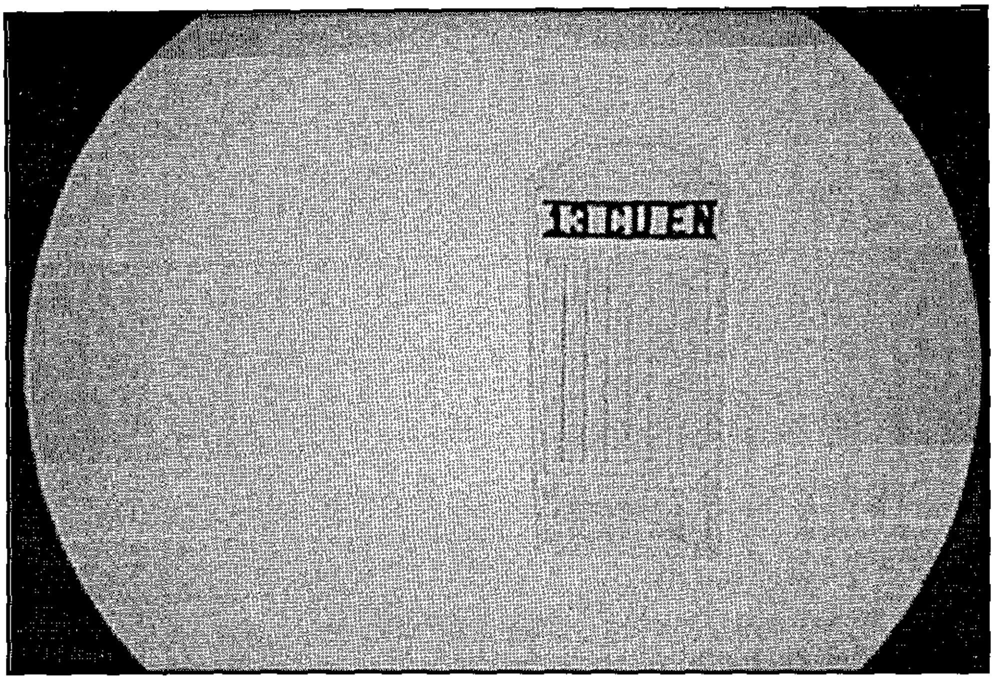


Figure 3.42 - Radioscopic Image of IQI at X-Ray Power Setting 2. Image taken over 300 frames (12 seconds).

3.4 - Random Noise

The radioscopic images consist of a true representation of the target, combined with signals from a number of noise sources. These noise sources can be divided into two types, constant noise and random noise. Constant noise sources are those that do not change between two images taken with the same settings. These sources include geometric distortion (see section 3.3.4), intensity variations across the image (section 3.3.5), the "inverse shadow" feature (section 3.3.2) and any dirt that may be on the optical equipment or in the X-ray chamber.

Random noise sources cause variations between two images taken with identical settings. Investigations show the random noise element includes a time dependent variation, frequency dependent noise, quantisation noise and intensity dependent variation. A random noise image can be obtained by taking the difference between two images taken under identical conditions.

Assuming the random noise images added to each of the radioscopic images are independent, the variance of the difference between the images is the sum of the variances of the two original images [126]. Thus the standard deviation of the noise components in the images is the standard deviation of the difference between the images divided by root two. However this is only valid if the noise images are independent. Although the noise is intensity dependent, an independent noise source can be obtained from images where the true image is uniform.

3.4.1 - Time Dependent Variations

Three sets of images were made to see how the system's output changes with time. The first set was with radioscopic images with no target in the X-ray chamber. The second set was radioscopic images of the rubber-tungsten sample, and the third set was with the camera pointed at a piece of white card. Images were taken fifteen seconds after switching on the X-rays (in the case of the third test, after switching on the camera), and every subsequent thirty seconds. The tests were run for ten minutes, as this is the longest time the X-ray machine can operate continuously. After each test, the equipment was left to cool down. Figure 3.43 plots the mean grey levels of the images, against time.

In both the radioscopic tests, the mean grey level decreases by one level approximately every thirty-six seconds. The third test has a more random nature, which could be explained by variations in the output of the bulb illuminating the card as a result of external power fluctuations. The tests indicate there is fading of the image with time, due to the X-ray source or the image intensifier. As the tests failed to show a steady state value within the time scale tested, all the radioscopic images must be assumed to have an unknown variations due to the time dependent image fading. This error is best reduced by taking the images in as short a time as possible.

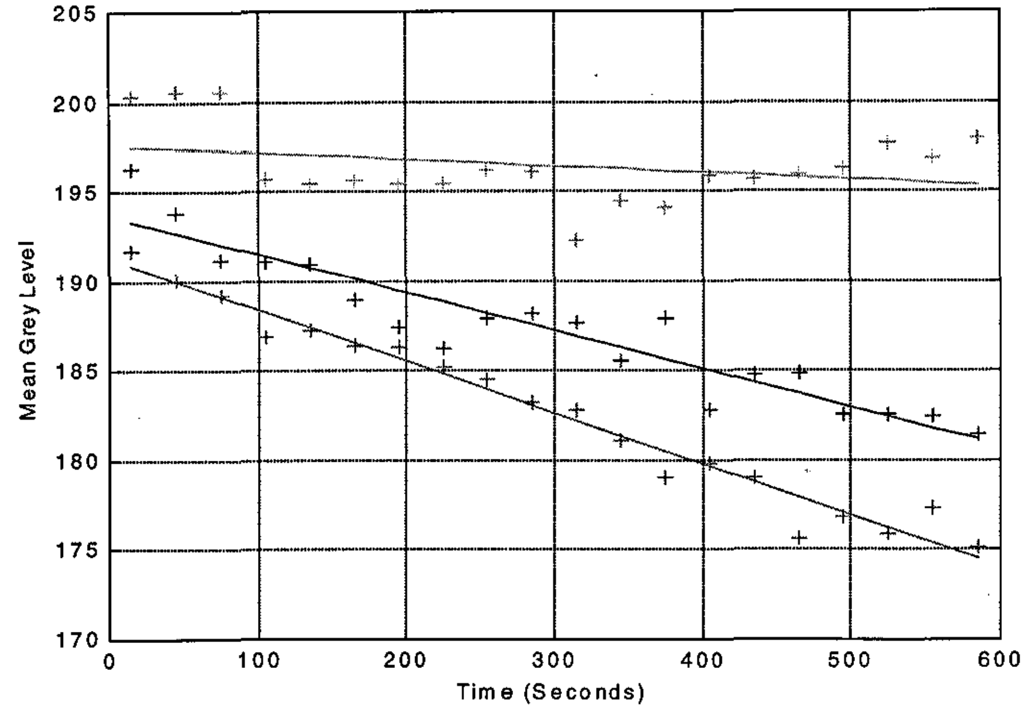


Figure 3.43 - Images Fading with Time. A plot of the mean grey level of images taken at timed intervals after switching on the equipment. Non Radioscopic Images: ······ , Blank Radioscopic Images: — , Rubber-Tungsten Radioscopic Images: — .

3.4.2 - A/D Conversion Noise

A plot (figure 3.44) was made of the grey level histograms of a number of the images taken for the tests in section 3.4.1. It can be seen that all the histograms peak at the same grey levels. These peaks are independent of the mean grey level of the images and are present in both radioscopic and normal images. The histograms could be explained by the A/D converter in the grabber board suffering from differential non-linearity [127]. This is when digital code increments take place at uneven increments in the input signal, which for a random input signal, results in some grey levels being more common than others. Histograms of images taken with a different model of camera but grabbed with the same board were also noted to have the same peak grey levels.

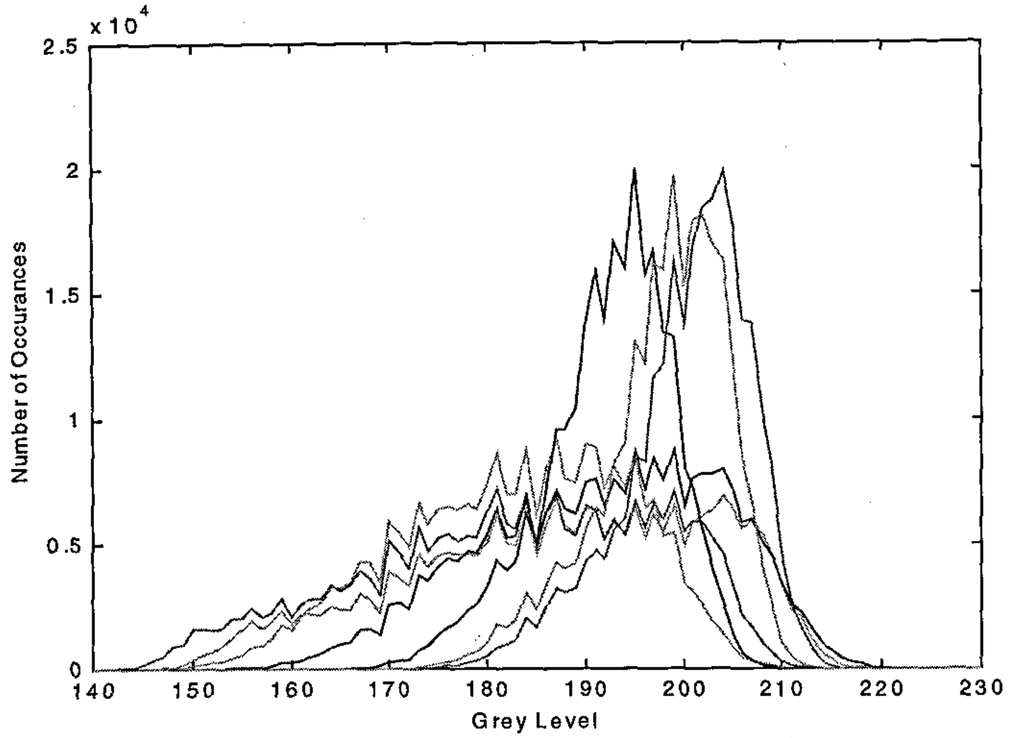


Figure 3.44 - Histograms of Grey Levels of Images. Histograms are of two blank radioscopic images, two radioscopic images of the rubber-tungsten sample, and three images of a blank piece of paper (Pulnix 525 camera) with normal illumination.

3.4.3 - Frequency Dependent Noise

A Fast Fourier Transform (FFT) was taken of a blank radioscopic image (figure 3.45). Of interest are the vertical intensity variations, which shows attenuation of higher frequencies in the horizontal direction. The attenuation is to be found in images taken with a second camera of non radiographic images. As the attenuation is in the direction of the image scanning, the variation indicates the scanned signal is subject to a frequency dependent filter in either, or both, the cameras and the image grabber. There would also appear to be some attenuation of low frequencies. This attenuation will result in noise in the final image.

The grey levels indicating the magnitude of the transform in figure 3.45 are displayed in a logarithmic scale to prevent the image being swamped by the DC signal. Figure 3.46 is an FFT of the difference between two blank images. Subtracting the images leaves only the transform of the two noise signals. A number of dark spots indicate there is likely to be some frequency dependent noise. The frequency dependent noise can be seen as a prominent and regular pattern in this image, which has normal scaling for the grey levels. This shows there is frequency dependent noise, and there is a phase shift between the noise in the two images. Further investigations with non radioscopic images, and a different camera indicated this noise source is not caused by the X-ray equipment, and probably originates in the image grabber board.

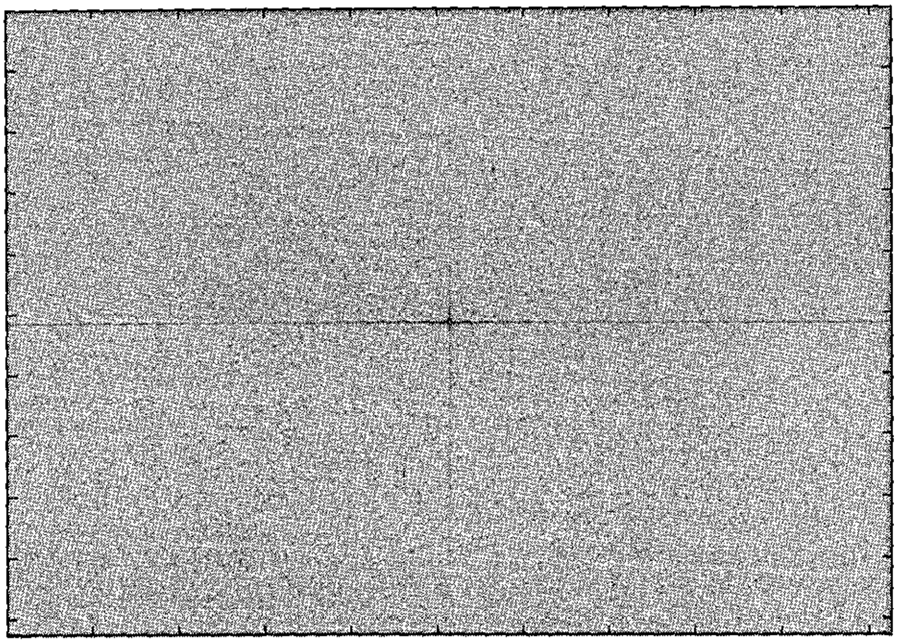


Figure 3.45 - Fourier Transform of Blank Radioscopic Image. Log grey scale. Zero frequency at centre.

The FFTs can be used to identify the frequencies of the noise. These frequencies can be deleted from a radioscopic image to remove this noise source. Figure 3.47 shows the FFT of the difference image after attenuating the sixteen most prominent frequencies. There are still signs of more frequency dependent noise, but the magnitude of the noise at these frequencies is very close to the background noise. It can be seen that the noise signal has been subjected to the same high frequency filtering as the image in figure 3.45. Tests on sixty-five pairs of images showed that removing the sixteen most noisy frequencies reduced the noise level by an average of 2%, with a maximum improvement of 5.3% and the minimum of 0%. The range of noise reduction is expected as it is dependent on the phase difference between the noise in the two images.

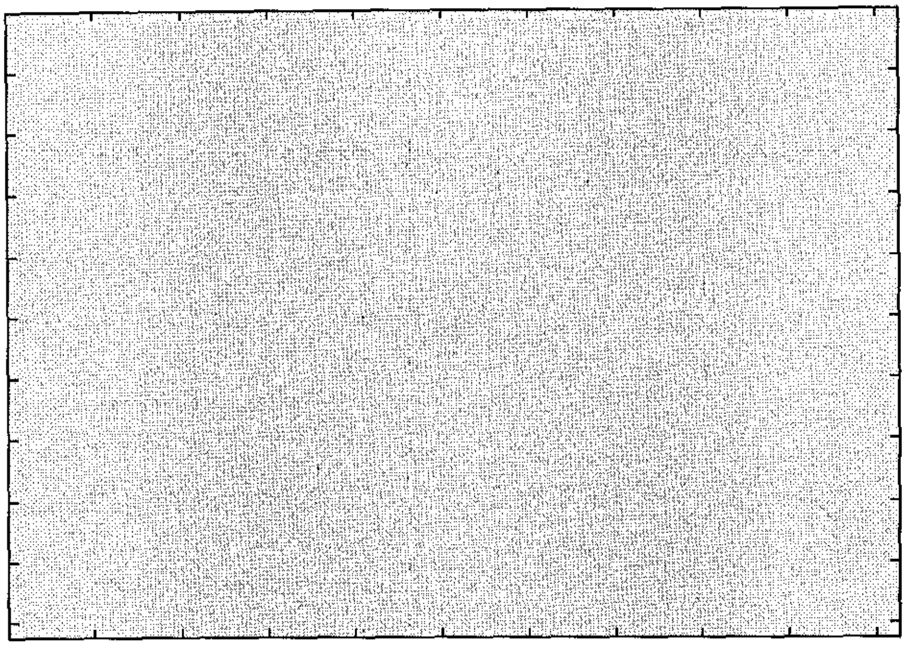


Figure 3.46 - Fourier Transform of the Difference between Two Blank Images. Zero frequency at centre.

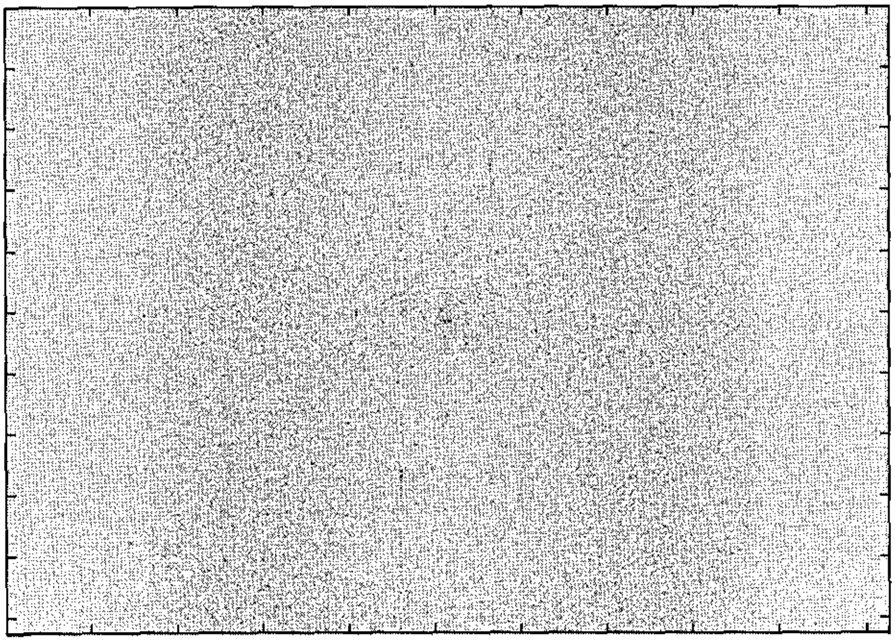


Figure 3.47 - Fourier Transform of the Difference between Two Blank Images After Removing some of the Frequency Dependent Noise. Zero frequency at centre.

3.4.4 - Noise Measurement

The quantity of noise in a given image can be estimated by calculating the standard deviation of the difference between two images taken under identical conditions. As mentioned earlier, the standard deviation of the noise in each image is the standard deviation of the noise in the difference image divide by root two, provided the noise images are independent. However, the noise source is not completely independent as it is known to include frequency dependent noise, and is likely to include grey level dependent noise from the camera [27].

The frequency dependent noise can be removed in the frequency plane, whilst the properties of the grey level dependent noise can be found by measuring the noise level at each grey level. To measure the grey level dependence, pairs of images were taken of blank radioscopic images with all three image intensifier zoom settings and non radioscopic images of a piece of white card. Blank images were used as they have a smaller grey level range, thus reducing the grey level noise. The images were taken with different apertures to measure the noise at different grey levels. Further pairs of images of the card were taken with a Pulnix 525 camera.

Figure 3.48 plots the standard deviation of the difference between two images (after removing the frequency dependent noise) against the mean grey level of the two images. It can be seen that the random noise has a component that is proportional to the grey level. This is independent of whether the image is of a radioscopic or non radioscopic target, and for all image intensifier zoom settings. These results indicate the typical standard deviation of the noise difference is 2 grey levels for black pixels, increasing to 3.5 grey levels for white pixels. This equates to the standard deviation of the noise in an image of between 1.4 grey levels and 2.5 grey levels.

The tests made with the Pulnix 525 camera, are noted to have significantly less noise than those taken with the normal camera. The Pulnix 525 camera has an auto iris function that explains why there is little variation in the mean grey level as the aperture is varied. However, with a very small aperture, the auto iris is unable to compensate, which results in one reading with a higher level of noise. The tests show that the Pulnix 765 camera is a significant source of noise.

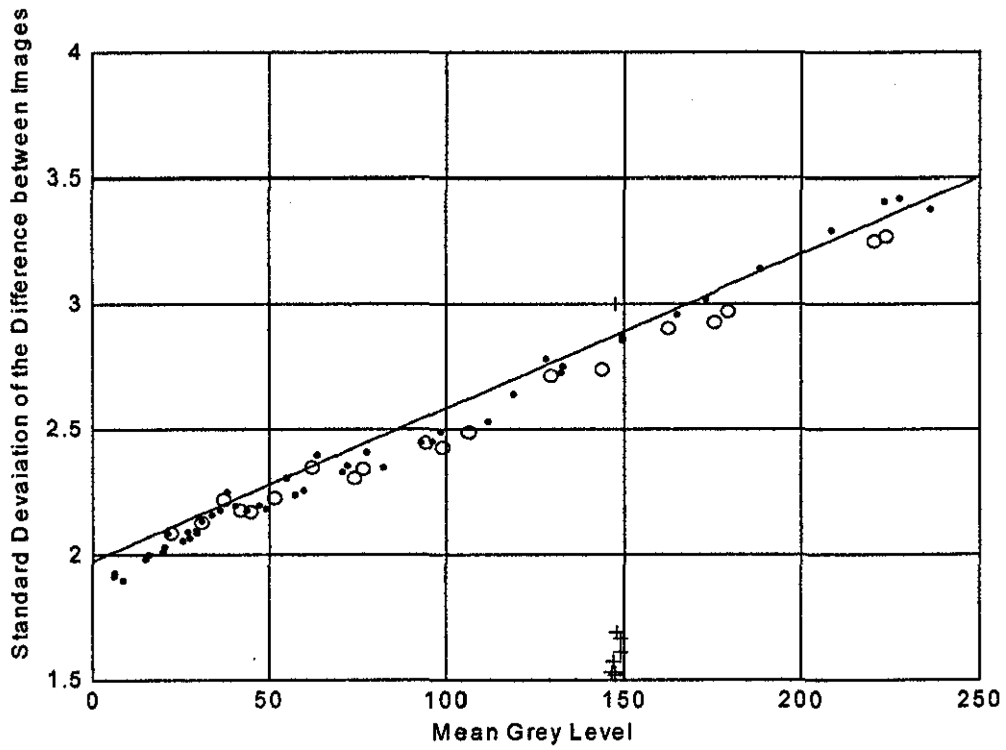


Figure 3.48 - Standard Deviation of the Difference in Grey Levels Between Pairs of Images taken with Identical Settings, for Different Apertures. Images are corrected to remove frequency dependent noise. • Radioscopic Images (all three zoom settings), O Normal Images, + Pulnix 525 Camera.

3.5 - Summary

Although there are a number of variables that can be used to control the radioscopic equipment, the images with the highest contrast will have similar settings. The brightest images result in the highest contrast, and are obtained with the widest aperture, the highest power setting, and the longest shutter time. Shutter times greater than one frame result in noise caused by variations between the brightness of the odd and even image fields. Apertures greater than about f11 are required for a sharp image. This means that in most circumstances full X-ray power is required, and the larger image intensifier zoom settings are an advantage. With certain targets, lower X-ray power results in better definition, and in these cases this improvement would have to be traded against the integration noise (and for very long shutter times, additive noise).

The equipment has a low energy X-ray energy output, more suitable for composite materials than for metallic objects. Thinner objects will produce brighter images. This means that provided an object is thin, materials with higher levels of absorbency may be suitable for imaging, and materials with a lower absorbency coefficient may be imaged with a lower X-ray power setting.

The zoom settings provide a wide range of image sizes, with the maximum definition almost matching the definition of the image intensifier. However a large quantity of image distortion is evident in certain circumstances, and has to be corrected for accurate spatial measurements. Other constant noise sources include the "inverse shadow" effect and intensity variations across the image.

Either the X-ray source, or the image intensifier, is responsible for the image fading with time. Most of the random noise is produced by the camera and the image grabber. The camera is noted to be a particularly large source of random noise, whilst the image grabber board has been shown to be responsible for noise caused by non-linearities in the analogue to digital converter, and probably the frequency dependent noise. It is possible to correct for the constant noise sources and the frequency dependent noise. The random noise could can be reduced by taking the average of a number of images taken under identical circumstances.

At present the system is held back principally by the camera and frame grabber. A new low light level camera would permit a greater range of X-ray energies to be used. It would be interesting to have a higher resolution camera so it can resolve the image at a higher rate than the image intensifier, as this would allow images that could be used to test the output of the image intensifier.

A new frame grabber board would be of great value. The problems with noise in the integration feature prevent the use of all but the most powerful X-ray settings. A new grabber board could also be specified with square pixels, and the gain and offset commands that were useful features of the VS-100 board. Square pixels would make correcting for distortion easier, and measuring distances within the images more accurate. The gain and offset commands would allow opaque targets to be visualised over a wider range of X-ray settings. Finally a servo controlled camera and mirror stand could be used to align the optics with the machine operating, which would make it easier to align all the components accurately.

4 - Distortion Correction

Chapter 3 described how radioscopic images are subjected to significant distortion caused by the image intensifier. In order to make accurate measurements of the radioscopic images, it is necessary to correct for the distortion. As the system is fitted with a zoom lens, every set of measurements will have unique distortion characteristics, so the correction parameters will have to be re-calculated regularly. Thus a reasonably fast and automatic method of correction is required.

A number of methods of correcting for distortion were discussed in chapter 2. These can be split into theoretical models based on the design of the optics, and measurement based methods that depend on the identification of features (normally in a grid pattern) on a phantom. The complex and variable nature of the zoom lens optics, and the distortion caused by the image intensifier, make a measurement method more suitable. The majority of measurement based techniques described in the literature use a manual method of finding the locations of features, and to identify their location in a grid. The automatic techniques for identifying grids all have the advantage of a grid of known dimensions, which makes these methods unsuitable for this system, as the grid dimensions alter with the zoom lens position. This chapter includes details of the phantom used to measure the distortion. Section 4.2 describes the software written to identify the features on the phantom.

Whilst some of the methods listed in chapter 2 removed distortion with an area by area method, the radial symmetry of the radioscopic distortion suggests the suitability of a single model covering the whole image. As most subsequent work requires the accurate location of points in the distorted image, the model was written to map a given point in the distorted image to an equivalent location in the corrected image. Section 4.3 gives details of the design of the global distortion model, and is followed by descriptions of the iterative procedure used to optimise the model's parameters.

Section 4.7 covers the method of mapping locations from a true image to the distorted image. The grey level at the locations can then be found by interpolation. This is used to produce undistorted radioscopic images.

4.1 - Distortion Measurement Phantom

The X-ray phantom consisted of a square grid of holes drilled in a 5mm x 400mm x 350mm plate of aluminium. The sheet was cut so that it fitted flush with the back left corner of the X-ray cabinet. Tests were undertaken to check that the 1mm holes were clearly visible with all zoom settings of the camera (figures 4.1 and 4.2). These tests also checked that a 5mm sheet of aluminium gave sufficient contrast. A 3mm grid spacing was chosen as this ensured that at least ten holes were visible in each direction under all circumstances (figure 3.20). However to cover all zoom settings, over four thousand holes would be needed. As this was considered excessive, only the central region of the imaging area was drilled with a 3mm hole spacing, and the rest was drilled at a 6mm spacing (figure 4.3). Wider angle images (figure 3.15) thus contain sufficient holes to measure distortion using only the 6mm grid.

The phantom plate was drilled on a CNC machine. The 6mm grid was drilled first, and the remaining holes in the 3mm grid were drilled later as three further 6mm grids. The grid was drilled in a 6mm thick sheet of aluminium and the top and bottom surfaces were machined down to 5mm to give a good surface finish. A number of holes in the bottom left corner (most of which are outside the radioscopic target area) where drilled in the wrong place, and had to be plugged and re-drilled. All the holes were de-burred with a 1mm drill to remove swarf.

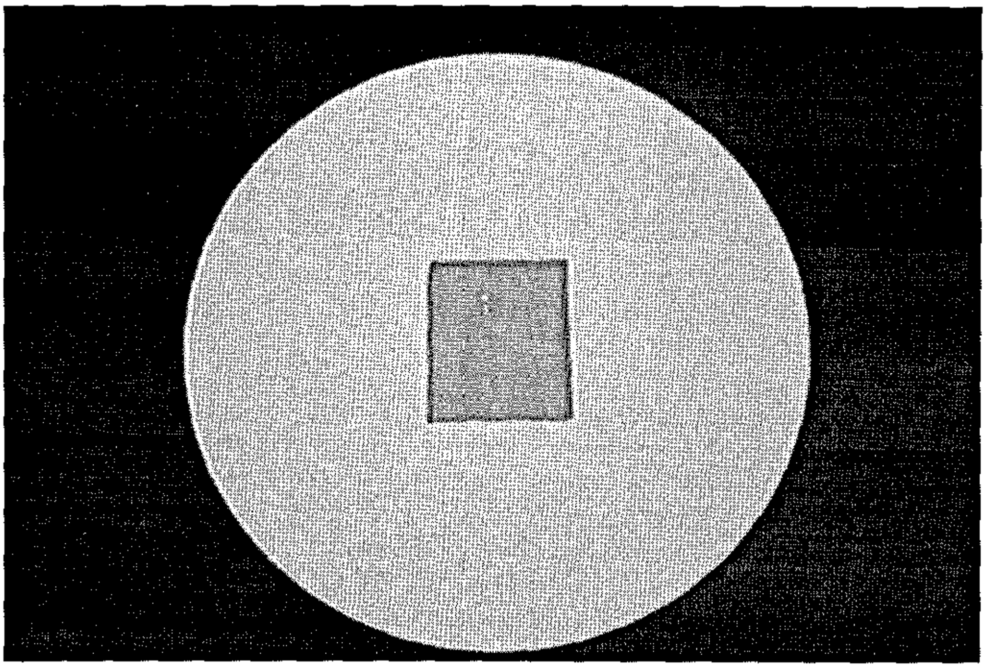


Figure 4.1 - Radioscopic Image of Piece of Aluminium Drilled with 1mm Holes.
215mm Image Intensifier Zoom Setting.

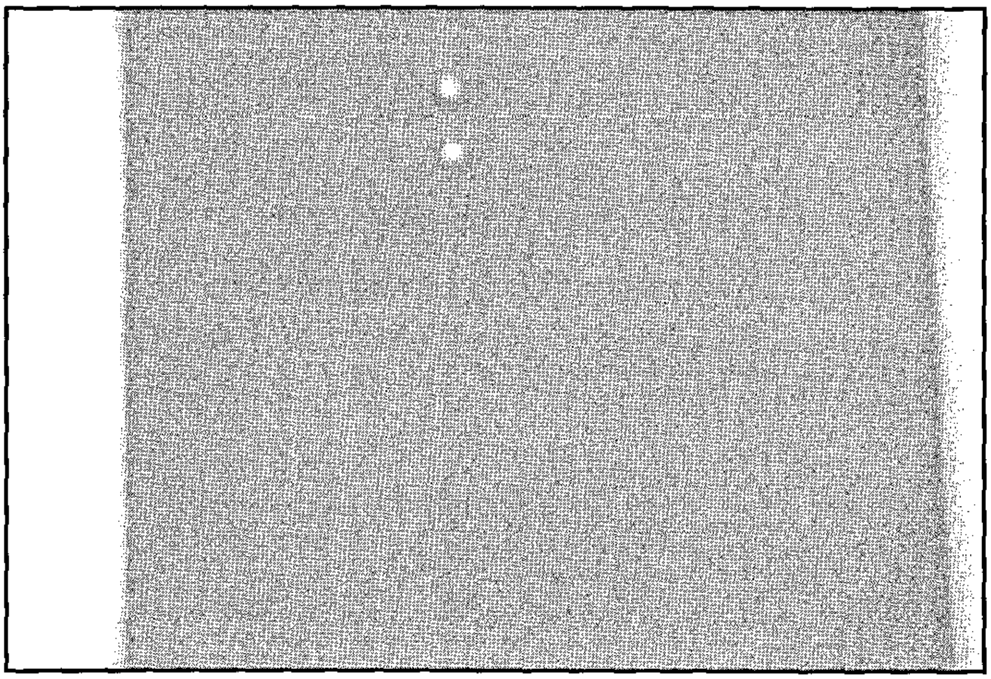


Figure 4.2 - Radioscopic Image of Piece of Aluminium Drilled with 1mm Holes.
Maximum Zoom.

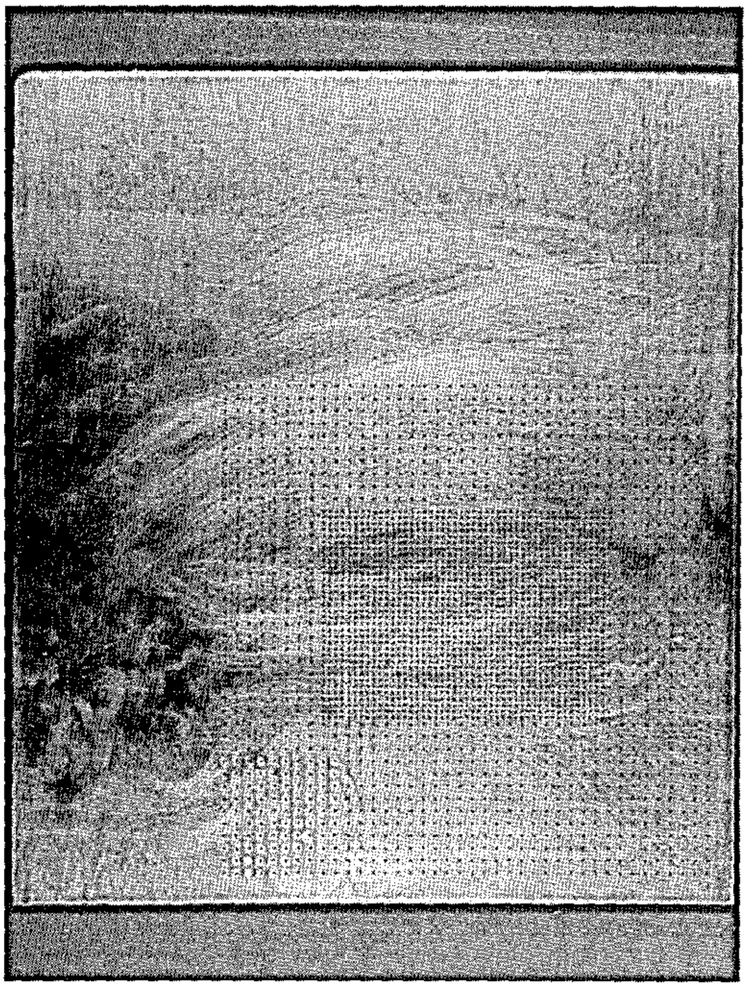


Figure 4.3 - Distortion Phantom.

4.1.1 - Calibration of the Phantom

Once manufactured, a micrometer was used to measure the spacing between consecutive holes in the completed phantom. This showed that the distances between adjacent holes have an error of up to 0.1mm. The plate was scanned to allow a fuller analysis of the variation in the hole spacing. The scans were taken with a resolution of 254 dpi (10 pixels per mm), the contrast and brightness set to maximise the contrast between the holes and the rest of the phantom (figure 4.4).

The holes in the target were identified using the method described in section 4.2. With holes of 1mm diameter, an average of 78 pixels will be within the boundary of each hole. If the hole is displaced 100μm, ten pixels will no longer be within the boundary of the hole, and ten new pixels will to be included within the hole boundary. This makes it possible to detect the location of the hole to an accuracy around 5μm. The centres of the holes were compared with an ideal rectangular grid with regular hole spacing. A modified version of the iterative method, described in section 4.6, was used to adjust the ideal grid to minimise the square of the errors between the ideal grid model and the measured grid. The modified iterative method assumed no radial or angular distortion in the image.

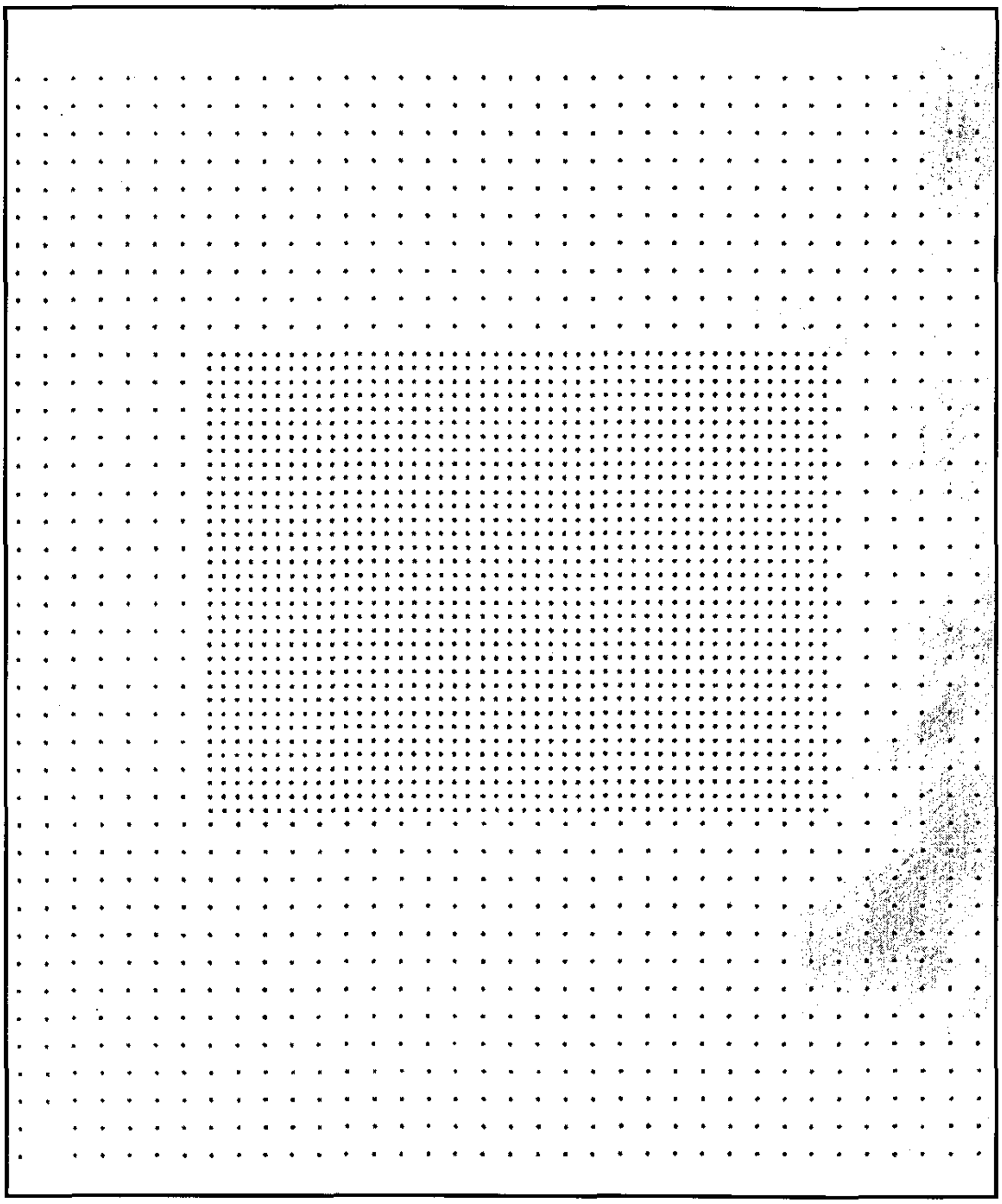


Figure 4.4 - Scan of Holes in Distortion Phantom.

Four scanned images of the target were processed, the first two of the top face, and the second two of the bottom face. In order to detect inaccuracies in the scanner, the second image was scanned with the target rotated by 90° to the first, whilst the last scan was of the target rotated through 180°. Scans were taken at different angles to detect systematic errors in the scanner. All the images were orientated the same way by the computer, before processing to find both the 6mm and 3mm grids. Details of the ideal grids, and the difference between the ideal and measured grids are shown in table 4.1. Referring to the equations in section 4.3.1, the grid dimensions are calculated from the inverse of parameters $\mathbf{k}_5$ and $\mathbf{k}_6$ and the angle between the ideal grid and the scanned grid, $\mathbf{k}_7$.

Table 4.1 - Measurements taken from Scans of the Distortion Phantom

Scan	Scanned Image	Grid Size	Grid Side X (Measured Average)	Grid Side Y (Measured Average)	Angle to Scanner (Measured)	Mean Error	Maximum Error
1	Top	6mm	6.009mm	5.997mm	.16°	130µm	493µm
2	Top, 90° rotation	6mm	6.010mm	6.009mm	.04°	130µm	464µm
3	Bottom	6mm	6.008mm	6.008mm	-.08°	140µm	483µm
4	Bottom, at 180°	6mm	6.008mm	6.001mm	-.06°	122µm	380µm
1	Top	3mm	3.004mm	3.000mm	.19°	92µm	266µm
2	Top, 90° rotation	3mm	3.006mm	3.004mm	.05°	99µm	290µm
3	Bottom	3mm	3.004mm	3.007mm	-.11°	100µm	298µm
4	Bottom, at 180°	3mm	3.004mm	3.005mm	-.10°	91µm	288µm

The greatest variation between the lengths of the **x** and **y** grid sides is 12µm, which is of the same order of magnitude as the level of tolerance of these tests. With all the measured axes close to the expected values, there is no reason to suggest that the scanner does not scan with square pixels. The angle between the scanner and the grid was very small, with variations of less than 0.05° between the two scans of each image. The results show that the 3mm grid is more accurate than the 6mm grid. It is noted that the measured average distance between the holes is slightly greater than the expected values.

Figure 4.5 is a histogram of the error distribution of the 6mm grid, whilst figure 4.6 is the error distribution of the 3mm grid. It can be seen that in both cases the errors form a Rayleigh distribution. Figure 4.7 plots the distribution of the locations of the errors. It can be seen that the model is centred close to the geometric centre of the errors, and that the errors are evenly distributed around the model. The random distribution of errors in the hole locations means that provided a large number of hole locations are used, the errors will be minimised with a global distortion model.

Figure 4.8 plots the difference in the **y** axis location between adjacent holes in the 3mm grid in the first image. It is noted that there is a strong correspondence between the locations of alternate holes. It is hypothesised that this was caused by drilling the 3mm grid as four separate 6mm grids. The errors between the grids would result in large differences between adjacent nodes and correspondence between alternate nodes. A similar result was noted with plots of the grid in the **x** axis.

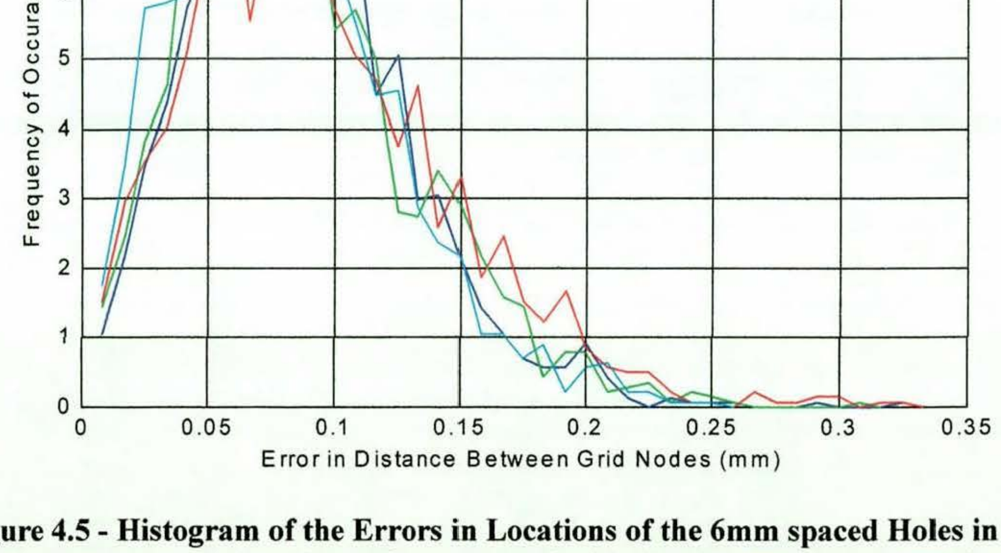


Figure 4.5 - Histogram of the Errors in Locations of the 6mm spaced Holes in the Distortion Phantom. Results are from scans of grids. Scan 1: —, Scan 2: —, Scan 3: —, Scan 4: —.

To correct this systematic fault, the "Matlab" function "mmodkx.m" (appendix III.1) was written to "juggle" the four overlapping 6mm grids to minimise the errors. This program uses the ideal grid model (section 4.3.1) as a measure of the accuracy of the grid. The offsets in the **x** and **y** directions for each of the four 6mm grids are varied to find the best values for the eight parameters. The parameter values are found in an iterative optimisation

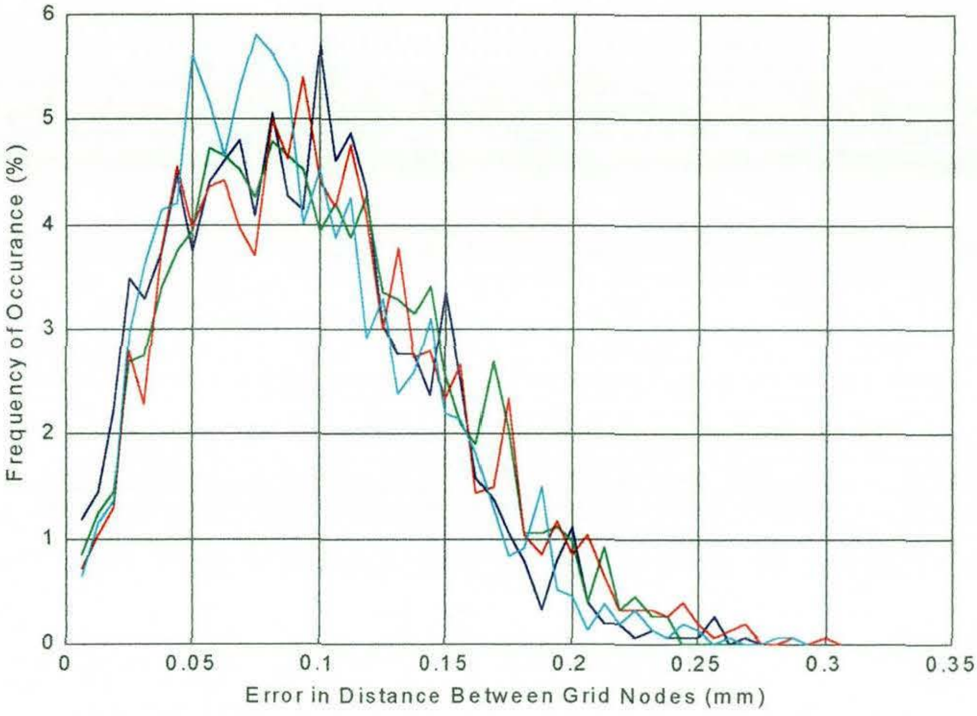


Figure 4.6 - Histogram of the Errors in Locations of the 3mm spaced Holes in the Distortion Phantom. Results are from scans of grids. Scan 1: —, Scan: —, Scan 3: —, Scan 4: —.

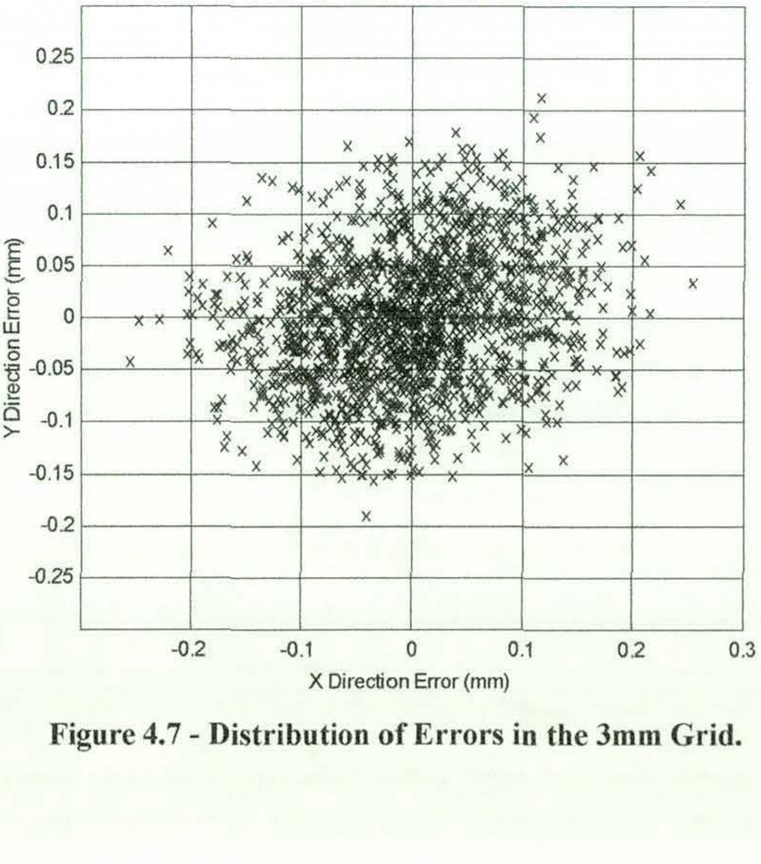


Figure 4.7 - Distribution of Errors in the 3mm Grid.

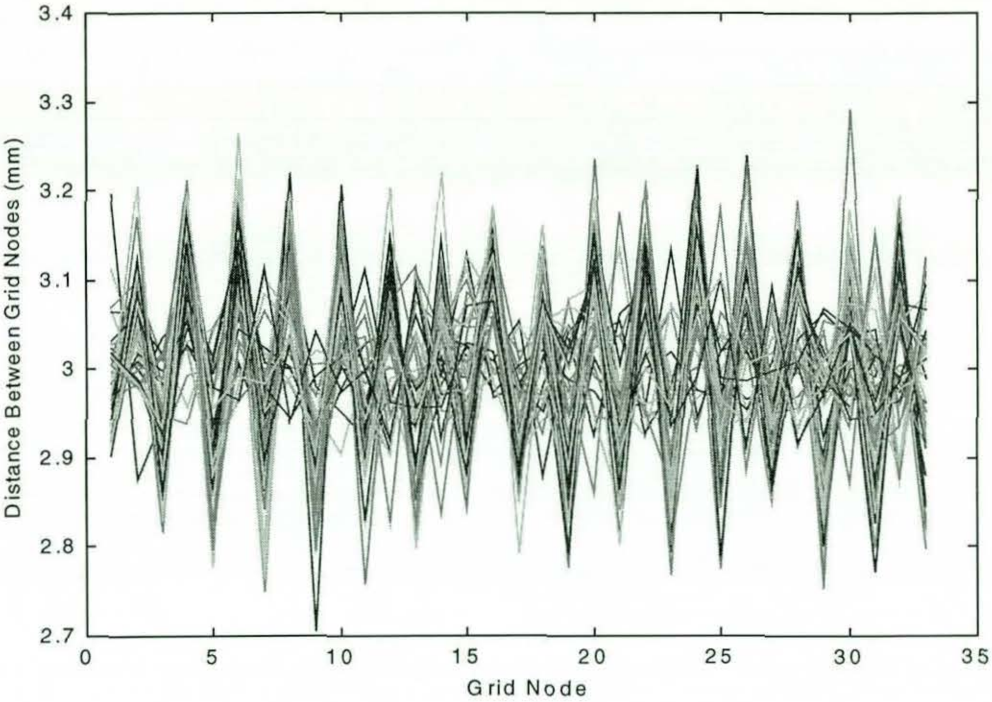


Figure 4.8 - Plot of Distance between Grid Nodes. Using the 3mm grid, and the first scan. Plots are made of each horizontal line in the grid.

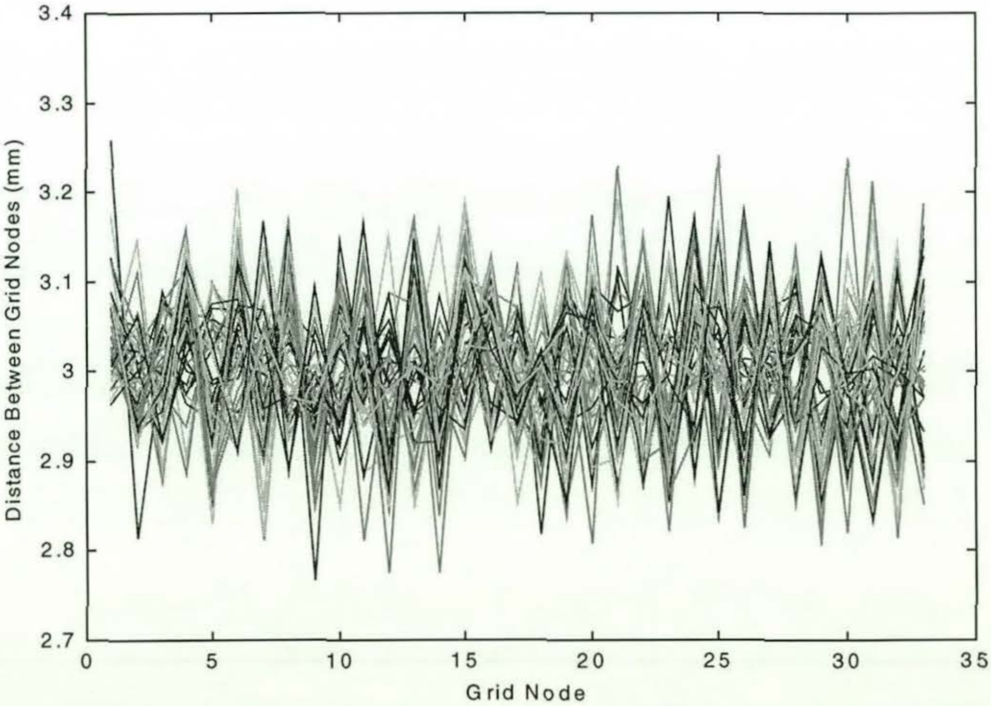


Figure 4.9 - Plot of Distance between Grid Nodes after Correction. Using the 3mm grid, and the first scan. Plots are made of each horizontal line in the grid.

method based on the one used in "mmodall.m" described in section 4.5. The distortion modelling software, runs the grid correction procedure once, near the start of the program.

Table 4.2 shows the results after applying this software to the four small grids. A 10% reduction in the mean error is noted in the grid scan of the two images of the top face. The results for the bottom surface scans are similar to the results without the grid correction. This indicates that the location of the hole in the top and bottom faces varies. This could be explained by drilling from the top surface with a drill that is bent, or not square to the plate. Figure 4.9 shows the effect of the grid correction on the difference in the **y** axis between adjacent nodes.

Table 4.2 - Measurements Taken from Scans of the Distortion Phantom after Correction for Manufacturing Error

Scan	Scanned Image	Grid Size	Grid Side X (Measured Average)	Grid Side Y (Measured Average)	Angle to Scanner (Measured)	Mean Error	Maximum Error
1	Top	3mm	3.004mm	3.000mm	.19°	82µm	231µm
2	Top, 90° rotation	3mm	3.006mm	3.004mm	.05°	90µm	251µm
3	Bottom	3mm	3.004mm	3.007mm	-.11°	99µm	304µm
4	Bottom, at 180°	3mm	3.004mm	3.005mm	-.10°	90µm	284µm

4.2 - Distortion Identification Software

Once a radioscopic image has been taken of the phantom, it has to be processed to measure the distortion. A couple of programs have been specifically written to identify the locations of the holes in the radioscopic images of the phantom. The first stage (4.2.1) is to pre-process the image to maximise the visibility of the holes. For this, the Matlab routine "dewarp.m" was written. The identification of the grid is undertaken using the executable file "findgrid.exe" that is compiled from "C" code. This is a large program that is described in sections 4.2.2 to 4.2.5. An analysis of the capabilities of the software follows in 4.2.6. Once the grid is defined, the software covered in section 4.3 can be used to produce a global model of the distortion.

4.2.1 - Image Pre-Processing

The Matlab file "dewarp.m" (see appendix III.2) loads two images. The first of these is the image of the phantom (figure 4.10), and the second is an image taken with nothing in the X-ray chamber (figure 4.11). The first image is divided by the second to reduce problems caused by the radial intensity variations. A cut off value is chosen by the operator so that the target is removed, leaving only the image of the holes. The results are displayed (figure 4.12) using the custom function "bimage.m" (appendix III.3), and saved as "output.raw" for use by the program "Findgrid.exe". "Bimage.m" is a modification of built in function "image.m", which displayed matrices as an image map. In the modified version, the image intensities are standardised, so that images with any range of pixel intensities can be displayed.

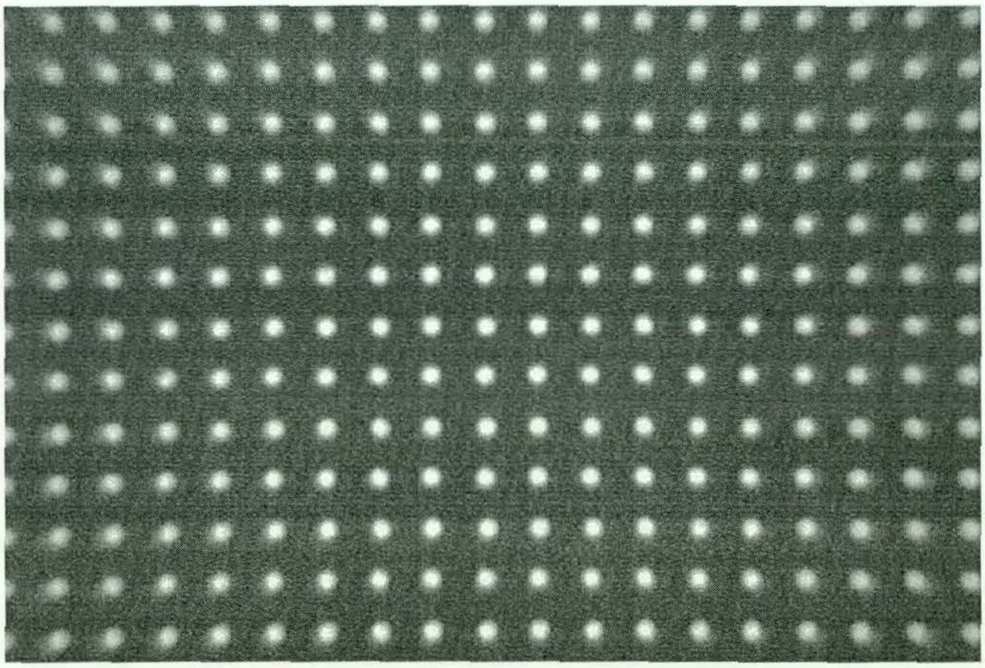


Figure 4.10 - Example of Radioscopic Image of Distortion Correction Phantom.

holes, and to sort them into a grid. The software is designed to cope with images of rectangular grids taken at any angle, and to identify and deal with erroneous and missing holes. The "main" procedure calls eight other procedures (figure 4.13). The tasks performed by these procedures are to identify the holes in the grid, map the boundary of the holes, find the centres of the holes, sort the holes into a grid, and save results. The "main" procedure also creates, and deletes some of the data structures. The procedures are described in 4.2.3 to 4.2.5.

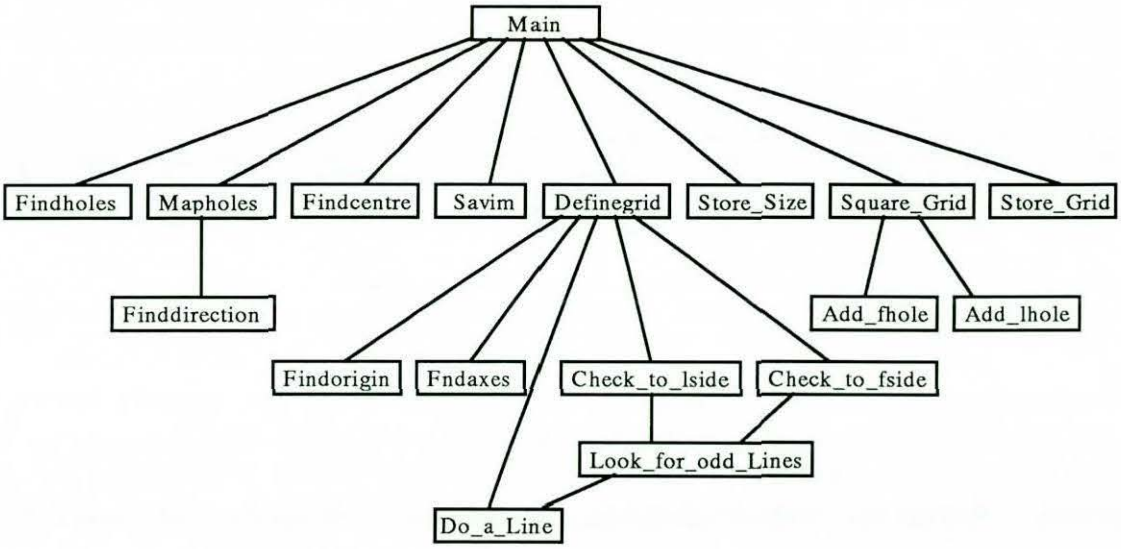


Figure 4.13 - "Findgrid.c" Software Structure Chart. Each procedure calls the procedures below it, working from left to right.

4.2.3 - Identifying the Centres of the Holes

The procedure "findholes", which is the first to be called, imports the processed image "output.raw", produced by "dewarp.m". "Findholes" filters the image to find pixels that are within the boundary of a hole, but whose neighbours above and to the left are not. These are placed in a list, which will include at least one pixel from each complete hole. The co-ordinates of the pixels ("ival", "jval") are stored as the data structure "pxl", which are linked together using the data structure "posit" (figure 4.14).

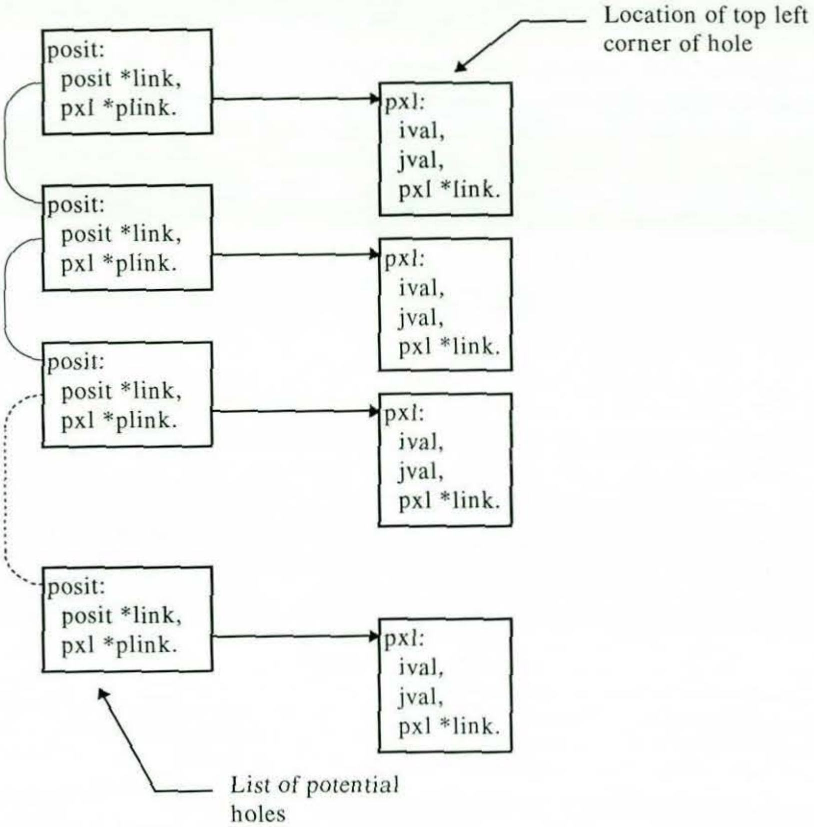


Figure 4.14 - Data Structure of "Posit" Returned by "Findholes"

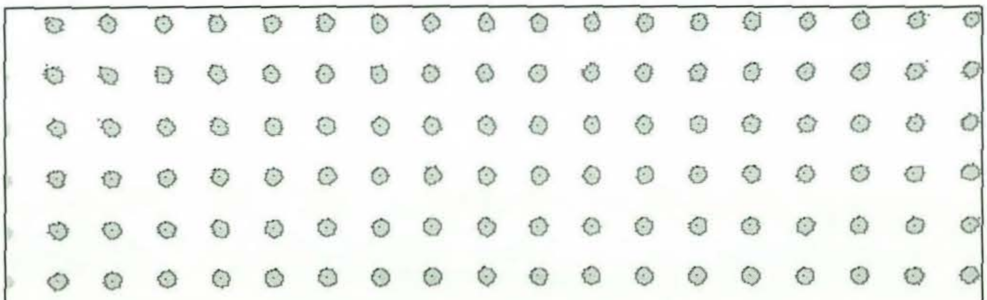


Figure 4.15 - Corners, Perimeters, and Centres of Holes found by "Findgrid.c".

The linked list is passed via the "Main" procedure to the procedure "Mapholes" that finds the perimeter of each hole (figure 4.15). Using the top corner pixels as starting points, "Finddirection" looks for the next pixel along the perimeter using four connectivity [80]. The co-ordinates of the next pixel are stored (figure 4.16), and linked from the previous pixel (pxl). Pixels added to the perimeter are checked against the list of top left corners. Any duplicates are removed to prevent holes being mapped several times.

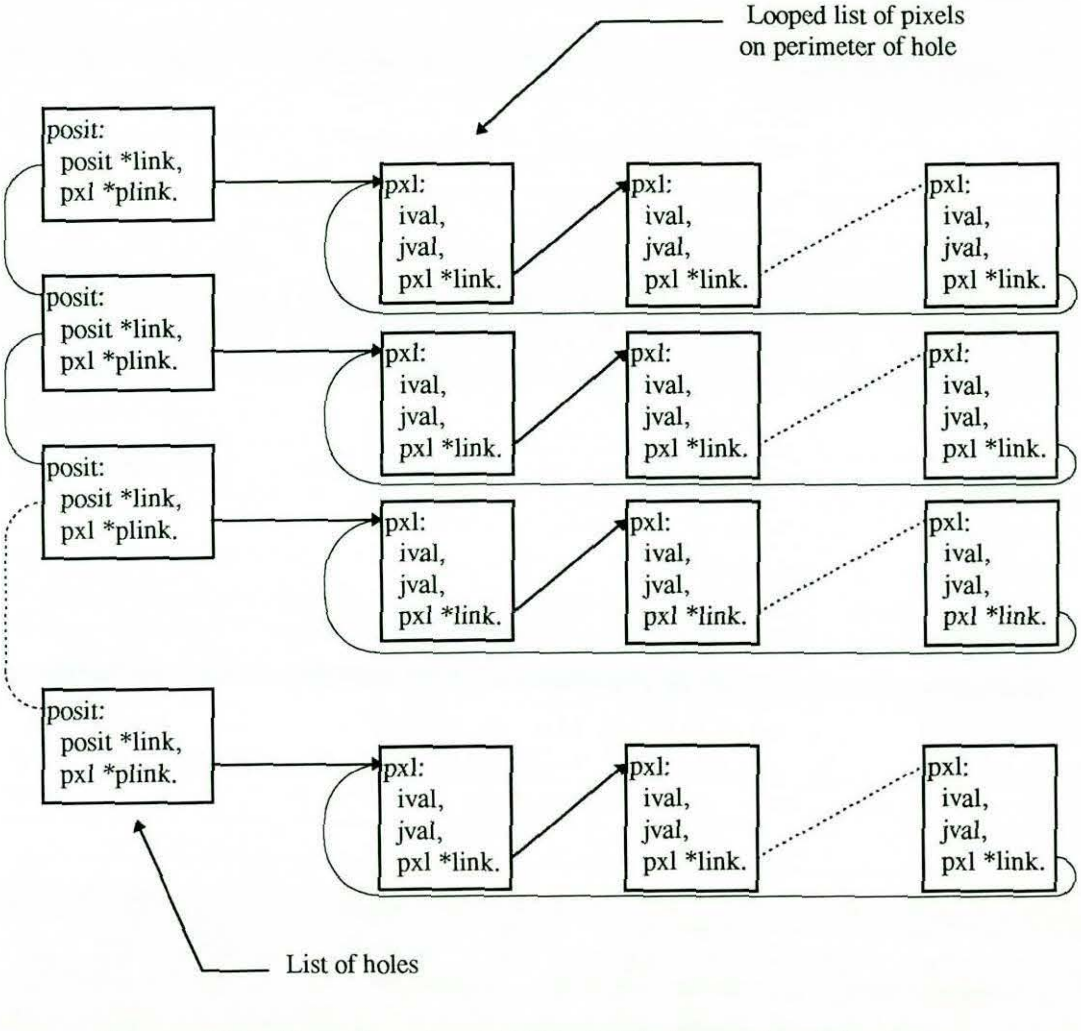


Figure 4.16 - Data Structure of "Posit" Returned by "Mapholes".

The "Main" procedure passes details of the holes to the procedure "Findcentre" (see figure 4.13). Here the maximum and minimum **x** and **y** values of the hole perimeter are found. The pixels within the rectangle created with these limits are used to find the centre of the hole. This is calculated as the centroid of all pixels within the square. The procedure rejects any hole that is on the edge of the image, or any that are too small. The locations of the centre of the holes are stored in another linked list, "fpxl" (figure 4.17), as floating point numbers ("ival", "jval"). The sum of the brightness of the pixels in the hole is also recorded ("size") to give a weighting should two holes be combined. The first location of a hole centre in the list is referenced from an "fposit" structure. Two dummy holes are added at the end of the list to avoid having to deal with an empty list if all the holes are allocated. Finally the last of the "fpxl" data structures is linked to the first to make them into a loop.

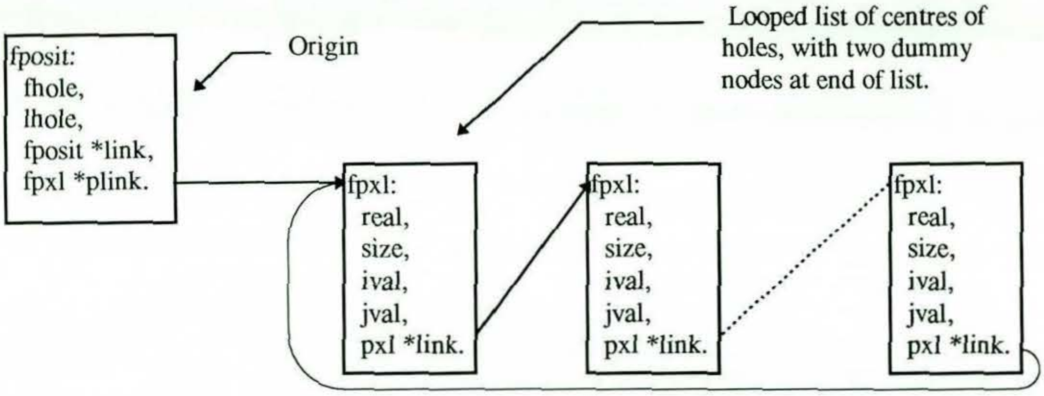


Figure 4.17 - Data Structure of "Fposit" Returned by "Findcentre".

4.2.4 - Sorting the Holes into a Grid Structure

It can be seen from figure 4.13 that "Definegrid" calls a number of other procedures to convert the list of hole locations to nodes in the grid structure. The first of these procedures looks for a suitable origin and axes to the grid. These are used to build up a grid by looking for nodes that are neighbours to nodes already in the grid. The data structure produced in "Definegrid" (figure 4.18) starts with an origin of data structure "fposit". Attached to this are the holes that have not yet been mapped, stored in a loop. The origin also links to a list of lines ("fposit"). Each of these lines stores the horizontal grid positions of the first and last holes in the line ("fhole", "lhole"), and links to the first of the holes in that row.

"Findorigin" takes the first hole in the list, and puts it at the start of the first line. "Findaxes" finds the two nearest orthogonal holes to the origin, and puts the hole nearer the vertical as the first hole in the second line, and the other as the first hole in the third line. A check is made in "Definegrid" to see if the two axes are close to 90° apart and of a correct length ratio. If the checks fail, the procedures are repeated with the next hole in the list of unused holes.

The two vertical holes are removed from the list of unused holes, and the third line is freed. The third hole is used to find the angle and distance for the horizontal components, and as the marker in the list of unused holes. This method of finding the origin will find the larger grid if it is visible, as the list of holes starts at the top of the image which is the first area to lose the fine grid.

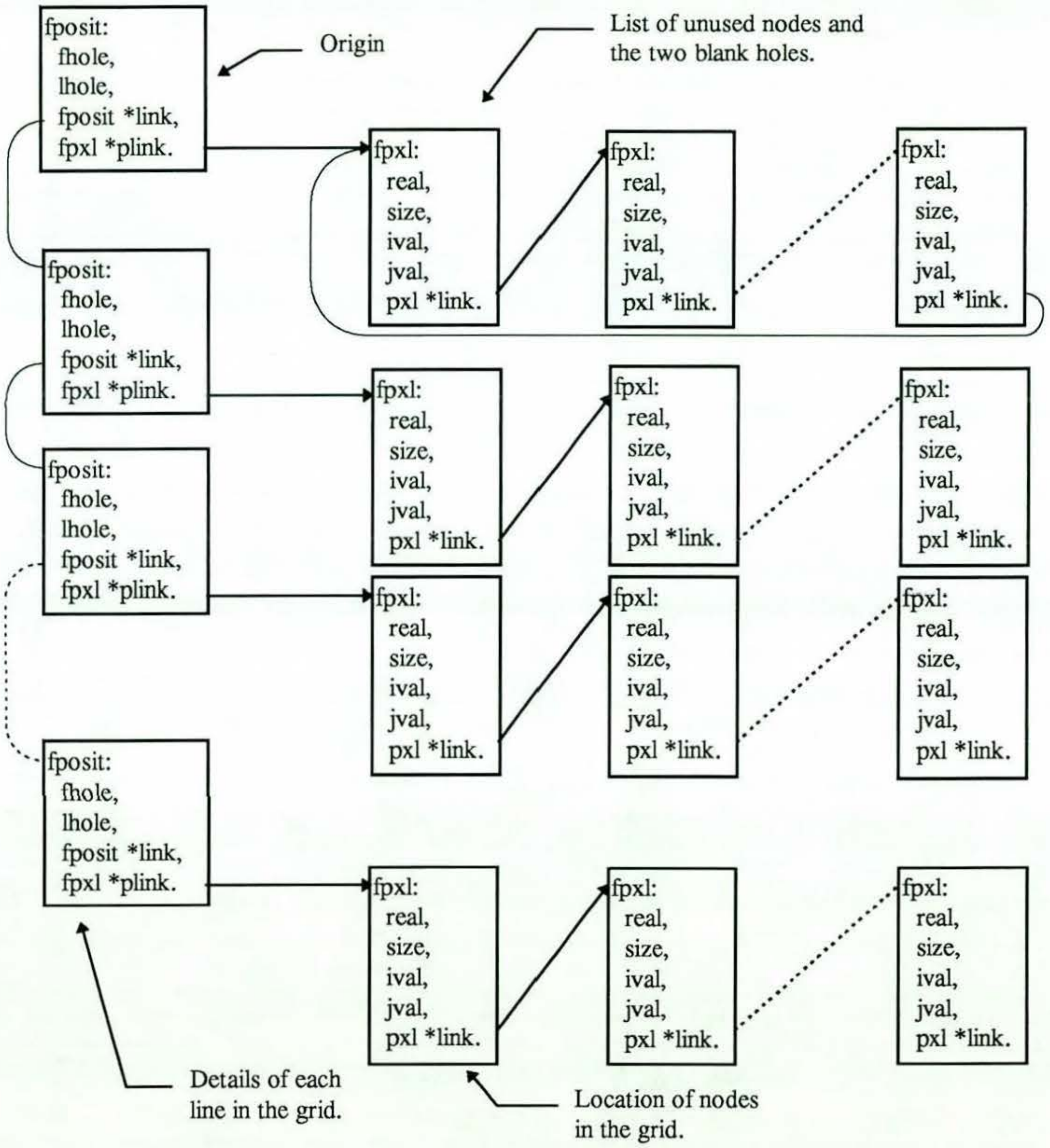


Figure 4.18 - Data Structure of "Fposit" Returned by "Definegrid".

"Do_a_line" is used to convert horizontal lines of holes to grid nodes. The procedure is given the start of the line and an angle and distance that are used to locate the next hole in the line. First the procedure moves to the end node of the line already found. Then it looks through the holes in the list that have not been allocated, for any holes within the target region for the next node. Any holes found in that region are combined, to form the new grid node, which is added to the end of the line. The procedure then looks for another node. When no more holes can be found to convert to nodes at the end of the grid line, the procedure starts at the near end of the grid and will extend the near end of the grid by looking for holes in the opposite direction, and putting them at the start of the list of grid nodes. The procedure updates the position of the first and last nodes in the "fposit" linked list. The procedures that call "Do_a_line" need to update the angles and distances as the grid is defined.

"Definegrid" finds the first two lines, using the third hole found by "Findaxes" to calculate the angle and distance between the lines. The grid is extended using "Check_to_fside" and "Check_to_lside" until no further grid nodes can be found. These procedures are given two consecutive lines that they use as references to extend the grid on an adjacent line. "Check_to_fside" extends the grid at the start of the next line and "Check_to_lside" extends the grid at the end of the next line. Both procedures can add an extra line to the grid, or extend existing lines around missing nodes.

These procedures start by looking to see if they will be creating a completely new line, or extending an existing line. The reference distance and angle are found from the two lines given, using the most suitable pair of nodes. The search for the new grid nodes is undertaken by the procedure "Look_for_odd_lines". This procedure looks through the list of unused holes, and uses the first (if any) hole that falls within the region of interest as the basis of a new line. The new line is created by "Do_a_line" and returned as the first line in the grid. The line is moved to the correct part of the grid by the calling procedure, and if it is an extension to another grid line, missing nodes are estimated by interpolation (marked in the data structure as not "real"), and overlaps deleted.

4.2.5 - Saving Grids

After "Definegrid" has completed the grid structure, "Store_Size" writes details of the dimensions of the grid to the ASCII file "gridata.m". This can be called by Matlab to load the grid dimensions. "Square_grid" fills in the gaps to make all the rows in grid start at the same column, and be of the same length. The additional holes used to create the grid are given locations extrapolated from the rest of the grid, and marked as not "real". "Store_Grid" creates three matrices in "gridata,m". The first is a matrix indicating which nodes are from holes, and which are interpolated or extrapolated. The other two matrices list the x and y dimensions of each node in the grid. For a 512 x 512 image, the dimensions are stored in the range 0 to 511.

The procedure "savim" saves the image "modg.raw" to disk as a raw image file. This can be viewed with any suitable imaging software. The data in "modg.raw" (for example figure 4.15) shows the area identified as being part of a hole, the boundaries of the holes, and the top left corners. Also the top left corners used to start hole mappings, and the locations of the centres of the holes are marked as different shades of grey. This data is taken from the array "mimage", which is updated in the procedures that identify the properties of the holes. The file is normally used for fault finding, and to ensure the holes are being correctly identified.

4.2.6 - Grid Identification Software Performance

The program "Findgrid.c" was tested with a number of X-rays of the grid with different zoom settings. Referring to the images listed in table 3.2, the images taken with maximum zoom took between six and twenty seconds to run, whilst the medium zoom images took between forty and seventy seconds to process with a Pentium 166. The software was noted to take five minutes for the 10 pixels/mm scans of the target (figure 4.4). The procedure that took the most time was "Mapholes", but "Definegrid" can take a while when large parts of the grid are missing.

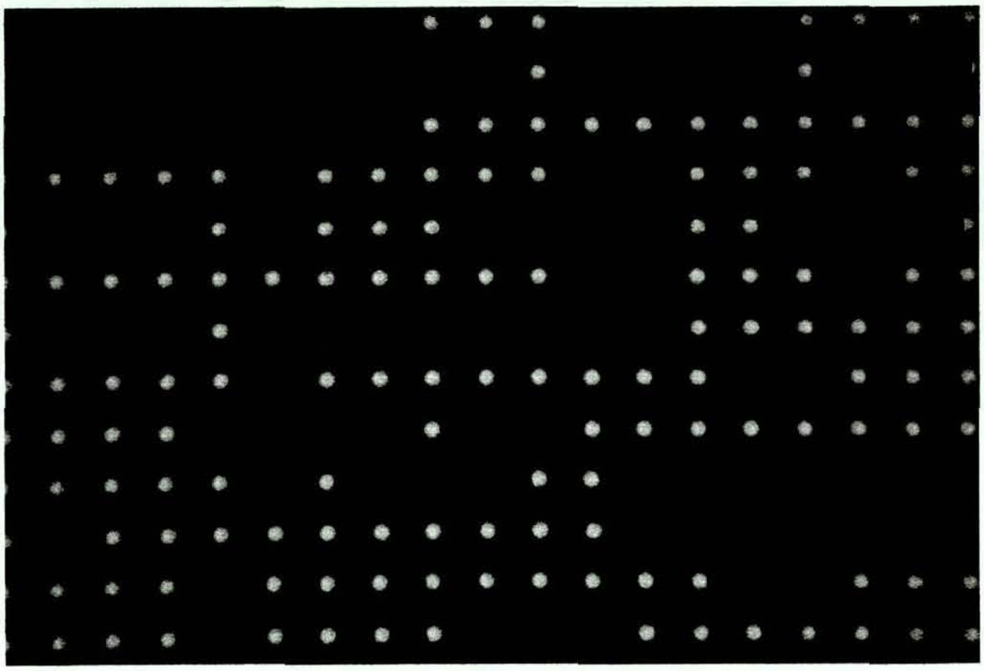


Figure 4.19 - Image Used to Test Capabilities of "Findgrid.c".

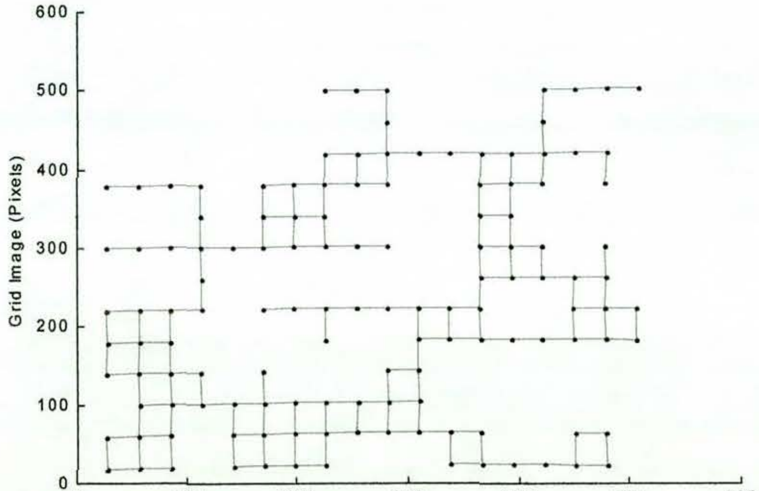


Figure 4.20 - Grid Successfully Found by "Findgrid.c".

The software's ability to find a grid with incomplete data was tested. The program was run with a number of input images that had holes manually blanked out (figure 4.19). Holes were removed to test the software's ability to deal with difficult data sets. This method identified a number of bugs, which were resolved, and the final version of the software was able to correctly identified the grid (figure 4.20). However it was found that the limits used to identify the origin have to be generous to deal with images with rectangular pixels. This could result in an incorrectly dimensioned grid. This problem could easily be solved by using an image acquisition board that produces square pixels.

4.3 - Modelling Spatial Distortion

Having found the location of the distorted grid nodes, they are used to make a model that can be used to remove distortion. The distortion model is in the Matlab procedure "zmod.m" (section 4.4). This procedure is able to measure the accuracy of the model by comparing the distorted grid after correction, to an ideal grid. The procedure "mmodall.m" (section 4.5) looks at models of the distortion within given parameter ranges, and finds the best model, using the measure of accuracy in "zmod.m". The parameter optimising procedure "mmod.m" (section 4.6) was written to search for the best model. It starts by finding a suitable initial model, and improves it iteratively using "mmodall.m".

4.3.1 - Design of the Distortion Model

The radioscopic images suffer pin-cushion distortion (figure 3.15), and "S" distortion. As pin-cushion distortion is radially symmetric, it can be modelled using a radial equation [63]. Tests were made with radial equations of up to the fifth order (equation 4.1) as the basis of the model.

$$R_N = r + k_9 r^2 + k_{10} r^3 + k_{13} r^4 + k_{14} r^5$$

(4.1

Where **k$_n$** are the model parameters defined below. The complete set of equations used to map a location in the distorted image (**x$_O$, y$_O$**, polar co-ordinates: **r, Ø**), including potential to correct for "S" distortion, to a location in the undistorted image (**X$_N$, Y$_N$**, polar co-ordinates **R$_N$, Ø$_N$**), is be expressed as equations 4.2 to 4.5:

$$X_N = \left[r + k_9 r^2 + k_{10} r^3 + k_{13} r^4 + k_{14} r^5\right].\cos\left(\phi + k_7 + k_{11} r + k_{12} r^2 + k_8 r^3\right) + k_4$$

(4.2

$$Y_N = \left[r + k_9 r^2 + k_{10} r^3 + k_{13} r^4 + k_{14} r^5\right].\sin\left(\phi + k_7 + k_{11} r + k_{12} r^2 + k_8 r^3\right) + k_3$$

(4.3

where

$$\tan(\phi) = \frac{k_5(y_O - k_1)}{k_6(y_O - k_2)}$$

(4.4

and

$$r = \sqrt{k_6^2(x_O - k_2)^2 + k_5^2(y_O - k_1)^2}$$

(4.5

Test images taken with the camera of a printed mesh illuminated with a white light source, did not suffer from noticeable distortion (figure 3.22) indicating the X-ray system is the major source of distortion. As it is difficult to accurately align the centre line of the camera with the centre line of the image intensifier, it is not possible to ensure that the centre of the distortion is at the centre of the image. This means two parameters (**k₁**, **k₂**) are needed by the model to identify the location of the centre of the distortion.

The FG-100 image grabber board uses rectangular pixels, which means a parameter is needed to set the ratio between the **x** and **y** axes. Two parameters, **k₆**, **k₅** scale the image in the **x** and **y** axes respectively. Scaling is used in the iterative procedure to model the grid to a grid with unit node spacing.

Two more parameters are used to locate the centre of distortion in the ideal grid (**k₃**, **k₄**). These parameters are only used to identify the model. When the model is used to remove distortion, these parameters can be set to zero, which means that the centre of distortion is mapped to the origin. A third parameter (**k**$_7$) is not required when removing distortion, as it measures the offset angle between the two grids.

The "S" distortion is modelled with a cubic equation (4.6), which relates the change in angle to the radial distance.

$$\phi_N = (\phi + k_7) + rk_{11} + r^2 k_{12} + r^3 k_8$$

(4.6

Figure 4.21 plots the error between an ideal grid, and the distorted grid after it is corrected for radial distortion only. The line at each grid node indicates the direction and magnitude of the error. Most of the image is subject to angular distortion, although the nodes to the left (which is noted to be close to electronic circuitry on the control panel) have an additional source of distortion.

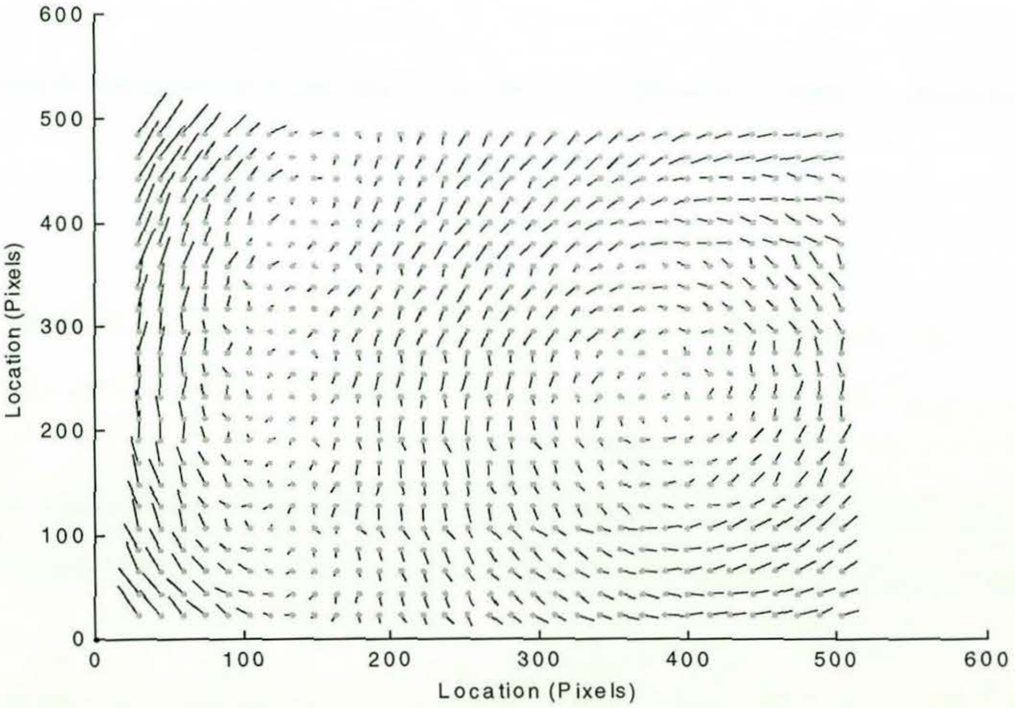


Figure 4.21 - Measured Angular Distortion Across Radioscopic Image. The lines indicate the direction, and magnitude of the difference between the measured and modelled distortion at locations across the image.

4.3.2 - Number of Parameters in the Distortion Model

Tests were made to see if all fourteen parameters are required for an accurate model. The iterative procedure "mmod.m" was modified to run with fifteen different combinations of parameters, to find the best combination. Tests were undertaken with the a small grid of 13 x 18 nodes, and a large grid of 23 x 33 nodes, both created from images of the area of the phantom with 3mm spaced holes. The two grids were taken with the same lens zoom position, the small grid uses the 120mm image intensifier zoom setting, and the larger grid is with the 215mm setting.

Table 4.3 shows which parameters are used in each test, and the final average error of the model after 120 iterations. The average error is calculated as the square root of the measure of accuracy divided by the number of nodes it is measured over, multiplied by the distance between nodes. Figure 4.22 plots the convergence of each test over time, and shows that sufficient iterations had been performed to get close to the best possible final models.

It will be noted that some of the parameter combinations include two entries for a parameter. This is because the optimising routine, "mmodall.m", deals with two parameters at a time, and expects an even number of parameters. If it finds that it is processing the same parameter twice, it optimises its search to look for improvements in the one parameter. In these tests, the initial model assumed no distortion, and gave the location of the centre of distortion for the distorted image as the centre of the image, and the corresponding location for the centre of distortion in the corrected grid.

The first test has a twelve parameter model including a third order radial model, and a third order angular model. The rest of the tests are performed on variations of this model. Tests 2 to 5 have a decreasing number of parameters in the angular model, with test 5 without even the angular offset between the two grids. Test 6 has a second order radial model, and test 7 is with a first order radial model. Test 8 is without the scaling parameters, but with the second and third order radial parameters. Test 9 is without optimisation of the centre of distortion in the corrected grid. Test 10 has no optimisation in the distorted grid, whilst test 11 has no optimisation of the location of the centre of distortion in either grid. Test 12 has a fifth order radial model, and test 13 has a fourth order radial model. Finally, test 14 has the fifth order radial model, and a first order angular model, whilst test 15 has fourth order radial and a second order angular model.

Table 4.3a Summary of Parameters Used in Each Test

Test	Description
1	Twelve Parameter Model - Third Order Radial Model
2	Three Parameter Angular Model
3	Two Parameter Angular Model
4	One Parameter Angular Model
5	No Angular Model
6	Two Parameter Radial Model
7	One Parameter Radial Model
8	No Scaling Parameters
9	No Optimisation of the Centre of the Corrected Grid
10	No Optimisation of the Centre of the Distorted Grid
11	No Optimisation of the Centres of the Grids
12	Fifth Order Radial Model
13	Fourth Order Radial Model
14	Fifth Order Radial Model, First order Angular Model
15	Fourth Order Radial Model, Second Order Angular Model

Table 4.3b Tests of the Accuracy of the Model Produced With Different Choice of Parameters

Test	Number of Parameters	Parameters Used	Average Error, (μm). Small Grid	Average Error, (μm). Large Grid
1	12	$k_2 k_4 k_1 k_3 k_5 k_6 k_9 k_{10} k_7 k_{12} k_8 \; k_{11}$	54.1725	169.8407
2	11	$k_2 k_4 k_1 k_3 k_5 k_6 k_9 k_{10} k_7 k_{12} k_{11} k_{11}$	54.2029	170.0016
3	10	$k_2 k_4 k_1 k_3 k_5 k_6 k_9 k_{10} k_7 \quad k_{11}$	55.0699	174.2028
4	9	$k_2 k_4 k_1 k_3 k_5 k_6 k_9 k_{10} k_7 k_7$	60.1088	199.6412
5	8	$k_2 k_4 k_1 k_3 k_5 k_6 k_9 k_{10}$	124.6148	218.7410
6	11	$k_2 k_4 k_1 k_3 k_5 k_6 k_9 k_9 \; k_7 k_{12} k_8 \; k_{11}$	54.5039	173.9405
7	10	$k_2 k_4 k_1 k_3 k_5 k_6 \quad k_7 k_{12} k_8 \; k_{11}$	86.5600	300.4566
8	10	$k_2 k_4 k_1 k_3 \quad k_9 k_{10} k_7 k_{12} k_8 \; k_{11}$	66.5982	230.6644
9	10	$k_2 \quad k_1 \quad k_5 k_6 k_9 k_{10} k_7 k_{12} k_8 \; k_{11}$	64.0729	192.1227
10	10	$k_3 \quad k_4 k_5 k_6 k_9 k_{10} k_7 k_{12} k_8 \; k_{11}$	64.2766	195.4597
11	8	$k_5 k_6 k_9 k_{10} k_7 k_{12} k_8 \; k_{11}$	91.5358	390.2761
12	14	$k_2 k_4 k_1 k_3 k_5 k_6 k_9 k_{10} k_7 k_{12} k_8 \; k_{11} k_{13} k_{14}$	54.1679	169.7077
13	13	$k_2 k_4 k_1 k_3 k_5 k_6 k_9 k_{10} k_7 k_{12} k_8 \; k_{11} k_{13} k_{13}$	54.1720	169.7228
14	12	$k_2 k_4 k_1 k_3 k_5 k_6 k_9 k_{10} k_7 \quad k_{11} k_{13} k_{14}$	55.0649	174.1176
15	12	$k_2 k_4 k_1 k_3 k_5 k_6 k_9 k_{10} k_7 k_{12} \quad k_{11} k_{13}$	54.2028	169.9132

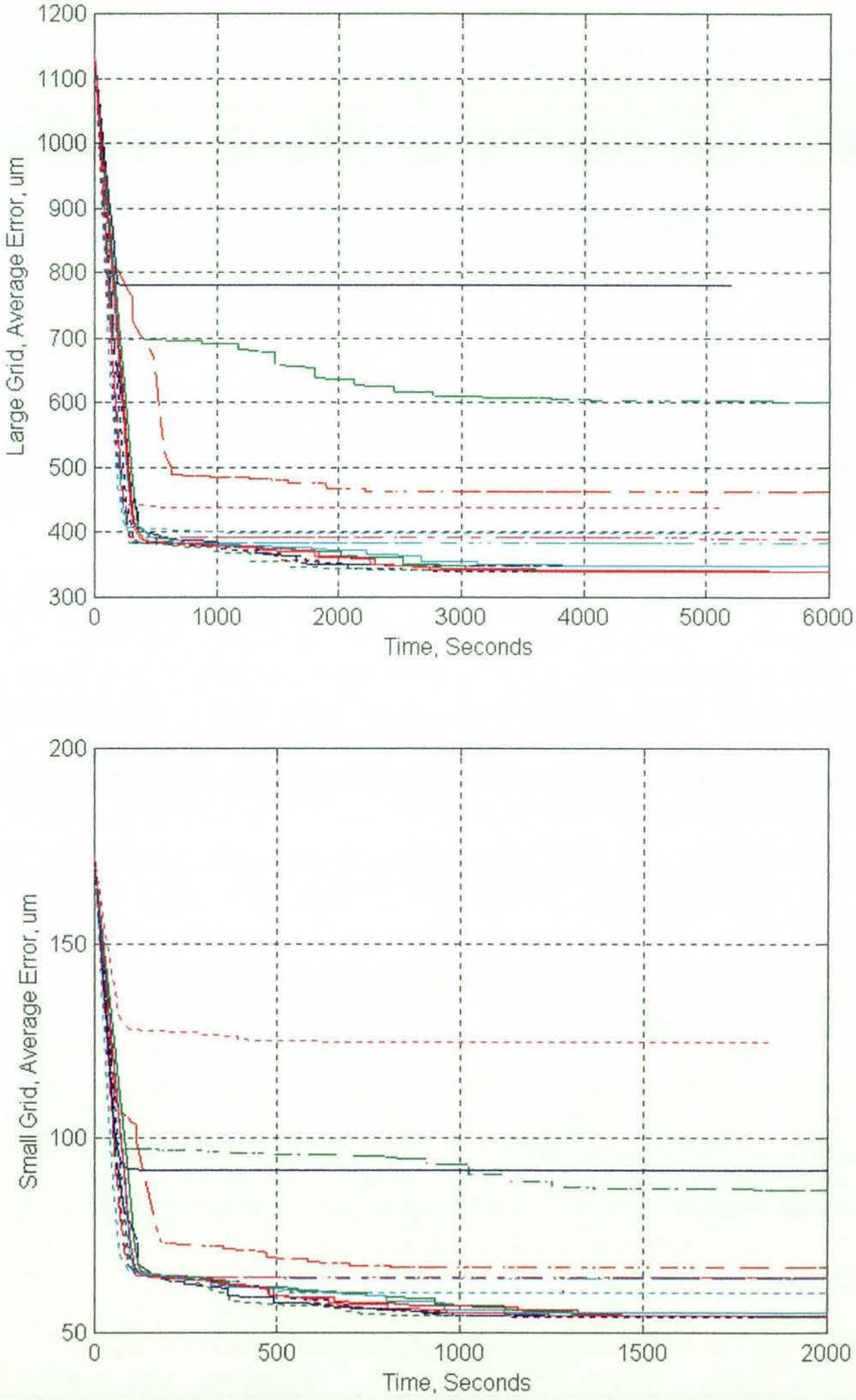


Figure 4.22 - Iterative Improvements in Distortion Model. Models listed in Table 4.3:

1: - - - - - 2: - - - - - 3: - - - - - 4: - - - - - 5: - - - - -
6: — — 7: — — 8: — — 9: — — 10: — —
11: ——— 12: ——— 13: ——— 14: ——— 15: ———

It has already been shown in chapter 3 that the large image intensifier zoom setting is subject to the most distortion. The smaller grid produced with the 120mm zoom setting has a more accurate model than the larger, more distorted grid taken with the 215mm zoom. Whilst the results from the two grids are not completely in correspondence, they are sufficiently similar to indicate the more accurate choice of model parameters.

Tests 1 to 5 indicate the model improves with more angular parameters, but with decreasing returns. Without even the angular correction (test 5), the results are very bad. A first order angular equation produces significant improvement. The improvement in the second order model is also noticeable, but there is limited advantage in the third order model. The smaller grid has proportionally greater improvement in the model with just the angular correction (Test 4), but the large grid improves more with additional angular parameters.

A lack of radial parameters (test 7) results in a very poor model. Using one radial parameter (test 6) results in a significant improvement. A second parameter (test 1) results in an improvement particularly with the large grid. Tests 13 and 12 show that the fourth and fifth order radial models result in minor improvements in the accuracy of the model. Test 8 shows that the model with no scaling parameters gives poor performance.

Test 11 results in arguably the most inaccurate result of all the tests, indicating the need to optimise at least one centre of distortion. Tests 9 and 10 show that optimising just the centre of the distorted image results in slightly better results than optimising only the centre of distortion in the model, but neither test results in a particularly accurate result, justifying the use of parameters $\mathbf{k}_1$ to $\mathbf{k}_4$ in all the other tests. Tests 14 and 15 show there is no advantage in swapping one or more of the angular model parameters with the fourth and fifth order radial model parameters.

The preferred model for future tests is that used in test 1, although it would be reasonable to omit either a radial or angular parameter when using the 120mm zoom setting. Test 1 has the advantage of an even number of parameters, which makes it easier to maintain the optimisation software.

4.4 - Processing the Distortion Model

The function "zmod.m" (appendix III.4) was written to measure the accuracy of a given model, and to map locations from a distorted image to an undistorted image. The inputs to the function are the model expressed as a twelve parameter array ("**k**"), a matrix indicating nodes in the grid to be ignored (the interpolated and extrapolated nodes from "findgrid.c"), matrices for the **x** and **y** distorted grid co-ordinates, and a control value that can be used to tell the procedure to produce a graphical output if required. There are three outputs to "zmod.m". The first is the measure of the accuracy of the model. The second is the remodelled grid. The final output is the ideal grid, which was compared with the remodelled grid to calculate accuracy of the model. Both output grids are expressed as complex numbers to reduce the number of variables.

The ideal grid is of the same size as the input grid, and of unit node spacing. A number of different measures of accuracy for the model were tested. These included the sum of the square of difference in locations, the quadratic of difference, and the product of the square of difference and quadratic of the difference between the ideal and remodelled grids. With all of these measures of accuracy, the smaller the result, the more accurate the model.

In addition to testing each measure separately, tests were undertaken to see if periodically switching between three different methods of calculating the accuracy would improve the speed of convergence of the model (figure 4.23). This was discounted after it was found that each measure tended to produce a different ideal distortion model. The higher order measures are more sensitive to larger errors, which normally occur at the periphery of the image, whilst the lower order measures result in more accuracy towards the centre of the image. Periodic changing of the measure could result in a degradation of results (figure 4.24). It was decided to use the sum of the square of difference measure in further tests as it was felt that greater accuracy was preferred towards the centre where the most measurements would take place, and this was the least complicated method.

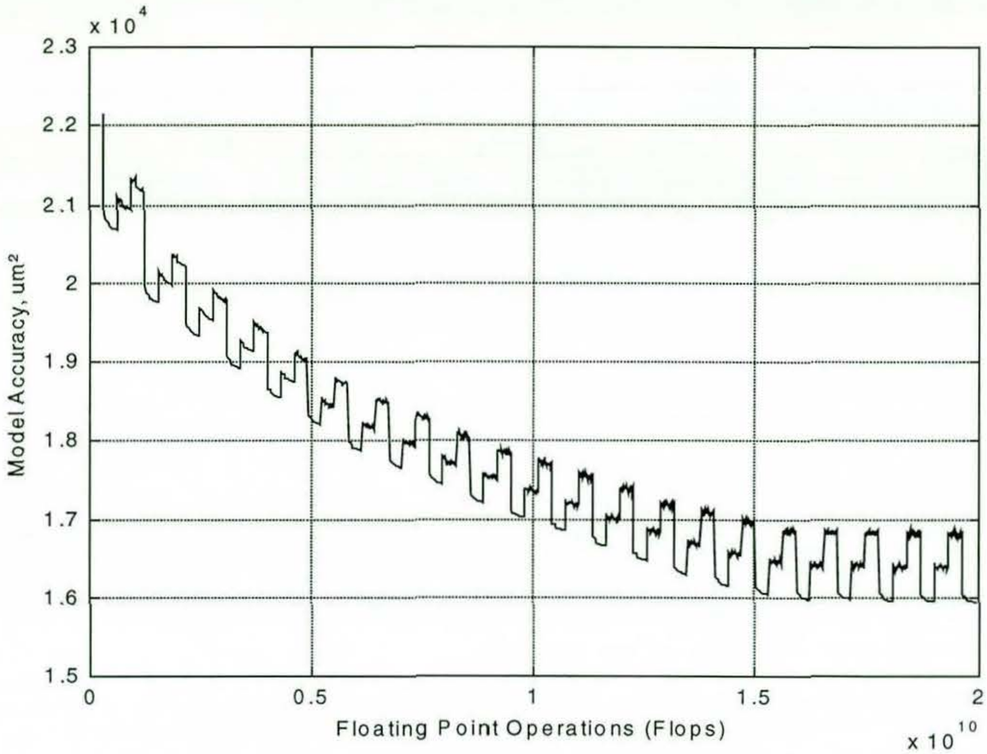


Figure 4.23 - Iterative Improvements in Distortion Model using Periodic Switching of Metric.

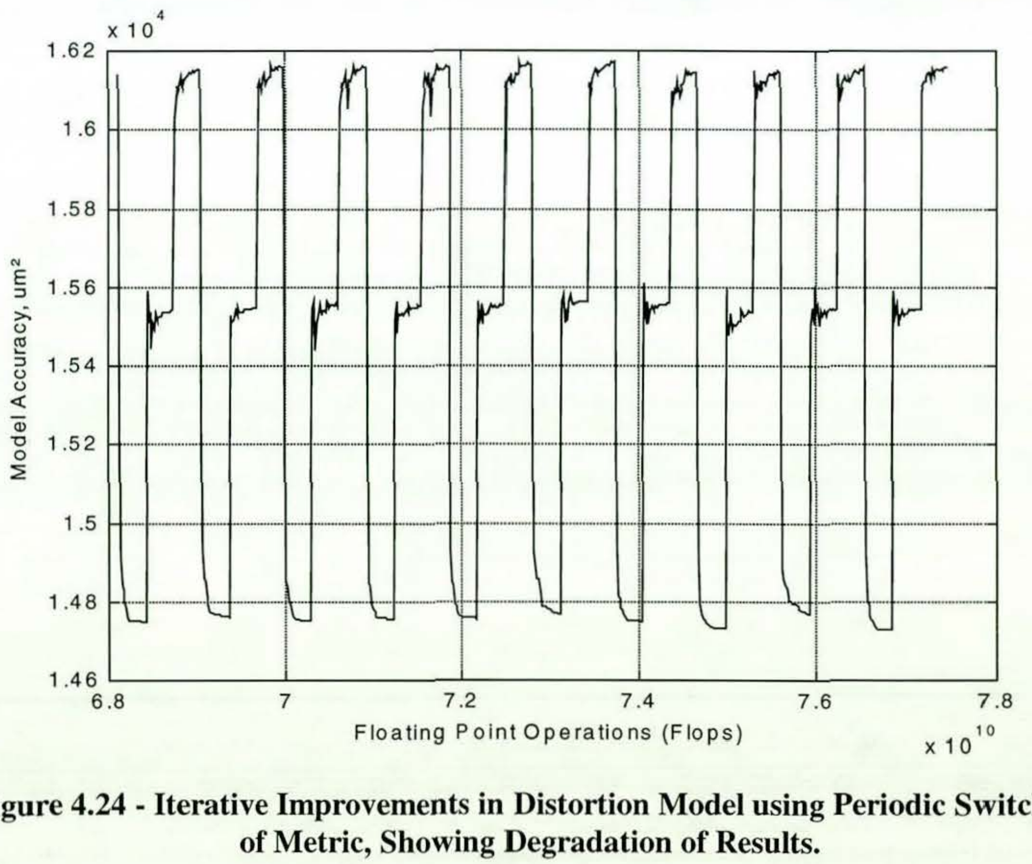


Figure 4.24 - Iterative Improvements in Distortion Model using Periodic Switching of Metric, Showing Degradation of Results.

The function “zmod.m” can be used to produce one of two graphical outputs. A control input of “3” results in the function “mmesh.m” being called (appendix III.5). This was written to plot lines indicating the difference in positions of each node in the grid (as shown in figure 4.21). The length of the lines can be scaled with a user input, whilst the direction of the line indicates the direction of the error. The ideal grid is used as the reference for these lines, and is identified by plotting a point at this end.

The second graphical output is “bmesh.m” and is obtained by making the control input equal to “4”. “Bmesh.m” (appendix III.6) is a procedure that was written to draw a grid between the node points (see figure 4.20). The procedure is called twice, so that the ideal grid, displayed in one colour, is superimposed by the re-modelled grid in a second colour. Nodes located at the origin are not plotted, which means that the image mask can be used to show only the real nodes.

4.5 - Optimising the Distortion Model Parameters

The optimisation of the model is undertaken in the “mmodall.m” procedure (appendix III.7). This calculates an improved set of parameters for given conditions. Inputs to this procedure are the values of the parameters, the order they are to be tested, the spacing between steps, the number of steps to be tested, the mask for the grid, and the grid itself. The procedure’s outputs are the improved model and the step sizes recommended for use in the next iteration.

The procedure takes two of the parameters at a time. The accuracy of every combination of values of these two parameters, over a given range and for a given step size, is tested using the function “zmod.m”. The new parameter values are taken as the most accurate combination. If the best parameter value is found to be at the limit of the range of tests, further tests of that parameter are made until a minimum is found. The parameter step size is reduced each time “mmodall.m” is called, unless reset by the calling routine, or unless the best parameter combination was found outside the normal range, in which case the step size is increased.

The new parameter values are refined with smaller steps to improve the accuracy. Five refinement tests are performed on the two parameters, each with a step size half that of the previous test. Each test attempts to improve the model by increasing or decreasing the parameters by one step size. Trials were made with up to 20 refinement tests. Figure 4.25 shows the final value of the trials. It can be seen that there is little advantage in running the refining routine more than four or five times.

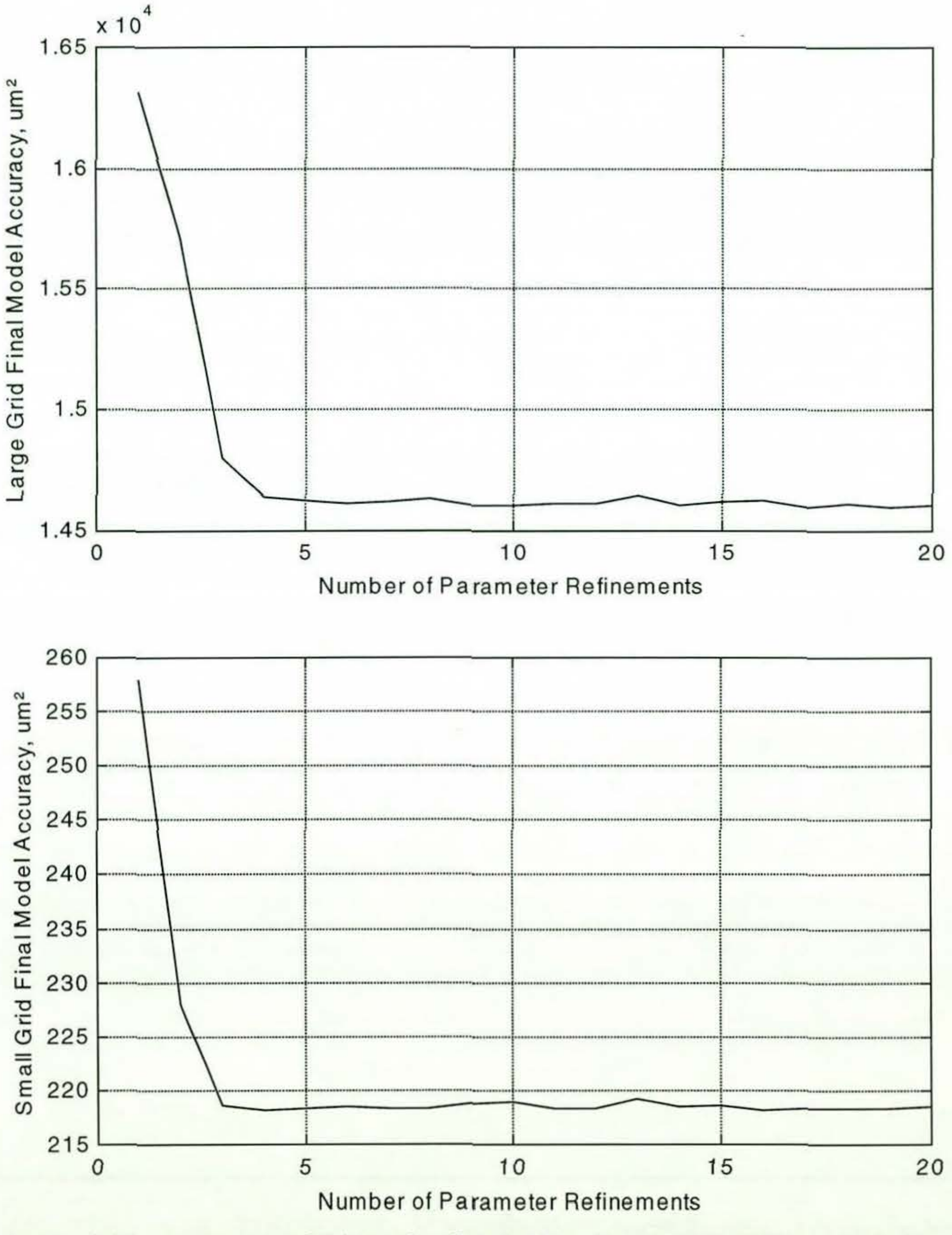


Figure 4.25 - Accuracy of Distortion Model Produced with Different Number of Parameter Refining Tests. Top plot is for the small grid, bottom plot is with the large grid.

Figure 4.26 shows how many operations are required to obtain a stable result. The number of refinement tests, affects the speed of convergence to a stable result, and the accuracy of the final model. The speed of convergence with more than five refinement tests were also noted, but not plotted for clarity. These tests took longer to achieve convergence. In some cases the trials with fewer refinement tests performed better to start with, but resulted in a poor final performance. It is noted that increasing the number of refining tests increased the processing time. Five trials is a reasonable compromise between speed and final result.

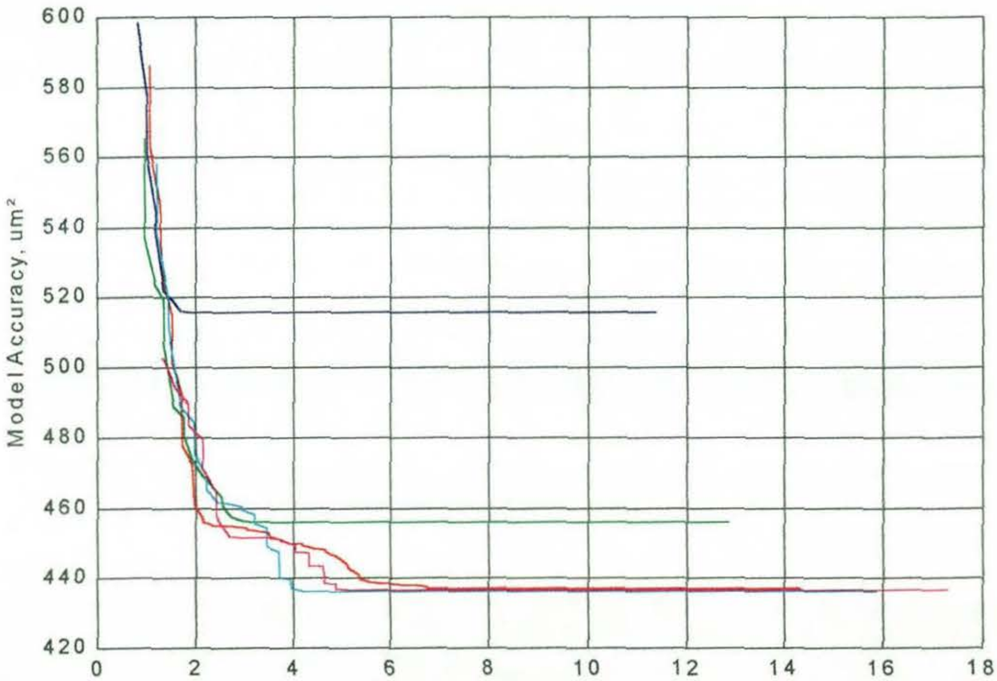


Figure 4.26 - Operations Required for Stable Results of the Distortion Model with Different Number of Parameter Refinements. Number of steps in refinement process: 1: —, 2: —, 3: —, 4: —, 5: —.

Each pair of parameters are processed in turn. Once all the parameters have been processed, further tests are performed to "slide" the parameters. A small fraction of the change in the parameter values over the optimising routine, is added to the parameters, and the result tested. If this produces a better model another fraction of the parameter difference is added. This frequently results in significant improvements in the model.

4.6 - Iterative Improvements

The procedure used to find an accurate model of a grid is "mmod.m" (appendix III.8), an iterative method used to improve the accuracy of the model. There are two parts to "Mmod.m". The first section finds a good initial set of values. Whilst the scaling is easy to estimate and the angular and radial models are initiated with the parameters for an undistorted model, finding good initial values for the centres of distortion is more difficult (see 4.6.1).

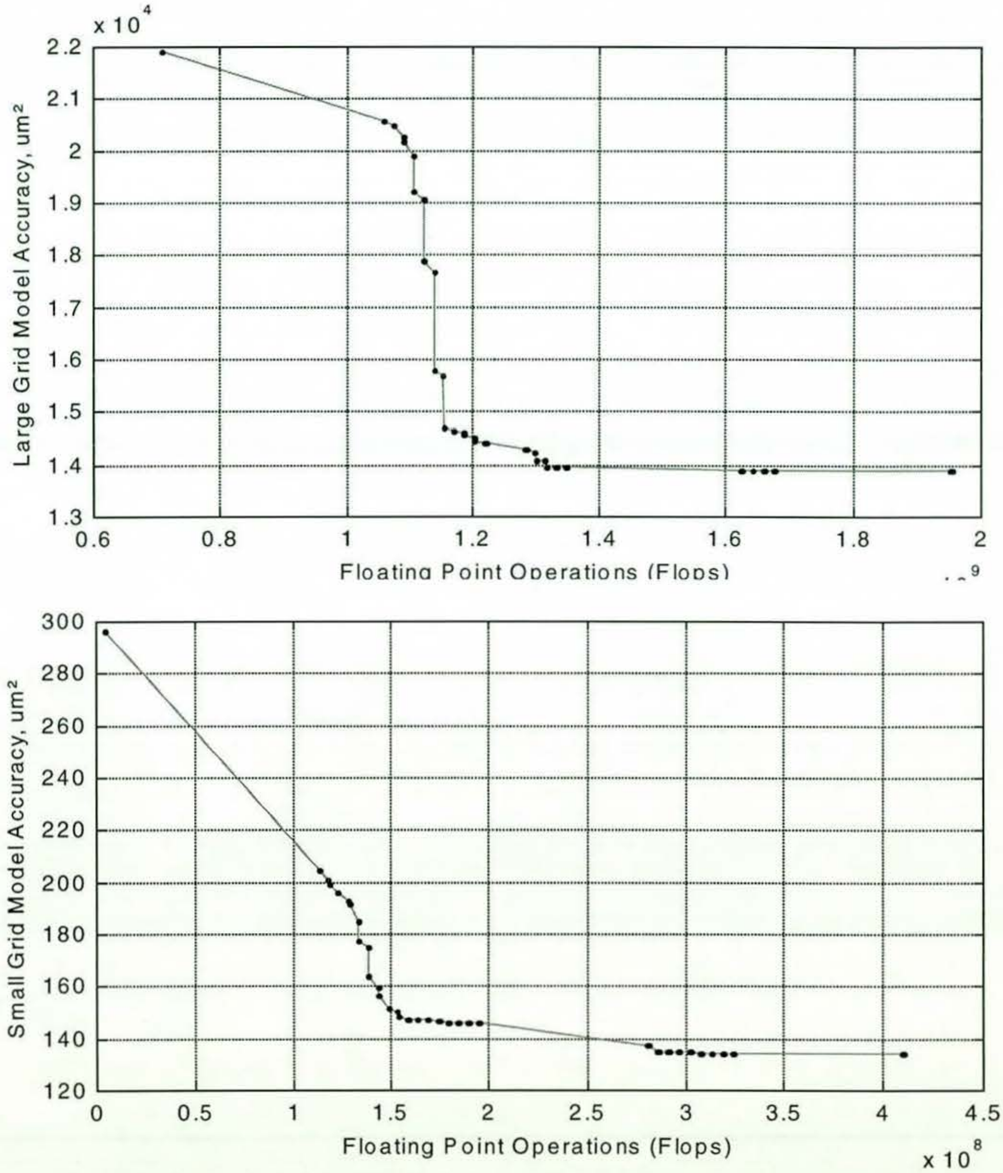


Figure 4.27 - Iterative Improvement of Deformation Models.

The second part of the procedure improves the parameters iteratively. Most of the improvements are undertaken in the "mmodall.m" optimising procedure (section 4.5), which is called by "mmod.m". Figure 4.27 shows how the accuracy of the models of the two grids improves with each iteration. These plots are produced from the final version of the software. After the first iteration, the grid correction procedure is implemented, as described in section 4.1.1. Iterative improvements to the model are made until convergence is achieved, or until the rate of improvement is negligible.

A number of variables are passed to the optimising procedure. These include the order of processing the parameters (section 4.6.2), the size of the steps (section 4.6.3) and the number of steps over which to optimise (section 4.6.4). At the end of each iteration, further improvements are undertaken by testing the average of the new set of parameters and all the previous sets of parameters found by the iterative process. If any of the averaged parameter sets result in an improved model, this is used. After the first iteration, "mmodkx.m" can be implemented to correct for the grid errors found in the 3mm grid (see section 4.1.1).

4.6.1 - Initial Parameter Values

The procedure "mmod.m" first loads the grid data that was saved by the "C" program as "gridata.m". The data is checked, and if necessary changed so that the axes are correctly orientated. The choice of the initial model for "mmod.m" is important, and affects the speed of convergence and the final accuracy. The radial and angular model parameters are initially set to zero whilst the scaling parameters are found from the mean distance between adjacent nodes. Unfortunately, as the optics are not perfectly aligned, the centre of distortion cannot be assumed to be at the centre of the image. A suitable initial location for the centre of distortion in the two grids is found by testing the accuracy of the model with a number of fixed centres of distortion.

Figure 4.28 shows the accuracy of models with fixed centres of distortion after one iteration, but with poorly chosen initial step sizes. The figure shows that there is more than one minimum in the mesh, indicating the need to find the centre of distortion before undertaking the iterative improvement process. The figure indicates that there are two significant sources of optical distortion. "Ripples" in the accuracy of the model radiating from the centre of the image (at 256, 256) were assumed to be caused by the camera and lens system. But the majority of the distortion would appear to be centred away from the visible image, and would be caused by the image intensifier.

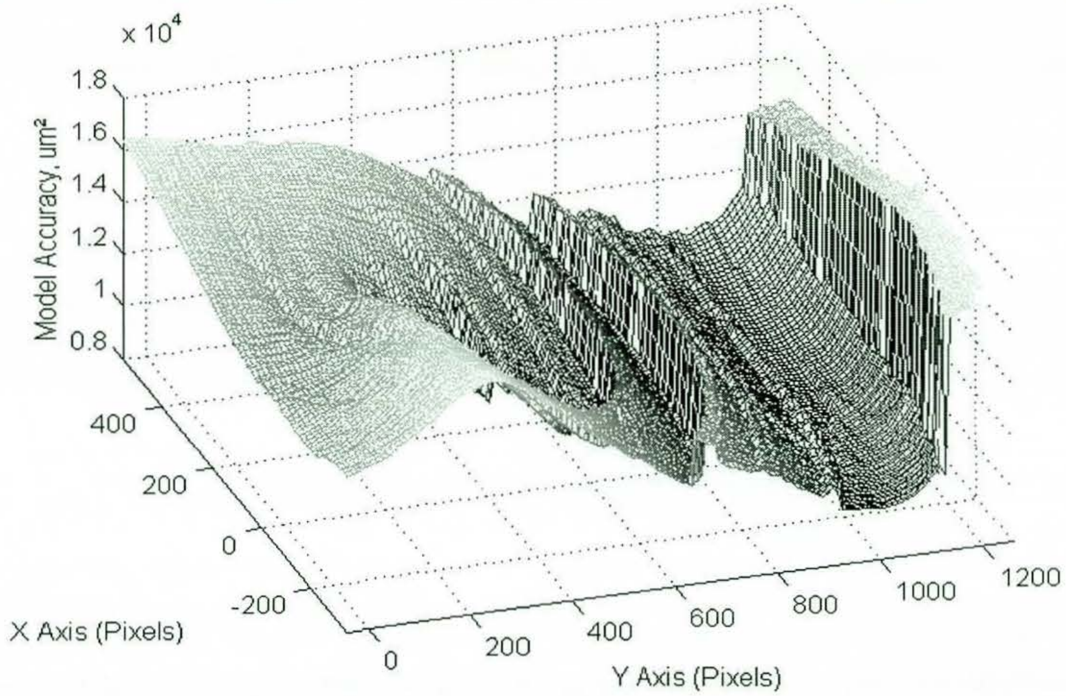


Figure 4.28 - Deformation Model Accuracy with Fixed Centres of Distortion.
Accuracy of model against centre of distortion after one iteration. Tests performed with poorly matched step sizes.

An investigation of the "ripples" indicate they are created by the optimising procedures as a result of cross coupling between parameters, and were removed by a better choice of the parameter step sizes. More tests were made to see how the models improve after further iterations. After twenty iterations (figure 4.29) it can be seen that the best initial value for the centre of distortion has changed from the best location found after only one iteration (figure 4.30). By tracking the performance of the results, it was found that location of the best initial value after twenty iterations was consistent with the result after the first three iterations (figure 4.31).

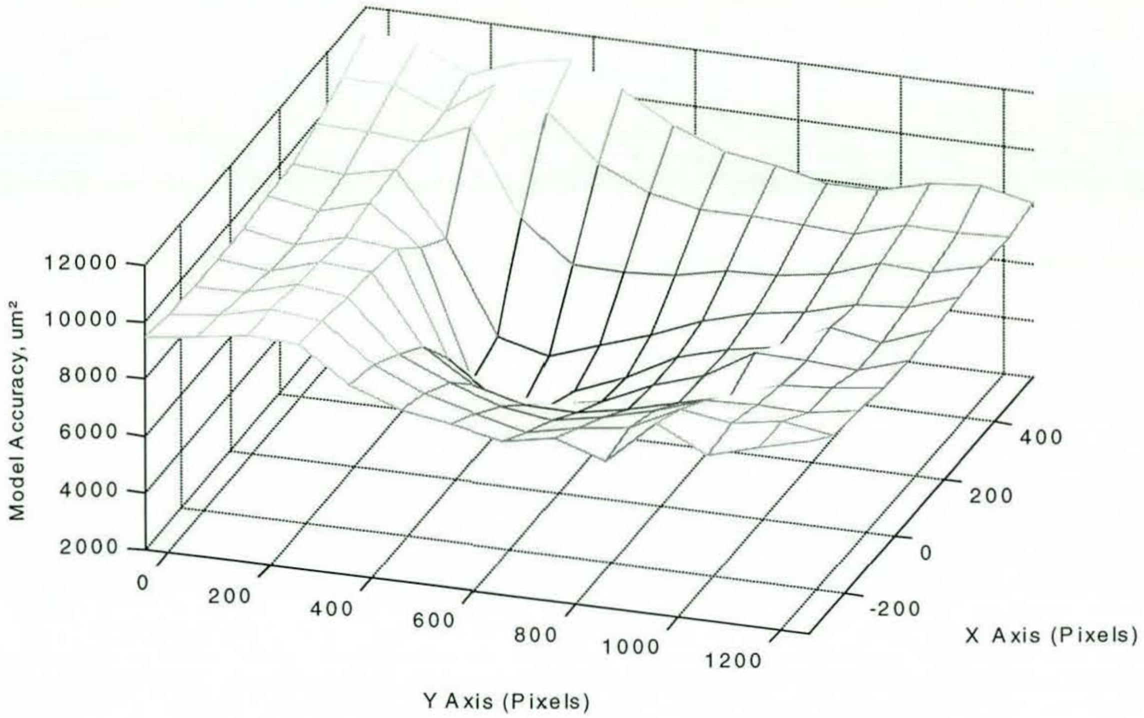


Figure 4.29 - Deformation Model Accuracy with Fixed Centres of Distortion.
Accuracy of model against centre of distortion after twenty iterations.

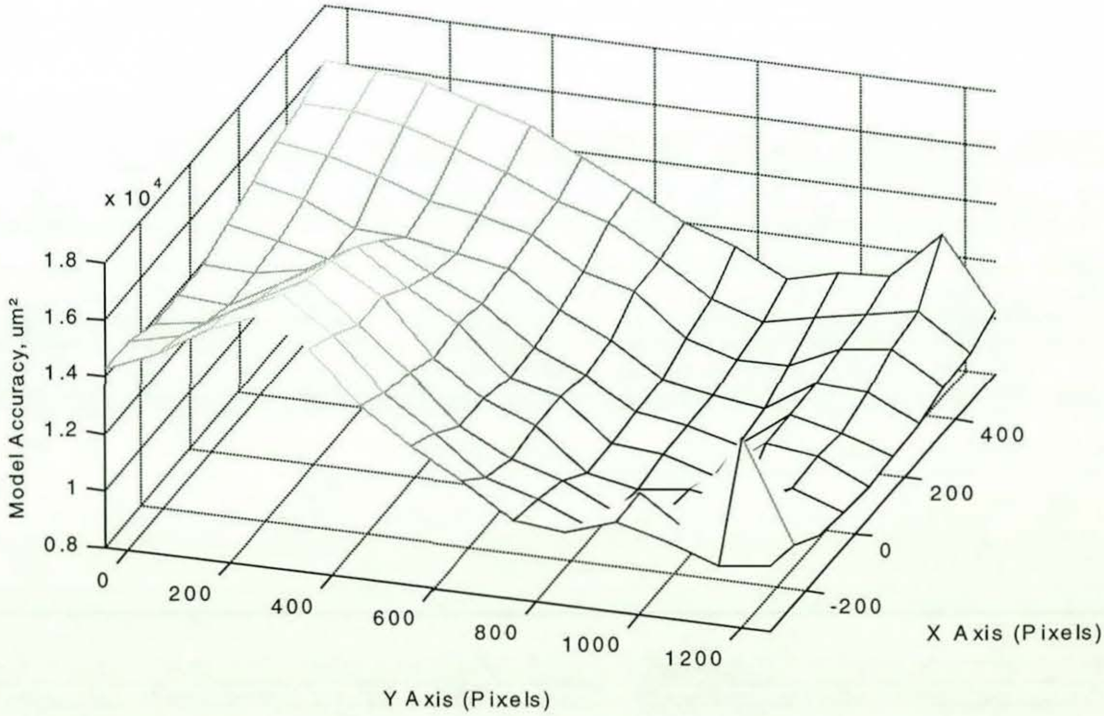


Figure 4.30 - Deformation Model Accuracy with Fixed Centres of Distortion.
Accuracy of model against centre of distortion after one iteration.

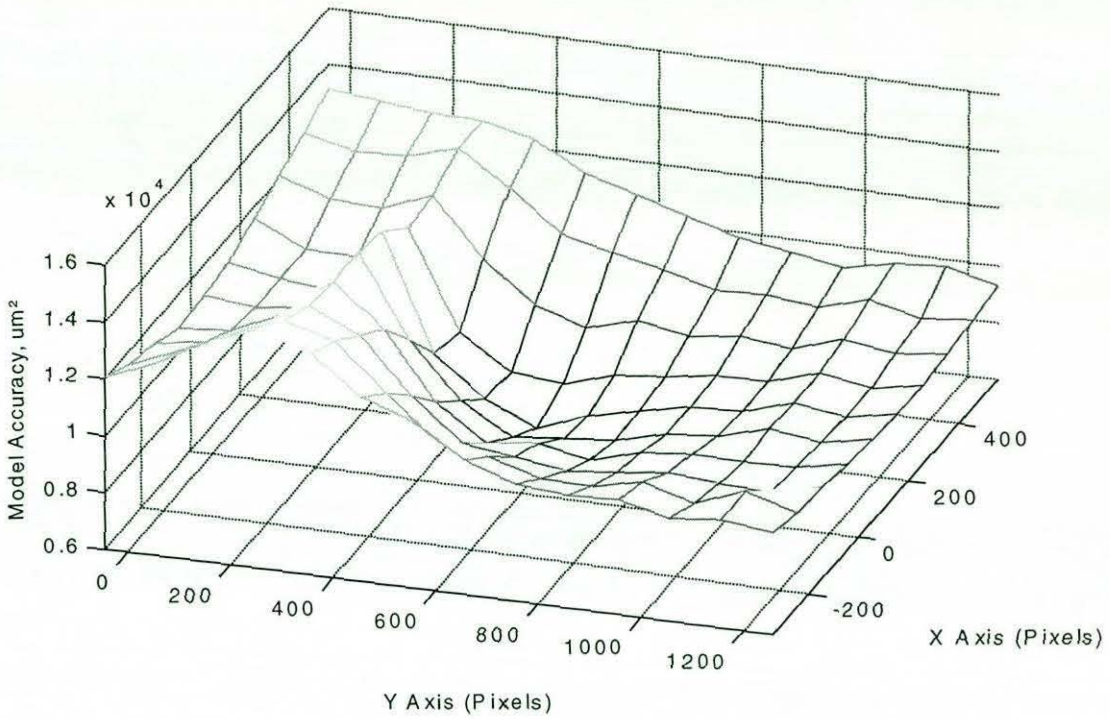


Figure 4.31 - Deformation Model Accuracy with Fixed Centres of Distortion.
Accuracy of model against centre of distortion after three iterations.

The initial location of the centre of distortion in the corrected image is calculated as the corresponding point to the centre of distortion in the distorted image. The relationship between the centres of distortion was checked by testing the relationship at locations across the image with suitable scaling parameters but no angular or radial components. After one iteration of the centre of distortion parameters in the undistorted image, the accuracy of the model was found to vary by less that 1% from the mean over an area of 800 by 800 pixels. This shows that after one iteration the model will have a relationship between the centres of distortion that is independent of the initial values.

The initial values for the centres of distortion are found using a grid search method. The first of the five 3 x 3 grids has distances between nodes of 250 pixels, and is centred at the middle of the image. Models at each grid node are refined with three iterations. In the first iteration, parameters 3 and 4 are tested with deliberately large step sizes to deal with any errors in the mapping of the centre of distortion from the distorted image. The best node in the grid was used as the centre of the second grid of 200 pixels spacing. The other three grids have a node spacing of 100 pixels, 60 pixels and finally 30 pixels. The overlapping

250 pixel grid and the 200 pixel grid are used to prevent localised minima adversely affecting the results.

4.6.2 - Order of Parameter Processing

The order for the processing of the parameters was decided after testing a number of combinations. Thirty iterations were run with each parameter combination. The measure of the model's accuracy before the iterative process was 172.6μm^2 for the small grid, and 569.3μm^2 for the large grid. Rather than testing all possible parameter combinations, the parameters were sorted into logical groups. Parameters 1, 3 and 5 are all in the vertical, **y**, direction whilst 2, 4 and 6 are all in the horizontal, **x**, direction. Parameters 9 and 10 are radial distortion, whilst parameters 7, 8 11 and 12 are angular parameters. As the iterative process deals with two parameters at a time, it was assumed that there would be no measurable difference caused by swapping the pairs of parameters.

Four tests were made with each initial parameter order, with the results after the thirty iterations, shown in table 4.4. The small and large grids were tested, first with the parameter order fixed, and then with the parameter order cycled using the Matlab command:

```
fq = rem(fq*7,12)+1;
```

This means that the parameters cycle through two sequences:

1 → 8 → 9 → 4 → 5 → 12 → 1

and

2 → 3 → 10 → 11 → 6 → 7 → 2

Thus the parameters rotate through a sequence of six different parameter orders. Whilst the cyclic parameter tests in some cases give improved performance (for example test 5), these improvements are only in situations where the fixed order test results are poor. All the best performing test results are from fixed order tests.

Test 1 is a control, with all the parameters in numerical order. Tests 2 to 8 compare different orders for parameters 1 to 6 with the order of the radial and angular parameters (7 to 12) fixed. The fixed parameters include the angular parameters first, in increasing order, followed by the radial parameters. Tests 3 and 7 are the best results from this group of tests. Both of these tests have the centre of distortion locations tested first, as the two **x** parameters together, and the two **y** parameters together, followed by the scale parameters.

Table 4.4 Tests of the Accuracy of Models Produced with Varying Parameter Orders, with and without Parameter Cycling

		Model Accuracy µm²			
Test	Parameter Order	Small Grid (fixed)	Large Grid (fixed)	Small Grid (cycled)	Large Grid
1	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12	63.3348	182.3693	63.0141	183.7454
2	1, 2, 3, 4, 5, 6, 7, 11, 12, 8, 9, 10	63.3461	184.6848	63.3602	183.5312
3	1, 3, 2, 4, 5, 6, 7, 11, 12, 8, 9, 10	56.1125	176.0934	62.2623	180.1107
4	1, 5, 2, 6, 3, 4, 7, 11, 12, 8, 9, 10	63.4673	179.2290	62.3812	183.8985
5	3, 5, 4, 6, 1, 2, 7, 11, 12, 8, 9, 10	63.4628	186.2996	61.7936	180.3300
6	5, 6, 1, 3, 2, 4, 7, 11, 12, 8, 9, 10	58.1052	178.2335	62.7833	183.0023
7	2, 4, 1, 3, 5, 6, 7, 11, 12, 8, 9, 10	56.4087	176.5508	63.1352	180.9675
8	5, 6, 2, 4, 1, 3, 7, 11, 12, 8, 9, 10	58.1885	175.1059	62.8520	184.0807
9	2, 4, 1, 3, 5, 6, 9, 10, 7, 11, 8, 12	56.1054	176.7700	62.5276	180.4569
10	2, 4, 1, 3, 5, 6, 9, 10, 7, 8, 11, 12	56.9333	175.0478	63.0527	180.8077
11	2, 4, 1, 3, 5, 6, 9, 10, 7, 12, 8, 11	56.9114	177.8929	62.9390	181.5343
12	2, 4, 1, 3, 5, 6, 7, 8, 11, 12, 9, 10	56.0408	174.8782	62.7908	183.2041
13	2, 4, 1, 3, 5, 6, 7, 12, 8, 11, 9, 10	56.2540	177.8568	63.1130	181.6910
14	7, 8, 11, 12, 9, 10, 2, 4, 1, 3, 5, 6	55.0980	172.7670	61.9107	177.3393
15	7, 8, 11, 12, 2, 4, 1, 3, 9, 10, 5, 6	56.6607	173.5215	60.9607	178.6626
16	7, 8, 11, 12, 2, 4, 1, 3, 5, 6, 9, 10	59.0914	172.9221	63.0788	181.5483
17	2, 4, 1, 3, 7, 8, 11, 12, 5, 6, 9, 10	56.1859	175.6208	62.7387	181.5252
18	2, 4, 1, 3, 7, 8, 11, 12, 9, 10, 5, 6	56.4116	172.9396	61.9831	181.9506

Tests 9 to 13 use the same order for parameters 1 to 6 as test 7, and vary the order of the radial and angular parameters. Tests 9 to 11 have the radial parameters first, followed by different orders for the angular parameters. Tests 7, 12 and 13 have varying order for the angular parameters, followed by the radial parameters. The best result (test 12) has the zeroth and third order angular parameters first, followed by the first and second angular parameters, and the radial parameters last.

Finally tests 14 to 18 are used to find which group of parameters should be calculated first. Including test 12, six combinations are tested, with all tests having each group of parameters in the same order. Tests are only made with parameters 1 to 4 before parameters 5 and 6 as this was shown to be preferable in tests 1 to 8. Similarly tests 9 to 13 were assumed to show that tests are only required with the angular parameters before the radial parameters. The best result with both the small and large grids was found to be test 14. This tests the angular parameters first, followed by the radial parameters, the centre of distortion parameters, and finally the scaling parameters. As both the grids performed well with the same tests in most cases, it is assumed that the best parameter order should be suitable for any grid.

4.6.3 - Step Sizes

The step sizes were chosen as ratios, and proportioned so that the effect of one step in any parameter was approximately the same as a step in any other parameter. This was checked by plotting a mesh of changes in the parameters. Parameters were found to be either coupled (figure 4.32) or uncoupled (figure 4.33). With coupled parameters changes in the value of one parameter affects the ideal value of the second. In this case the step sizes were chosen to try and produce diagonal plots of minimum values. With uncoupled parameters, the correct step size ratio is indicated by a near to circular minimum to the mesh.

With the step size of each parameter having a similar effect on the model, all the parameters are multiplied by 0.15. This multiplication factor was chosen after testing values between 0.01 and 0.6. Figure 4.34 shows how the models improve after a certain number of runs, against the multiplication factor. As the multiplication factor has little effect on the processing time per iteration, this plot indicates which factors result in the quickest convergence to the best model. Both the tests with the small and large grid indicate the quickest convergence occurs with the factor in the region of 0.15.

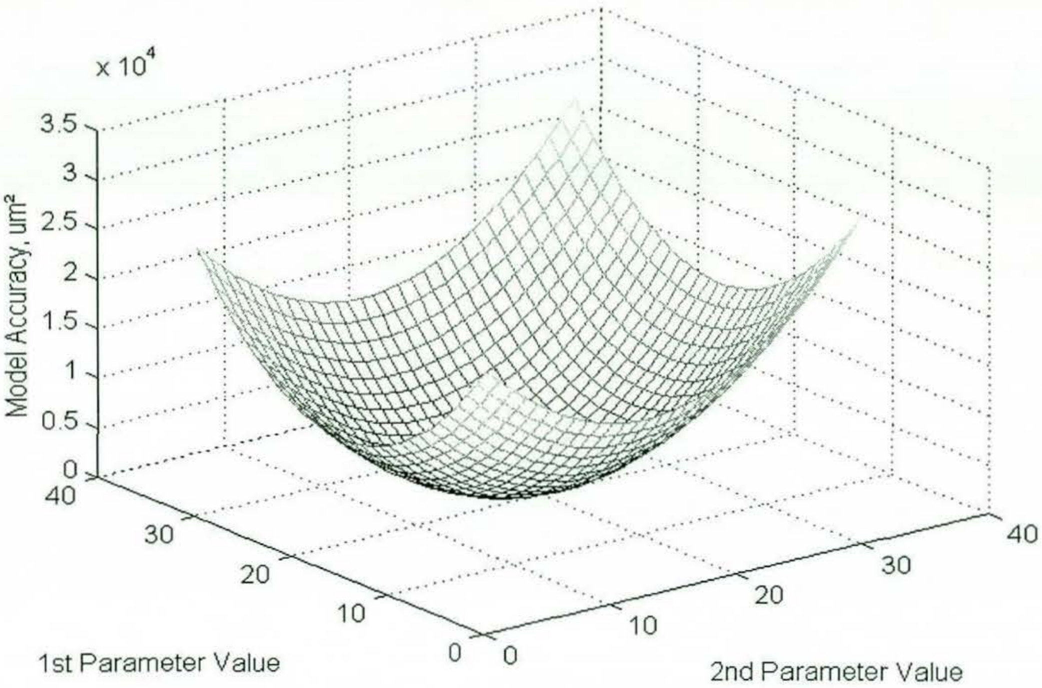


Figure 4.32 - Example of Finding the Optimum Parameter Value for Coupled Parameters.

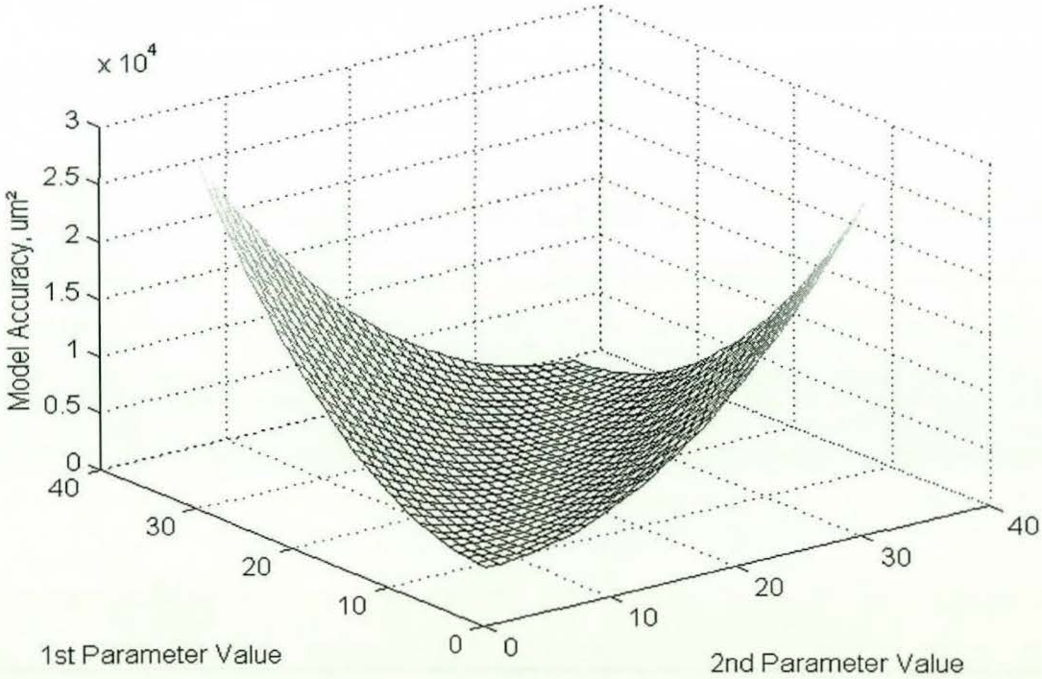
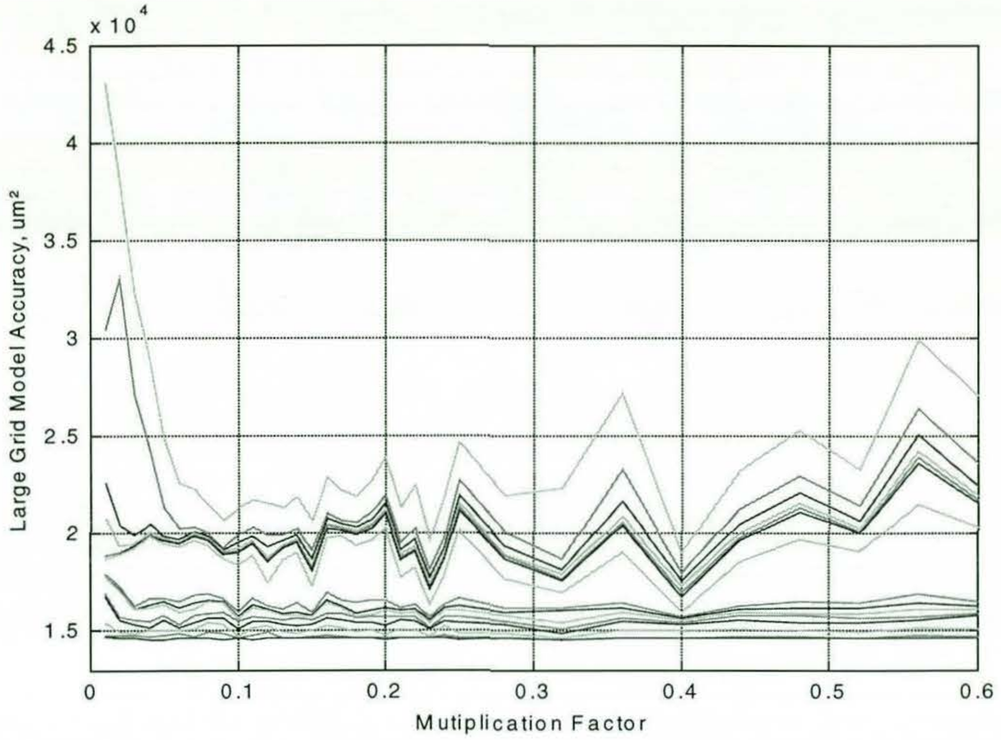


Figure 4.33 - Example of Finding the Optimum Parameter Value for Uncoupled Parameters.



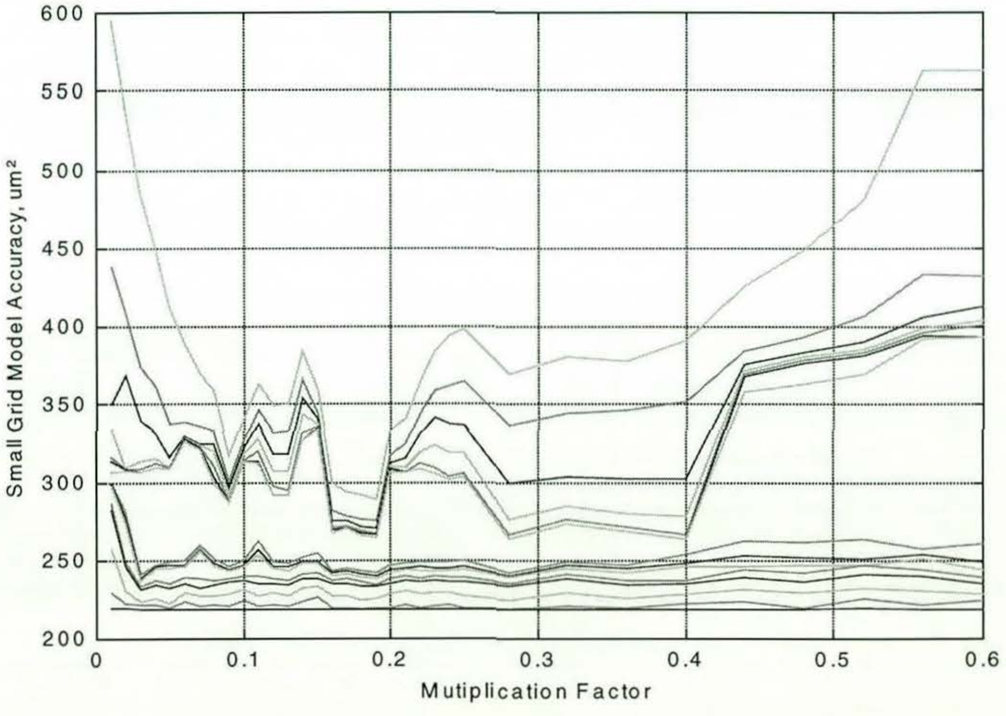


Figure 4.34 - Accuracy of Distortion Model with Different Step Multiplication Factors. Plots are for different number of runs of the iterative program.

4.6.4 - Number of Steps

The procedure "mmod.m" is able to change the number of steps over which each iteration tests. Both the small and large grids were used in trials with step sizes of between 3 and 25. Figure 4.35 shows that at least 11 steps are required to get the most accurate model. However, figure 4.36 shows how long it takes to acquire each result. It will be noted that using 11 iteration steps takes about four times as long to converge as the three iteration process with the large grid and provided a 1% improvement. The eleven iteration process takes twice as long with the small grid producing a 3% improvement.

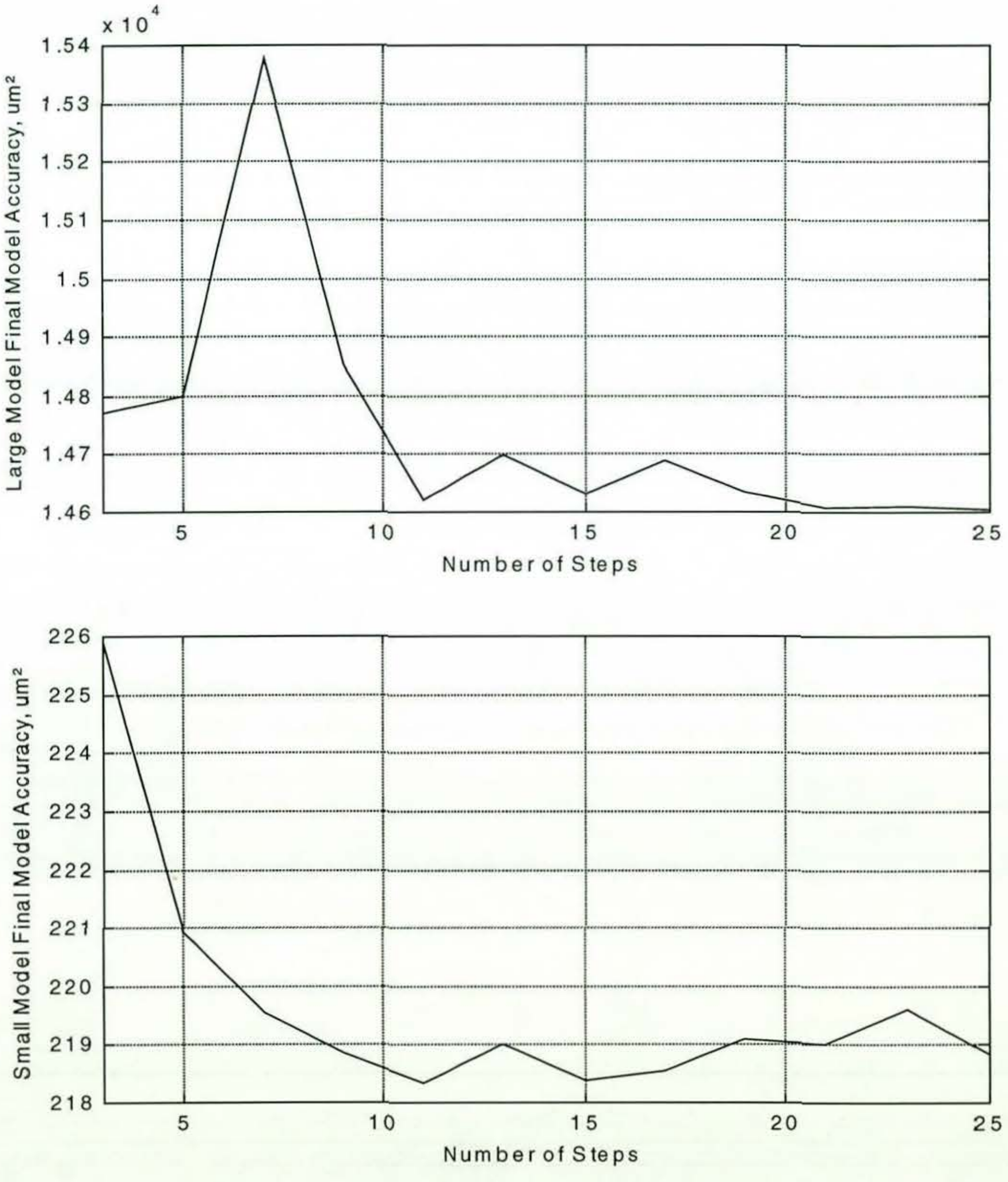


Figure 4.35 - Accuracy of Final Distortion Model Produced with Different Number of Steps in Each Iteration.

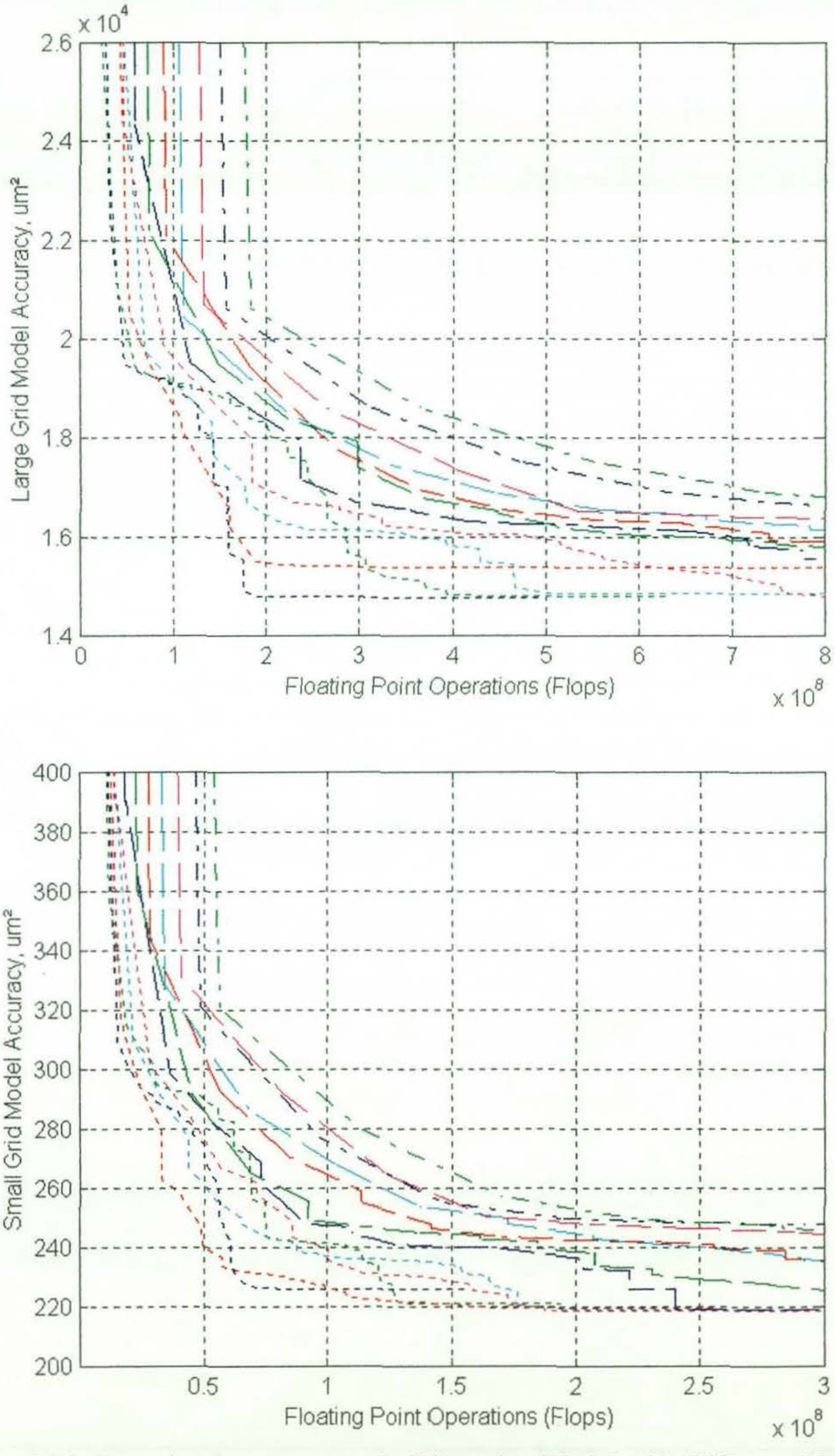


Figure 4.36 - Iterative Improvement in Distortion Model with Different Number of Steps in Each Iteration. Number of steps used:

3: - - - - -   7: - - - - -   9: - - - - -   11: - - - - -   13: - - - - -
15: — —   15: — —   17: — —   19: — —   21: — —
23: — -   25: — -

Further tests were undertaken showed that an eleven step iteration was unable to make additional improvements to the best model produced by the three step iteration process. However, a thirty-one step iteration resulted in a major improvement. Thus, the final version of "mmod.m", runs with a three step model to convergence, and then uses a thirty-one step iteration, followed by more three step iterations.

4.7 - Reverse Transformations

The distortion model "zmod.m" was written to provide the true location of a point in the distorted image. However it is desirable to be able to remove all the distortion from a distorted image. This means that the equivalent position to locations in the undistorted image must be found in the distorted image. This requires a mapping from the undistorted image to the distorted image. The locations of pixels in the undistorted image can then be founded in the distorted image, and their grey level values can be found by interpolating from the neighbouring pixels in the distorted image.

The Matlab procedure "imod.m" (appendix III.9) was written to map from the undistorted image to the distorted image. It calculates the locations of all the pixels in a 512 x 512 image after distortion. The procedure starts by finding the limits to the undistorted 512 x 512 image that will allow the whole of the image to be filled by the corrected image. It is easy to reverse map most of the distortion algorithm, however the third order radial equation has to be found on a pixel by pixel basis using Matlab's built in polynomial function, "roots.m". The last parameter of the radial equation varies across the image, so the roots have to be found for the location of each pixel.

Being a third order equation, this produces three roots, and the appropriate root needs to be identified. The required root is positive, real, and assuming the centre of distortion is within the image area, will be close to zero. The version of "imod.m" in the appendix uses the smallest positive real root. In tests on the grids processed, the software provided the correct root, but there is a possibility of error. Other methods of finding the correct root were tried. In cases where there are two imaginary roots, it is easy to find the correct root.

The correct root can be double checked by mapping back onto the original location with "zmod.m". However this method was found to fail on occasion, as more than one root maps can map to the correct location, and rounding errors could give precedence to the wrong root. It was noted that for all locations on both grids, the third root was correct with the chosen method.

Having found the location each pixel distorts to, the procedure "cout1.m" (appendix III.10) is used to remove distortion from a distorted image. Each distorted image has to be processed by "cout1.m" separately, but the pixel mappings found in the "imod.m" only has to be run once per zoom setting. This procedure finds a value for each pixel in turn, as a function of the four nearest pixels in the distorted image. Figure 4.37 shows the corrected version of the grid in figure 4.10. It can be seen that following correction, the majority of the spatial distortion has been removed.

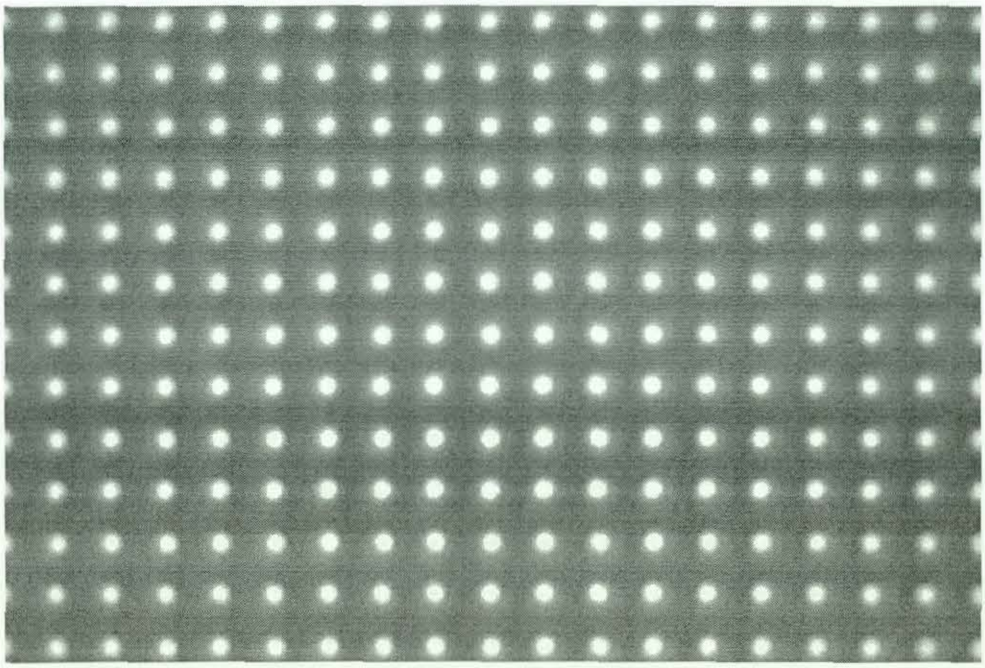


Figure 4.37 - Radioscopic Image of Distortion Correction Phantom after Correction.
The original image is in figure 4.10.

4.8 - Summary

The phantom used for distortion correction was found to have measurable variations in the distance between the holes. Some of the variations in the location of the holes in the 3mm grid were identified as being predictable, and a method of correcting for this error was produced. The rest of the errors were assumed to be random, and centred on the expected location of the hole. Provided that a large number of holes are used in the distortion model, the error in the location of the holes will be averaged out across the whole image.

A twelve parameter model was found to be suitable for modelling the distortion. The model includes four terms to identify the centre of the distortion in the distorted and the corrected image, a three parameter radial model (plus one parameter to scale between the two axes), and a four parameter angular distortion model (including angular offset). Work on the iterative procedure method saw significant improvements in the speed of optimisation. The first design took several days to produce an accurate model. By careful choice of the initial values of the centres of distortion parameters, and other minor modifications, the software now produces a final model in under thirty minutes.

This chapter gives details on the measurement, modelling and correction for spatial distortion. In order to measure the distortion, a blank radioscopic image and a radioscopic image of the phantom are required. These are combined in the Matlab routine "dewarp.m". There may be a need to make alterations to this function, depending on the contrast in the radioscopic images. The Matlab routine produces a composite image, which is used by the program "findgrid.exe". This program identifies a grid that is saved as an ASCII file which can be read by Matlab.

The distorted grid can be modelled using the Matlab routine "mmod.m". This routine produces a twelve parameter model of the distortion. The model can be used to produce undistorted radioscopic images, or to identify the true location of features in the radioscopic images. In the majority of cases the model will be used to find the true location of points in radioscopic images. However, the chapter also gives details of how to obtain an image corrected for spatial distortion.

The distortion modelling process has been applied to a number of radioscopic images. The most difficult to accurately model were found to be images that were taken with the 215mm zoom setting. Figure 4.39 shows the grid produced by the holes in the image taken with the 215mm zoom setting that is used in chapter 7. This grid displays classic pin-cushion distortion. Figure 4.39 shows the same grid after the grid has been corrected with the final version of the distortion model. It will be noted the grid has been scaled to give it a unit spacing. It can be seen that the corrected grid suffers from very little pin-cushion distortion when compared to the original grid. Some distortion is still present in image 4.39, particularly in the horizontal lines. The distortion correction process is able to remove significant quantities of distortion from the radioscopic image, but being a global model, it is unable to remove all the distortion across the whole of the image. With the present design of the model, and the way it is determined, the centre of the image is the most accurately modelled.

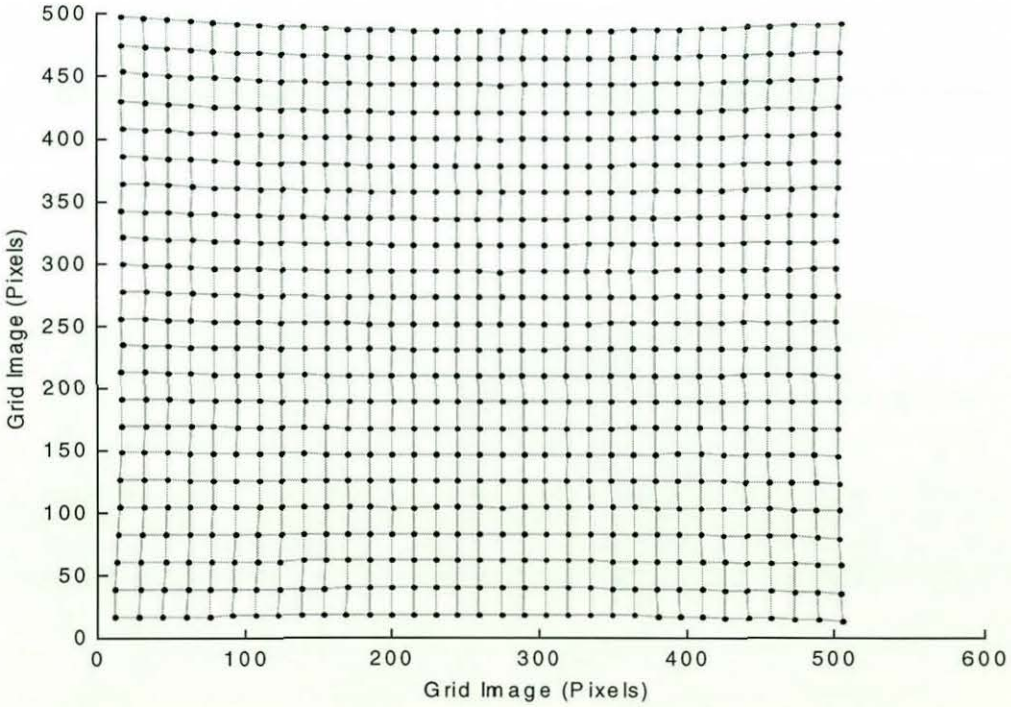


Figure 4.38 - Locations of Holes in Phantom before Correction.

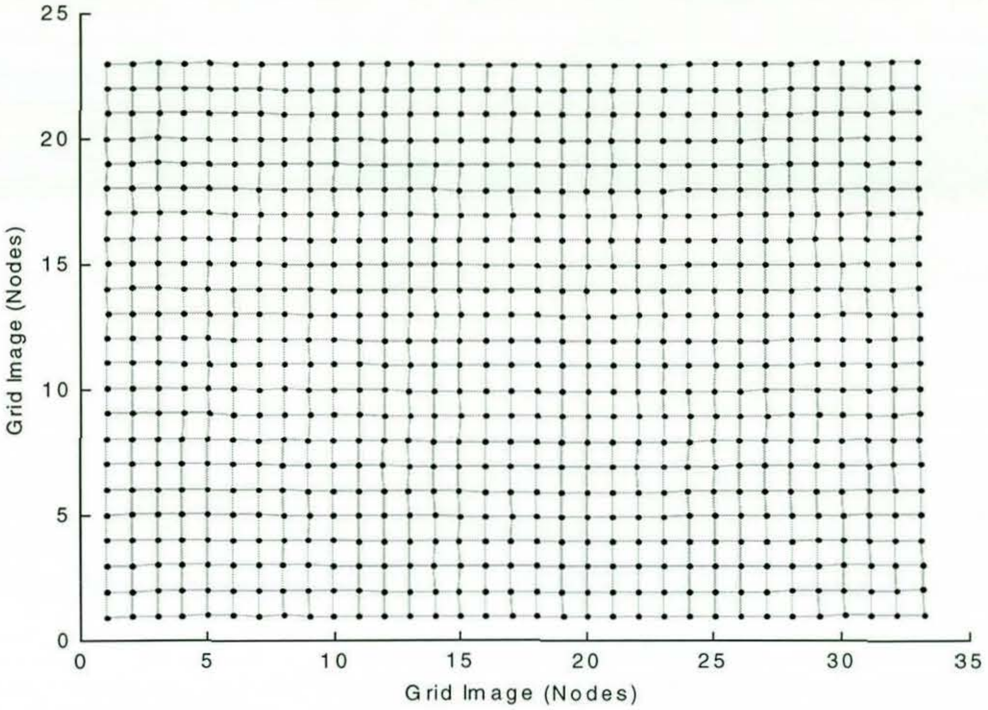


Figure 4.39 - Locations of Holes in Phantom after Correction.

5 - Test Sample for Measuring Deformations

Before any radioscopic strain measurements could be made, a suitable material had to be found for testing, and a method of applying strains needed to be identified. As the sample was intended to test the performance of the pattern matching methods, the material would need to repeatably withstand measurable strains. This would enable the optimisation of the pattern matching method to maximise the chance of the method making the correct match.

The sample needs to exhibit sufficient contrast in the radioscopic images to enable pattern matching. As few materials provide natural contrast, the contrast will need to be added as an X-ray opaque seed material. The seeds have to be introduced into the host material, preferably at the manufacturing stage. Host materials that set to a solid are particularly suitable for the tests as the seeds can be mixed in before the material sets. A number of materials were tested, including plaster of paris and car body fillers. The tests were to find how easy it was to introduce seeds, and to determine the distribution of the seeds once the material had set. However, these materials were discounted for the strain measurements as they are unsuitable for large strains. A rubber sample was finally chosen as the most suitable host material as it could be used to determine the maximum and minimum measurable strains of the technique.

The seed material has to give contrast to the radioscopic images. The material needs to be chosen to maximise the contrast. The most suitable type of seed was considered to be small particles, although tests were made with ball bearings. The ball bearings could be added to the material in a structured manner, allowing direct measurements of distances between the seeds, but they were found to be too intrusive when introduced to the host material.

The size and number of seeds in the material need to be chosen to maximise the accuracy of the pattern matching methods without adversely affecting the host material's properties. The size of the seeds in the radioscopic images is dependent on the zoom settings. This means that by changing the zoom, one seed sample size can be used to test a range of radioscopic particle sizes. However, it should be noted there will be better contrast with larger seeds because of their greater thickness. Contrast is also affected by the choice of camera and frame grabber, and large particles are more likely to affect the host material's properties. As the apparent size of the seed could be altered by the zoom settings, no tests were undertaken to find the optimum seed size, but tests were undertaken with simulations of the radioscopic images to find the optimum percentage of the image covered by seeds.

Many composite materials are ideal for manufacture with radioscopic seeds embedded within them. The seeds could, for example, be mixed with the resin in one layer of a carbon fibre composite material. Alternatively, seeds could be fixed to internal surfaces that could be periodically X-rayed to detect impending failure.

5.1 - Filler Materials

The first test samples were made of plaster of paris and car body fillers. As these materials start life in a powder, liquid or paste form and set to solid, it is possible to introduce seeds, and to create a desired shape. Samples were made to see how easy it was to produce a sample with sufficient contrast detail to allow deformations to be measured.

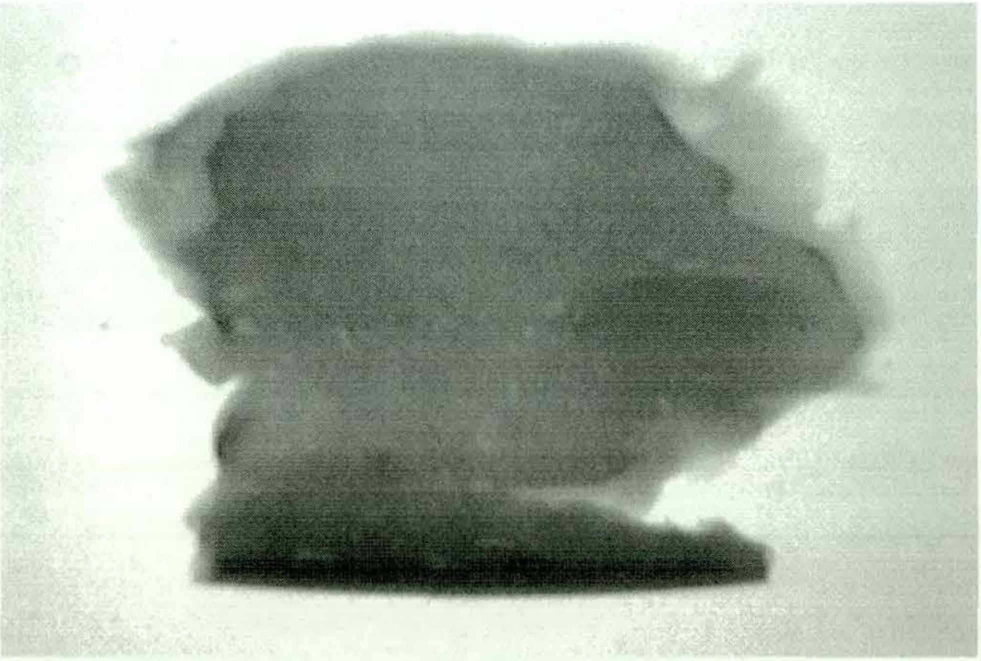


Figure 5.1 - Radioscopic Image of Plaster of Paris Sample.

A small quantity of plaster of paris was mixed with water and allowed to set. This first test piece (figure 5.1) was used to prove the radioscopic system could take reasonable images of plaster of up to 10mm thickness. With the maximum X-ray energy, the aperture could be kept acceptably small to prevent blurring of the image. It was possible to detect the thicker sections of the plaster in the radioscopic image, and also to identify a number of air bubbles. The top surface of this image was found to be quite rough when set, despite efforts to give it a smooth finish. Whilst the rough surface will slightly improve the radioscopic image contrast, it was considered undesirable for strain tests, as it would result in material weaknesses, and local strain variations.

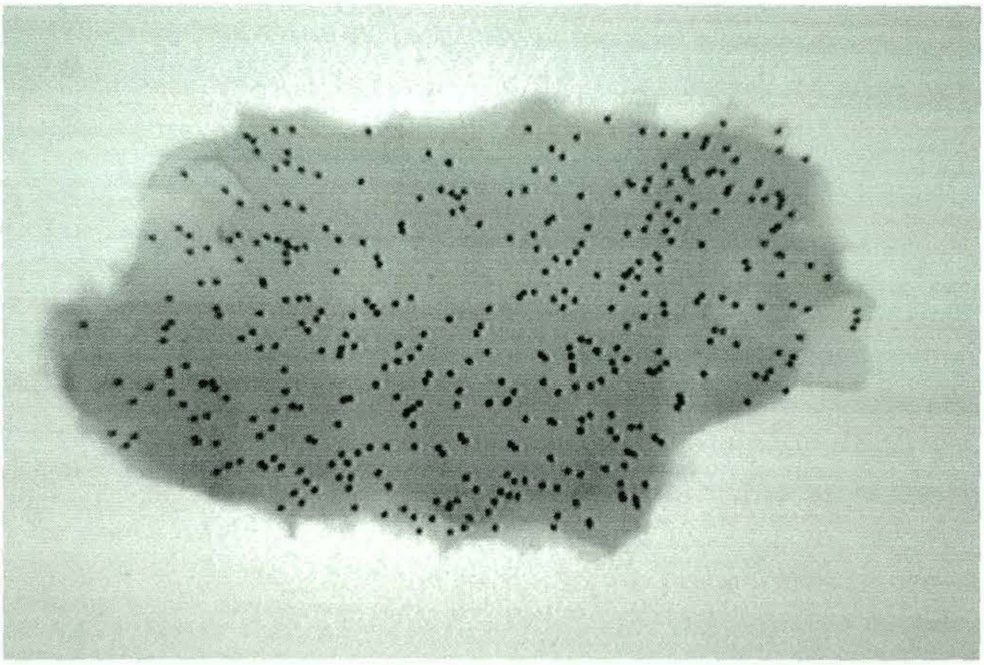


Figure 5.2 - Radioscopic Image of Plaster of Paris Sample with Ball Bearings.

Tests were made on how to introduce contrast into the images of the samples. Impurities were added to the plaster to see how much detail they introduced to the radioscopic images. Tests were made with 1mm steel ball bearings (figure 5.2) and a sample of aluminium oxide powder (figure 5.3). The ball bearings are very easy to identify, and proved good radioscopic seeds. However, their large size meant the plaster mix was no longer a smooth paste. It was very hard to distinguish the particles of aluminium oxide in the other sample. This was due to their smaller size, and the smaller difference between the radiographic opaqueness of aluminium and the plaster.

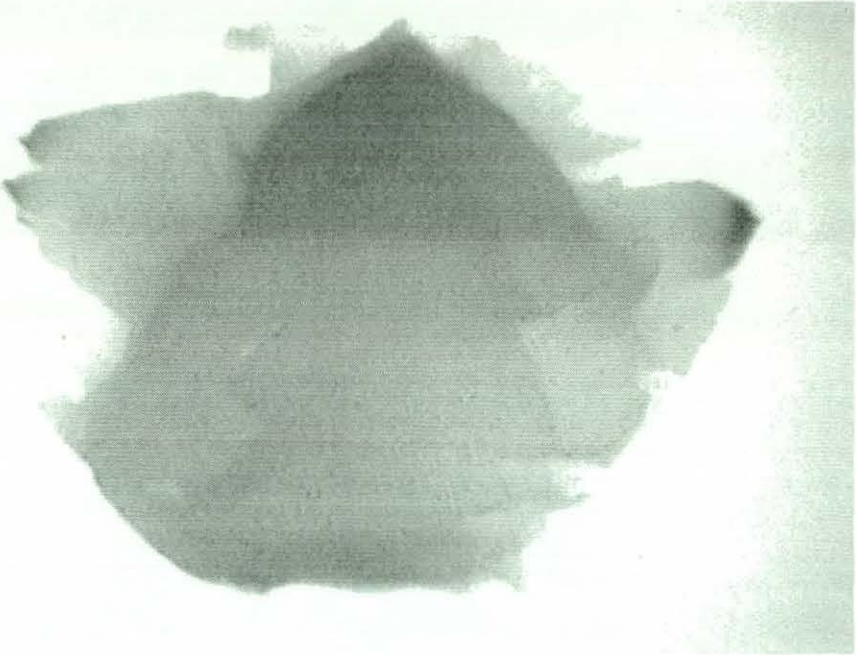


Figure 5.3 - Radioscopic Image of Plaster of Paris Sample with Aluminium Oxide.

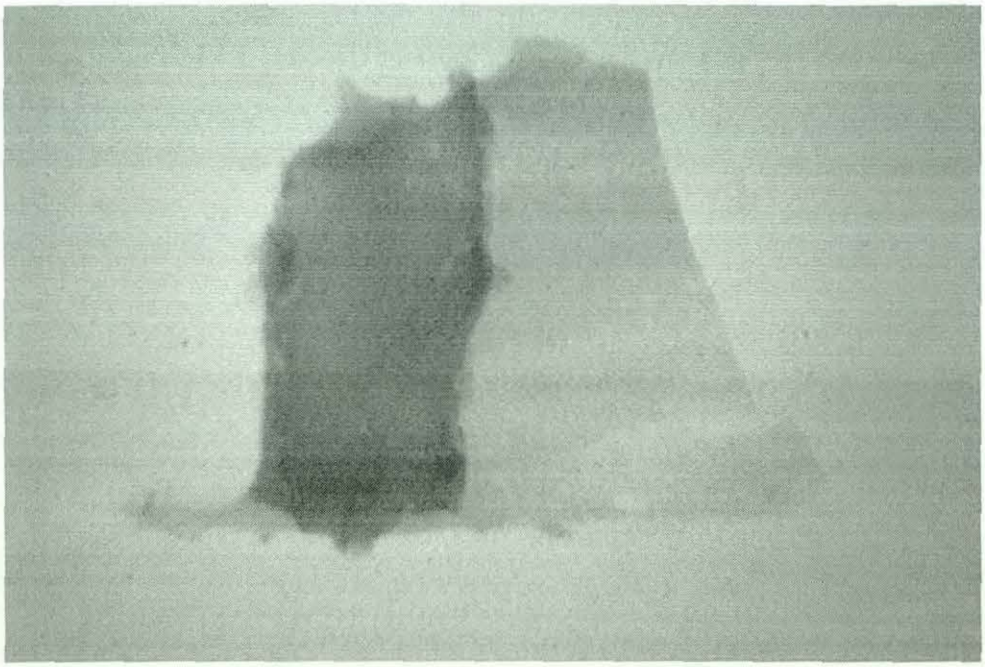


Figure 5.4 - Radioscopic Image of P38 Filler Material Sample.

Three car body filler materials were tested for their suitability. Their greater flexibility, compared with plaster, made them more suitable for strain measurements. The first filler sample was a material called "No-Mix". This sample was discounted as it was found to take several weeks to set properly, and to have very little strength. Despite making a sample with two thicknesses, the second filler sample, P38 (figure 5.4), was found to be too thin to give much image contrast. The manufacture of the P38 sample indicated that a smooth surface finish would also be difficult to obtain due to its sticky consistency. The sample was found to be brittle. The third sample, made with P40 body filler, was also found to have little contrast. Being a fibre and resin material, it was very sticky until set. This resulted in a rough surface finish. Of the samples tested, it was the best suited to withstand strains, but the material's high strength would need to be considered when building a rig to apply the strains. More samples were used to test the ease of mixing the aluminium oxide samples with the car body fillers. Mixing the seeds in before adding the hardener, was found to produce a good seed distribution.

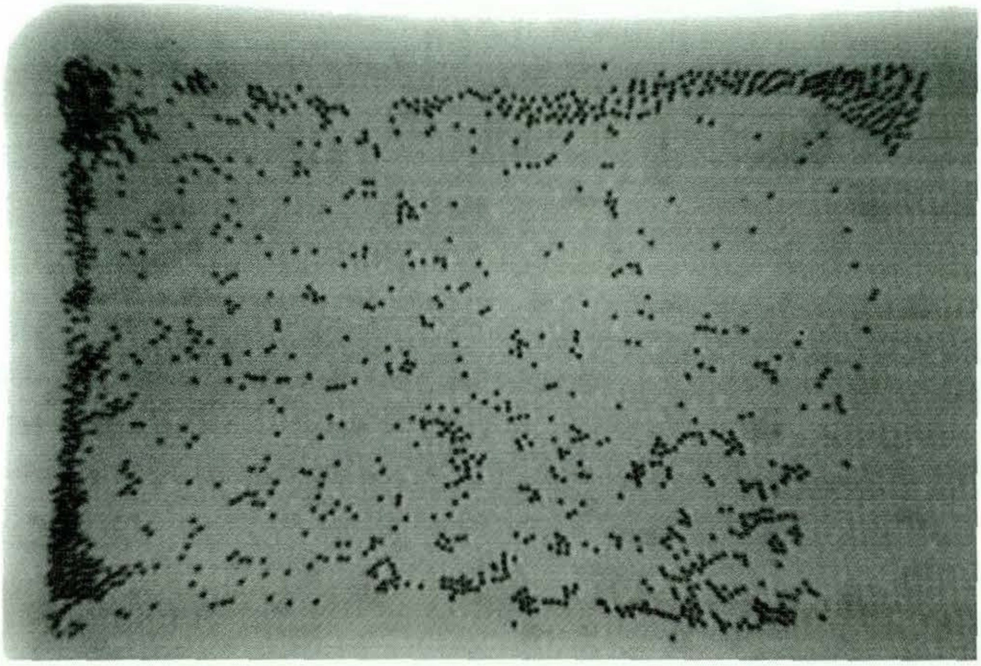


Figure 5.5 - Radioscopic Image of Moulded Plaster of Paris Sample with Ball Bearings.

Further tests were undertaken with cast samples. These tests used a mould to try and improve the shape and surface finish. The mould was rectangular with rounded corners to assist the removal of the sample. Figure 5.5 is of a plaster of paris sample with 1mm steel ball bearings. Looking at the bottom of the sample, the ball bearings are visible, indicating they sank to the bottom of the casting. It can be seen that the ball bearings have migrated to the edges of the mould, leaving a very poor distribution across the sample. This may have been caused by tapping the sample to try to force out air bubbles. Despite the efforts to remove the air bubbles, they are still distinguishable in this sample.

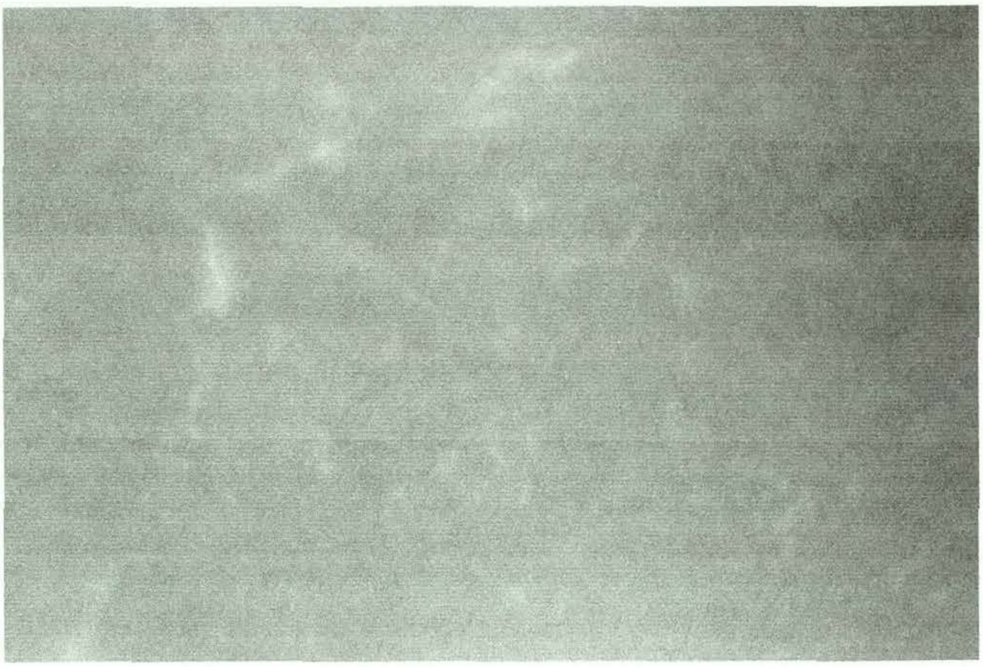


Figure 5.6 - Radioscopic Image of Moulded Plaster of Paris Sample (Dry Mix).

More samples were made with different plaster-to-water ratios to try and minimise the air bubbles. A very dry sample (figure 5.6) was found to suffer from cracks. This sample had a rough surface finish. The runniest sample resulted in dark patches (figure 5.7), which it was suspected indicated areas where the plaster was not properly mixed with the water. An intermediate consistency mix was found to be easy to work with, and a sample with the aluminium oxide powder was produced (figure 5.8). But once again the aluminium oxide proved to be difficult to detect.

The materials tested were not considered to be suitable for investigating the radioscopic strain measurement method as none were likely to withstand measurable strain. The darkness of the radioscopic images was dependent on the cross section (although local inhomogenities were sometimes detectable). Air bubbles were noted, but they provided little contrast.

The ball bearings were easily detectable in the radioscopic image, but were badly distributed The tests indicated the desire for an easily stretched material host with low X-ray absorbency, seeded with small particles of a highly opaque material.

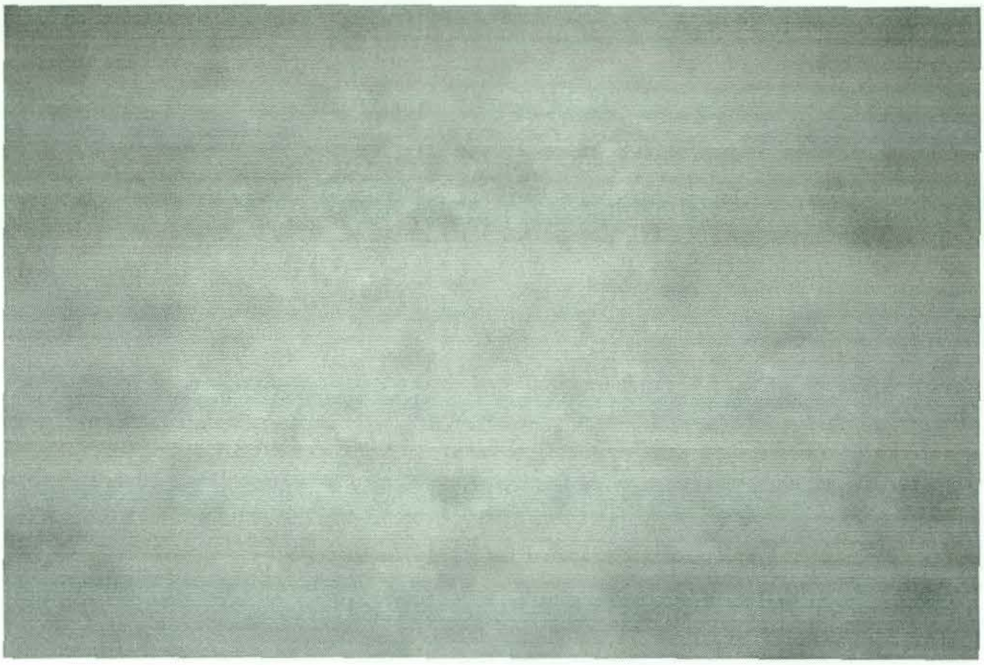


Figure 5.7 - Radioscopic Image of Moulded Plaster of Paris Sample (Wet Mix).

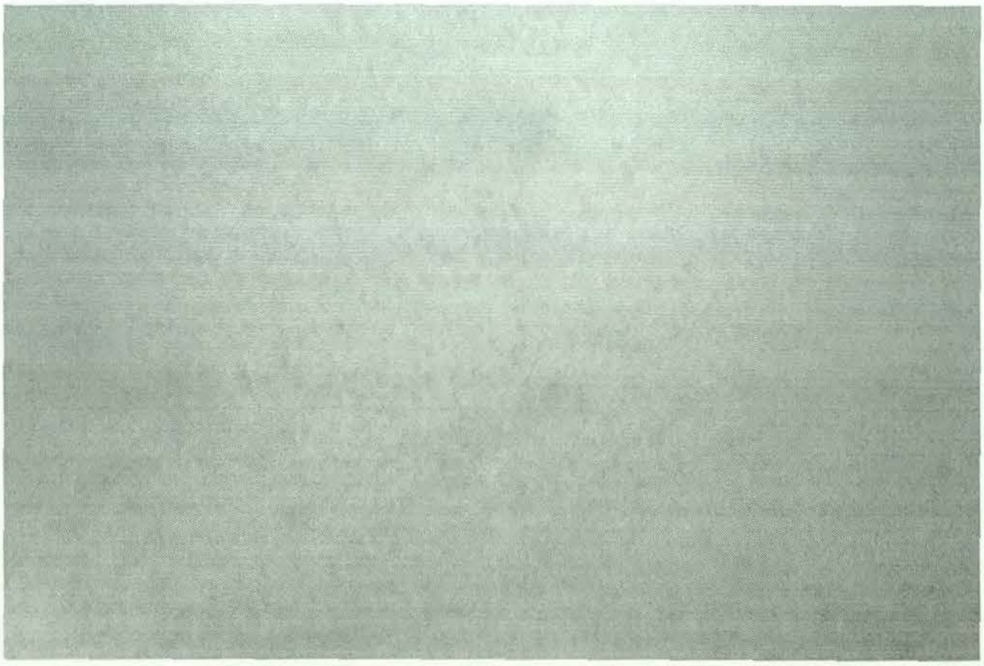


Figure 5.8 - Radioscopic Image of Moulded Plaster of Paris Sample with Aluminium Oxide.

"Sylgard 170" is a two part silicone elastomer normally used to pot electrical components. On mixing the two liquids, "Sylgard 170" sets to a flexible rubber. The maximum tensile strength is 3.5MPa, and the maximum elongation is 150%, although this value decreases with time [128]. "Sylgard 182 and 184" are transparent potting compounds with a tensile strength of 6.2MPa, and maximum elongation of 100% [129]. A price of £75.20 was quoted for a 1.1kg pack of "Sylgard 184". Details were also obtained for "Sylgard 186". This material has a tensile strength of 5MPa, and maximum elongation of 420% [130]. Of the three, the lower tensile strength of the "Sylgard 170" would make this the easiest to work with. Any of these materials would be suitable as a test sample providing a way could be found of introducing the seed material. However, the University's Institute of Polymer Technology and Materials Engineering has rubber processing equipment, which was utilised to make the final test piece.

5.2 - Seeds

The ball bearings were found to be too big, whilst the particles of aluminium oxide provided little contrast in the radioscopic images. The ideal seed would have particles the size of the aluminium oxide, but with the contrast provided by the ball bearings. A quantity of a material that is highly absorbent to low energy X-rays was required. Gold and lead are commonly used for radiation shielding, but investigations showed that tungsten is a more practical material in the X-ray energy levels used in these experiments.

The absorption coefficient of a number of elements against X-ray energy [131] is plotted in figure 5.9. Details for over a dozen elements were found, but only the five most interesting elements are plotted for clarity. Tungsten has the best absorption coefficient around the 70keV point. Iridium has the best coefficient at higher energies, whilst plutonium is better at lower energies. None of the other elements performed as well as these three. The gold and lead coefficients are included for comparative purposes. Iron would also have been included in the graph, but other than for the 50keV coefficient (14.1 cm^{-1}), no values were available from the reference.

The Goodfellows chemical catalogue was consulted to find the price of powders of the elements. The details of a number of the powders available are listed in table 5.1. Although their coefficients of absorbency were investigated, plutonium, mercury, and thallium are not listed below as there were no suitable samples listed in the catalogue. It can be seen that there is a considerable range in the prices of the powders, which made a number of the

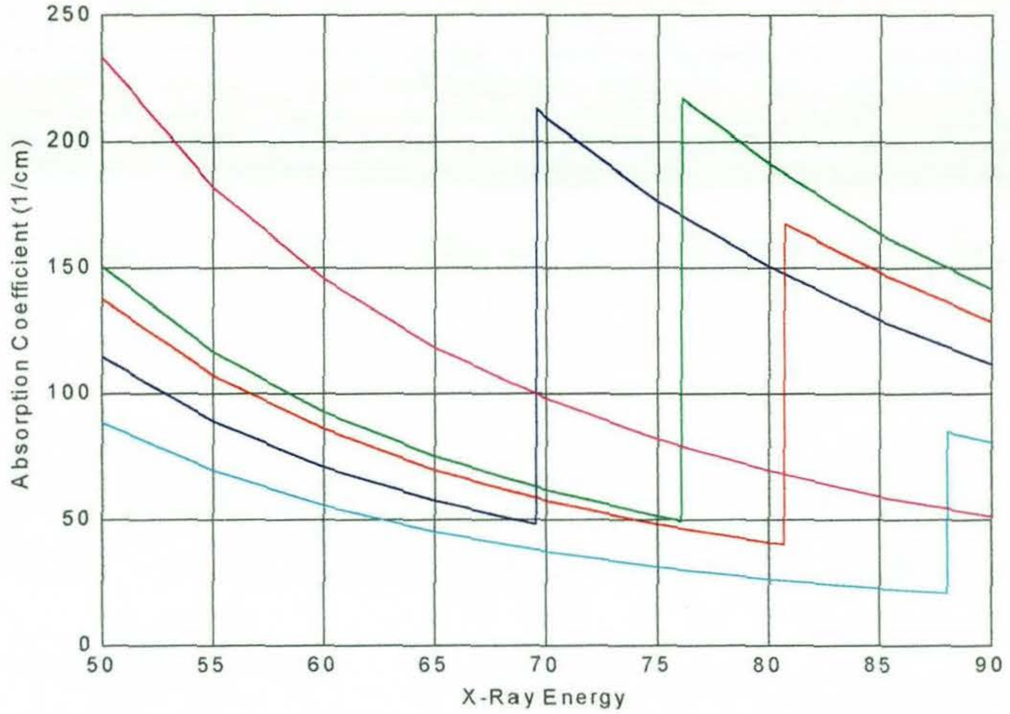
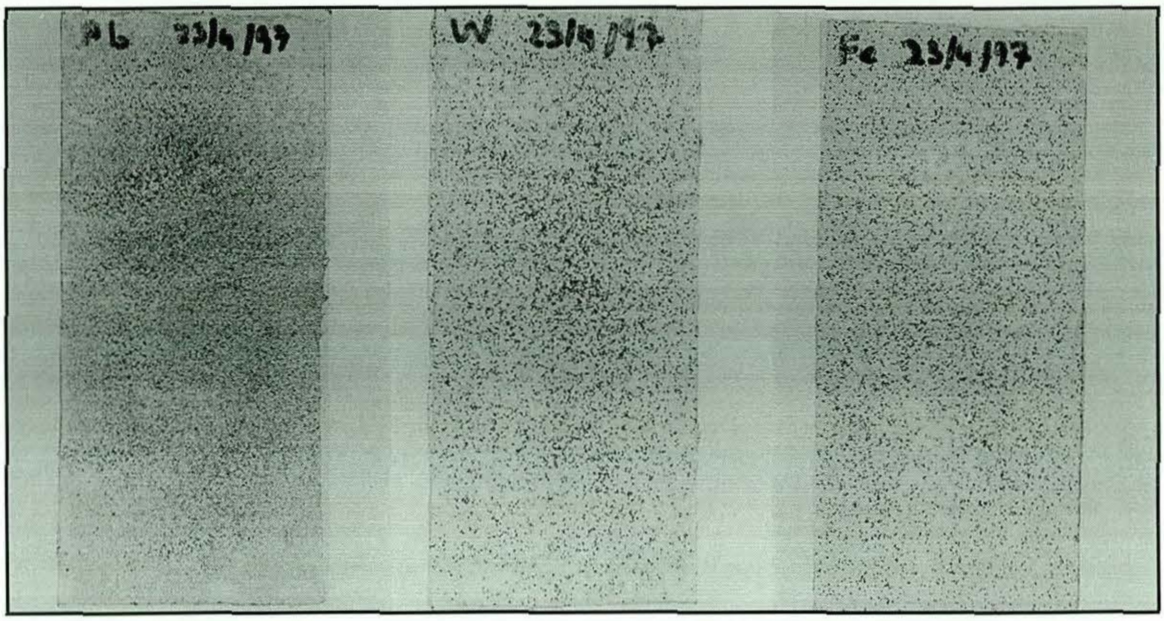


Figure 5.9 - X-Ray Absorption Coefficients for Potential Seed Materials. Element: Tungsten: ▬, Iridium: ▬, Gold: ▬, Lead: ▬, Plutonium: ▬.

Table 5.1 - Potential Seed Materials

Element (Atomic Number)	Catalogue Code	Minimum Size (μm)	Mean Size (μm)	Maximum Size (μm)	Purity (%)	Price
72	HF006010			40	95.0+	£175/10g
73	TA006030			75	99.9	£144/100g
73	TA006010	45		350	99.85	£261/100g
74	W006015		0.5		99.95	£109/100g
74	W006020			5	99.9	£116/100g
74	W006030			100	99.95	£80.50/100g
74	W006011			250	99.75+	£73.50/100g
75	RE006011			45	99.99	£548/100g
76	OS006010			150	99.9	£118/g
77	IR006010			850	99.995	£220/g
78	PT006020		2		99.95	£133/g
78	PT006010			400	99.95	£93/g
79	AU006020		2		99.95	£118/g
79	AU006015			53	99.95	£87.50/g
79	AU006010			250	99.95	£81/g
82	PB006025			150	99.5	£39/100g
26	FE006045			60	99.0+	£27.60/100g
26	FE006010			450	99.0+	£24.20/100g

elements impractical. However the prices of Tungsten (W) and Lead (Pb) were considered acceptable. Orders were placed for 100g of W06011 and for 100g of Pb006025. In addition, 1kg of fine iron filings were ordered from Fischer Scientific to be used to model the radioscopic images.



5.10 - Photo of Seed Sample Slides. Left to right, Lead, Tungsten and Iron.

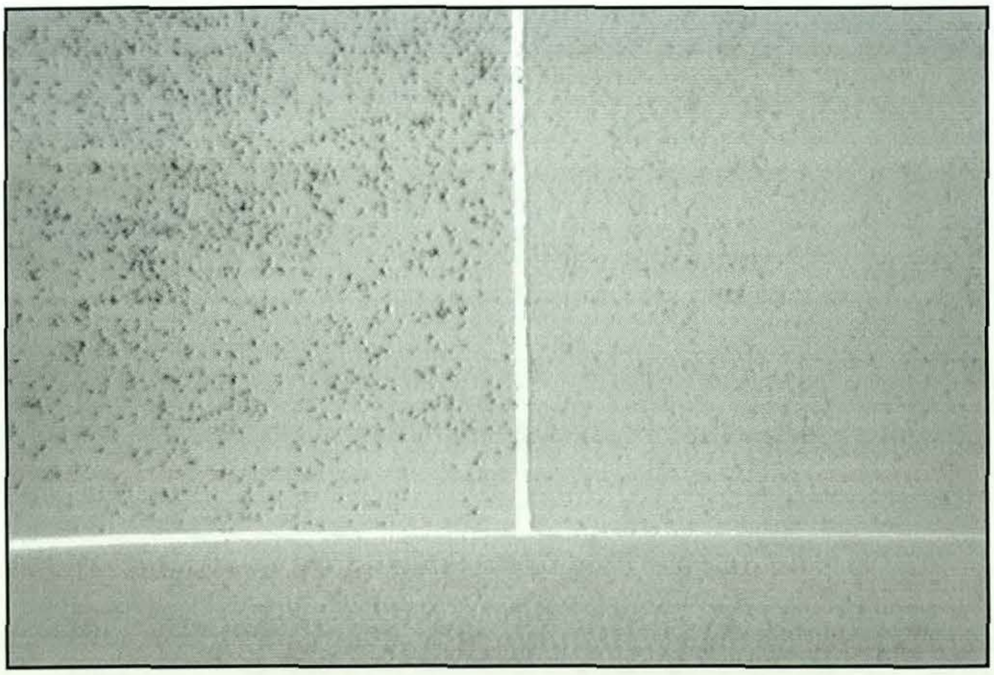


Figure 5.11 - Radioscopic Image of Samples of Potential Seed Materials. Lead (bottom), Tungsten (right) and Iron (left).

Glass slides were sprinkled with the three powder samples and sealed with sticky backed plastic (Figure 5.10). A radioscopic image of the three slides was taken (figure 5.11). The tungsten in the top left slide is easily visible. The iron filings (top right) can be identified, but without any clarity. The bottom slide has the lead powder, which is not easily identifiable. The difficulty in seeing the lead sample is explained by the smaller particle size (the largest tungsten particles are at least four times as big as the largest lead particle), and by the lower absorption coefficient.

Radioscopic images of the slides were taken with a range of apertures at X-ray settings 7 and 9 to find the ideal setting for best image contrast. Figure 5.12 shows the grey level range within windows of each slide. As the range is plotted against the minimum pixel value in the window, the plots suffer from a couple of spurious glitches, particularly with the darker images. But, it can be seen that the best contrast is obtained with the higher X-ray setting, and with the aperture open as wide as possible without clipping the image. Of the samples tested, the tungsten particles are the most promising seed material.

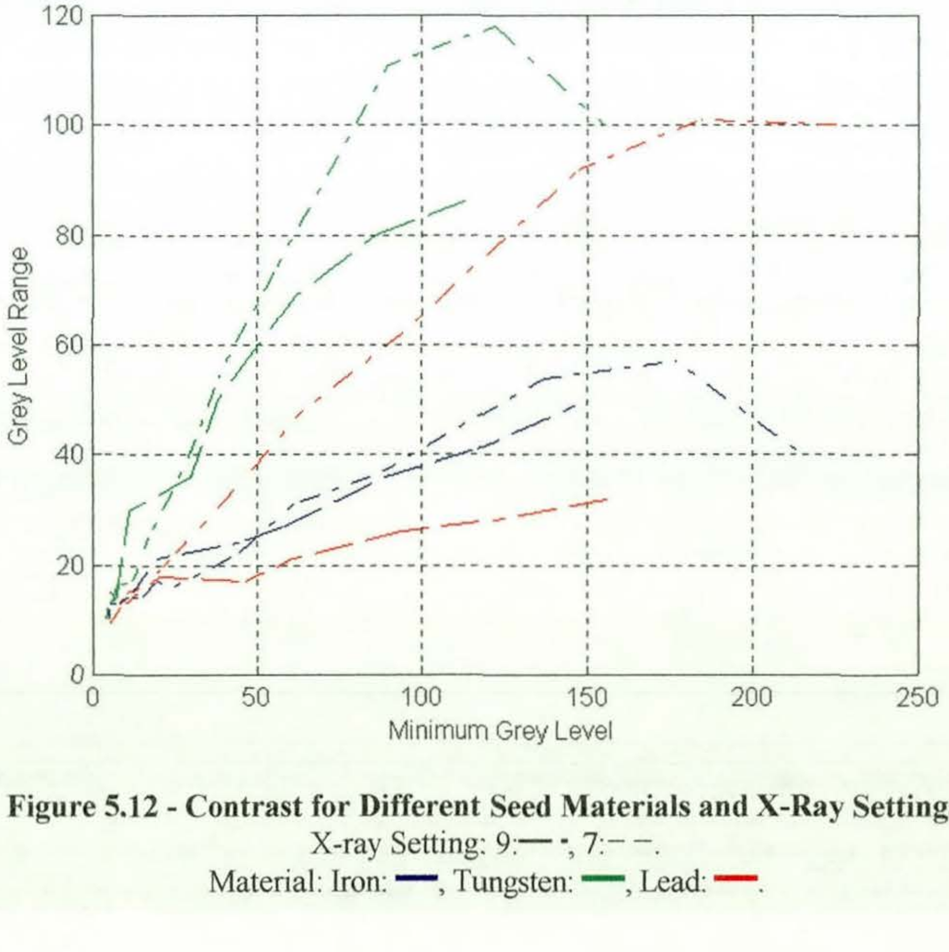


Figure 5.12 - Contrast for Different Seed Materials and X-Ray Settings.
X-ray Setting: 9:— ·, 7:– –.
Material: Iron: 🟦, Tungsten: 🟩, Lead: 🟥.

5.3 - Rubber-Tungsten Sample

Rubber was considered to be the most suitable test material. It is cheap, easily available, and able to withstand large strains. Equipment at the University's Institute of Polymer Technology and Materials Engineering was used to process the rubber sample. A 1mm thick sheet of rubber of 125mm by 125mm, seeded with 1.4g of tungsten particles to 50g of NRL was manufactured at the institute. The finished sample has a mass of 29.8g, indicating a seed density of 52g per m^2 surface area. Previous research [132] shows a seed volume of less than 3% has little effect on the properties of the host material, although other research has given different acceptable seed densities [54]. The rubber sample has a mass ratio of 2.8%, and a lower volume ratio.

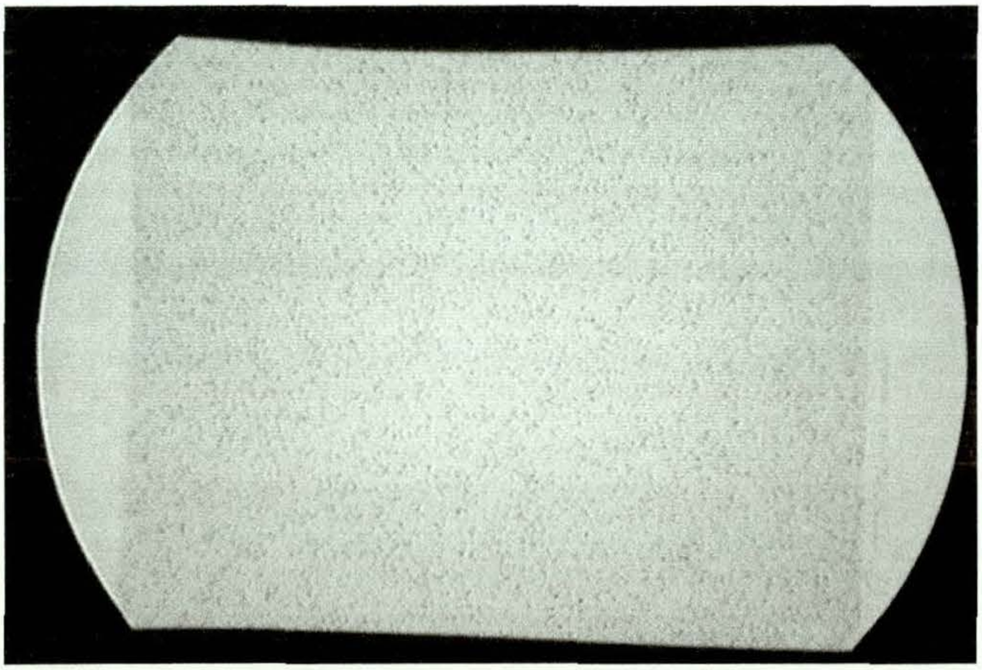


Figure 5.13 - X-Ray of Sheet of Rubber with Tungsten Seeds.

A radioscopic image of the rubber sample (figure 5.13) shows that the tungsten seeds provide identifiable features in the sheet of rubber. In this figure, the sheet of rubber is held in the strain rig. The typical grey level for this image outside the image intensifier's field of view is 21. The typical grey level where the rig is in the image is 25. The grey level for the areas with nothing in-between is 195. For the rubber sample, the grey level is about 185 for the rubber areas, with the tungsten taking the grey level down to 110. This indicates a reasonable level of contrast within the rubber sample.

5.4 - Strain Rig

A rig was designed and constructed to apply strain to the rubber sample for this project (figures 5.14 and 5.15). The rig is able to apply both longitudinal strains and also lateral skews to the sample. The rubber sample is clamped at the top and bottom (as seen in the radioscopic images) by two sets of clamps. These clamps are mounted within a frame that allows the top jaw to move sideways, and the bottom jaw to move up and down, thus applying strains and skews to the test piece. The clamps are moved, and held, by wing nuts on a 1mm pitch thread.

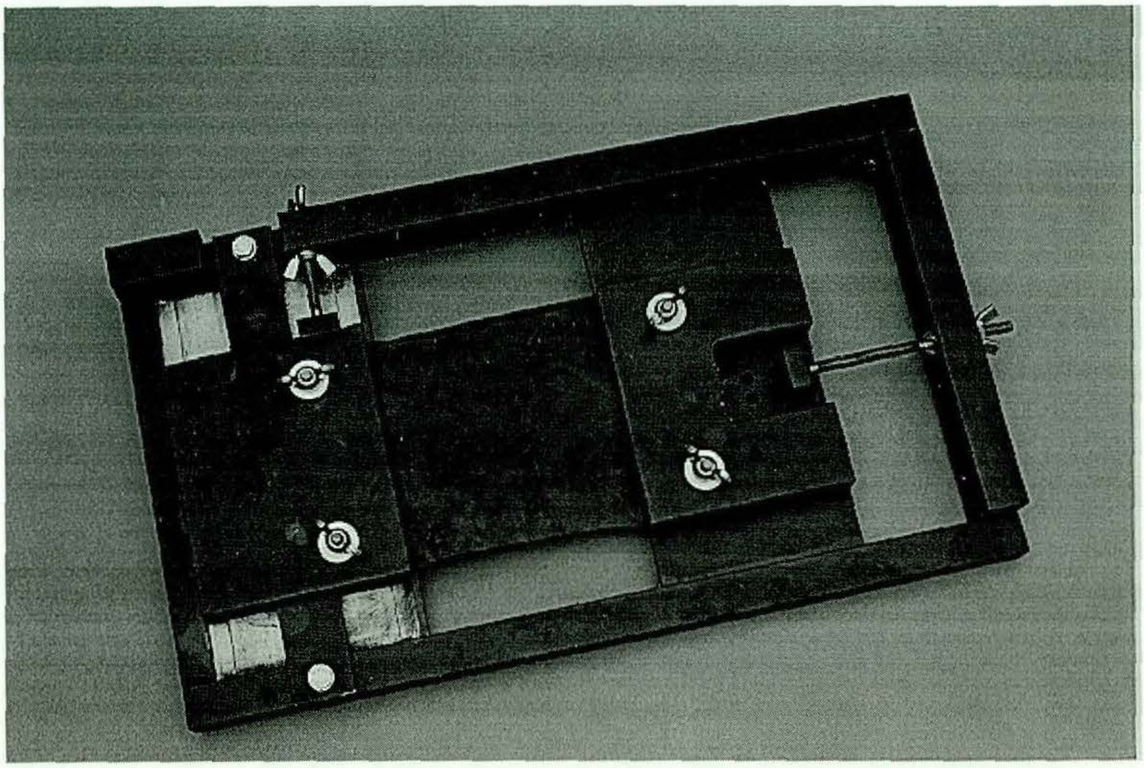
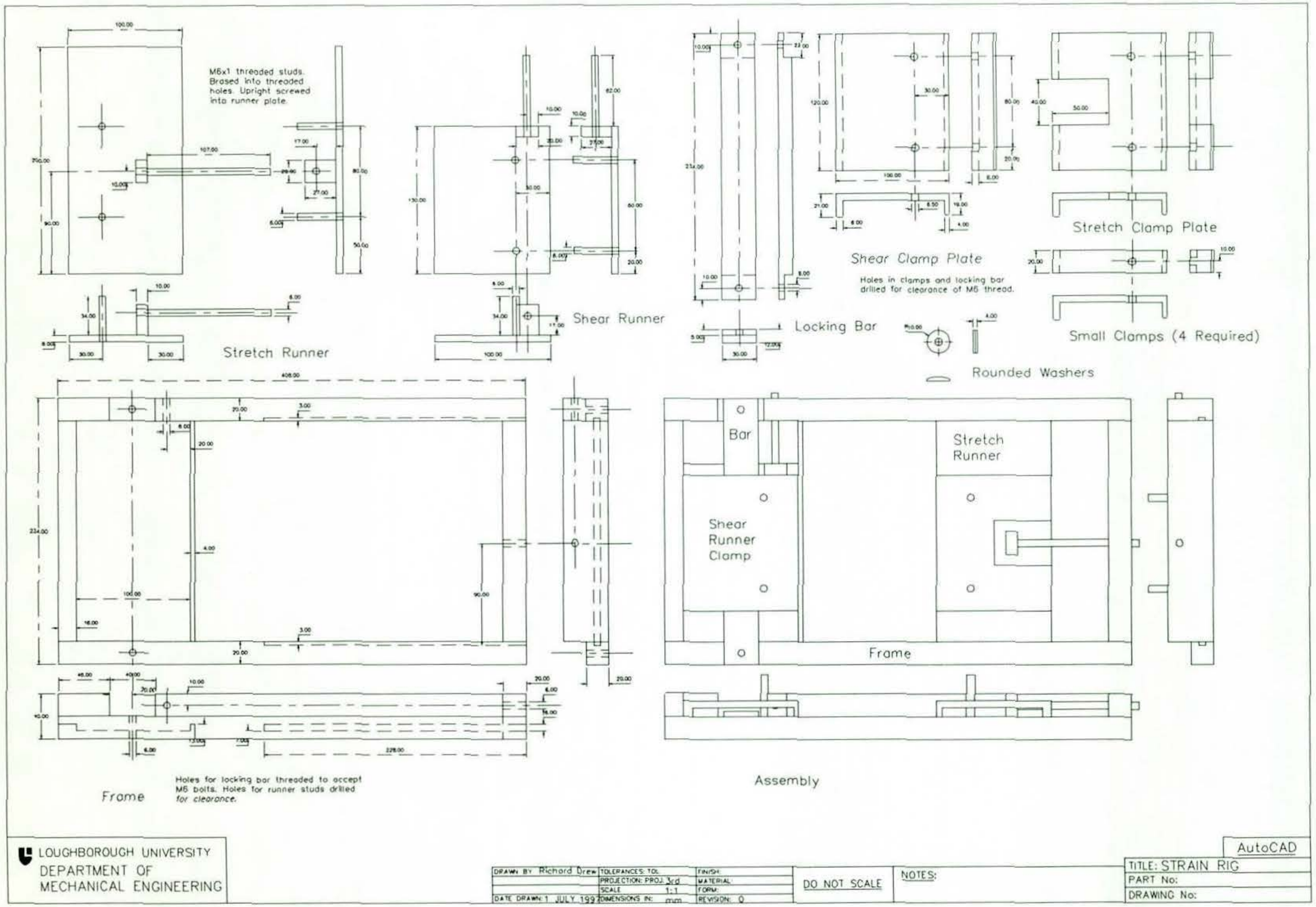


Figure 5.14 - Strain Rig.

The clamps are guided by grooves in the frame. This was found to be a source of error. There was no play in the lateral skew guide, but the longitudinal strain clamp was found to move laterally by up to 1mm. With a little lateral skew, the clamp is held to one side, so the problem is only evident when no skew is applied. However, by forcing the lateral clamp the appropriate way when no skew is applied, the error can be reduced.



M6x1 threaded studs. Brased into threaded holes. Upright screwed into runner plate.

Holes in clamps and locking bar drilled for clearance of M6 thread.

Holes for locking bar threaded to accept M6 bolts. Holes for runner studs drilled for clearance.

DRAWN BY Richard Drew
DATE DRAWN: 1 JULY 1997
TOLERANCES: TOL
PROJECTION: PROJ. 3rd
SCALE 1:1
DIMENSIONS IN: mm
FINISH:
MATERIAL:
FORM:
REVISION: 0
NOTES:
TITLE: STRAIN RIG
PART No:
DRAWING No:

Figure 5.15 - Strain Rig Drawing.

5.5 - Ideal Seed Density

The concentration of tungsten in the rubber samples will affect the accuracy of the pattern matching methods. In addition, the average size of the particles is expected to have an effect, but this variable will depend on the zoom settings. Experiments were made to try and find the seed density that maximises the probability of the pattern matching method identifying the correct location.

Due to the expense of the tungsten seed material, the tests used back lit iron filings as an approximation to the rubber-tungsten radioscopic image (figure 5.16). A sheet of glass, lit from behind with a diffused white light, had quantities of iron filings sprinkled over it. A camera was located so the iron filings were of similar size to the radioscopic images of the tungsten seed. Twenty-one images were taken with varying concentrations of iron filings.

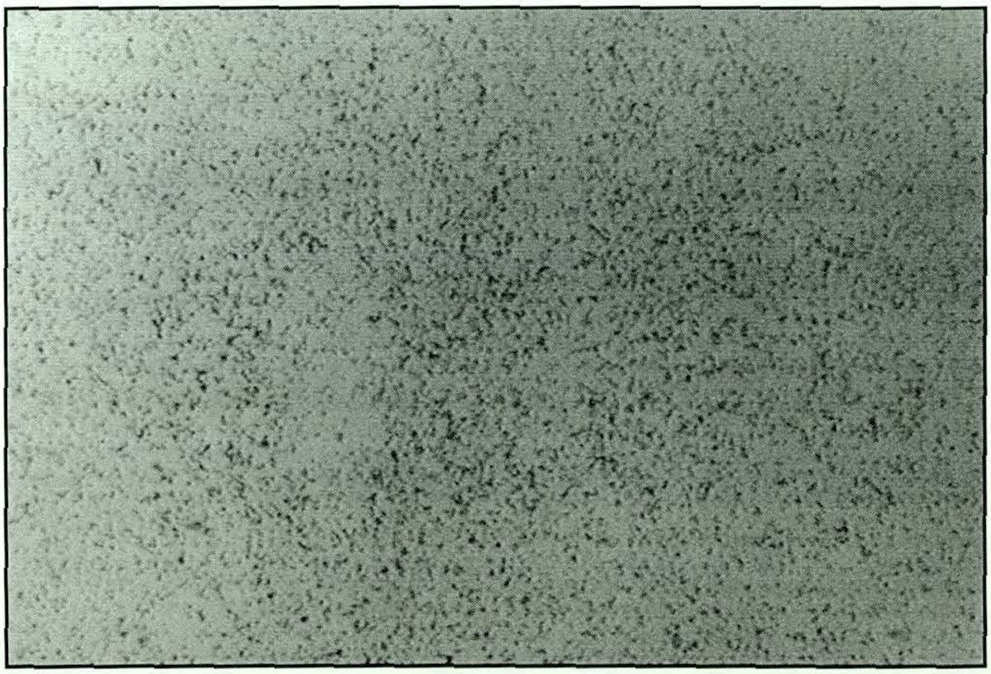


Figure 5.16 - Simulation of the Radioscopic Images Using Back Lit Iron Filings. Image has 16% of area covered by seeds (rubber-tungsten radioscopic images have 15% coverage).

In order to equate the results of the iron filing tests to the radioscopic images, they were classified by the percentage of the area of the image covered by seeds. Whilst the seeds are visible in the images, it is not easy to estimate the area covered. A simple threshold method (figure 5.17) is more likely to identify intensity variations across the image rather than the seeds. To prevent this, the Matlab routine "improc.m" (appendix III.11) was written to process the image. This normalises the image so it has a mean of zero, and a standard deviation of one. The image is filtered with a 7 x 7 high pass filter, which removes low frequency variations across the image.

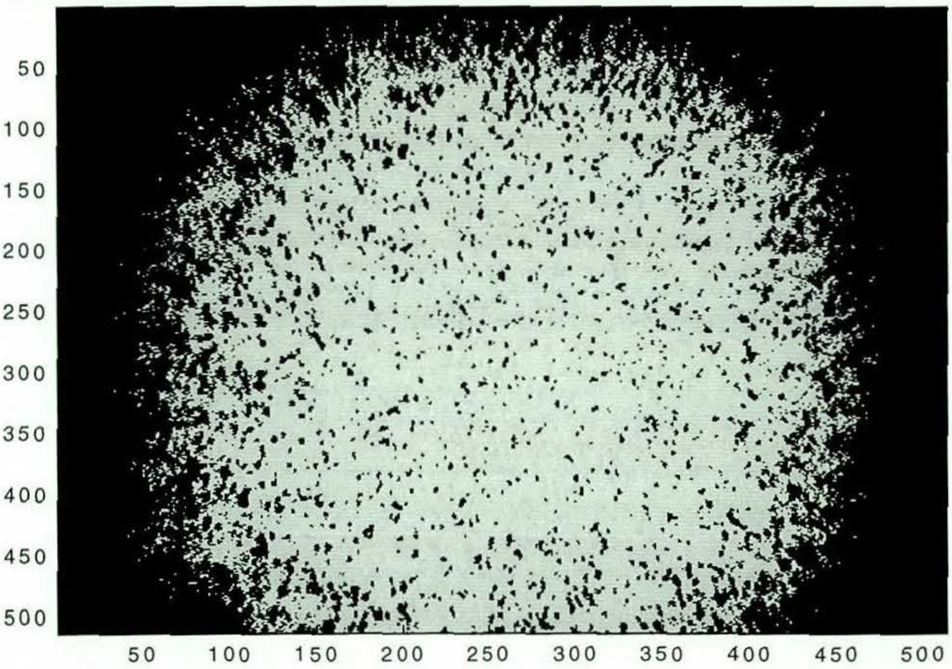


Figure 5.17 - Radioscopic Image of Rubber Tungsten Sheet, Thresholded at Mean.

After filtering, the image is thresholded to find the area covered by seeds. However, the use of the high pass filter means that the centres of areas that are totally covered by seeds will be lost. To prevent this, the image of the seeds is combined with a version of the original image, thresholded to give only the most dense concentrations of seeds. The seed area is measured as the percentage of the image covered by the seeds.

The result is dependent on the threshold value. Consideration was given to using a variable threshold dependent on the standard deviation of the filtered image. But this would result in seeds being missed in images with high percentage seed areas. A fixed threshold was found to be reasonable, provided all the images were normalised before filtering. However, tests with the radioscopic images of the rubber-tungsten sample found problems with noise sensitivity. Only radioscopic images taken at the 215mm image intensifier zoom setting produced reasonable results, indicating about 15% of the image were covered by seeds.

The other two image intensifier zoom settings produced images that needed a higher threshold value to remove noise. However this makes the area of the seeds smaller, making the results difficult to compare with the iron filing images.

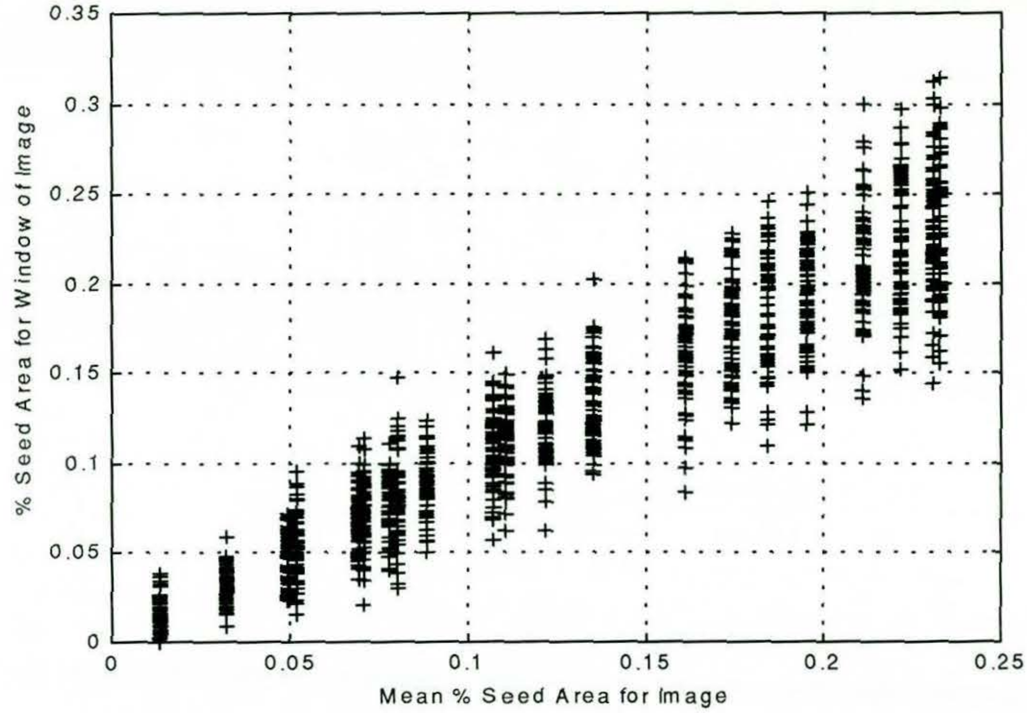


Figure 5.18 - Seed Density Variation Across Images.

A variation in the density of filings across the images was noted (figure 5.18). This is a result of the difficulty in sprinkling the iron filings evenly across the glass plate. This problem could be solved by using a suitably buoyant opaque seed held in suspension in a liquid. A glass bottomed tank would be used to allow the suspended particles to be back lit. The seed density could easily be controlled as it would be proportional to the depth of the seed/liquid suspension in the tank.

A second version of each image was created by stretching the image by 5% using Paint Shop Pro 5. Sixty-four windows (of 32 x 32 pixels) from each image were pattern matched with windows of 256 x 256 pixels taken from the stretched version. Tests were undertaken using the normalised cross correlation pattern matching equation (see chapter 2).

A measure of the performance of each pattern matching equation was taken for each window. The measure is one minus the ratio between the potential for improvement in the pattern matching method at the point of match, and the next smallest potential for improvement. This can be expressed as the difference between the lowest pattern matching equation minimum and the second lowest pattern matching equation minimum divided by the second lowest minimum. This shows how prominent a peak the correct match makes.

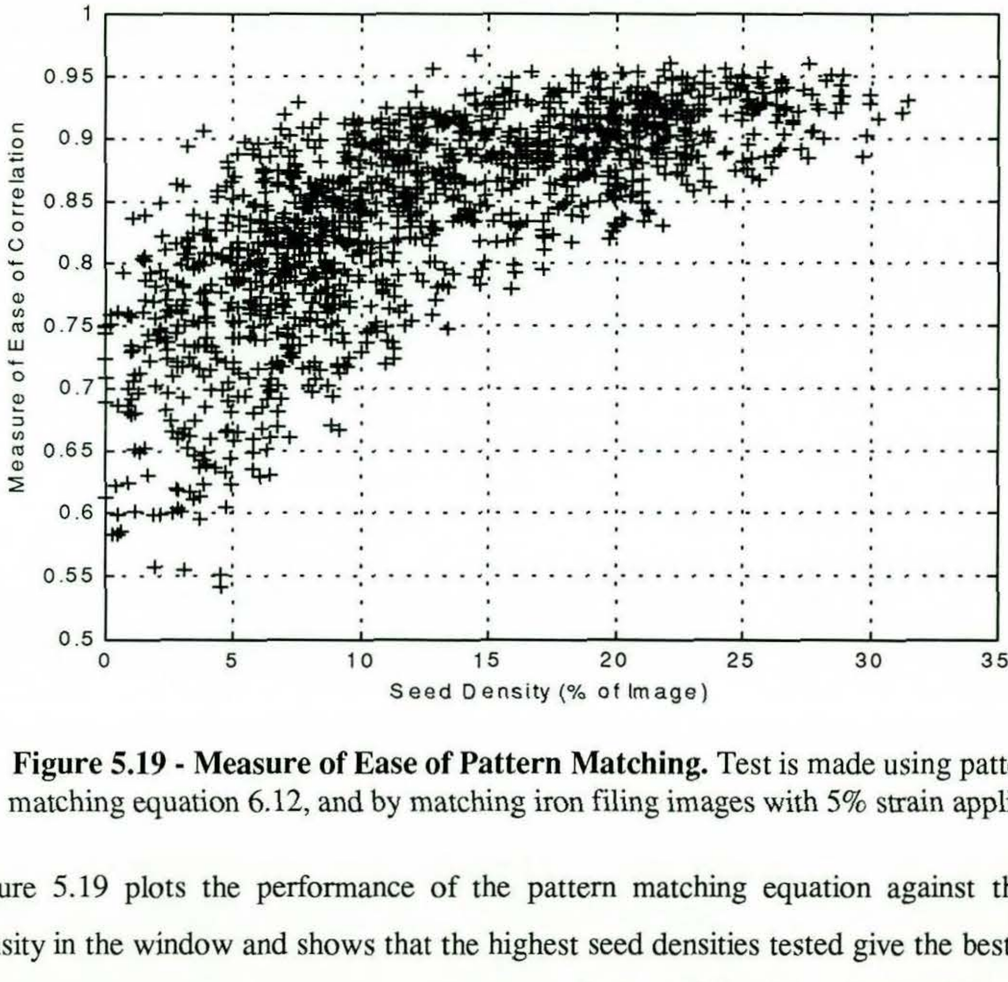


Figure 5.19 - Measure of Ease of Pattern Matching. Test is made using pattern matching equation 6.12, and by matching iron filing images with 5% strain applied.

Figure 5.19 plots the performance of the pattern matching equation against the seed density in the window and shows that the highest seed densities tested give the best results from the pattern matching equations. The images have seed densities of up to 32%. Higher densities up to 50% are expected to improve the performance of the pattern matching methods, but further increases will reduce the amount of texture in the images, and are expected to reduce the performance. However, there are often other considerations when choosing the seed density for a given application. In particular, the affect of the seeds on the host material.

5.6 - Summary

This chapter investigates the hardware required to test radioscopic strain measurement. A number of materials were tested whilst looking for a suitable material for testing. As it is desirable to test the robustness of the deformation measurement technique, a material was desired that could withstand large strains. In addition, the material needs to provide contrast in the radioscopic images to allow the detection of deformations. It was decided it was best to introduce the contrast as small particles of a highly X-ray absorbent seed material.

The final choice of material was a rubber host seeded with tungsten particles. The rubber allows large strains to be applied. The tungsten particles provide a reasonable level on contrast to the images. The chapter gives details of a strain rig built for use with the rubber-tungsten sample. An investigation was made to find the best seed density. This was undertaken with simulated images. There were some problems with the distribution of the seed material in these images, but the experiment suggested half the image area should be covered with seeds to ensure the highest likelihood of an accurate pattern match.

6 - Deformation Measurement Methods

A search for pattern matching methods resulted in a discussion of several equations in chapter 2. This chapter looks more closely at these equations, and identifies those suitable for the measurement of deformations applied by the rig described in chapter 5. In addition to finding a number of different pattern matching equations, several of the methods of pre-processing the images were found that are used either to standardise the images, or to enhance features that would improve the probability of a correct match. A number of comparative tests are included in this chapter to compare the ability of the methods to identify the correct deformation. These tests applied the chosen pattern matching equations to the images after pre-processing with a number of techniques

The tests were made with computer generated deformations of the image. This means the deformation between any two images is known, and makes it easy to identify correctly matched windows. Simulated deformations were used as the real deformations on the rubber-tungsten test sample were noted to be non-uniform across the sample, making it difficult to confirm results. In each test, the best match to a small window from the deformed image was found in an un-deformed image. The tests were made with a wider range of potential matches than would be necessary when measuring the distortion for most strain measurement applications. This makes the test more onerous, but gives an idea of the robustness of the different methods.

The suitability of each method was found by looking at the speed of implementing the method, and by comparing the accuracy of the matches with a number of the simulated images. This chapter looks at the results from the different methods to see if they can be used to provide a more accurate match, or give a better level of confidence in a particular situation.

6.1 - Review of Pattern Matching Methods

The pattern matching literature review in section 2.3 found no less than seventeen different equations, used or proposed, for matching images. However quite a few of these equations are very similar, or provide identical results. Equations 2.6, 2.7, 2.10, 2.11, 2.13 and 2.18 all use a difference method. Of these, 2.6, 2.7 and 2.11 use the absolute difference equation, with 2.6 and 2.7 measuring the sum, and 2.11 taking the maximum difference. If the means are subtracted globally from the image before processing (a pre-processing technique), equation 2.7 can be expressed as 2.6. Using the maximum difference, equation 2.11 is likely to be unreliable, especially with very sharp images that have been slightly distorted. Equations 2.10, 2.13 and 2.18 are all expressions of the sum of squares method, and will produce the same location for the best match.

Equations 2.5, 2.8, 2.9, 2.12, 2.14, 2.15, 2.16, 2.17, 2.19, 2.20 and 2.21 are cross correlation (multiplicative) methods. All these equations are normalised except for equation 2.8, which is discounted as it will produce unreliable results for images with large mean variations across the image. Equation 2.9 is the most common version of the equation, and will provide the same location for best match as equations 2.12, and 2.15. Pre-processing the images to remove the mean makes equation 2.14 the same. Equation 2.5 is very similar to equation 2.9, but requires a local mean value to be calculated for each possible window, which increases the computational requirement.

Equation 2.19 is the fully normalised method, whilst equation 2.17, (which can be expressed as equation 2.16), is the new normalised method. Equations 2.20 and 2.21 are assumed to be misprints, as they are not correctly normalised, and are not implemented in this chapter.

This means that of the seventeen equations, five have been identified for further investigation. All five equations could be pre-processed to remove the mean, enhance edges or normalise the images in other ways. The methods are listed in equations 6.1 to 6.5 in a form where a minimum value gives the best match. This is to make it easier to compare the results from the different methods. In these equations, the general expression **A(a, b)** is the grey level value of the pixel at position **(a, b)** in image **A** (size **M**, **N**), and **B(a, b)** is the grey level value of the pixel at **(a, b)** in image **B** (also of size **M**, **N**). The variables (**k**, **l**) represent the size of the pattern matching windows. **C_{m,n}(x, y)** is the measure of match between the window in **A** with an origin at (**m**, **n**) and the window in **B** with an origin at (**x**, **y**). The equations are expressed so that the best match between the window in **A** at (**m**, **n**) and **B**, is the location in **B** with the smallest value of **C_{m,n}(x, y)**, an exact match will give a result of zero. The equations tested are:

Sum of differences (was equation 2.8):

$$C_{m,n}(x,y)=\sum_{i,j=0}^{k-1,l-1}\left|A(m+i,n+j)-B(x+i,y+j)\right|$$

(6.1

Sum of Square of Differences (was equation 2.13):

$$C_{m,n}(x,y)=\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)-B(x+i,y+j)\right)^2$$

(6.2

Normalised Cross Correlation (was equation 2.15):

$$C_{m,n}(x,y)=1-\frac{\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)\right)\left(B(x+i,y+j)\right)}{\sqrt{\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)\right)^2\sum_{i,j=0}^{k-1,l-1}\left(B(x+i,y+j)\right)^2}}$$

(6.3

New Normalised Cross Correlation (see equation 2.17):

$$C_{m,n}(x,y)=1-\frac{2\times\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)\right)\left(B(x+i,y+j)\right)}{\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)\right)^2+\sum_{i,j=0}^{k-1,l-1}\left(B(x+i,y+j)\right)^2}$$

(6.4

Fully Normalised Cross Correlation (see equation 2.19):

$$C_{m,n}(x,y)=1-\frac{N.M\sum\limits_{i,j=0}^{k-1,l-1}A(m+i,n+j)B(x+i,y+j)-\sum\limits_{i,j=0}^{k-1,l-1}A(m+i,n+j)\sum\limits_{i,j=0}^{k-1,l-1}B(x+i,y+j)}{\sqrt{\left[N.M\sum\limits_{i,j=0}^{k-1,l-1}A(m+i,n+j)^2-\left\{\sum\limits_{i,j=0}^{k-1,l-1}A(m+i,n+j)\right\}^2\right]}}\cdot\frac{1}{\sqrt{\left[N.M\sum\limits_{i,j=0}^{k-1,l-1}B(x+i,y+j)^2-\left\{\sum\limits_{i,j=0}^{k-1,l-1}B(x+i,y+j)\right\}^2\right]}}$$

(6.5

All the valid equations in the literature search are covered by these five equations. Equations 6.2 to 6.4 are all suitable for implementation with Fast Fourier Transforms (FFTs), as they are functions of equations 6.6 to 6.8:

$$`AB_{m,n}(x,y)=\sum_{i,j=0}^{k-1,l-1}\big(A(m+i,n+j)\big)\big(B(x+i,y+j)\big)$$

(6.6

$$`AA_{m,n}=\sum_{i,j=0}^{k-1,l-1}\left(A(m+i,n+j)^2\right)$$

(6.7

$$`BB_{m,n}(x,y)=\sum_{i,j=0}^{k-1,l-1}\left(B(x+i,y+j)^2\right)$$

(6.8

Equation 6.5 can also be implemented with FFTs, but would require the additional calculation of equations 6.9 and 6.10:

$$`A_{m,n} = \frac{\sum_{i,j=0}^{k-1,l-1} A(m+i, n+j)}{\sqrt{N \times M}}$$

(6.9

$$`B_{m,n}(x,y) = \frac{\sum_{i,j=0}^{k-1,l-1} B(x+i, y+j)}{\sqrt{N \times M}}$$

(6.10

Equation 2.5 is unsuitable for implementation with FFTs as the mean has to be calculated for every sub image. In order to compare the accuracy of the matching equations, equations 6.6 to 6.10 can be calculated for all values of (**m**, **n**), and used to find the best matches using the various methods. The absolute difference method needs to be calculated separately.

Using equations 6.6 to 6.10, the matching equations can be rewritten. The sum of squares of differences method (equation 6.2) will be expressed as:

$$C_{m,n}(x,y) = `AA_{m,,n} + `BB_{m,n}(x,y) - 2 \times `AB_{m,n}(x,y)$$

(6.11

Normalised Cross Correlation becomes:

$$C_{m,n}(x,y) = 1 - \frac{`AB_{m,n}(x,y)}{\sqrt{(`AA_{m,,n})(`BB_{m,n}(x,y))}}$$

(6.12

New Normalised Cross Correlation is:

$$C_{m,n}(x,y) = 1 - \frac{2 \times `AB_{m,n}(x,y)}{(`AA_{m,,n}) + (`BB_{m,n}(x,y))}$$

(6.13

Fully Normalised Cross Correlation is rewritten as:

$$C_{m,n}(x,y)=1-\frac{{}^{\backprime}AB_{m,n}(x,y)-{}^{\backprime}A_{m,n}\times {}^{\backprime}B_{m,n}(x,y)}{\sqrt{\left({}^{\backprime}AA_{m,n}-{}^{\backprime}A_{m,n}{}^{2}\right)\left({}^{\backprime}BB_{m,n}(x,y)-{}^{\backprime}B_{m,n}(x,y)^{2}\right)}}$$

(6.14

Whilst the equations listed above use the correlation between the two windows to measure the accuracy of the match, chapter 2 included other ways of measuring the match. The SSDA (sequentially similarity detection algorithm) method only calculates part of the equations to determine the suitability of the match. Equation 2.11 was noted to use the maximum difference between the two images. Another reference in chapter 2 [78] mentions counting the number of sign changes in the pointwise intensity to measure the match.

6.2 - Simulated Test Images

Tests were made to find the suitability of the various pattern matching equations to measure the deformation of the rubber-tungsten sample. The equations were tested with simulated images. The test images were created from a radioscopic image of the rubber-tungsten sample. The image was processed to remove noise. First the frequency dependent noise was reduced by attenuating sixteen frequencies in the frequency plane that are known to be noisy. Intensity variations were corrected by dividing the grey level values by the grey level values in a radioscopic image taken with nothing in the radioscopic chamber (which had also had the frequency dependent noise reduced). The processed image (figure 6.1) was stretched to take up the full grey scale.

Paint Shop Pro version 5 was used to distort the processed image. Fifteen images were stretched using the bilinear re-sizing option. The image was stretched vertically in equal steps from 16 to 240 pixels, which produces images of up to 752 x 512 pixels. This equates to deformations of 0% to 46.875% in equal steps of 3.125%. A further ten images were taken with the image skewed by up to 20°, in steps of 2°. The Paintshop Pro skew deformation feature keeps the image the same size, and the centre in the same position, but results in the loss of the top right and the bottom left corners.

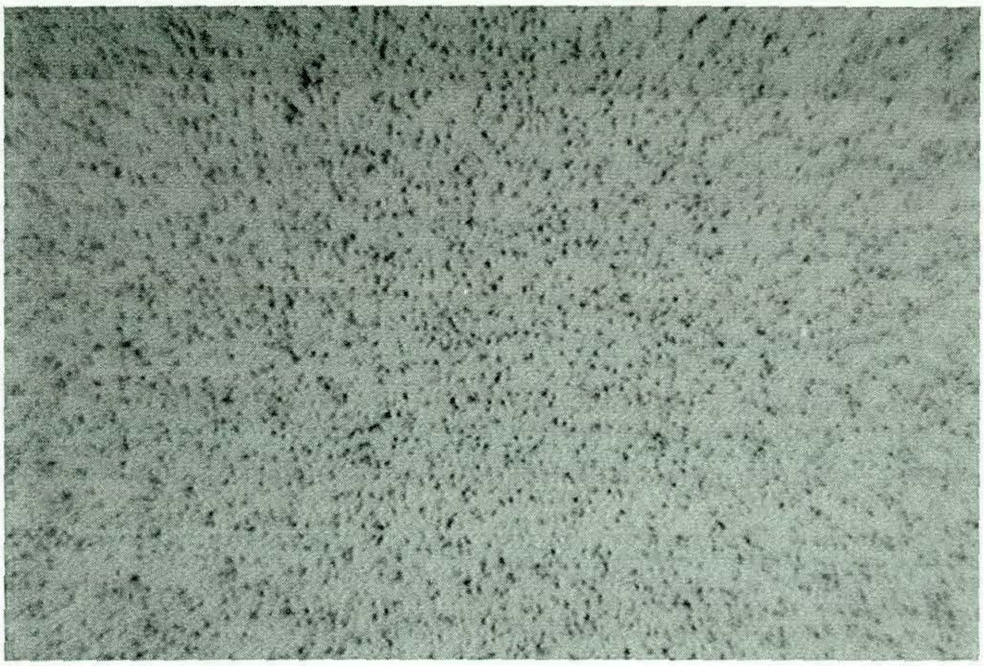


Figure 6.1 - Radioscopic Image of Rubber-Tungsten Sample with No Strain. Image Processed to reduce shading and frequency dependent Noise.

6.3 - Software to Test Deformation Measurement

Matlab was used to compare the deformation measurement methods. In each test, a hundred small windows (size varied from test to test) from one of the modified images were compared with a 64 x 64 pixel window from the original processed image, centred on the equivalent area of the image. Each comparison produces a measure of the match for every possible matching location of the smaller window in the larger 64 x 64 window. For each test, the result is taken as the location of the best match.

The Matlab routine "corl1.m" (appendix III.12) includes the code used to find the frequency dependent noise mask and to create the original processed image. The routine is designed to add normally distributed random noise to the image, and to pre-process the images if needed. Each of the 26 distorted images are compared in turn with the original processed image. The comparisons are undertaken in the routine "correl1.m". The results saved indicate the location of the best match between each pair of windows, as measured by each pattern matching method. The windows were chosen so that a correctly identified match gave the location of match as (0,0).

The routine "correll.m" (appendix III.13) is used to identify the windows that are to be matched in the tests. Tests are made on a grid of 10 x 10 windows, with a spacing in the original processed image of 33 pixels. Early tests were with a spacing of 32 pixels, but were found to be affected by the phase of the images. The centres of windows in each test are passed in turn to the routine "sbcor1.m". The centres are chosen so they will coincide if the best match is correct. The calculation of the best match is undertaken in routine "sbcor1.m" (appendix III.14). This calculates the pattern matching equations 6.11 to 6.14 using the function "corfunc1.m" (appendix III.15) to calculate the equations 6.6 to 6.10. The locations of best matches using each equation are found, and returned as offsets from the centre of the large window.

If the best match is within one pixel of the calculated location, the test is considered successful. This would be sufficiently accurate to allow a second stage interpolated test to find the correct result using a finer scale. Tests were undertaken to compare the accuracy of the equations, the effect of pre-processing the images, the best window dimension in the modified image, and the noise tolerance of the methods. Minor changes to "corl1.m" are required to test these variables.

6.4 - Comparison of Deformation Measurement Methods using Simulated Images

The first test (see section 6.4.1) was to compare the speed of the different methods of matching the images. Simulations of the processing required to calculate the correlation and difference methods were compared with the processing required for the SSDA, the maximum difference and the sign change techniques. The correlation and difference methods were found to be several times faster than the other methods, so only these methods were taken further.

Section 6.4.2 investigates the probability of finding a correct match with the different pattern matching equations, combined with a range of pre-processing techniques. With the simulated images, the pre-processing techniques were not found to reduce the probability of a correct match, so further tests were carried out without pre-processing. In addition the sum of absolute difference method was discarded at this stage, as this equation performed no better than the other equations, and testing it increased the processing time of the test by several times.

Section 6.4.3 investigates the optimum pattern matching window size. Larger windows were found to be more likely to match, but smaller window sizes allow a finer resolution to the deformation measurement. Smaller windows are also quicker to process as the size of the window in the original image can be reduced. The last section, 6.4.4, investigates the noise tolerance of the different methods.

6.4.1 - Speed Tests

The processing overheads of the pattern matching methods vary widely. Before testing the accuracy of the equations, the time required to process each equation was measured. The Matlab function "tbcor.m" (appendix III.16) was written to measure the time and operations (measured in Matlab "flops") required by each method for different sizes of the larger window. The tests were carried out with random images, using a window size of 6 x 32 pixels matched to an image of between 32 x 32 pixels and 256 x 256 pixels. The timings include the time to process the measure of match for each potential window location, but not the time required to identify the best match. Seven methods were tested:

SSDA
sum of absolute differences
sign changes
fully normalised cross correlation
square of differences
normalised cross correlation
new normalised cross correlation

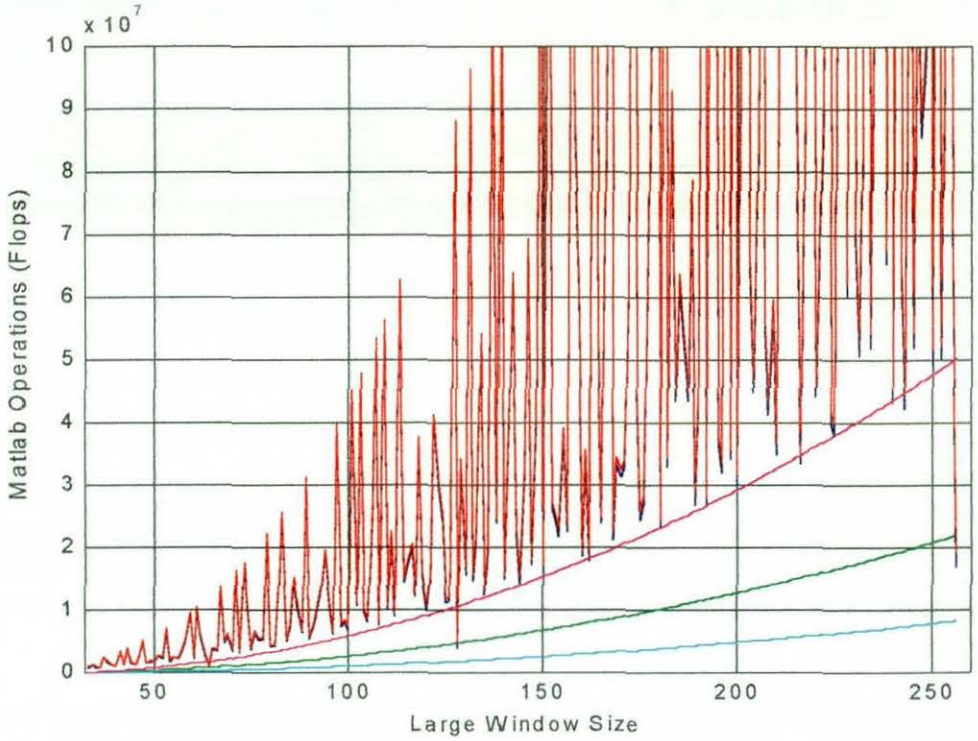


Figure 6.2 - Operations Required to Process Each Method. Method: Correlation/Square of Difference: ▬, Sum of Difference: ▬, Fully Normalised Correlation: ▬, SSDA: ▬, Sign Change: ▬.

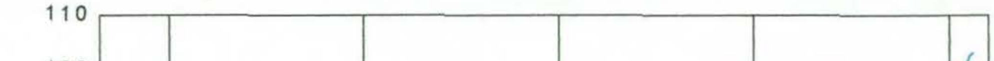


Figure 6.3 - Time to Process Each Method. Method: Correlation/Square of Difference: ▬, Sum of Difference: ▬, Fully Normalised Correlation: ▬, SSDA: ▬, Sign Change: ▬.

The number of operations required are plotted in figure 6.2, whilst the time required to process each method is in figure 6.3. The SSDA threshold was set so that each result was calculated after six calculations.

The square of difference equation, normalised cross correlation equation and the new normalised cross correlation equation can all be obtained from the same set of calculations, and can also be determined if the fully normalised cross correlation equation is implemented. Calculation of the SSDA, sum of absolute difference or sign change methods must be performed separately.

The function "tbcor.m" measures the time taken to calculate the results for each possible match of the two windows for each of the deformation measuring equations. The equations 6.6 to 6.10 are calculated separately, and the times are added together to provide the results for the correlation equations. It will be noted that there are two methods used to calculate equation 6.8 in "tbcor.m". Earlier tests were undertaken on a Pentium 166 computer, and the loop version of the algorithm was found to be faster for all window sizes. However, when run on a Pentium 333 computer, the FFT method was found to be faster with some window sizes. The choice between these two ways of calculating equation 6.8 is therefore dependent on the size of the windows, and on the speed of the computer.

Despite requiring more operations, the FFT methods were fastest. This is because the Matlab code and modern computer architectures are optimised for large matrices, rather than loops. Conversely, the SSDA method requires the least number of operations, but takes the longest to process. This method was originally designed to speed up pattern matching and, when introduced, was a couple of orders of magnitude faster than other methods. It would be more suitable for implementation with a lower level programming language as it requires extensive use of loops, where it may be faster than the other methods. The sign change method also proved to be slow and so this method, and the SSDA, were not used in further tests. The absolute difference method was the fastest of the methods based on loops, and was used in the later accuracy tests.

The results show that the number of operations required for the Matlab FFT methods is not proportional to the window size. Small changes in the window size can change the processing overheads dramatically. Further investigations showed there are preferred window sizes, and the input into the FFT algorithms should be padded out to be the next largest of these sizes. The preferred window sizes were found to be:

1, 2, 4, 8, 9, 16, 18, 32, 36, 64, 128, 135, 256, 270, 512

It is noted that not all of the preferred window sizes are 2 to the power of an integer. The list was also noted to change from one computer to the next (the above list is from a Pentium 333 machine). This suggests the internal code of the Matlab built in FFT function is not fully optimised.

6.4.2 - Pre-Processing Techniques

All the pre-processing tests were undertaken with a 32 x 32 window from the distorted image. A number of techniques were tested to find their effect on the ability of the pattern matching equations to find the correct match. Tests were made with images pre-processed to standardise the mean and standard deviation, and also to enhance features:

1. No pre-processing
2. Images divided by standard deviation
3. Means subtracted
4. Means subtracted and image divided by standard deviation
5. Edge detection filter.
6. Statistical mask operator.

It was noted in chapter 2 that the three different methods of normalising the cross correlation equation would each give a different set of images a result that indicates the two images matched. The fully normalised method would accept images with different mean levels, and different standard deviations. The normalised method would accept images with different standard deviations, whilst the new method would only accept images that were identical.

The pre-processing techniques tested include methods of normalising the standard deviation and the mean to see what effect this has on the accuracy of the pattern matching methods, bearing in mind some of the equations will be unaffected. The edge detection filter used was the filter used in section 5.5. The statistical mask operator is used as a pre-process filter for statistical correlation [80], and assumes there is no observation noise. This is a valid assumption for the first run of tests with the simulated images. The mask operator chosen assumes the "adjacent pixel correlation" is maximum.

Two runs were made, one with normally distributed noise with a standard distribution of 10 added to the images, and the other with no added noise. The noise level in the first of these tests was calculated to be similar to the level of noise to be found in the radioscopic images.

Table 6.1 - Accuracy of Deformation Measurement Methods with Standard Deviation of Noise of 10.

Method	Incorrectly Located Matches (%)									
	Stretched Images					Skewed Imaged				
	Sum of Difference	*Fully normalised*	*Square of Difference*	*New normalised*	*Normalised Correlation*	*Sum of Difference*	*Fully normalised*	*Square of Difference*	*New normalised*	*Normalised Correlation*
Original Image	13.5625	11.3125	11.6875	12.0625	12.6250	27.8182	30.8182	30.9091	31.1818	31.3636
STD=1	14.0625	11.3125	11.8125	12.3750	12.6250	73.0909	30.8182	61.8182	56.5455	31.3636
Mean=0	15.1875	11.3125	12.8125	11.5625	11.5625	31.6364	30.8182	34.0000	32.6364	32.0000
STD=1, Mean=0	15.1875	11.3125	12.7500	11.5625	11.5625	37.0000	30.8182	39.4545	32.3636	32.0000
Edge Filter	22.4375	21.6250	22.8750	21.6250	21.5625	40.6364	43.8182	46.5455	43.8182	43.8182
Statistical Filter	89.4375	88.6875	89.4375	88.6250	88.6875	81.2727	81.1818	82.2727	81.1818	81.1818

Table 6.2 - Accuracy of Deformation Measurement Methods with no Noise.

Method	*Incorrectly Located Matches (%)*									
	Stretched Images					Skewed Imaged				
	Sum of Difference	Fully normalised	Square of Difference	New normalised	Normalised Correlation	Sum of Difference	Fully normalised	Square of Difference	New normalised	Normalised Correlation
Original Image	12.5000	10.5625	11.8125	12.0625	12.0625	28.9091	29.5455	29.4545	29.7273	29.9091
STD=1	12.8750	10.5625	11.8125	12.0625	12.0625	77.8182	29.5455	66.0909	59.7273	29.9091
Mean=0	13.6250	10.5625	12.3125	11.8125	11.6250	32.4545	29.5455	32.7273	31.3636	31.4545
STD=1, Mean=0	13.6250	10.5625	12.4375	11.8125	11.6250	38.5455	29.5455	39.3636	32.0909	31.4545
Edge Filter	22.5000	21.0625	22.0000	21.0625	21.1250	37.2727	41.8182	44.2727	41.7273	41.8182
Statistical Filter	64.5000	77.6250	79.8750	77.6875	77.6250	52.9091	70.5455	70.9091	70.6364	70.5455

The tests were made with the five pattern matching equations, as shown in the tables 6.1 and 6.2. The results are the average percentage of the windows that match correctly for the stretched images, and for the skewed images. The results decreased in accuracy with increasingly deformed images (figures 6.4 and 6.5). The tests with noisier images gave less accurate results in virtually every test. This is particularly noticeable in the statistical mask tests. The probability of an accurate match with the stretched images was found to be greater than the probability of a correct match of the skewed images, except with the statistical filter tests. The filtered images have poor results in all cases.

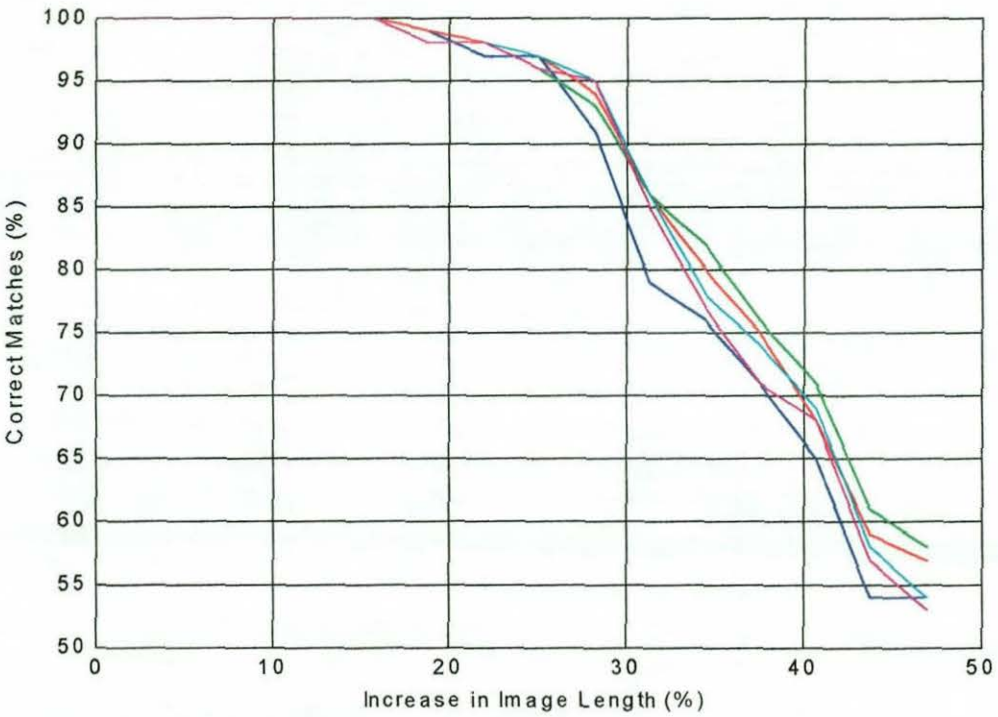


Figure 6.4 - Accuracy of Pattern Matching Methods for Varying Stretch. No pre-processing. Pattern Matching Method: Difference: 🟦, Fully Normalised: 🟩, Square of Difference: 🟥, New Normalised: 🟦, Normalised: 🟪.

As expected, the results for the fully normalised cross correlation equation were independent of the mean and standard deviation of the images, whilst the normalised correlation equation was not affected by dividing by the standard deviation. The square of difference equation was badly affected by pre-processing, especially by dividing by the standard deviation with the skew images, whilst the new correlation equation also had a problem with the skewed images when divided by the standard deviation, unless the mean was also subtracted. The results for the sum of difference equation were poor in all cases, except for detecting skews in images without pre-processing.

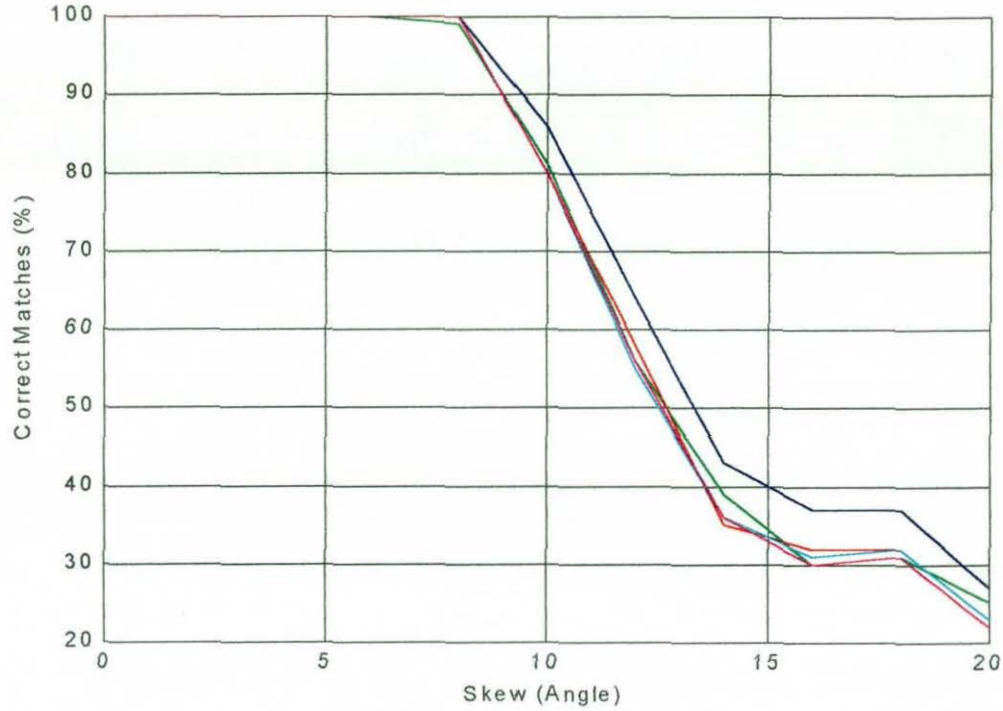


Figure 6.5 - Accuracy of Pattern Matching Methods for Varying Skew No pre-processing. Pattern Matching Method: Difference: —, Fully Normalised: —, Square of Difference: —, New Normalised: —, Normalised: —.

Although the sum of difference equation gave the most accurate result with the skewed image when there was no pre-processing, it more than doubled the processing time for the tests. The small improvement in accuracy was not considered justification for the extra processing time, so it was decided not to undertake further tests with this equation. Whilst the cross correlation equations were more accurate with pre-processing for the stretched images, all other tests performed better with no pre-processing. Further experiments in this chapter were undertaken without pre-processing.

6.4.3 - Optimum Window Size

The accuracy of the match is dependent on the size of the window in the deformed image. Research [88] has shown that in cases where the deformation is directional, rectangular windows are more accurate than square windows. The optimum window size in each direction is dependent on the amount of deformation in that direction. As these tests were made with simulated images deformed in only one direction, the optimum window size in the other direction is as large as possible. However, the upper window size is limited by the desire to measure deformations over small areas.

The accuracy of a number of window sizes was tested with both the skewed and the stretched images. These tests were undertaken iteratively, with tests made to refine the optimum size. A list of the tests and results are in table 6.3.

Table 6.3 - Window Sizes

		Correctly Located Matches (%)							
Window Size		Stretched Images				Skewed Imaged			
Y	X	Fully Normalised	Square of Difference	New Normalised	Normalised Correlation	Fully Normalised	Square of Difference	New Normalised	Normalised Correlation
1	12	87.5625	78.8125	79.6875	85.9375	85.7273	76.6364	77.1818	84.6364
2	12	61.8750	48.6250	49.0000	60.9375	60.2727	47.0000	48.1818	58.8182
4	12	29.8750	16.4375	16.3750	24.3750	29.5455	20.0000	20.8182	28.1818
8	12	11.5000	4.7500	4.8125	7.8125	17.7273	11.3636	12.1818	16.1818
16	12	7.5625	4.9375	5.1250	7.4375	26.1818	23.6364	24.1818	28.6364
8	8	31.0000	19.0000	19.1250	28.1875	36.6364	27.6364	27.7273	36.4545
8	16	4.6250	1.1875	1.0000	2.0625	8.4545	5.0909	5.2727	7.3636
8	20	1.5000	0.0625	0.0625	0.3125	3.0000	1.2727	1.5455	2.6364
8	24	0.5625	0.0625	0.0625	0.0625	1.1818	0.5455	0.6364	1.0000
8	32	0	0	0	0	0.3636	0	0.0909	0.1818
8	40	0	0	0	0	0.0909	0	0	0.0909
8	48	0	0	0	0	0.0909	0	0	0
2	32	3.3125	0.6250	0.5625	1.3125	3.4545	1.8182	1.7273	2.1818
4	32	0.2500	0.1875	0.1875	0.1875	0.1818	0	0	0
6	32	0	0	0	0	0.4545	0.0909	0.0909	0.0909
10	32	0	0	0	0	1.0000	0.4545	0.4545	0.6364
12	32	0	0	0	0	2.0000	1.6364	2.0909	2.2727
6	28	0.1250	0	0	0	0.5455	0.2727	0.2727	0.3636
6	30	0.1250	0	0	0	0.4545	0.2727	0.3636	0.3636
6	34	0	0	0	0	0	0	0	0
6	36	0	0	0	0	0	0	0	0

The first five tests were with the **X** window direction fixed to 12 pixels, and a choice of **Y** window dimensions. The fourth test, with a **Y** dimension of 8, produced the best result, particularly with the skewed images. A range of **X** dimensions were tested with a **Y** dimension of 8. This showed that the larger **X** dimensions produced better results. Further tests show a window of 8 x 32 produces acceptable results for the simulated deformation in the images tested.

6.4.4 - Noise

Tests were made to see how resilient the pattern matching equations are to noise. Normally distributed noise with standard deviations up to six times that measured in the radioscopic images was added to the simulated images. The square of difference and the new normalised equations were more accurate with the less noisy images, whilst the square of difference and the fully normalised equations performed better with the noisier images (see tables 6.4 to 6.6). Tests were made with three window sizes: 4 x 32, 6 x 32 and 8 x 32. The larger window size was the most resilient to noise.

Table 6.4 - Noise Tolerance with 4 x 32 Window

Noise	Correctly Located Matches (%)							
Level	Stretched Images				Skewed Imaged			
(std. dev.)	Fully Normalised	Square of Difference	New normalised	Normalised Correlation	Fully Normalised	Square of Difference	New normalised	Normalised Correlation
0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
10.0000	0.2500	0.1875	0.1875	0.1875	0.1818	0	0	0
15.0000	6.3125	4.5625	4.4375	5.5625	6.3636	4.1818	4.5455	5.7273
20.0000	24.1875	21.3750	22.3125	26.5625	25.0909	24.3636	25.8182	30.636
30.0000	63.0625	67.9375	70.6875	75.5625	64.7273	69.2727	75.0909	78.363
40.0000	83.3750	87.0625	89.0000	92.6875	85.2727	88.3636	91.9091	94.454
60.0000	95.9375	96.5000	98.4375	98.8125	96.2727	97.2727	98.4545	99.000

Table 6.5 - Noise Tolerance with 6 x 32 Window

Noise	Correctly Located Matches (%)							
Level	Stretched Images				Skewed Imaged			
(std. dev.)	Fully Normalised	Square of Difference	New normalised	Normalised Correlation	Fully Normalised	Square of Difference	New normalised	Normalised Correlation
0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
10.0000	0	0	0	0	0.4545	0.0909	0.0909	0.0909
15.0000	2.0000	1.0625	0.8750	1.2500	3.5455	2.0000	2.0909	2.6364
20.0000	12.6875	9.8750	10.6250	13.6250	14.7273	13.8182	14.8182	18.636
30.0000	48.6875	52.3750	58.1250	63.6875	51.2727	55.5455	60.0000	67.636
40.0000	74.6875	80.7500	83.6250	87.8125	75.0909	82.0909	87.3636	90.363
60.0000	93.6875	95.3750	97.7500	98.5000	95.0000	96.6364	98.2727	98.727

Table 6.6 - Noise Tolerance with 8 x 32 Window

Noise	Correctly Located Matches (%)							
Level	Stretched Images				Skewed Imaged			
(std. dev.)	Fully Normalised	Square of Difference	New normalised	Normalised Correlation	Fully Normalised	Square of Difference	New normalised	Normalised Correlation
0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
10.0000	0	0	0	0	0.3636	0	0.0909	0.1818
15.0000	0.8125	0.3750	0.4375	0.7500	2.8182	2.0909	2.0000	2.7273
20.0000	6.6875	4.7500	5.2500	7.3125	12.0909	9.0000	10.4545	13.727
30.0000	36.8750	40.1875	44.6250	49.1875	42.3636	47.5455	52.4545	57.909
40.0000	65.0625	72.3750	76.6250	82.5000	69.2727	75.2727	81.1818	86.727
60.0000	91.7500	94.1250	97.1250	98.0000	92.0909	94.6364	96.7273	97.454

6.5 - Using all the Pattern Matching Equations

Calculating the fully normalised cross correlation equation means the best matches for the other three equations can be calculated with little extra processing. This produces four locations for the best match. Some or all of these may be incorrect. It is worth comparing the results from the four methods to see if they can be cross referenced to either improve the accuracy of the match, or to give a level of confidence for the match.

Figure 6.6 plots the percentage of measurements that match for each pair of methods. It can be seen that the fully normalised method is more likely to have a unique result for the noisier images, whilst the sum of squares of differences method produces very similar results to the new normalised cross correlation method.

For each measurement, each of the four methods can produce either a correct or incorrect result. This means there are 16 potential combinations of the results. The number of occurrences of each of these combinations for the tests collated in table 6.5, are to be found in table 6.7:

Table 6.7 Methods Providing Correct Results (6 x 32 window tests)

Methods Providing Correct Result				Noise Level (Number of Occurrences)							
Fully Normalised	Square of Difference	New Normalised	Normalised Correlation	0	5	10	15 (std	20 dev)	30	40	60
✓	✓	✓	✓	2600	2600	2595	2511	2058	762	228	26
✓	✓	✓		0	0	0	6	70	132	82	12
✓	✓		✓	0	0	0	0	1	2	2	0
✓	✓			0	0	0	2	27	111	83	38
✓		✓	✓	0	0	0	3	6	10	9	3
✓		✓		0	0	0	1	6	7	5	0
✓			✓	0	0	0	0	0	2	1	0
✓				0	0	0	6	75	265	228	69
	✓	✓	✓	0	0	4	38	117	100	29	7
	✓	✓		0	0	0	3	13	26	7	1
	✓		✓	0	0	0	0	1	1	2	0
	✓			0	0	0	2	12	54	45	22
		✓	✓	0	0	0	0	2	9	11	2
		✓		0	0	0	2	4	7	6	2
			✓	0	0	0	0	2	1	1	0
				0	0	1	26	206	1111	1861	2418
4 Correct Results				2600	2600	2595	2511	2058	762	228	26
3 Correct Results				0	0	4	47	193	244	122	22
2 Correct Results				0	0	0	6	50	156	109	41
1 Correct Results				0	0	0	10	93	327	280	93
0 Correct Results				0	0	1	26	206	1111	1861	2418

Table 6.8 - Incorrectly Identified Matching Locations

Methods Providing Incorrect Matches				Noise Level (Number of Occurrences)							
Fully Normalised	Square of Difference	New Normalised	Normalised Correlation	0	5	10	15 (std	20 dev)	30	40	60
✓	✓	✓	✓	0	0	1	9	70	304	508	676
✓	✓	✓		0	0	0	0	5	6	2	1
✓	✓		✓	0	0	0	0	0	0	0	0
✓	✓			0	0	0	0	0	2	1	0
✓		✓	✓	0	0	0	0	1	2	1	1
✓		✓		0	0	0	0	0	0	0	0
✓			✓	0	0	0	0	0	0	0	0
	✓	✓	✓	0	0	0	0	0	0	0	0
	✓	✓		0	0	0	0	0	0	0	0
	✓		✓	0	0	0	0	0	0	0	0
		✓	✓	0	0	0	1	3	5	0	1
				2600	2600	2599	2590	2521	2281	2088	1921
4 Matching Errors				0	0	1	9	70	304	508	676
3 Matching Errors				0	0	4	0	6	8	3	2
2 Matching Errors				0	0	0	1	3	7	1	1
0 Matching Errors				2600	2600	2595	2590	2521	2281	2081	1921

The results can be compared with those in table 6.8. This table gives details of results that match at incorrect locations. Comparing the two tables show that for the noisier images, having all four results matching does not mean the correct result has been found. The results show that the same incorrect match is often found by all four methods. In fact, two or three matching results are more likely to produce a correct result than when all four results are the same. The noise can produce an incorrect location more similar to the original than the true location. This makes it difficult to use all four methods to cross check the result, as the similar, but incorrect, locations would still be found. Accurate results are best obtained by using the most accurate of the equations. Alternative checks on the accuracy could be to undertake matches with different window sizes, or possibly with pre-processed images.

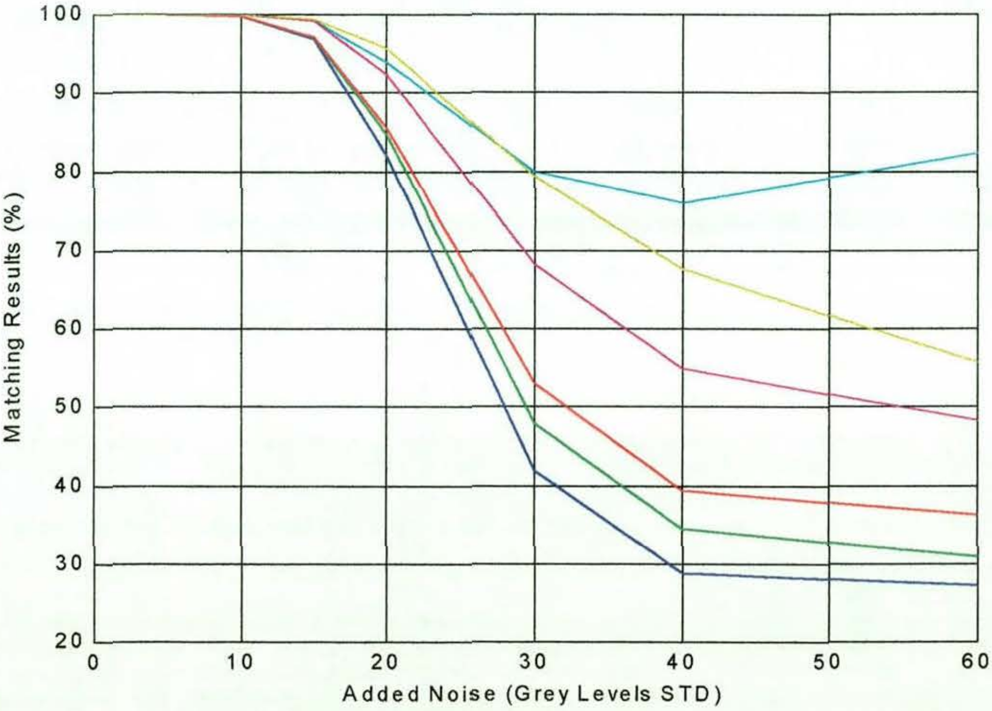


Figure 6.6 - Comparison of Results from Pattern Matching Equations. Results where two or more methods produce identical results. Pattern Matching Methods: Fully Normalised and Normalised: ▬, Fully Normalised and New Normalised: ▬, Fully Normalised and Square of Difference: ▬, Normalised and New Normalised: ▬, Normalised and Square of Difference: ▬, New Normalised and Square of Difference: ▬.

6.6 - Summary

This chapter investigated the pattern matching equations identified in chapter 2. A number of common features of the methods were identified. These were the pattern matching equation, the pre-processing method (if applicable), and the method of identifying the best match. The chapter looked at identifying which combination of these features was the most robust for use with the radioscopic deformation measurement images.

Tests were undertaken using images with simulated deformations applied. These have the advantage of having a known deformation, making it easy to confirm an accurate match. In the tests, several methods were rejected due to the length of time they took to calculate with Matlab. All combinations of the remaining pattern matching equations and pre-processing techniques were tested to find the combination most likely to correctly identify the matching location. The pre-processing methods were not found to improve the robustness of the technique with the simulated images used in these tests.

Further tests were made with the correlation equations and the square of difference method as there was very little overhead in calculating all four equations. A reasonable pattern matching window size was found to be 8 x 32 pixels. This is a compromise between the robustness of the method, the speed of processing and the ability to make fine scale deformation measurements. Of the methods tested, the square of difference was found to provide the most accurate results for images with the noise level expected in the radioscopic images.

7 - Deformation Measurement for Radioscopic Images

The last experiments in this programme of research were to test the capabilities of the pattern matching methods to measure the deformation of real radioscopic images of the rubber-tungsten sample. Whilst the pattern matching methods were tested with simulated images in the previous chapter, these were known to have a number of properties that would be different from the real images. The simulated images had a uniform deformation across the whole image, whilst the real images were found to have non-uniform deformations. The simulated images also failed to accurately represent any spatial distortion and shading that remained after processing.

This chapter gives details of the experiments undertaken to measure the deformation of the rubber-tungsten sample. Section 7.1 gives details of over a hundred radioscopic images of the sample that were used to test the deformation measurement technique. The optimum pattern matching method from chapter 6 was found to be less reliable with these radioscopic images, than it was with the simulated images. Trials were therefore instigated to optimise the technique for use with the radioscopic images. In addition to testing the different methods and window sizes, a modification to the method of correcting for shading was found to improve the probability of finding the correct match. Details of the optimisation of the pattern matching method for the real radioscopic images are in section 7.2. This section also has details of the method used to measure the deformation and details of how incorrectly located matches were identified. The deformation was taken as the percentage change in the distance between matching features.

The optimised pattern matching method was applied to the radioscopic images, and measurements were made of the deformation of the sample. Section 7.3 looks at the measured deformation, and compares them with the predicted values.

7.1 - Acquisition of Images

The rubber-tungsten sample was clamped into the strain rig. The sample was located so that it was not under tension when the distance between the jaws was 100mm. The rig was placed in the radioscopic chamber with the rubber-tungsten sample in the viewing region. The camera zoom setting was adjusted so the rubber-tungsten sample covered the whole of the image for all of the deformations tested. The camera was focused using a copper IQI, and the aperture was adjusted to maximise the brightness of the 215mm image intensifier zoom setting without clipping the images.

Thirty-seven images were taken with each of the three image intensifier settings and with varying quantities of deformation applied to the sample. All the images were taken at maximum X-ray power. Stretch and skews were applied to the sample by turning the wing nuts (the screw thread has a pitch of 1mm). Readings were taken with up to 20mm stretch and up to 20mm skew. For each deformation setting, an image was taken with each of the three image intensifier zoom settings. Images were also taken at each image intensifier zoom setting of the grid phantom, and with no subject in the radioscopic chamber.

Additional images were taken of the rubber-tungsten sample with the 160mm and 120mm image intensifier zoom settings, with the aperture optimised for the brightness of each setting (table 7.1). Eleven images were taken at each setting, together with a blank image and an image of the grid phantom.

Table 7.1: Images optimised for 160mm and 120mm zoom settings.

Zoom Setting:			160mm	120mm	160mm	120mm	160mm	120mm
Optimum Aperture:			160mm	120mm	160mm	120mm	160mm	120mm
Image Number	Stretch (mm)	Skew (mm)	Standard Deviation of Image (grey levels)		Mean Value of Image (grey levels)		Range of Image (grey levels)	
1	0	0	8.49	6.32	180.7	145.7	63	52
2	10	0	8.36	6.28	175.7	145.8	63	52
3	20	0	8.40	6.43	175.5	151.1	67	50
4	20	10	8.39	6.43	175.7	150.1	63	54
5	20	20	8.51	6.37	177.7	149.3	64	54
6	10	20	8.43	6.31	176.7	147.8	67	54
7	0	20	8.52	6.48	174.7	144.8	64	51
8	0	10	8.31	6.41	174.6	145.9	63	49
9	10	10	8.27	6.33	176.5	147.8	64	51
10	10	0	8.30	6.33	176.8	146.7	63	53
11	0	0	8.00	6.34	165.9	145.4	60	55

Table 7.2 Radioscopic Images of Rubber-Tungsten Sample:

Zoom Setting (mm)			215	160	120	215	160	120	215	160	120
Optimum Aperture (mm)			215	215	215	215	215	215	215	215	215
Image Number	Stretch (mm)	Skew (mm)	Standard Deviation of Image (grey levels)			Mean Value of Image (grey levels)			Range of Image (grey levels)		
1	0	0	13.06	5.22	3.08	186.4	96.7	52.2	81	44	31
2	1	0	13.12	5.22	3.12	187.4	96.8	53.0	86	46	30
3	2	0	13.07	5.30	3.05	186.7	98.4	52.1	86	46	27
4	5	0	13.16	5.23	3.14	187.9	97.2	53.4	85	44	31
5	10	0	13.18	5.34	3.08	188.4	99.6	52.7	83	51	29
6	15	0	12.82	5.18	3.00	185.0	95.9	51.3	82	45	30
7	18	0	12.93	5.22	3.07	185.9	96.8	52.7	84	43	29
8	19	0	13.06	5.29	3.08	188.1	98.4	52.8	81	45	29
9	20	0	13.24	5.36	3.10	190.2	100.3	53.2	81	46	31
10	10	0	13.21	5.27	3.18	189.2	97.9	54.0	83	45	31
11	0	0	13.27	5.39	3.12	189.3	100.5	52.9	89	45	31
12	0	1	13.08	5.23	3.13	187.1	97.2	53.0	81	46	30
13	0	2	13.06	5.28	3.04	186.6	98.2	51.9	81	47	30
14	0	5	12.87	5.15	3.04	184.1	95.2	51.8	80	45	32
15	0	10	12.79	5.23	2.94	182.8	96.6	50.2	81	48	28
16	0	15	13.31	5.43	3.21	193.6	100.3	54.6	92	48	31
17	0	18	12.80	5.41	3.14	191.2	99.6	52.8	85	48	31
18	0	19	12.62	5.38	3.16	189.2	98.3	53.3	84	48	31
19	0	20	12.46	5.39	3.10	188.3	98.4	52.3	82	45	30
20	0	10	12.81	5.15	2.99	182.9	95.1	51.1	82	44	30
21	0	0	12.81	5.20	2.99	184.0	96.5	50.9	80	43	29
22	0	10	13.22	5.27	3.14	188.7	97.8	53.3	81	46	31
23	5	10	13.07	5.26	3.05	188.3	98.5	52.6	82	46	29
24	10	10	13.06	5.27	3.07	189.0	98.9	52.9	81	43	30
25	15	10	13.04	5.33	3.10	190.5	100.1	52.8	83	45	30
26	20	10	13.02	5.29	3.13	191.4	99.4	53.8	85	45	30
27	10	10	13.20	5.37	3.10	190.8	101.2	53.1	85	48	30
28	10	0	13.31	5.28	3.20	190.5	98.7	54.4	86	47	31
29	10	5	13.26	5.37	3.11	190.4	100.6	53.2	82	46	30
30	10	10	13.28	5.29	3.18	191.6	99.8	54.4	85	45	32
31	10	15	12.88	5.43	3.11	192.3	101.5	53.3	85	48	29
32	10	20	12.84	5.31	3.13	191.9	98.4	53.9	91	46	33
33	20	20	12.26	5.45	3.11	194.3	101.9	53.8	85	47	30
34	20	15	12.60	5.39	3.17	194.3	101.2	54.7	82	46	30
35	20	10	12.97	5.36	3.08	192.3	101.3	53.2	84	49	30
36	20	5	13.33	5.32	3.17	193.0	100.0	54.2	84	42	31
37	20	0	13.26	5.37	3.06	191.6	101.6	52.8	83	47	31

Tables 7.1 and 7.2 give details of all the radioscopic images. The tables include statistical data about the images. This information shows that unlike the simulated images, there are variations in the brightness and grey level range for images taken with the same zoom and aperture settings. It is also noted that the 215mm zoom setting images have the best contrast, although the range of grey levels in these images is only a third of the full range. When looking at the properties of the images, it is worth noting that section 3.4.4 found the noise level in the radioscopic images had a standard deviation of at least 2 grey levels, indicating the images of the rubber-tungsten sample have low signal to noise ratios.

7.2 - Measuring Deformations

The first measurements of the deformation of the rubber-tungsten sample using the radioscopic images used the optimum method found in chapter 6. This chapter found that a pattern matching window of 8 x 32 pixels, produced accurate matches with the simulated images, when implemented with no pre-processing and the sum of square of difference pattern matching equation. The tests found that with the real radioscopic images, the results were considerably less accurate (and more variable) than the results in chapter 6 (figures 7.1 and 7.2). Results obtained with three other pattern matching methods are also plotted. The other methods indicate the sum of square of difference equation is not necessarily the most accurate method and that the simulated images did not give a good approximation to the radioscopic images. This section gives details of tests performed with the real radioscopic images to find a better pattern matching method.

In order to obtain a measure of the deformation of the sample, small windows in one radioscopic image (**A**) were pattern matched with larger windows in a second radioscopic image (**B**). The difference in the locations of the matching windows in the two images shows the difference in the deformation of the sample between the two images. The windows in image **A** were located in a 32 pixel spaced grid. The matching locations in the image **B** form a grid that was used to determine the deformation. The advantage of using simulated images was that the location of an accurate match was known. In order to measure the accuracy of the tests with the radioscopic images, software was required that could identify correctly matched windows (see section 7.2.1). Details of the tests used to optimise the method are to be found in section 7.2.2. The tests included trials for different types of pre-processing, and tests to find the ideal window size. The method of correcting for the shading of the radioscopic images was also found to need optimisation.

The newly optimised deformation measurement method was used to measure the distortion of the rubber tungsten sample in the radioscopic images. Section 7.2.3 describes the software used to measure the deformation of the rubber-tungsten sample. Section 7.2.4 covers the methods used to check the accuracy of the results. The tests to optimise the pattern matching methods used a known number of windows, but the number of matches in the deformation measurements varied, making it necessary to write new software.

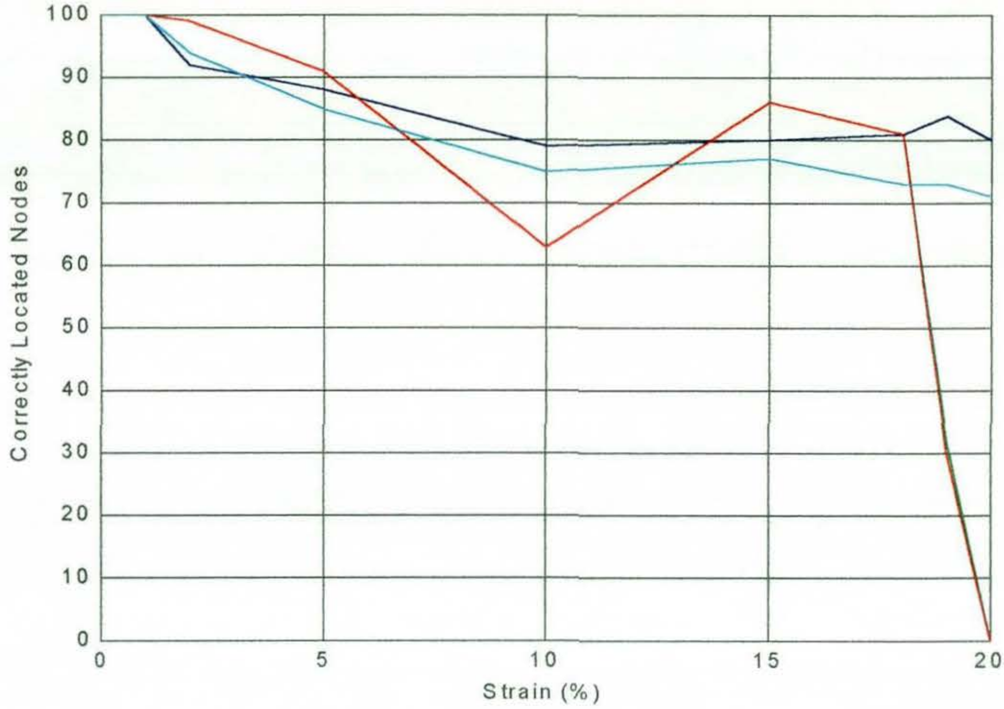


Figure 7.1 - Accuracy of Pattern Matching Method with Varying Stretch. Best pattern matching method from chapter 6 used on radioscopic images. Pattern Matching Method: Fully Normalised: —, Square of Difference / New Normalised: —, Normalised: —.

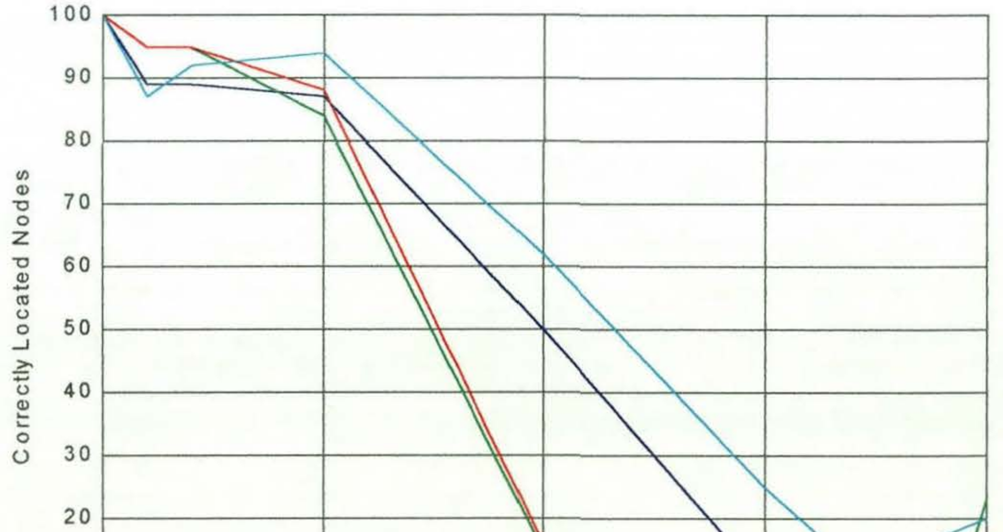


Figure 7.2 - Accuracy of Pattern Matching Method with Varying Skew. Best pattern matching method from chapter 6 used on radioscopic images. Pattern Matching Method: Fully Normalised: —, Square of Difference: — , New Normalised: — , Normalised: —.

7.2.1 - Measure of the Accuracy of Deformation Measurement Tests

Although the deformation applied to the rubber-tungsten sample is known, the material does not deform uniformly. Figure 7.3 shows ripples in the rubber sheet when subjected to a skew displacement, and figure 7.4 shows necking in the sheet when subject to stretch. In addition, the displacement of the radioscopic image is dependent on which part of the sample is being imaged. The variation in displacement means the equivalent location to the centre of image **A** could be anywhere in image **B**. There are also likely to be local variations in the positions of the nodes in the measured grid. A method of measuring the accuracy of the distortion measurement must be able to cope with these variations in the results.

The Matlab function, "gridmet.m" (appendix III.17) was written to measure the accuracy of results from the version of "corl1.m" (described in section 7.2.2) used to test the pattern matching equations with radioscopic images. These tests were undertaken with 100 pattern matching windows located in a regular 10 x 10 grid pattern in image **A**. The software produces a second (measured) grid of matching locations in **B** image. The software counted the number of nodes in the measured grid that could be identified as being part of a regular grid. A different function (see section 7.2.4) was written to identify correct grid nodes in the final deformation measurement software as these results produced a variable size grid.

The function "gridmet.m" measures the angle and distance between nodes, to see which nodes are regularly spaced and at a consistent angle. First, the function processes vertical lines. The angle of the line between each pair of nodes in a vertical line is calculated. With the 10 x 10 grid this gives a total of 450 angles. The function, "histogra.m" (Appendix III.18) is used to find the most common (mode) angle. Any line that lies at an angle close to the most common angle is identified. A count is made, at each node, of the number of lines that are at an angle close to the most common angle. All nodes with a count of three or more are marked as correct.

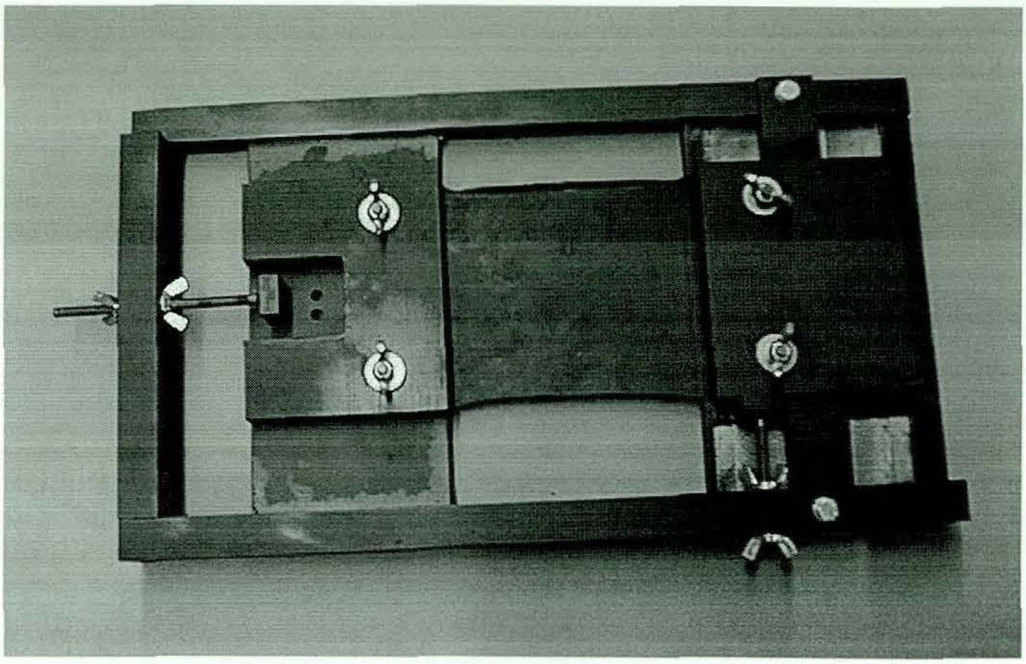


Figure 7.3 - Rubber-Tungsten Sample Subject to Stretch. Note "necking" of rubber sheet

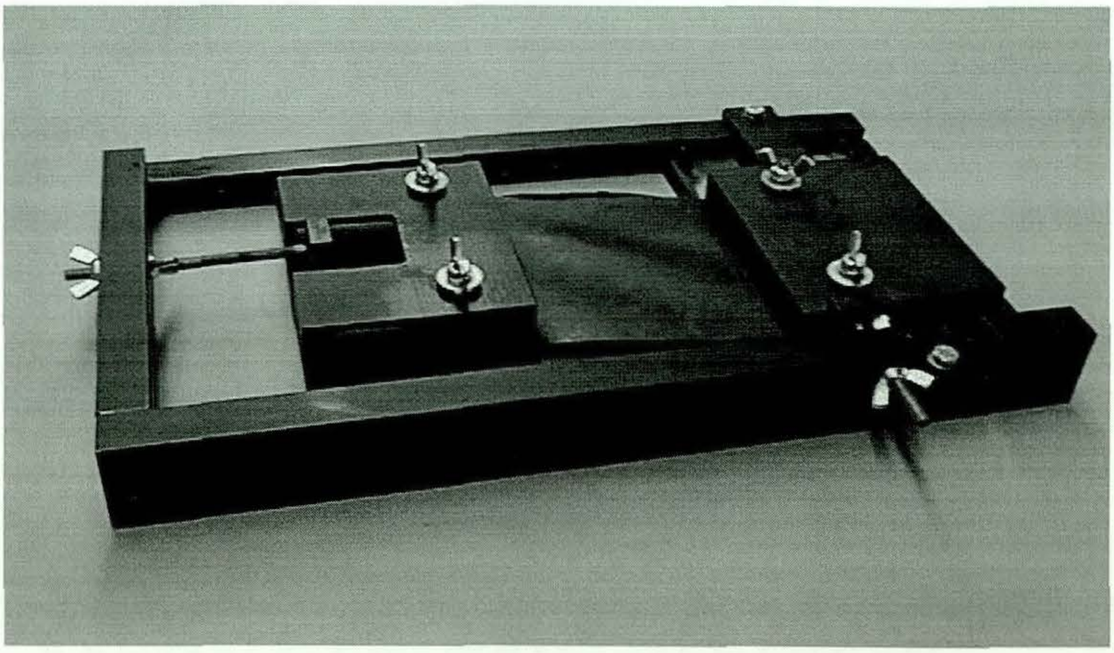


Figure 7.4 - Rubber-Tungsten Sample Subject to Skew. Note the "ripples" in rubber sheet.

The distance between the nodes in vertical lines are considered next. The distance between every pair of nodes in each line is measured, and divided by the number of intermediate nodes plus one, to give a unit node spacing. The most common value of the unit node spacing is found. This is used to identify nodes that are located at the correct distance from other nodes in the line. Any node that is at the correct distance from three or more other nodes in the vertical line is marked as correct. Correctly located nodes in the vertical direction are taken as nodes with correct results for the angle and the distance tests. The procedure is repeated with horizontal lines. Correct nodes in the image are taken as any node that has a correct result for either the horizontal or vertical test. The measure of accuracy of a given grid is the number of correct nodes found in the grid.

If a second argument is passed to "gridmet.m" with a value of one, a graphical output is produced. This uses the function "bmesh.m" (appendix III.6) to display the grid nodes found in image **B**. Different colours are used to distinguish between parts of the grid that were considered incorrect, parts that were found to be correct in both the horizontal and vertical directions, and parts of the grid that were correct in only one direction. All the correctly identified nodes are ringed.

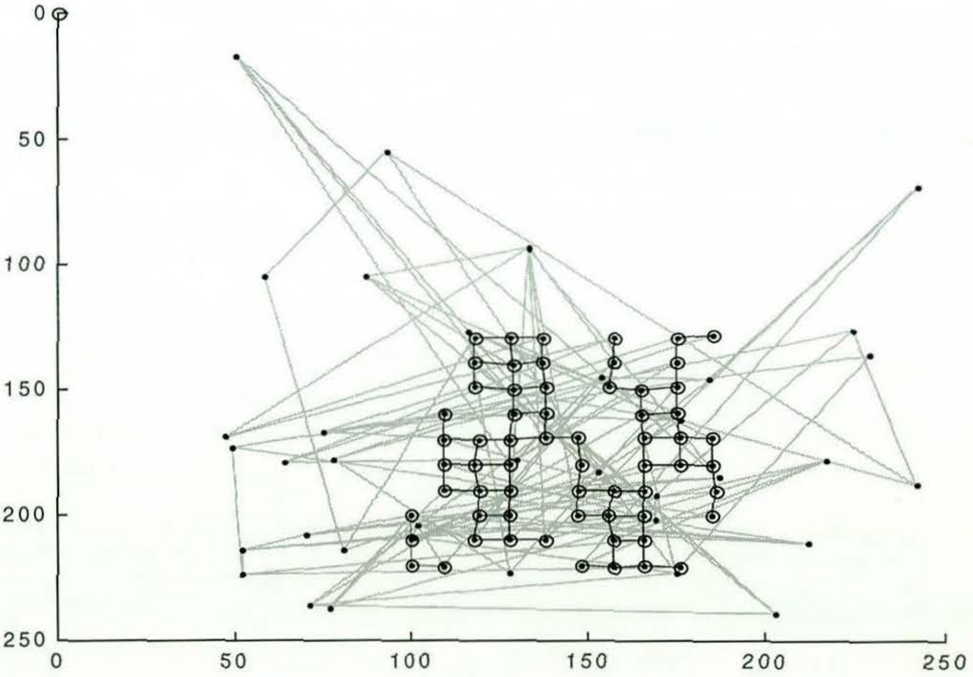


Figure 7.5 - Example of Successful Identification of a Partially Accurate Grid. Image 1 compared with image 4, 4 x 32 window size.

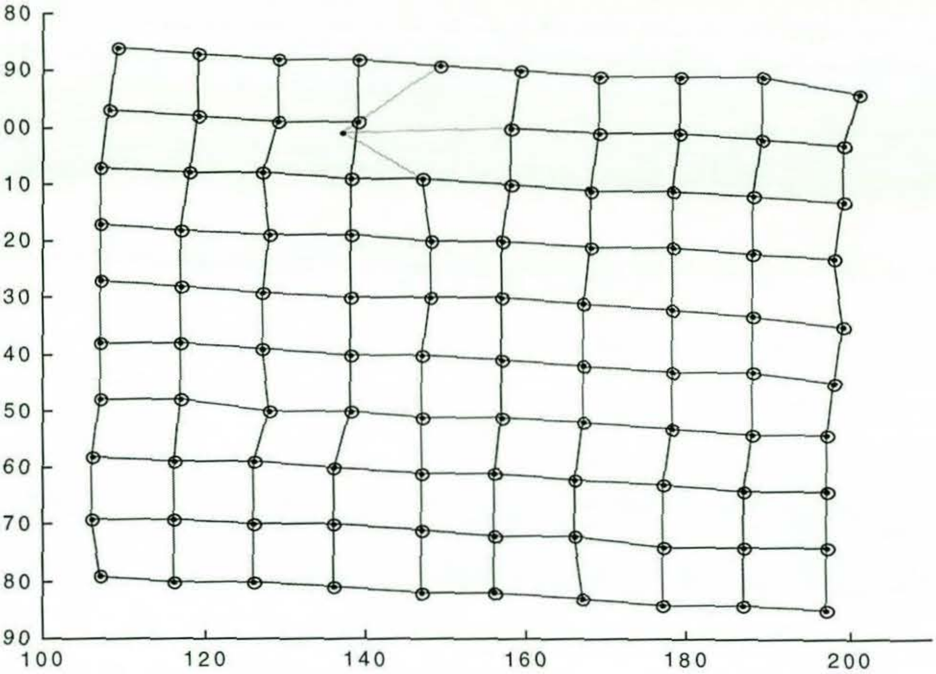


Figure 7.6 - Example of Grid with One Node Rejected. Image 1 compared with image 13, 20 x 32 window size.

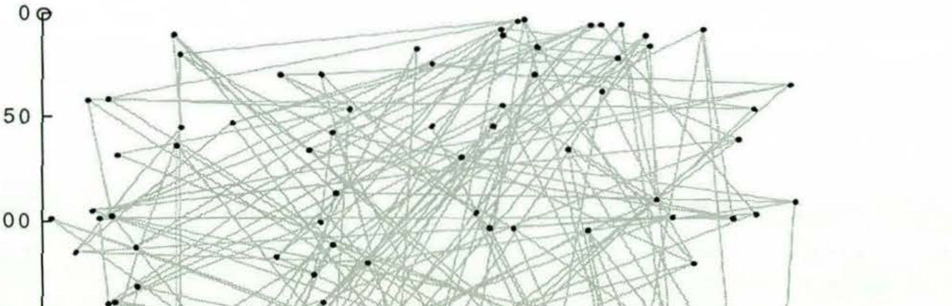


Figure 7.7 - Example of Grid with No Structure Found. (But there may be a structure present). Image 1 compared with image 2, 4 x 4 window size.

Figure 7.5 is a typical grid that has only been matched correctly for about half of the nodes. It can be seen that the function has successfully found the structure of the correctly located nodes. Some of the nodes were only found once, and few nodes in the identified grid look to be offset by a few pixels. The "gridmet.m" function provides an accurate result for grids with a high proportion of correctly located nodes, for example figure 7.6, which shows a grid with one node badly placed. However, the function will under-estimate the accuracy of grids with only a few correctly located nodes, as the line lengths, and angles will not be properly identified (figure 7.7).

7.2.2 - Optimisation of the Deformation Measurement Method

As the deformation measurement method designed for use with the simulated images was found to perform badly with the radioscopic images, a series of tests were undertaken to find the optimum pattern matching method for use with the radioscopic images. The tests were undertaken with eighteen of the radioscopic images taken with the 215mm image intensifier zoom setting. The images used in the test were in order: the image with no applied loading, eight images with increasing amounts of applied stretch, eight with increasing skews, and one with the maximum stretch and skew applied. These are listed in table 7.2 as images 1 to 9, 12 to 19 and image 33.

A modified version of "corl1.m" (see section 6.3) was used to test the deformation measurement methods. The locations of the small windows in the deformed image were chosen so they would all map within a large 256 x 256 window in the image with no load applied, even after maximum deformation. This was found to restrict the maximum size of the small window to 60 x 72 pixels.

Tables 7.1 and 7.2 show that unlike the simulated images in chapter 6, there are mean and standard deviation variations in the radioscopic images. The first of the tests was to see if removing these variations by normalising the images with a pre-processing technique would produce more accurate results. The six pre-processing techniques listed in section 6.4.2 were all tested with four pattern matching equations, and the results are plotted in figure 7.8. It will be noted the statistical filter technique has a near perfect result for all the tests. However, on investigation it was found that the results from this method showed there was no deformation in any of the tests. This was found to be caused by the equation matching to the blank image that was used to correct for shading, thus invalidating this result. Ignoring this erroneous result, the most accurate pre-processing method was the normalisation of the mean and standard deviation, closely followed by the subtraction of mean method. Normalisation of both the mean and standard deviation was used for all later tests to find an optimum deformation measurement method.

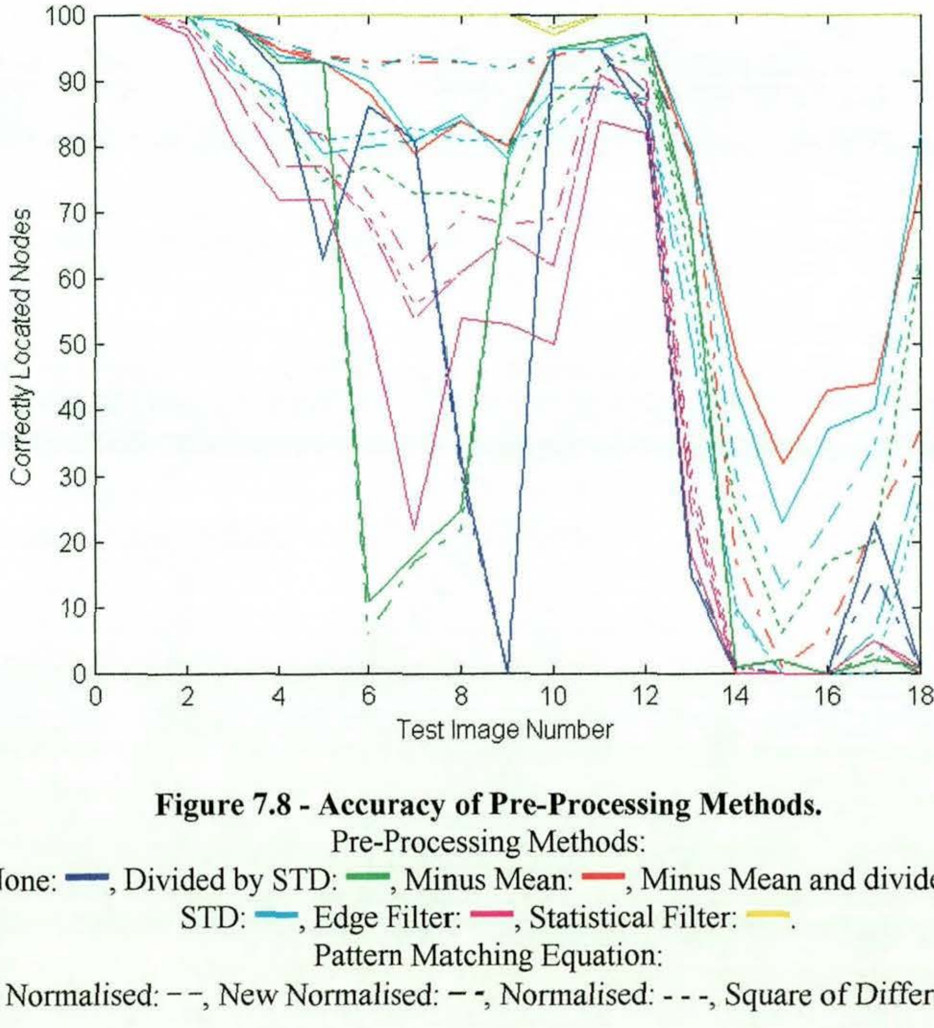


Figure 7.8 - Accuracy of Pre-Processing Methods.
Pre-Processing Methods:
None: —, Divided by STD: —, Minus Mean: —, Minus Mean and divided by STD: —, Edge Filter: —, Statistical Filter: —.
Pattern Matching Equation:
Fully Normalised: – –, New Normalised: – -, Normalised: - - -, Square of Difference: —.

Further investigations showed that in some situations other methods also detected part of the blank image used to correct for shading (figure 7.9). It was therefore desirable to modify the method of shading correction. A pair of images that were noted to identify the blank image were used to test the modifications. The first test was made with a softened blank image, but this gave very poor results. A second, more successful test, was with a constant was added to the blank image. Figure 7.10 plots the accuracy of the results

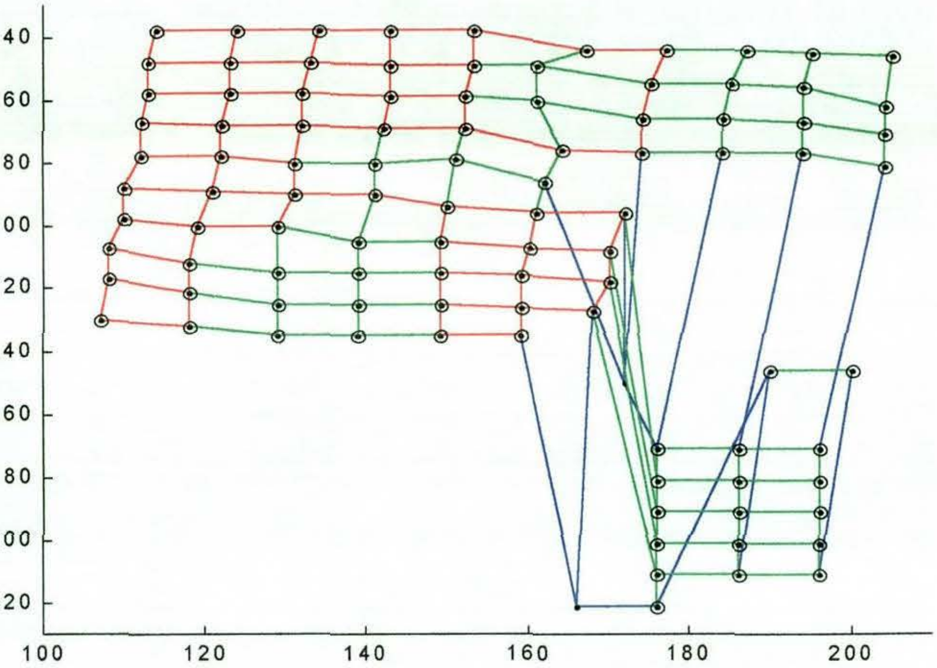


Figure 7.9 - Example of Grid where Part of the Shading Image has been Identified. Red sections are identified as being in the grid, Green areas are identified in one axis only. Blue indicates areas not identified as part of the grid. Shading image found at bottom right corner. Image 1 compared with image 19.

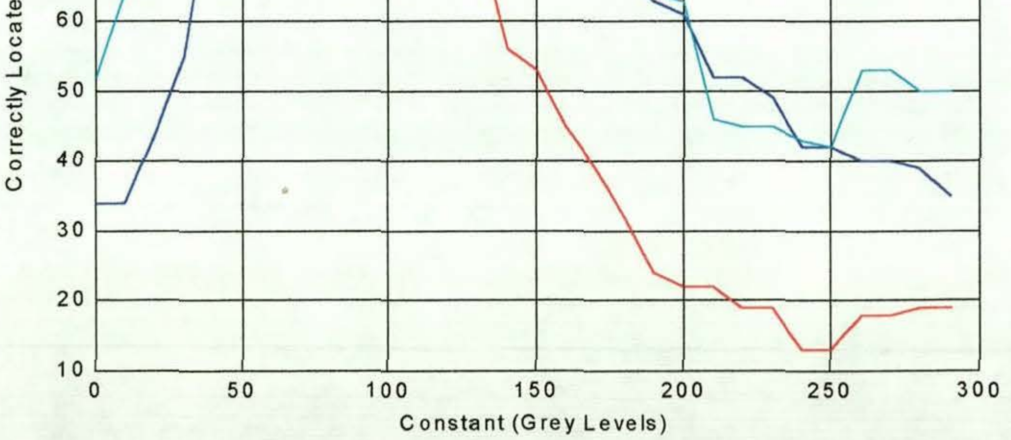


Figure 7.10 - Affect of Adding a Constant to Shading Image. Pattern Matching Method: Normalised: ▬, Fully Normalised: ▬, Square of Difference: ▬, New Normalised: ▬. 215mm zoom images.

against the value of the added constant. The figure shows that a constant of 80 added to the blank image gives an accurate result for all four pattern matching equations, although the fully normalised cross correlation is tolerant of larger values for the constant.

Tests were undertaken to find the best size of the small window (with the constant added to the blank image was 80). Details are also given here of tests undertaken with a constant of 185 added to the blank image. These were obtained after an earlier, erroneous, attempt to find the best value for the constant added to blank image. The results from these tests were corrected to avoid the affect of differing number of potential error locations. This was achieved by calculating the probability of an erroneous match at a given location, and using this to calculate the probability of error if there were 256 x 256 potential locations of best match. Without this correction, larger window sizes have an advantage as there are less potential erroneous locations.

A number of tests were undertaken to find the optimum pattern matching window size for each set of images. The first tests were to find a square window that produced accurate matches with all four pattern matching methods, but allowed some error which could be used to compare the four methods. The shape of the window was found next. This was done by comparing windows of different shapes, but with a similar area. This produces a ratio between the two window sides. Finally, a number of windows of this shape were tested to find the optimum window size for accurate pattern matching.

Figure 7.11 plots the percentage of windows that were correctly matched for the test with 80 added grey levels to the blank image. The figure shows a window of 40 x 40 pixels provides accurate results. The window dimensions were tested with windows that had an area as close to 1600 pixels as possible. Figure 7.12 shows that a ratio of around 1:1.5 ($n \cong 48$) provides the most accurate results. Finally, a number of windows were tested which had a horizontal to vertical ratio of 1:1.5. Figure 7.13 indicates that windows larger than 50 x 75 pixels should be suitable for reliable measurement of the deformation in the radioscopic images. The new normalised correlation method was chosen to measure the deformation of the images as it is the most accurate method in figure 7.13.

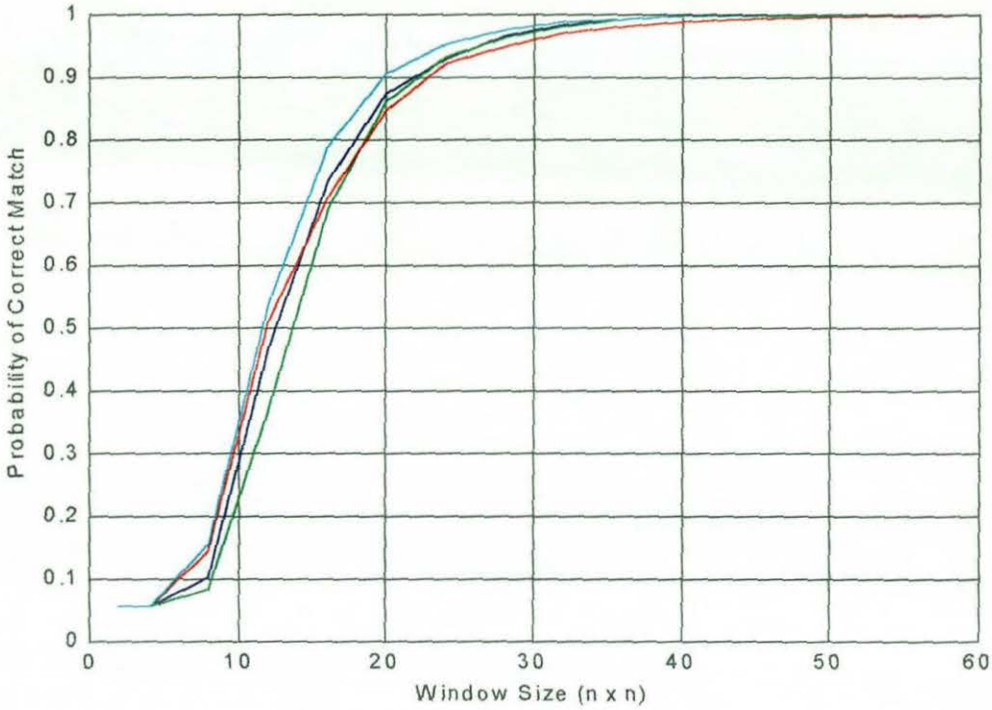


Figure 7.11 - Optimum Window Size with Square Windows. Pattern Matching Method: Normalised: —, Fully Normalised: —, Square of Difference: —, New Normalised: —. Shading constant of 80.

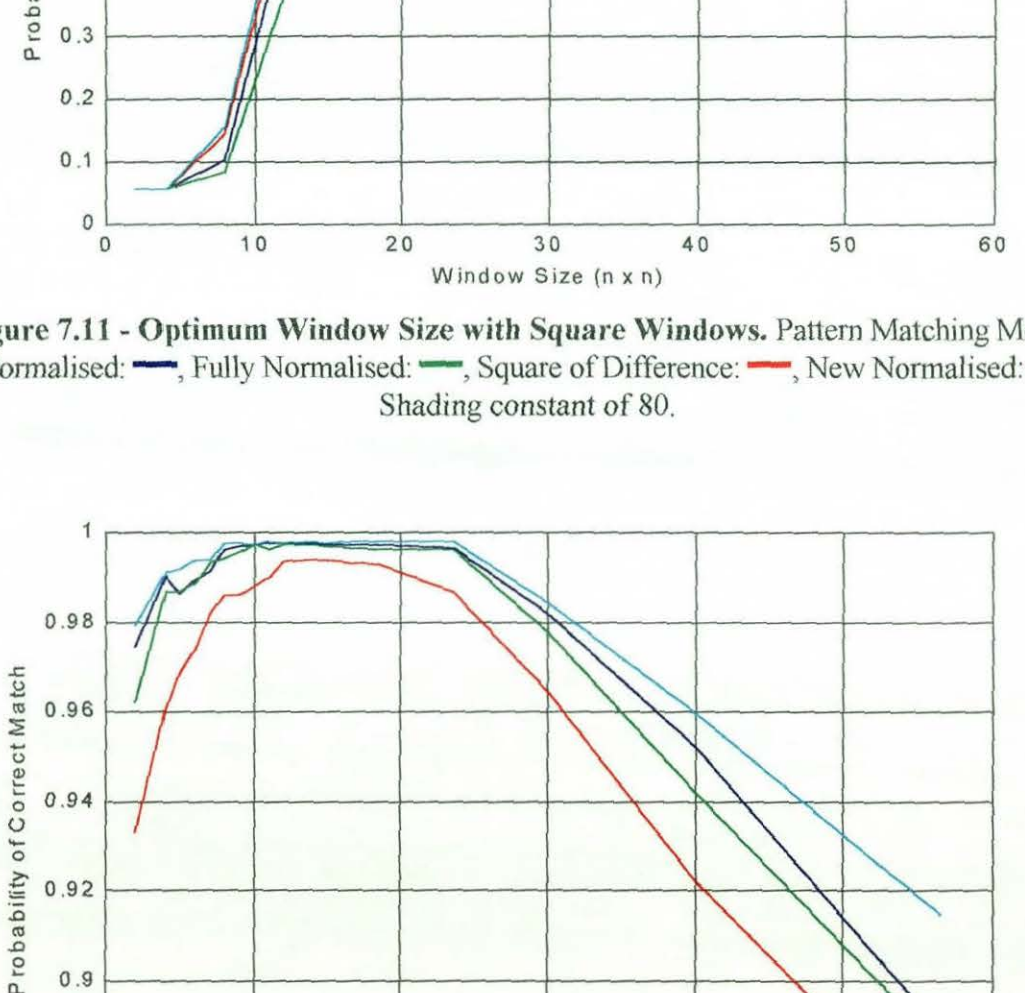


Figure 7.12 - Optimum Window Size with Different Ratios. Pattern Matching Method: Normalised: —, Fully Normalised: —, Square of Difference: —, New Normalised: —. Shading constant of 80.

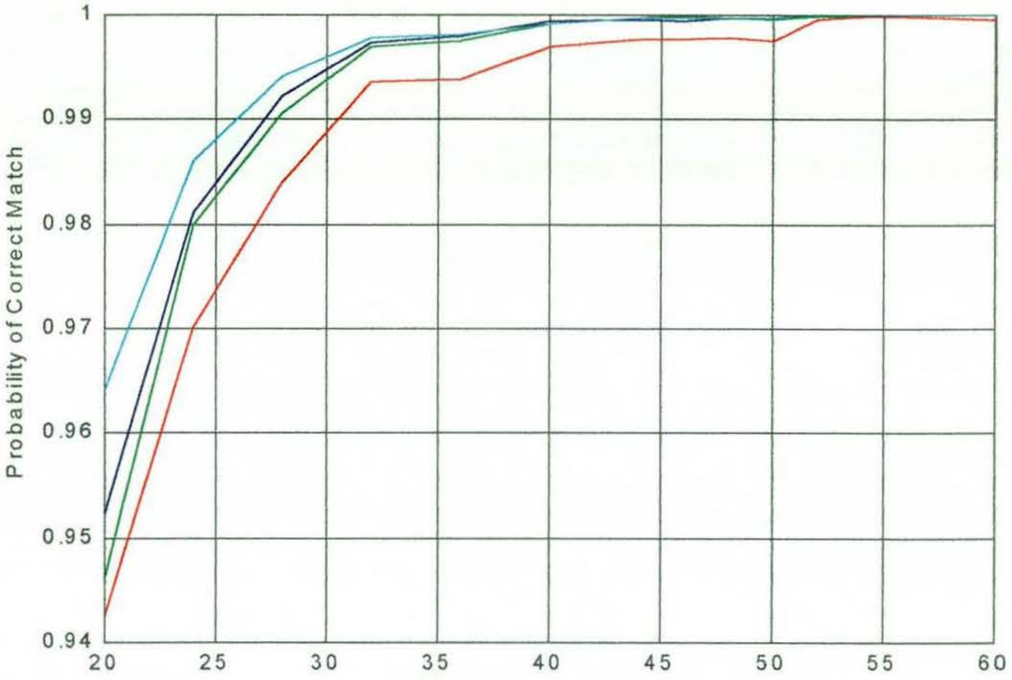


Figure 7.13 - Window Size with 1:1.5 Ratio. Pattern Matching Method: Normalised: —, Fully Normalised: —, Square of Difference: —, New Normalised: —. Shading constant of 80.

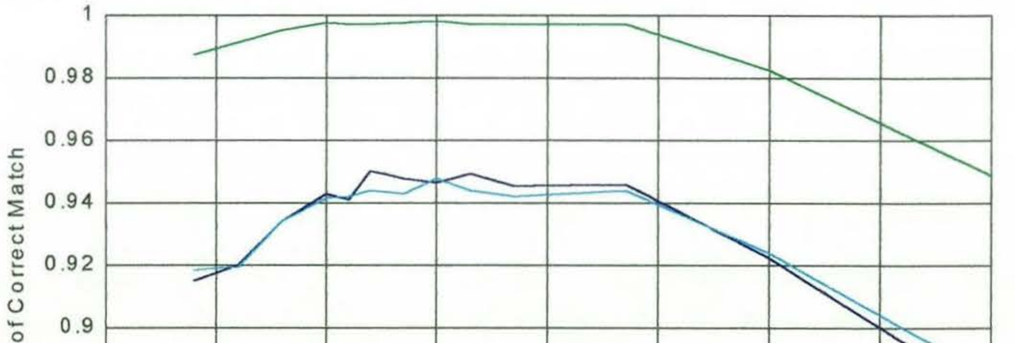


Figure 7.14 - Optimum Window Size with Different Ratios. Pattern Matching Method: Normalised: —, Fully Normalised: —, Square of Difference: —, New Normalised: —. Shading constant of 185.

Another set of tests were undertaken with a constant value of 185 grey levels added to the blank image. Figure 7.14 is a plot of the optimum window dimension ratio. A ratio in the region of 1:2 (n≅56) is noted as giving the best results, which is a higher ratio than the optimum for the test with 80 grey levels added to the blank image. Figure 7.15 shows that a window larger than 32 x 64 pixels can be expected to give accurate results. It is noted that the fully normalised correlation equation is at least 4% more accurate with the higher value of the constant, than the other three pattern matching equations.

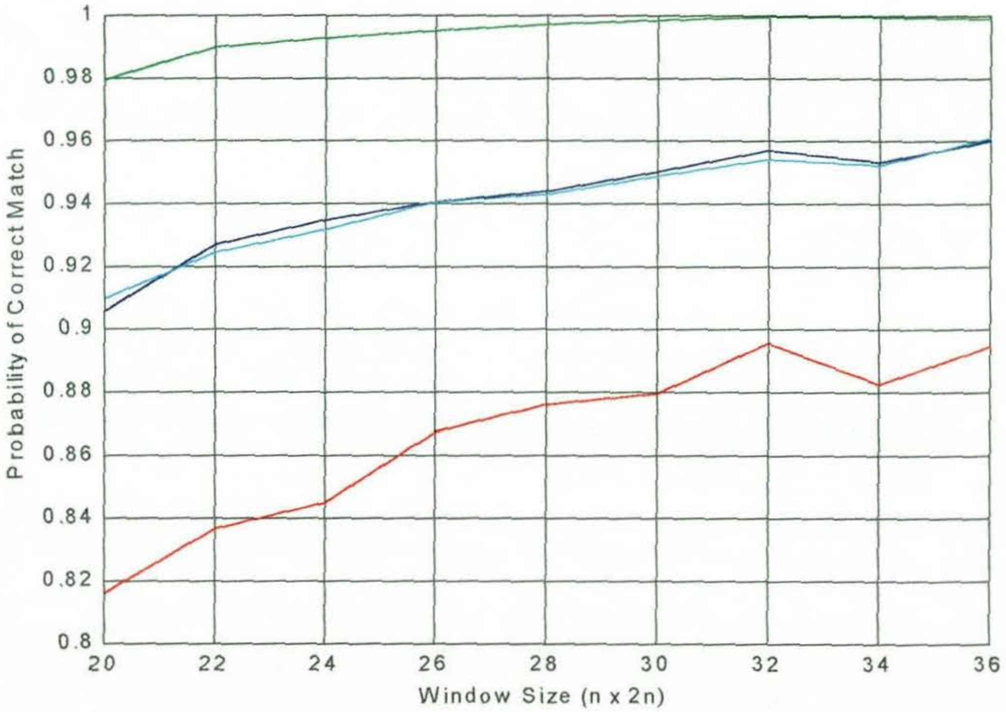


Figure 7.15 - Window Size with 1:2 Ratio. Pattern Matching Method: Normalised: —, Fully Normalised: —, Square of Difference: —, New Normalised: —. Shading constant of 185.

The preferred window size was also found for the eleven radioscopic images taken with the aperture optimised to the 120mm zoom setting. Figure 7.16 shows the effect of adding a constant to the shading image. The fully normalised cross correlation equation is noted to be unaffected by the value of the constant, whilst the other three equations perform better with a smaller value for the constant than the optimum value found in the other tests. A value of 30 was chosen for further tests.

Trials were undertaken with a number of different size windows. Figure 7.17 shows the average results for all eleven images with square windows of different sizes. The results

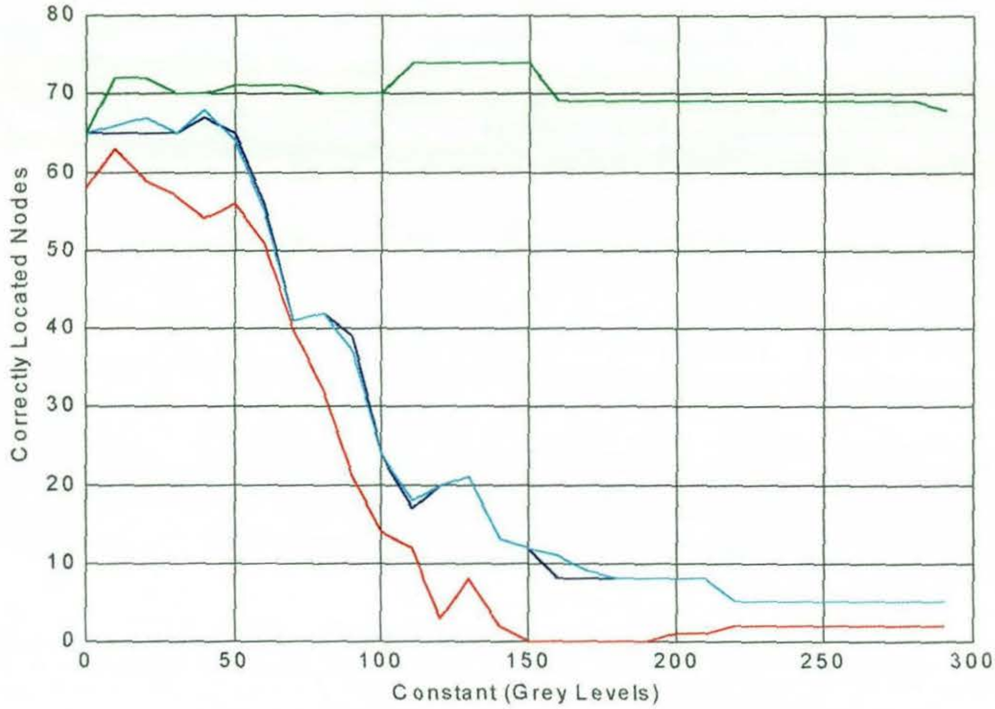


Figure 7.16 - Constant Added to Shading Image. 120mm zoom images. Pattern Matching Method: Normalised: ▬, Fully Normalised: ▬, Square of Difference: ▬, New Normalised: ▬.

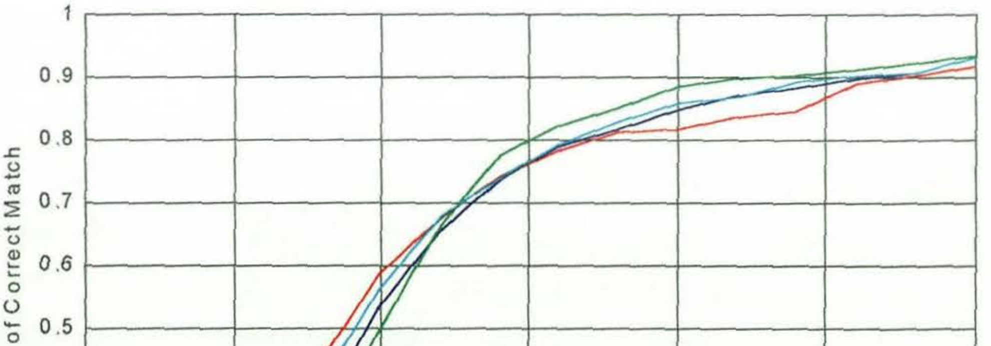


Figure 7.17 - Optimum Window Size with Square Windows. 120mm zoom images. Pattern Matching Method: Normalised: ▬, Fully Normalised: ▬, Square of Difference: ▬, New Normalised: ▬. Shading constant of 30.

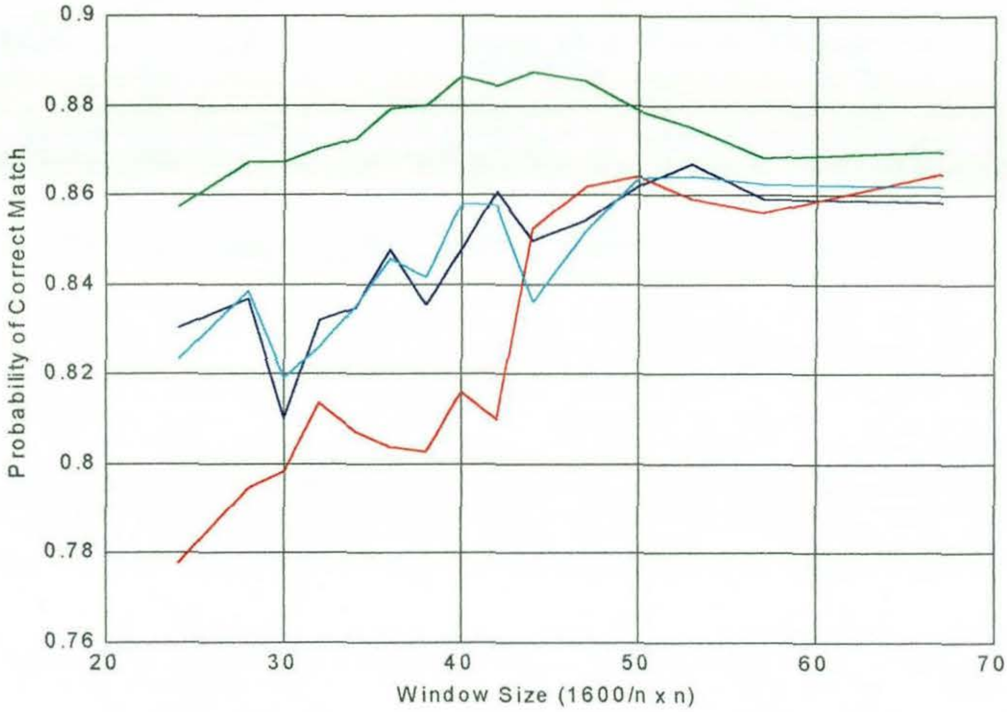


Figure 7.18 - Optimum Window Size with Different Ratios. 120mm zoom images. Pattern Matching Method: Normalised: —, Fully Normalised: —, Square of Difference: —, New Normalised: —. Shading constant of 30.

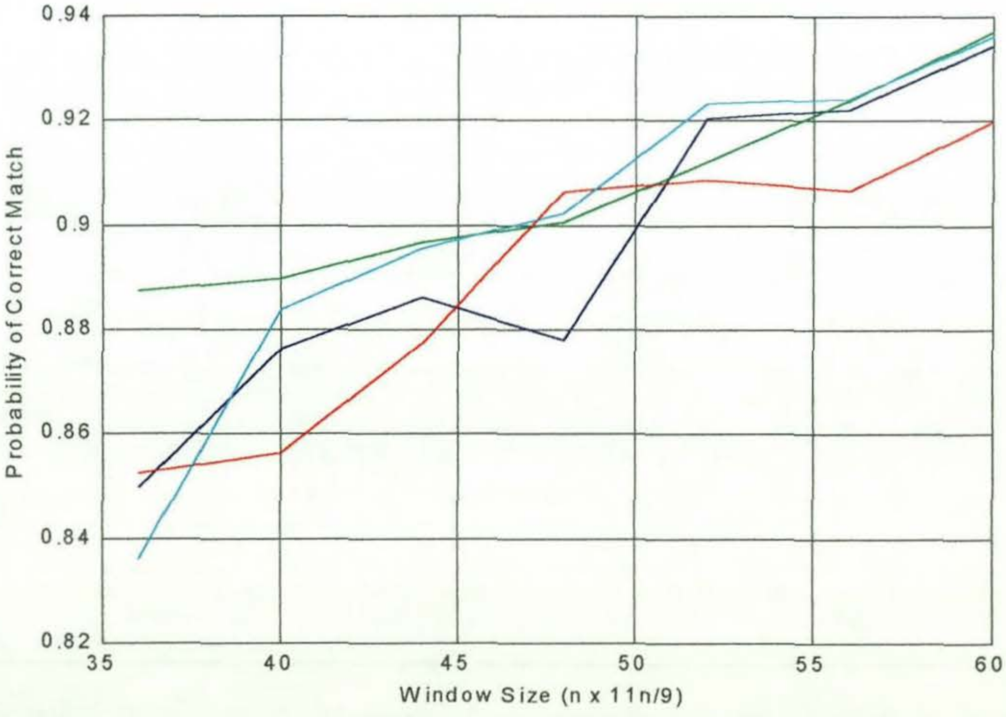


Figure 7.19 - Window Size with 9:11 Ratio. 120mm zoom images. Pattern Matching Method: Normalised: —, Fully Normalised: —, Square of Difference: —, New Normalised: —. Shading constant of 30.

are noted to be less accurate than those for the 215mm zoom setting. Figure 7.18 plots the results for windows of similar sizes (area of 1600 square pixels), but with different ratios between the sides. The best results from this plot have windows in the region of 36 x 44 pixels. More windows with a ratio in the region of 9:11 were used to produce figure 7.19. This shows the larger window sizes give more accurate results, and a window of 60 x 73 nodes was chosen for use in further tests of these images.

7.2.3 - Matching Locations in the Radioscopic Images

The software used to measure the deformation of the rubber-tungsten sample was very similar to the software described in section 6.3. The major difference was the replacement of "correll.m" by "correl.m" (appendix III.19). The new procedure is considerably more complicated than the old version, as it is designed to measure the deformation between two images with an unknown displacement.

The old procedure "correll.m" was used to find the best matches in image **B**, to 100 windows in image **A**. The new procedure, "correl.m", was written to find the best matches to all the grid nodes in the region that appear in both images. The research in section 7.2.2 found the optimum window size in image **A** depended on the images being tested (see tables 7.3 and 7.4 for window sizes used). The windows in image **A** are centred on a 32 x 32 pixel grid, and are all of the same size. The search window in image **B** is either 256 x 256 pixels (used to find the displacement between the two images), or 64 x 128 pixels. The number of grid nodes produced for each pair of images varies because the deformation between the two images is different for each pair of images, resulting in a different shape of the region that appears in both images.

The larger size of window in the **B** image is used in the first three matches. These are matched with a window at the centre of image **A**, and windows for the nodes above, and to the left of the centre node. They are used to find the displacement between the two images and to identify the axes. The smaller window size is used for all the other matches used to make up the grids of corresponding locations. The grid nodes are found first in a vertical line above, and then below the centre node. Further vertical lines of nodes are found, working from the centre. The lines to the left are found first, followed by those to the right. With each line, the nodes above the centre line are found first, followed by the nodes below the centre. For each match, the centre of the window in image **B** is the location of the expected best match. This is found by referring the nearest part of the grid that has already been matched. Searches for matches are only performed for nodes where both the windows used for matching are entirely within the respective images.

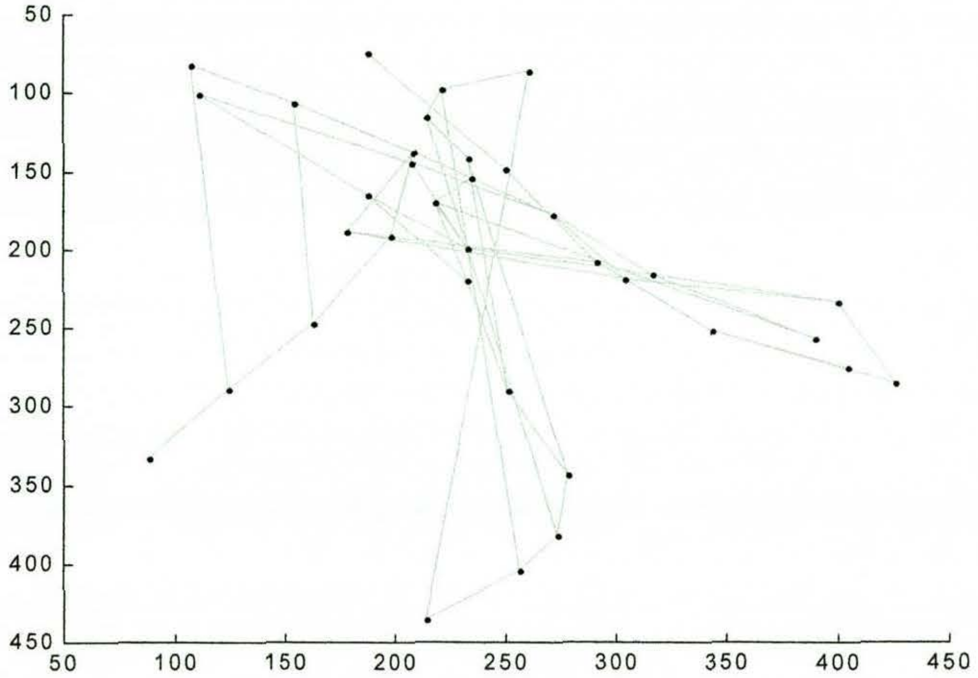


Figure 7.20 - Example of a Failed Grid. Image 1 compared with image 33, 160mm grid.

Other changes to the pattern matching software is the use of "sbcor.m", which is very similar to "sbcor1.m", but only calculates the fully normalised correlation equation. The other significant change to the software is in "corl.m" which is the equivalent to "corl1.m", but compares each image with every other image.

The sequential nature of "correl.m" can cause errors. If any of the first three matches are incorrect, the whole grid structure is likely to be wrong (figure 7.20). Also, if a node within a line is wrong, further nodes added to the line are often incorrect as this results in an incorrect location for the expected match (see figure 7.24). Finally, some nodes can be missed at the sides of the images.

7.2.4 - Measuring the Deformation

Other software was written to measure the deformation of the grids. The windows in image **A** are located in a regular (ideal) grid pattern. The equivalent location of each node in image **B**, is found using the software described in section 7.2.3. By measuring the deformation of the (measured) grid of the best matches in image **B**, the difference in the deformation between the two images can be calculated.

The procedure "mest.m" (appendix III.20) was written to measure the average deformation across the whole of the grid. It loads each set of results, and processes each pair of grids in turn. The ideal and measured grids are passed to "nevu.m" (appendix III.21) which identifies erroneous nodes. This function is similar to "gridmet.m", but was written to deal with grids of an unknown size. It also looks for the most common distance and angles between nodes, but only considers adjacent nodes. Correct nodes are taken as those with one or more adjacent node that is at the correct distance and angle.

The function "mest.m" corrects both grids for spatial distortion, using "zmod.m" (see chapter 4 for details of model). The function assumes the axes of the strain rig are the same as the axes of the grid on the distortion phantom, but parameter 7 of the distortion model could be used to rotate the grids if they are not square. After spatial distortion correction, the function "mdomptx.m" (appendix III.22) is used to measure the deformation of the grids. The stretch is taken as the mean increase in length of the vertical grid lines, whilst the skew is taken as the mean change in horizontal displacement of the vertical grid lines, divided by the vertical length of the vertical grid lines.

A measure is taken of the error in the results. This gives a value that indicates how consistent the results are. It is calculated from the variance of the errors of the deformation measurements. Grids with no structure (for example figure 7.20) give very large error values. Grids where a number of incorrect nodes are in the identified grid used to measure the deformations give a value in excess of 10 (for example 7.24). Images with some badly placed nodes, but no serious errors give a value of around 2, whilst good grids give a value several orders of magnitude less than 1.

The function "mdomptx.m" can produce images of the mesh showing which parts of the grid were used to measure the deformation. The saved results are the measured stretch and skew, the expected strain and skew, the size of the grid, the number of nodes in the grid, the number of nodes used in the measurement and the measure of accuracy of the result.

7.3 - Deformation Measurement Results

Each of the five sets of images were compared using 50 x 75 pixel windows with the new normalised correlation equation, and using 32 x 64 pixel windows with the fully normalised correlation equation. In addition, the two sets of images listed in table 7.1 were also tested with a window size of 60 x 73 pixels with the fully normalised correlation method. Details of the tests and the results are in tables 7.3 and 7.4.

The tables include the number of grids that are rated as failures. This is where less than half of the nodes were identified as being part of a recognisable grid. Figure 7.21 is an example of a failed grid. Although 34% of the nodes were sorted into a grid, this grid does not represent the deformation between the two images.

The following entry in the tables is the number of badly matched grids. In these grids between 50% and 85% of the nodes have been identified as being part of a consistent grid. These grids were found to have sufficient correct nodes to give a good measurement of the deformation, but these measurements were discounted as many of these grids had no measurements over large areas of the images. Figure 7.22 shows an example where the right hand side of the grid was lost as a result of an error in the location of the search window in image **B**. All the grids used to calculate the deformation had over 85% of the nodes identified as correctly located. This still allows grids with up to 15% of nodes rejected. The tables includes the number of rejected nodes in the grids used for the measurements, together with the number of the grids that included rejected nodes.

A comparison was made between the deformations measured using the grids, and the expected value of deformations. The tables includes the average difference between the measured and expected values of stretch and skew, together with the standard deviation of the difference. The values in the tables are the absolute difference between the deformations, expressed as the percentage change in length.

Table 7.3 Summary of Results (New Normalised Cross Correlation Equation Test)

Image Intensifier Zoom:			215	160	120	160	120
Aperture Optimised to:			215	215	215	160	120
Small Window Size:			50 x 75				
Constant Added to Blank Image:			80				
Number of Totally Failed Grids			76	50	226	11	35
Number of Badly Matched Grids			2	1	2	2	0
Number of Nodes Used			1291	1318	1141	108	86
Number of Erroneous Nodes in Grids Used			352	54	8	16	17
Number of Grids Used with Erroneous Nodes			119	22	6	5	4
Difference when Compared with Expected Result (Deformation %)	Stretch	Mean	1.02	0.86	0.70	0.60	0.52
		Std Deviation.	1.30	1.11	0.93	0.78	0.68
	Skew	Average	1.24	0.99	0.86	1.41	0.99
		Std Deviation	1.54	1.35	1.18	1.90	1.37
Difference between Grids of the same Pair of Images (Deformation %)	Stretch	Mean	0.20	0.19	0.13	0.21	0.14
		Std Deviation.	0.25	0.19	0.16	0.22	0.19
	Skew	Average	0.86	0.81	0.54	0.91	0.46
		Std Deviation	1.22	1.28	0.74	1.48	0.71

Table 7.4 Summary of the Results (Fully Normalised Cross Correlation Equation Test)

Image Intensifier Zoom:			215	160	120	160	120	160	120
Aperture Optimised to:			215	215	215	160	120	120	120
Small Window Size:			32 x 64					60 x 73	
Constant Added to Blank Image:			185					30	
Number of Totally Failed Grids			0	19	187	3	30	4	39
Number of Badly Matched Grids			0	17	6	1	0	0	0
Number of Nodes Used			1369	1333	1176	117	91	117	82
Number of Erroneous Nodes in Grids Used			82	226	249	31	34	17	2
Number of Grids Used with Erroneous Nodes			27	56	72	6	7	8	1
Difference when Compared with Expected Result (Deformation %)	Stretch	Mean	1.09	0.86	0.73	0.61	0.59	0.58	0.49
		Std Deviation.	1.38	1.11	0.95	0.78	0.78	0.77	0.66
	Skew	Average	1.30	0.96	0.86	1.58	1.12	1.67	0.99
		Std Deviation	1.60	1.29	1.19	2.15	1.53	2.29	1.39
Difference between Grids of the same Pair of Images (Deformation %)	Stretch	Mean	0.40	0.30	0.21	0.42	0.26	0.31	0.19
		Std Deviation.	0.39	0.31	0.24	0.44	0.30	0.43	0.24
	Skew	Average	1.00	0.85	0.56	1.15	0.55	1.37	0.44
		Std Deviation	1.27	1.27	0.84	1.71	0.76	1.93	0.65

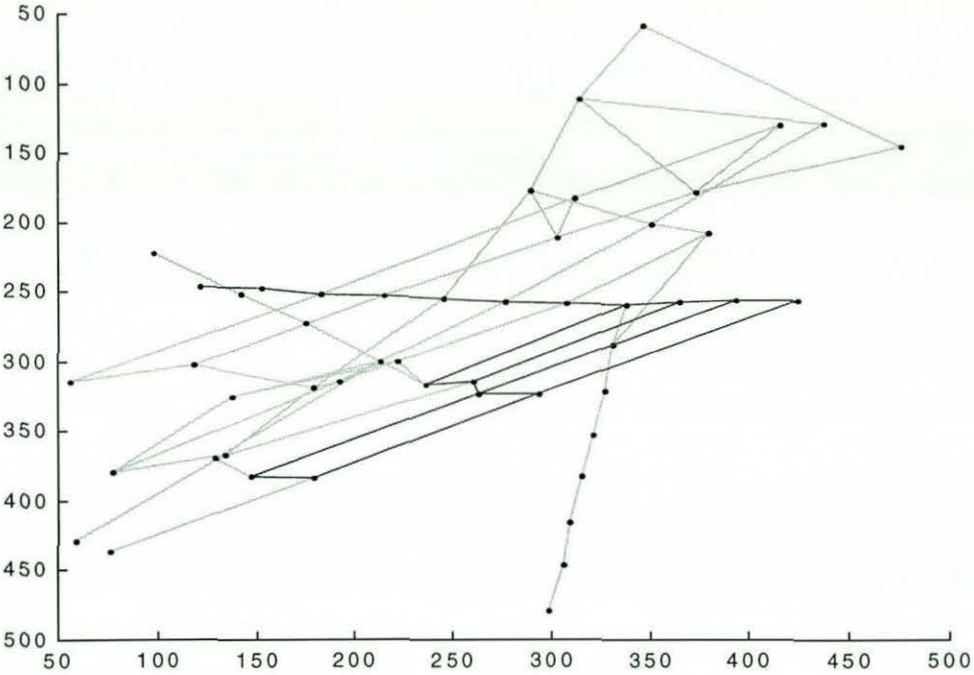


Figure 7.21 - Example of Failed Grid. Light Grey indicates areas identified as incorrect. 34% of nodes found in the grid structure, image 18 compared with image 1, 160mm zoom setting.

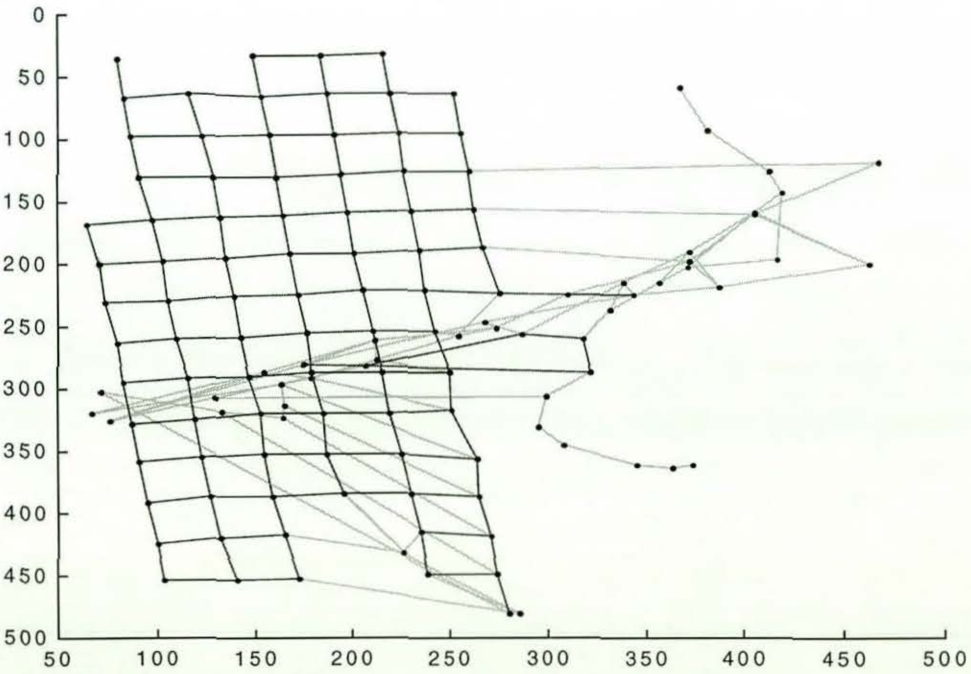


Figure 7.22 - Example of Partly Found grid. Light Grey indicates areas identified as incorrect. Image 1 compared with image 17, 160mm zoom setting.

As every image was compared with every other, the deformations were measured both from one image to the second, and from the second to the first. This gives two measurements for each pair of images. For the two sets of results to agree, the two measured results are the inverse of each other. The differences between the two sets of measurements were calculated, and are tabulated. This gives an measure of the repeatability of the measurements.

From these tests, it can be seen that the pattern matching techniques are able to detect the change in deformation between two images. The fully normalised correlation equation results are slightly more reliable, with less erroneous nodes and failed grids. However, the new normalised correlation equation results are slightly more accurate, being both closer to the expected results, and being more consistent.

The results for the 160mm and 120mm zoom settings have a higher failure rate. However, it should be noted these images have greater average deformations than the images taken with the aperture optimised to the 215mm zoom setting. The larger window size resulted in a higher rate of grid failure, but in most cases the larger windows produced improved accuracy of the results, particularly with the 120mm zoom setting images.

The results of the tests with the images taken with the 215mm zoom setting are noted to have good pattern matches in almost every case, particularly with the fully normalised correlation equation tests. The measured values of deformation vary more than the results taken with the other zoom settings. The larger area visualised in the 215mm images may explain the more variable results, as the non uniform nature of the deformation is likely to have greater effect. With the fully normalised results, the highest value for the measure of error was 3.05, (figure 7.23). Section 7.2.4 found this value for the measure of error indicated some badly positioned nodes. The greatest number of incorrect nodes in any grid found by "nevu.m" was 6 (figure 7.24). In this set of measurements, an average of 161 nodes were found in each experiment. In addition to the 82 erroneous nodes found by the procedure "nevu.m", 4 more were found to be incorrect by manual inspection (an example of an error that was not detected is in figure 7.25). It will be noted that these erroneous nodes have a location and angle that are similar to the rest of the grid, and therefore do not have a large impact of the measured stretch and skew, or the measure of error value.

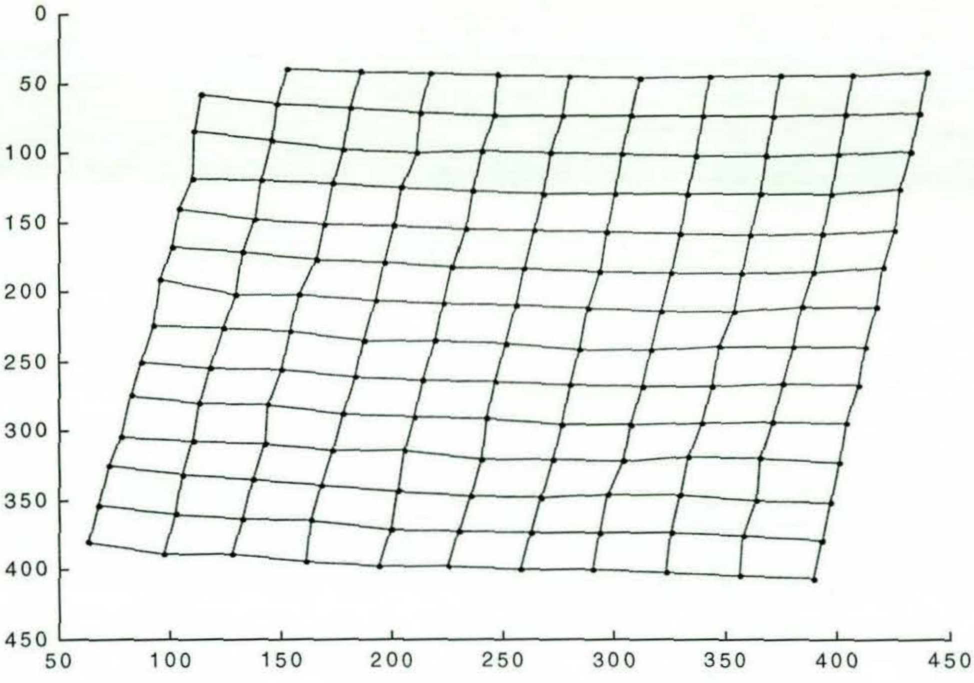


Figure 7.23 - Grid with Worst Merit Value for 215mm Images. Image 19 compared with image 37.

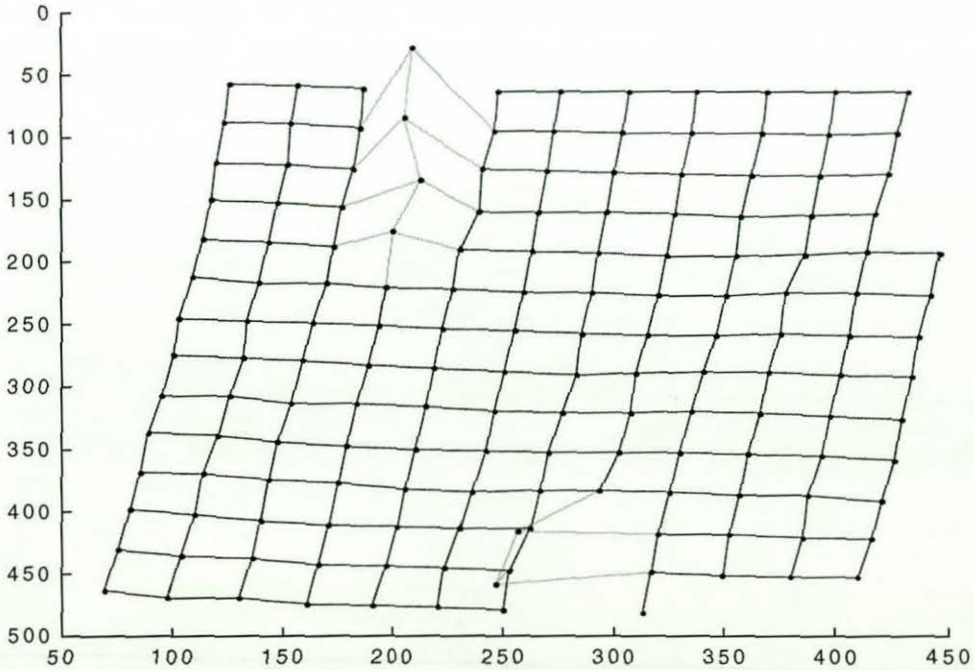


Figure 7.24 - 215mm Image Grid with the Most Incorrect Nodes. Light Grey indicates areas identified as incorrect. Image 18 compared with image 1.

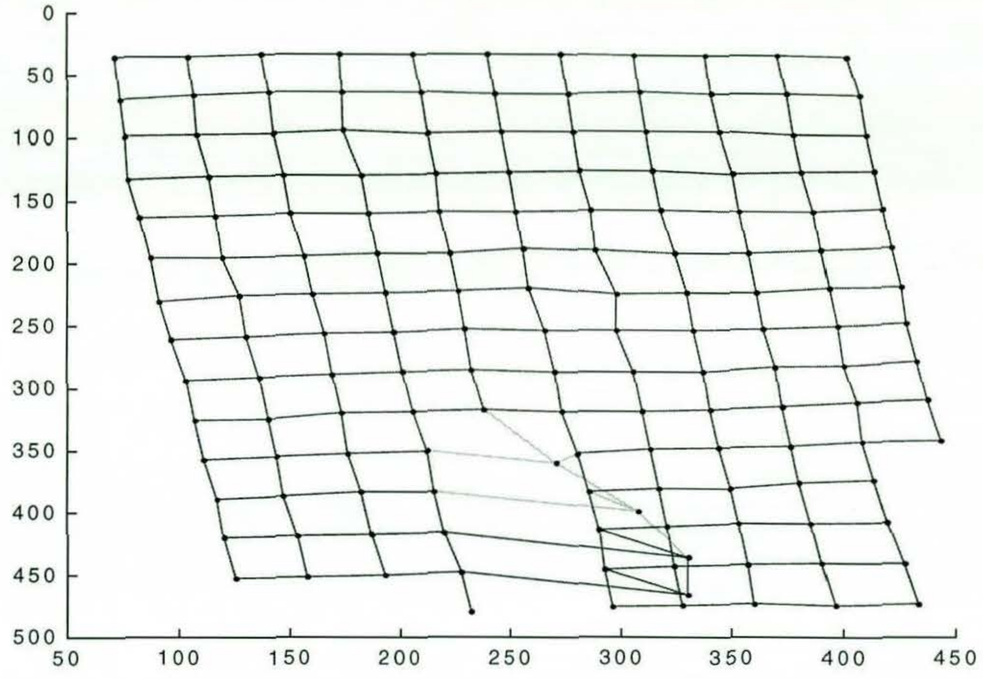


Figure 7.25 - Example of Grid with an Error that is not Identified. Light Grey indicates areas identified as incorrect. Image 11 compared with 17.

The deformations measured using the pattern matching equations show a good correspondence with the displacements applied by the strain rig. Using the results produced by the fully normalised correlation equation tests and 215 mm zoom setting, figures 7.26 and 7.27 shows the measured stretch and skew for the image of the rubber-tungsten sample without deformation (image 1), when compared with the other images. The measured deformations can be seen to closely follow the applied deformations.

Looking at figures 7.26 and 7.27, the measured deformation can be seen to be less than predicted. This could be explained by the clamps allowing some movement of the rubber-tungsten sheet within the jaws of the clamps. This would have the same affect as making the initial distance between the jaws greater than the measured 100mm. Rather than using the measured distance between the clamps, an equivalent clamp distance could be used to take into account the movement of the rubber. A plot of the accuracy of the measured results against different values of the initial distance between the clamps is given in figure 7.28. The best result for the stretched results is with a distance of 115.9mm, whilst the best distance for the skewed results is 107.6mm.

The different values for the best equivalent initial clamp displacement for stretch and skew suggests the skew and stretch loads cause different movements at the jaws. Using the new equivalent clamp distances, the average difference between measured and expected stretch improved from 1.09% to 0.38%, with standard deviation reduced from 1.38% to 0.44%. The improvement in the average difference in the skew measurement was from 1.3% to 1.18%, and standard deviation improved from 1.6% to 1.49%

Figures 7.29 and 7.30 show the results with the optimised clamp spacing. The measured stretch results are very close to the applied stretch. However, the skewed results are too large in the early results, and too small for the later images. It is possible that the rubber tungsten sample could have moved in the clamps, causing the smaller measured skews for the later images. In addition it is noted the skewed results are slightly less than expected for images with larger vales of stretch. The results for all the measured deformations are in figures 7.31 and 7.32. The expected deformations are in figures 7.33 and 7.34.

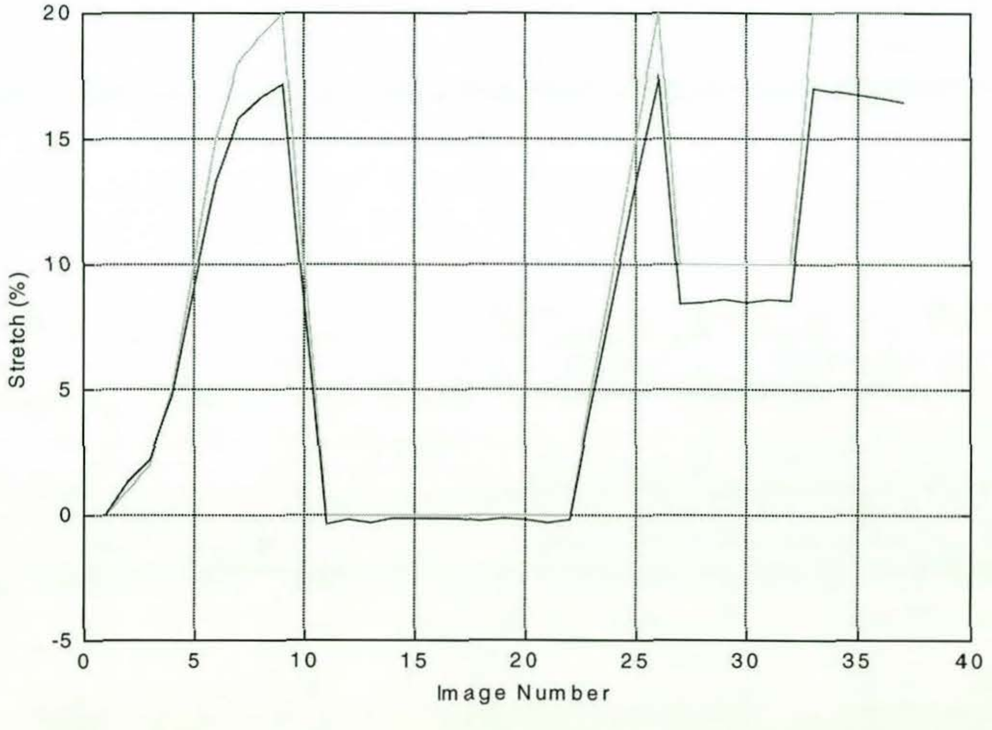


Figure 7.26 - Measured and Expected Stretch. Expected Result: —, Measured Result: —. All images compared with image 1.

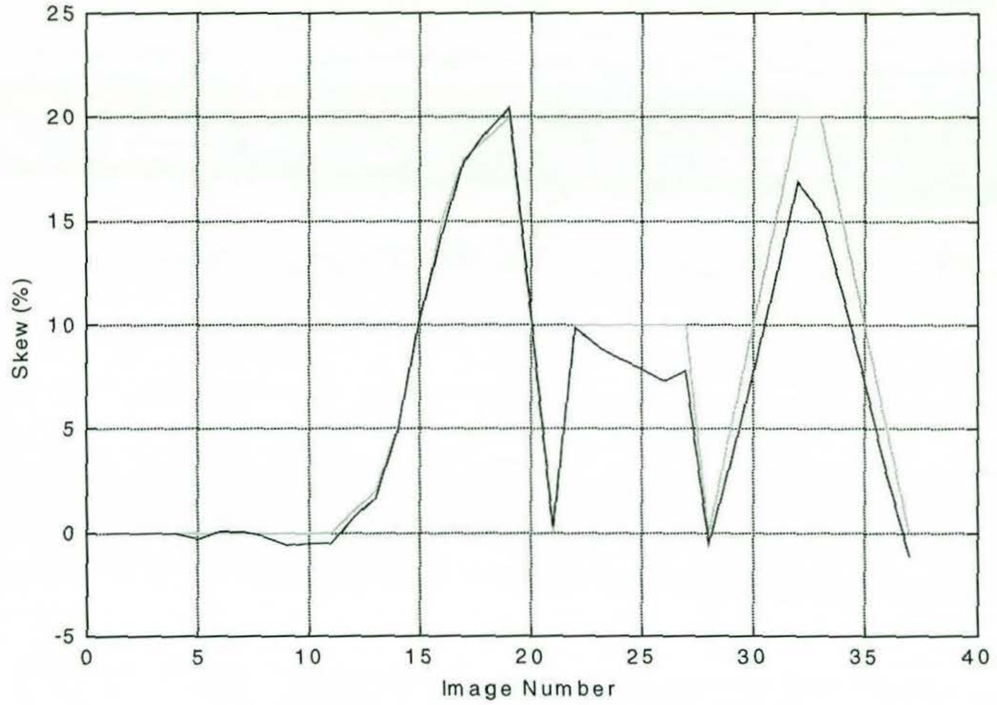


Figure 7.27 - Measured and Expected Skew. Expected Result: —, Measured Result: —. All images compared with image 1.

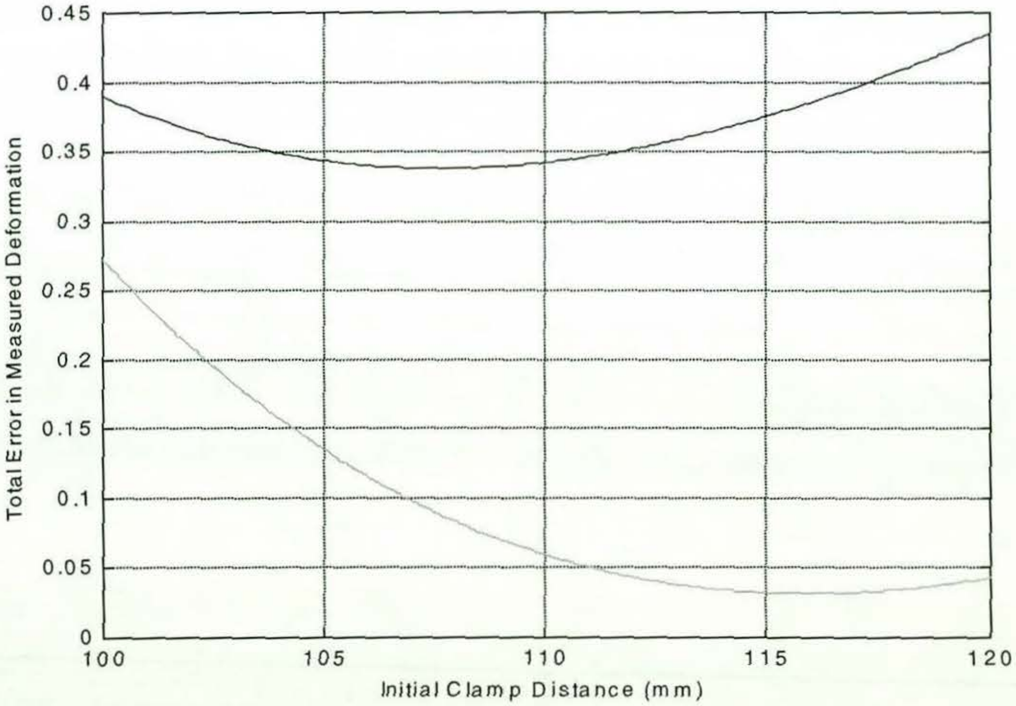


Figure 7.28 - Accuracy of Results When Value for Initial Clamp Distance is Varied. Skew: —, Stretch: —.

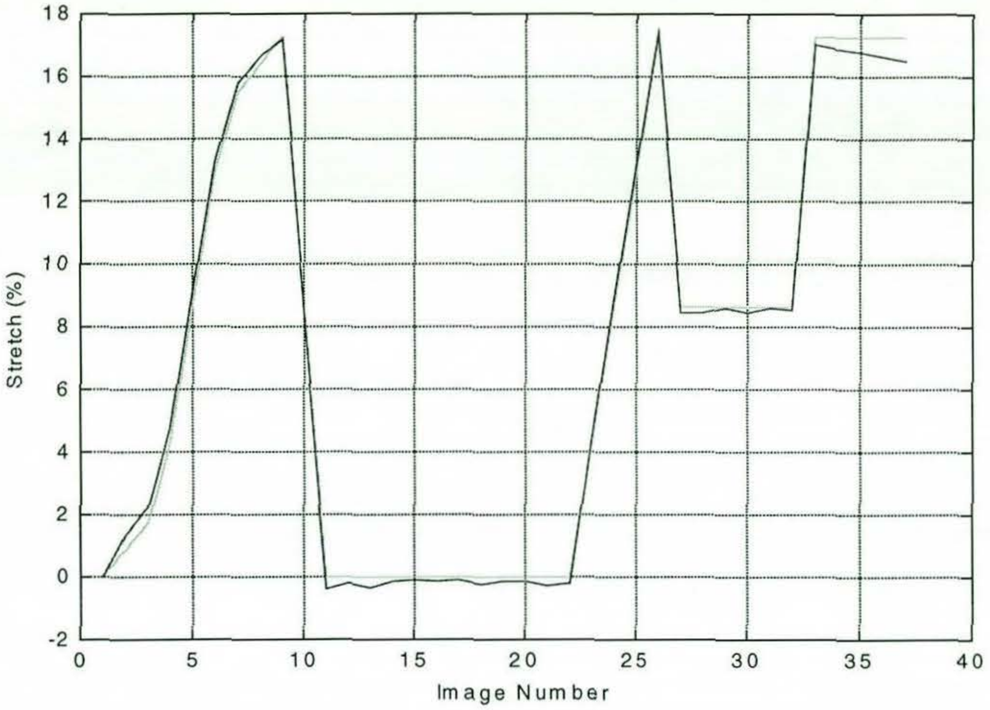


Figure 7.29 - Measured and Expected Skew, with Corrected Clamp Distance.
Expected Result: —, Measured Result: —. All images compared with image 1.

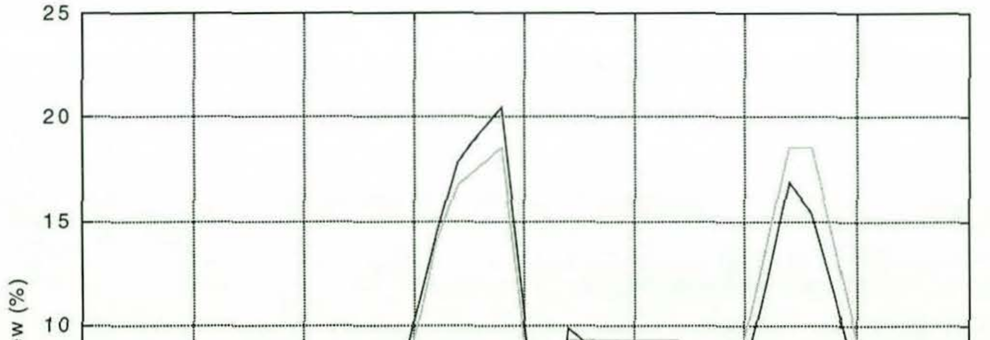


Figure 7.30 - Measured and Expected Stretch, with Corrected Clamp Distance.
Expected Result: —, Measured Result: —. All images compared with image 1.

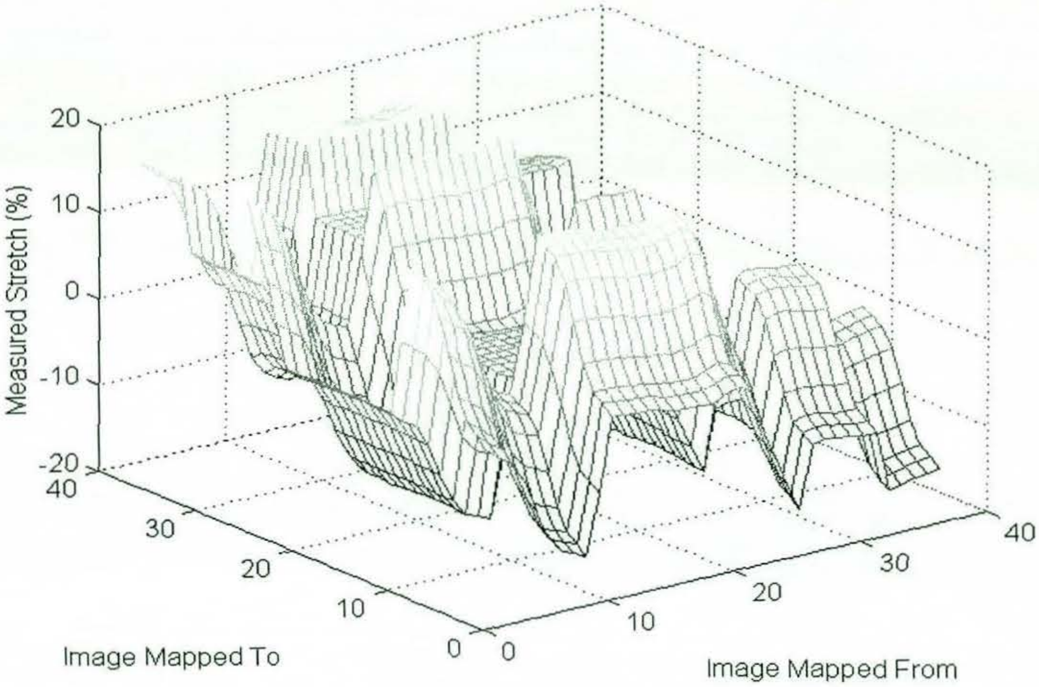


Figure 7.31 - Measured Stretch.

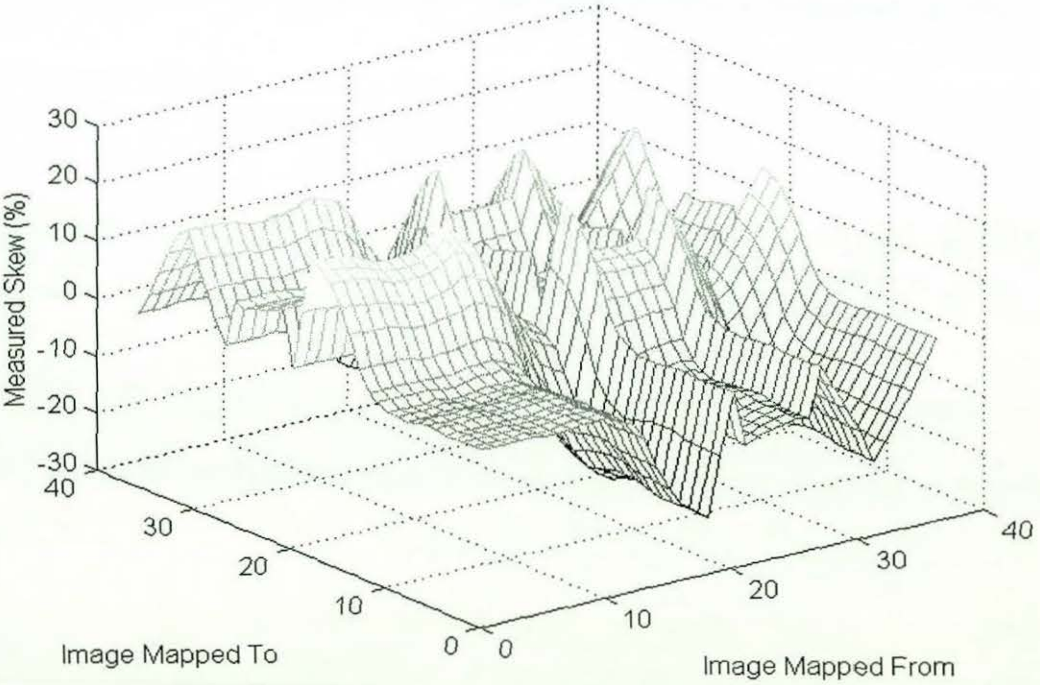


Figure 7.32 - Measured Skew.

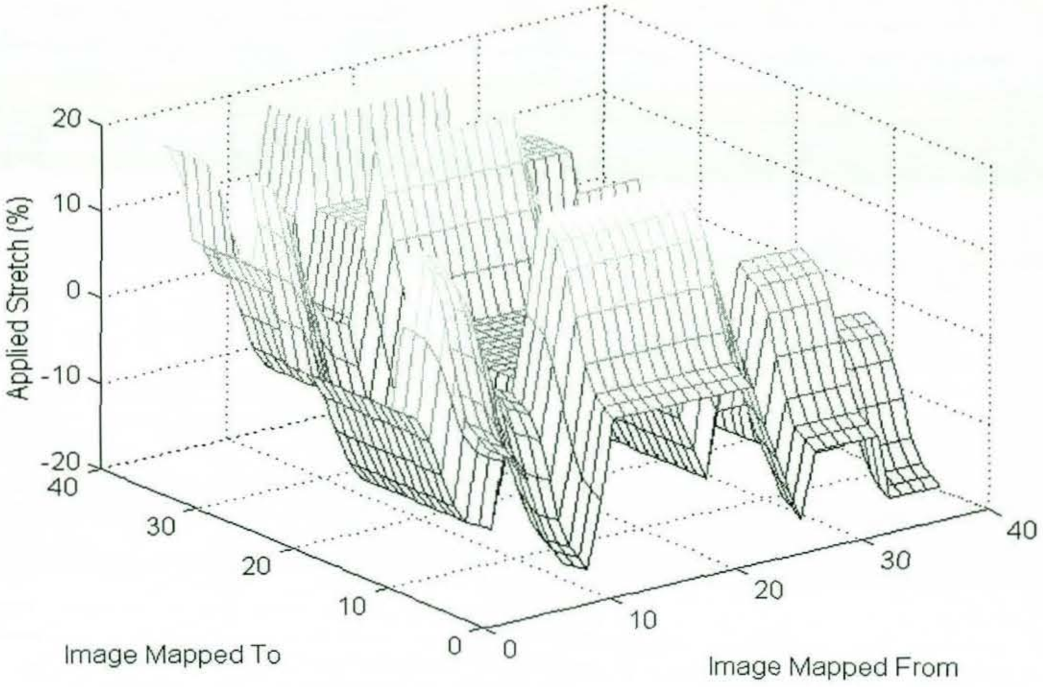


Figure 7.33 - Expected Stretch.

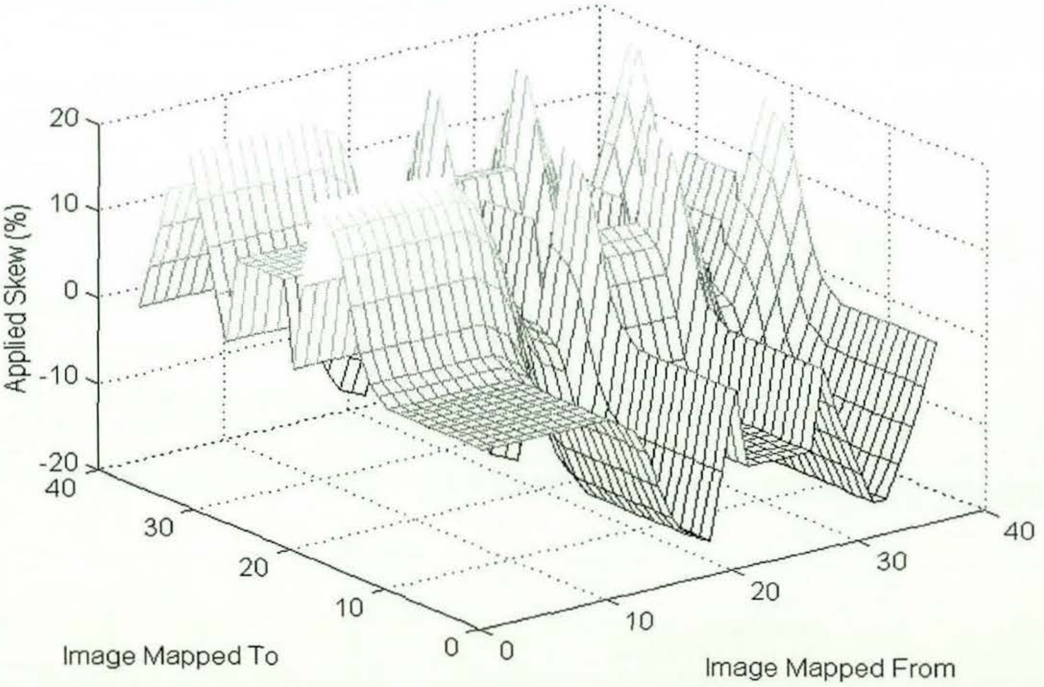


Figure 7.34 - Expected Skew.

Figure 7.35 gives details of the deformations applied to the 193 failed grids from the 120mm images taken with the aperture optimised for the 215mm image intensifier setting and with the fully normalised correlation method. For each failed grid the figure plots the expected value of the stretch against the expected value of the skew. It can be seen that the deformation measurement method is better at measuring stretched images than skewed images.

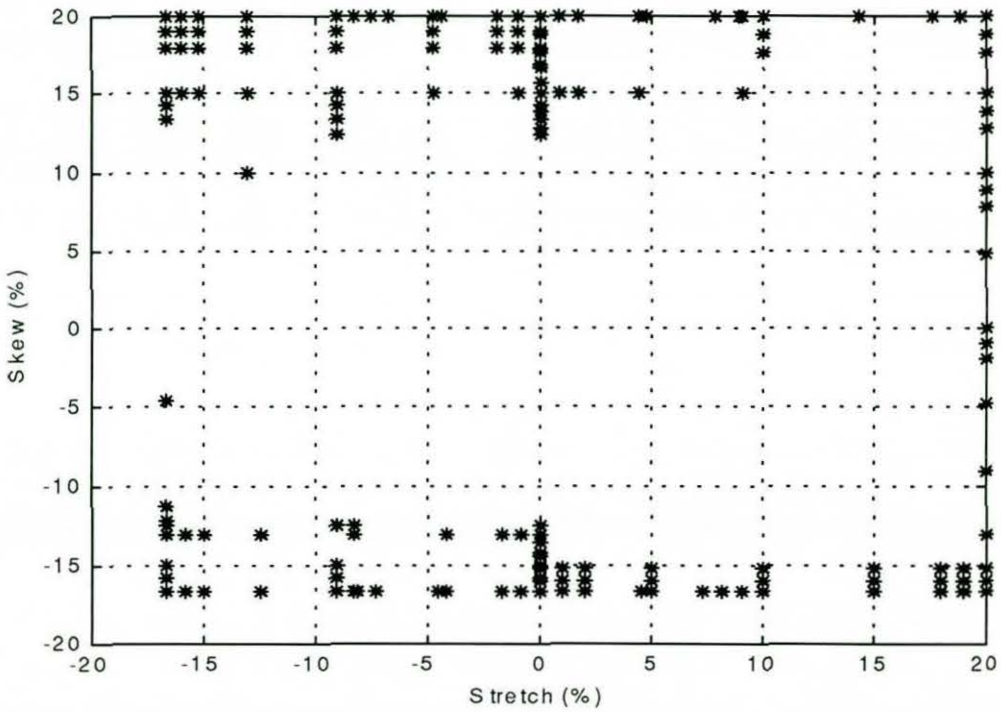


Figure 7.35 - Expected Stretch and Skew of Rejected Grids. 120mm images.

8 - Conclusions

This programme of research looked at the feasibility of performing sub-surface strain measurement using the Department of Mechanical Engineering's radioscopic equipment. The technique compares two images of a seeded sample to determine the change in deformation. Pattern matching methods were used to measure the movement of small windows in the images.

The radioscopic equipment consists of a low energy X-ray tube and an image intensifier. The image intensifier has three zoom settings, with the largest setting giving the brightest image. The output of the image intensifier is detected by a video camera fitted with a motorised zoom lens. Live images are displayed on a monitor located next to a 286 computer which is able to grab individual frames for further investigation.

The output energy of the X-ray tube is suitable for imaging composite materials or thin metal samples that have low absorbency coefficients in the 50keV to 90keV energy range. Brighter images were found to have better contrast. Unfortunately the camera's integration function was found to suffer from excessive noise. This meant there was a trade off between the aperture setting, and the X-ray energy. Wider aperture settings produced less detailed images, which meant the radioscopic images had to be taken with the higher X-ray energy levels. However the lower energy levels were noted to visualise more detail in some materials, particularly the plastic casing of the IQIs. By adjusting the zoom settings it is possible to provide a wide range of image sizes, from an image that includes the whole of the image intensifier output window to images with a definition almost matching the definition of the image intensifier.

An investigation into the image noise identified a number of sources. The camera and frame grabber were noted to be the cause of a number of different types of noise. These included additive noise in the camera and odd/even field variations for images taken with the integration feature. The additional noise in the integration function precluded its use to overcome the lack of brightness in the lower energy X-rays. The analogue to digital converter was found to have non linear step sizes, whilst the system was found to suffer from frequency dependent noise and the camera had an "inverse shadow" effect. Radioscopic systems are known sources of spatial errors. These errors are shading of the images further away from the centre, and pin cushion spatial distortion.

In order to make accurate deformation measurements, the spatial distortion has to be corrected. A literature search indicated the use of a distortion phantom would be the best way to measure the distortion. This was made of a sheet of aluminium with holes drilled in it, in a regular grid pattern. On inspection, the location of the holes were found to have measurable errors. Although distortion correction methods have in the past used precise locations of points on a phantom to measure the distortion, the errors were considered random enough for a global distortion measurement to reduce their effect.

None of the methods found in the literature search gave a good description of a suitable automatic method of identifying the features on the phantom. This meant a new method was required. This was designed so it could identify the holes in the phantom, irrespective of the zoom settings. The only manual input for the distortion correction technique is needed to ensure shading is properly corrected. The shading is corrected by combining the radioscopic image with a blank radioscopic image.

Twelve parameters were chosen for the mathematical modelling of the distortion. This was found to be sufficient to correctly model the majority of the distortion, although errors were noted towards the edges of the images. Since designing the correction model, it has been noted that "S" distortion was not of the type originally assumed [64]. A more accurate model would be produced by replacing the angular model parameters with parameters able to properly model the "S" distortion. In addition to the spatial noise, the frequency dependent noise has also been reduced by attenuating the appropriate frequencies.

Although a range of pattern matching methods have been used in the field of image processing, there have been few papers that compare the methods, and no comprehensive list of all the methods has yet been published. This thesis fills this gap by classifying the methods, and testing the robustness of a number of them using radioscopic images.

The pattern matching methods were classified in three ways. Firstly some methods pre-process the images. The most common method is to delete the mean value from each image. Secondly, each method used an equation to measure the match of every potential location. These equations were either correlation or difference equations. Finally, there are a number of different methods for finding the measure of the match. These included the location with the maximum difference between the two windows, or the sum of the measures across the window. Depending on the method, the location of the best match is taken as the location with the maximum or minimum value of the matching equation.

The pattern matching equations were tested to see which equation was able to measure the widest range of deformations of the target. Initially, two difference equations and three correlation equations were compared, although the absolute difference method was later discounted as it was unsuitable for implementation with Fast Fourier Transforms.

A suitable sample and a strain rig were required to test the robustness of the pattern matching techniques. A number of samples were X-rayed before settling on the rubber-tungsten sample sheet. The two properties required from the sample were that it would be flexible enough to test the capabilities of the pattern matching methods, and that it should have sufficient radioscopic contrast to allow matching. None of the earlier material samples were as flexible as the rubber sheet. The contrast in the earlier samples was variable. Ball bearings gave good contrast, but were considered too intrusive in the host material. Some of the samples made with plaster exhibited some contrast, either from air bubbles, or inhomogenities in the mix. The ability to visualise these features is potentially of use in a number of industries, for example food processing.

Tungsten particles were chosen as the seed material because of their high coefficient of absorption in the relevant X-ray energy levels, and their comparatively low cost when compared to the other elements with high absorption coefficients. In the radioscopic images of the 5mm thick rubber sheet they provide a good level contrast, sufficient to measure the deformation of the target. Consideration was given to the optimum seed density. As the system uses a zoom lens, this was only measured as the percentage area covered by the seeds. Tests suggested a 50% seed area would provide the maximum probability of a correct match, but a concentration this high would probably be too intrusive in the host material.

A strain rig was designed and built to apply stretch and skew to the rubber-tungsten sample. This was able to provide large deformations to the sample, but play was noted in the rig, which would make it difficult to accurately apply small deformations.

The best set of radioscopic images of the rubber-tungsten sample has a grey scale range of about a third of the full dynamic range, and a typical standard deviation in the region of 13 grey levels, whilst the noise level of the radioscopic images was previously found to be in the region of 2 grey levels. This indicates the major factor of the comparison of the pattern matching methods is their noise tolerance. A number of tests were made with the pattern matching methods. These were with both simulated images and real radioscopic images. The tests were designed to find the most accurate combination of pre-processing technique, pattern matching method and window size. The tests with simulated images found pre-processing was not required, whilst the results with the real images performed better with images with a mean value of zero, and a standard deviation of one. This indicates there are significant differences between the simulated images and the real images. The real images were noted to have variations in the mean levels. This was not modelled in the simulated images, which only included normally distributed noise across the images. A more accurate model, that included variations in the mean level, is likely to favour the use of the pre-processing methods.

The five pattern matching methods tested were found to provide similar levels of accuracy, and three methods were identified as the most accurate in different circumstances. The absolute difference method was not tested in every situation as it is unsuitable for implementation with Fast Fourier Transforms, making it more difficult to calculate. The other four methods were all obtained from the same set of calculations. The only differences between the methods were their suitability for use with some of the pre-processing techniques, and the effect of adding a constant to the shading correction image.

Although the other methods performed better in some circumstances, the fully normalised cross correlation method was found to be the most robust with the widest range of variations between the two images. This method is unaffected by pre-processing the image to normalise the mean and standard deviation. The other equations, which are affected by the pre-processing, can be calculated with a small amount of additional processing.

Unfortunately, the attempt to use the other methods to give a measure of confidence in the accuracy of the match, was a failure .

Of more significance to the probability of locating the correct match was the size and shape of the matching window. The simulated images were found to match with smaller windows than the real radioscopic images. The method used to optimise the window size found different window sizes for the different sets of images. Trials with these windows indicated the larger windows were more accurate. The method of optimising the window size is subjective. Each test of a possible window size took an hour of processing on a 333MHz Pentium computer. It is possible a better comparison of window sizes would be obtained by testing a wider range of potential windows with a quicker test. This could be achieved by comparing fewer pairs of images.

In these tests, the larger pattern matching windows were found to be more accurate. However, smaller windows allow greater precision of measurements. The larger windows were found necessary for the large quantities of deformation applied to the images to test the robustness of the pattern matching methods. Most strain measurements are with high tensile materials that will only exhibit micro-strains. As there would be less variation between images measuring micro-strain, these would be suitable for use with smaller windows. The tests on the pattern matching methods were to compare the robustness of the methods. It is assumed the more robust methods will also be more suitable for measurements in the order of micro-strains. However, the tests will also be of interest in other applications of pattern matching, for example object recognition and speech recognition.

The more reliable deformation measurements were obtained when using the larger windows and the higher contrast images. The techniques chosen to measure the deformations in the rubber-tungsten sample were found to be better at dealing with stretch in the sample than with skew. It is likely that a method with a narrower window would be more efficient at detecting skew. The results show the displacement at locations across the sample to the nearest pixel. A value of the deformation was found from these results, and was found to be close to the applied deformation.

This thesis successfully demonstrates a method of measuring deformation in suitable radioscopic images. The work is characterised by a need to make measurements on images with a very low signal to noise ratio. The method chosen was optimised for the radioscopic images being tested, and a similar method of optimisation of the technique would be required for different types of images, and deformation. To date, the work has only proved the feasibility of the method of measurement, and there is significant opportunity for refinement to ensure accurate strain measurements.

8.1 - Potential Improvements and Further Work

There is potential for further work to improve the present results, and to enable the method to be used to measure small scale strains. The work to date has been characterised by limitations caused by the imaging system. There is significant scope to improve the contrast in the images, and to reduce the level of noise. In particular, the camera, image grabber board and computer are components that should be considered for replacement.

A replacement camera could be specified with better sensitivity to lower light levels, and with a higher resolution image (1024 x 1024 pixel cameras are now widely available). Some of the noise sources could be reduced by using a different image grabber board. In addition a camera with gain and offset commands would improve low light level images. The most important feature at present not used is the camera's integration function. A new board would remove the problems with the variations between the odd and even fields. Alternatively, it may be possible to remove the odd/even field variations in software. The additive noise problem with longer integration times could be corrected. It is suggested a median filter could be applied to the locations where the additive noise is known to be a problem. The integration function would allow the use of the full range of X-ray energies.

Random noise can be reduced by combining several images. This was not implemented with the present equipment, principally because of the limitations on the memory and the speed of the controlling computer. Another noise source that could be removed is the inverse shadow.

There is potential for further improvement in the correction of the spatial errors. The method of shading correction may be improved by using a model, or by using a phantom that ensures the brightness across the image is constant. However, it is possible different phantoms would be required for different image intensifier zoom settings and X-ray energies, and the energy spectrum would vary across the image. The distortion correction model may also be improved after further study of the nature of the "S" distortion.

The deformation measurement technique has been shown to correctly match images with large variations in the deformation. For most practical strain applications, the method needs to be applied to smaller deformations. The present method of measuring deformations could be improved by incorporating some features of the Digital Image Correlation method [94]. This method uses interpolated images to give a more accurate location of match. Strains are found by matching images that are deformed mathematically. It is unlikely the previous research used images that suffered from the low levels of contrast, and high levels of noise found in the radioscopic images, but further development of the strain measurement method, combined with an improved imaging system should result in strain measurements of a similar resolution and accuracy to that found with Digital Image Correlation.

A radioscopic strain measurement system has many potential applications, particularly with composite materials. The materials could be manufactured with regions seeded with a material high absorbency to X-rays. These could be used as strain gauges, and would be measured at regular intervals using portable radioscopic equipment. The gauge could be located anywhere within the structure. Regular measurements of the gauge would be used to check the long term performance of the structure.

References

1. Halmshore, R.: “Industrial Radiology, Theory and Practice”, Applied Science Publishers, 1982.

2. Kerr, F. M.: “Simplified Radiography of Low Density Sheet Materials”, British Journal of NDT, Volume 31, Number 3, p131-133, March 1989.

3. Taylor, G. M.: “The 'Optimum' X-Ray Absorption Coefficient Myth”, Materials Evaluation, Volume 28, p244-245, 1970.

4. Garrett, J. A., Smithson, P. H.: “Conventional X-ray imaging”, IEE Proceedings, Volume 134, Part A, Number 2, p107-114, February 1987.

5. Moores, B. M.: “Digital X-ray imaging”, IEE Proceedings, Volume 134, Part A, Number 2, p115-125, February 1987.

6. Fujita, H., Giger, M. L., Doi, K.: “Investigation of basic imaging properties in digital radiography. 12. Effect of matrix configuration on spatial resolution”, Medical Physics, Volume 15, Number 3, p384-390, May/June 1988.

7. Khuri-Yakub, T.: “Ceramics: Non destructive Evaluation”, in: Encyclopaedia of Materials Science and Engineering, Volume 1, A-Co, p595-600, Pergamon Press, 1986.

8. Chan, H. P., Lam, K. L., Wu, Y.: “Studies of performance of antiscatter grids in digital radiography: Effect on signal-to-noise ratio”, Medical Physics, Volume 17, Number 4, p655-664, July/August 1990.

9. Tucker, D. M., Souto, M., Barnes, G. T.: “Scatter in Computed Radiography”, Radiology, Volume 188, Number 1, p271-274, July 1993.

10. Fassbender, R. H., Hagemaier, D. J.: “Low-Kilovoltage Radiography of Composites”, Materials Evaluation, Volume 41, p831-838, June 1983.

11. Clarke, E. T.: "Cobalt-60 Radiography of Concrete", Materials Evaluation, Volume 47, p1200-1203, October 1989.

12. Farman, A. G., Perez, C., Jacobson, A., Kelly, M. S., "Evaluation of aluminium-yttrium filtration for intraoral radiography", Oral Surgery, Oral Medicine and Oral Pathology, Volume 67, Number 2, p224-226, February 1989.

13. Tetradis, S., Scaf, G., Lurie, A. G., Freedman, M. L.: "Niobium filtration of conventional and high-frequency x-ray generator beams for intraoral radiography", Oral Surgery, Oral Medicine and Oral Pathology, Volume 80, Number 2, p232-241, August 1995.

14. Tait, W. H.: "Digital colour radiography", NDT&E International, Volume 26, Number 4, p171-176, 1993.

15. Zananiri, F. A. F., Speller, R. D.: "Dual energy radiography using X-ray films and intensifying screens", Journal of Biomedical Engineering, Volume 12, p239-247, May 1990.

16. Lahure, P.: "The International System (SI) of Units in Gamma-Radiographic Inspection", Materials Evaluation, Volume 43, p920-921, July 1985.

17. Miller, R. L. Jr.: "Gamma-Radiography: Technical Safety", Materials Evaluation, Volume 43, p806-811, June 1985.

18. "The Ionising Radiation Regulations 1985", Statutory Instruments 1333, HMSO 1985.

19. "The protection of persons against ionising radiation arising from any work activity", Health and Safety Commission Approved Code of Practice, L58, 1985.

20. "Local Rules for the Protection of Persons Exposed to Ionising Radiations", Loughborough University, January 1996.

21. Hesse, P. W., Criscuolo, E. L.: "Spatial Frequency Filtering of Radiographs", Materials Evaluation, Volume 31, p43-46, March 1973.

22. Miettunen, R. H., Korhola, O. A.: "The effect of digital unsharp-mask filtering on the signal-to-noise ratio in computed radiography", European Journal of Radiology, Volume 13, p225-228, 1991.

23. Hunt, B. R., Janney, D. H., Zeigler, R. K.: "Radiographic Image Enhancement by Digital Computers", Materials Evaluation, Volume 31, p1-5, January 1973.

24. Kehoe, A., Parker, G. A., LeBlanc, A.: "Image Processing for Automatic Defect Detection in Industrial Radiographic Images", Third International Conference on Image Processing And it's application, IEE conference publications, Number 307, p202-206, July 1989.

25. Gayer, A., Saya, A., Shiloh, A.: "Automatic recognition of welding defects in real-time radiography", NDT International, Volume 23, Number3, p131-136, June 1990.

26. Gonzalez, R. C., Wintz, P.: "Digital Image Processing", Second Edition, Addison Wesley, 1987.

27. Jain, A. K.: "Fundamentals of Digital Image Processing", Prentice Hall Information and System Sciences Series, 1989.

28. Jähne, B.: "Digital Image Processing, Concepts, Algorithms, and Scientific Applications", Third Edition, Springer-Verlag, 1995.

29. Boettinger, W. J., Burdette, H. E., Kuriyama, M.: "X-ray magnifier", Review of Scientific Instruments, Volume 50, Number 1, p26-30, January 1979.

30. Kuczumow, A., Larsson, S.: "Scheme for x-ray tracing in capillary optics", Applied Optics, Volume 33, Number 34, p7928-7932, 1 December 1994.

31. Macherauch, E.: "X ray Stress Analysis", Experimental Mechanics, Volume 6, p140-153, March 1966.

32. Norton, J. T.: "X-Ray Determination of Residual Stress", Materials Evaluation, Volume 31, pA21-A41, February 1973.

33. Webster, P. J.: "Strain Scanning Using X-Rays and Neutrons", Materials Science Forum, Volumes 228-231, p191-200, 1996.

34. Moore, M. G., Evans, W. P.: "Mathematical Correction for Stress in Removed Layers in X-Ray Diffraction Residual Stress Analysis", SAE Transactions, Volume 66, p340-345, 1958.

35. Duh, J. G., Wu, Y. S.: "Residual stress effects and t→m transformation in ion-implanted yttria-stabilized zirconia", Journal of Materials Science, Volume 26, p6522-6526, 1991.

36. Wright, P. K.: "Measurement of Residual Stresses in Metal Matrix Composites", Journal of Engineering for Gas Turbines and Power, Volume 116 p605-610, July 1994.

37. Nishida, M., Hanabusa, T., Fujiwara, H.: "X-Ray residual stress measurement of laminated coating layers produced by plasma spraying", Surface and Coatings Technology, Volume 61, p47-51, 1993.

38. Kupperman, D. S., Majumdar, S., Singh, J. P.: "Neutron Diffraction NDE for Advanced Composites", Journal of Engineering Materials and Technology, Volume 112, p198-201, April 1990.

39. Majumdar, S., Singh, J. P., Kupperman, D., Krawitz, A. D.: "Application of Neutron Diffraction to Measure Residual Strains in Various Engineering Composite Materials", Journal of Engineering Materials and Technology, Volume 113, p51-59, January 1991.

40. Ceretti, M., Michaud, H., Perrin, M., Lodini, A.: "Residual Stress Measurement in a Plasma Semi- Transferred Arc (PTA) Coating by Neutron and X-Ray Diffraction", Experimental Techniques, Volume 19, p17-21, May/June 1995.

41. "Inspection of Composite Structures", Aerospace Engineering, Volume 12 Number 5, p9-13, 1992.

42. "New Composite Materials Demand Changes in Testing Strategies", Materials Evaluation, Volume 44, p53-54, January 1986.

43. Moore, L., Kline, R., Madaras, E., Ransone, P.: " Nondestructive Evaluation of Changes in Mechanical Properties in Carbon-Carbon Composites during Processing", Journal of Composite Materials, Volume 28, Number 4, p352-366, 1994.

44. Navarre, G., Rouais, J. C., Rouby, D.: "Observation of crack path in an SiC-SiC fibre composite by X-ray radiography and SEM", Journal of Materials Science Letters, Volume 9, p636-638, 1990.

45. Kortschot, M. T., Zhang, C. J.: "Charaterization of Composite Mesostructures and Damage by De-ply Radiography", Composites Science and Technology, Volume 53, p175-181, 1995.

46. Becker, G. L.: "A Brief Look at the Low-energy Radiography of Composite Materials", Materials Evaluation, Volume 43, p596-598, May 1985.

47. Crane, R. L., Chang, F., Allinikov, S.: "The Use of Radiographically Opaque Fibers to Aid the Inspection of Composites", Materials Evaluation, Volume 36, p69-71, September 1978.

48. Lahure, P.: "Determination of Thickness Variations by Measuring Film Densities on Two Adjacent Areas of a Radiograph", Materials Evaluation, Volume 43, p354-356, March 1985.

49. Martin, B. G.: "An Analysis of Radiographic Techniques for Measuring Resin Content in Graphite Fiber Reinforced Epoxy Resin Composites", Materials Evaluation Volume 35, p65-68, September 1977.

50. Crane, R. L.: "Measurement of Composite Ply Orientation Using a Radiographic Fringe Technique", Materials Evaluation, Volume 34, p79-80, April 1976.

51. Gayer, A., Saya, A.: “The use of X-radiography and computer software for detecting defects during the manufacture of steel-belt tyres”, NDT International, Volume 21, Number5, p333-336, October 1988.

52. “X-ray Tire Monitoring System”, Materials Evaluation, Volume 42, p42, January 1984.

53. Sekine, I., Fujinawa, M.: “Exploratory Tests of Corrosion of Reinforcing Steel in Concrete by X-radiography” Materials Evaluation, Volume 42, p121-126, January 1984.

54. Jordan, E. H., Ochi, S. C. U., Pease. D., Budnick, J. I.: “Microradiographic Strain Measurement Using Markers”, Experimental Mechanics, Volume 34, p155-165, June 1994.

55. Bay, B. K.: “Texture Correlation: A method for the Measurement of Detailed Strain Distributions Within Trabecular Bone”, Journal of Orthopaedic Research, Volume 13, Number 2, p258-267, 1995.

56. Cosslett, V. E., Nixon, W. C.: “X-Ray Microscopy”, Cambridge University Press, 1960.

57. Sharpe, R. S.: “Projection Microradiography”, Journal of Microscopy, Volume 117, Part 1, p123-143, September 1979.

58. Berger, H., Kupperman, D.: “Microradiography to Characterize Structural Ceramics”, Material Evaluation, Volume 43, p201-205, February 1985.

59. Mondy, L. A., Graham, A. L., Bryant, L. E. Jr., Lucero, J. P.: “Real-Time Radiography and Particle Dynamics in Optically Opaque Suspensions”, Material Evaluation, Volume 44, p120-124, January 1986.

60. Rikard, D.: “Determining Internal Component Displacement by Using a Radiographic Reference”, Materials Evaluation, Volume 45, p788-791, July 1987.

61. Hartmann, F.: “X-Ray Mapping of Flaws by Computer Graphics”, Materials Evaluation, Volume 27, p169-179, August 1969.

62. Hasenkamp, F. A.: "Radiographic Laminography", Materials Evaluation, Volume 32, p169-174, August 1974.

63. Rudin, S., Bednarek, D. R., Wong, R.: "Accurate characterization of image intensifier distortion", Medical Physics, Volume 18, Number 6, p1145-1151, November/December 1991.

64. Solzbach, U., Wollschläger, H., Zeiher, A., Just, H., "Optical Distortion due to Geomagnetism in Quantitative Angiography", Computers in Cardiology Conference p355-357, 25-28 September 1988.

65. Casperson, L. W., Spiegler, P., Grollman J. H. Jr.: "Characterization of aberrations in image-intensified fluoroscopy", Medical Physics, Volume 3, Number 2, p103-106, March/April 1976.

66. Baltzopoulos, V.: "A videofluoroscopy method for optical distortion correction and measurement of knee-joint kinematics", Clinical Biomechanics, Volume 10, Number 2, p85-92, 1995.

67. Ning, R., Riek, J. K., Conover, D. L.: "An Image Intensifier-Based Volume Tomographic Angiagraphy Imaging System: Geometric Distortion Correction", SPIE Physics of Medical Imaging, Volume 2163, p199-210, 1994.

68. Pietka, E., Huang, H. K.: "Correction of Aberration in Image-Intensifier Systems", Computerised Medical Imaging and Graphics, Volume 16, Number 4, p253-258, 1992.

69. Chakraborty, D. P.: "Image intensifier distortion correction", Medical Physics, Volume 14, Number 2, p249-252, March/April 1987.

70. Boone, J. M., Seibert, J. A., Barrett W. A., Blood, E. A.: "Analysis and correction of imperfections in the image intensifier-TV-digitizer, imaging chain", Medical Physics, Volume 18, Number 2, p236-242, March/April 1991.

71. Tehrani, S., LeFree, M. T., Sitomer, J., Bourdillon, P. D. V.: "High-Speed Digital Radiographic Pincushion Distortion Correction Using an Array Processor",

Computers in Cardiology Conference, p618-618, Boston, MA, USA, 7-10 October 1986.

72. Butler, D. A., Pierson, P. K.: "A Distortion-Correction Scheme for Industrial Machine-Vision Applications", IEEE Transactions on Robotics and Automation, Volume 7, Number 4, p546-551, August 1991.

73. Dijak, J. T.: "A Method of Correcting Geometric Distortion in Video Cameras", Proceedings of the IEEE International Aerospace and Electronics Conference, Volumes 1&2, p1382-1388, 1985.

74. Bernstein, R.: "Digital Image Processing of Earth Observation Sensor Data", IBM Journal Research and Development, Volume 20, p40-57, January 1976.

75. Green, W. B., Jepsen, P. L., Kreznar, J. E., Ruiz, R. M., Schwartz, A. A., Seidman, J. B.: "Removal of Instrument Signature From Mariner 9 Television Images of Mars", Applied Optics, Volume 14, Number 1, p105-114, January 1975.

76. Harris, J. S., Harding, K. G., Mersch, S. H.: "Techniques for evaluation of aircraft windscreen optical distortion", Optical Engineering, Volume 20, Number 1, p115-122, January/February 1981.

77. Boult, T. E., Wolberg, G.: "Local Image Reconstruction and Subpixel Restoration Algorithms", CVGIP: Graphical Models and Image Processing, Volume 55, Number 1, p63-77, January 1993.

78. Gottesfeld Brown, L.: "A Survey of Image Registration Techniques", ACM Computing Surveys, Volume 24, Number 24, p325-376, December 1992.

79. Svedlow, M., McGillem, C. D., Anuta, P. E.: "Image Registration: Similarity Measure and Preprocessing Methods Comparisons", IEEE Transactions on Aerospace and Electronic Systems, Volume 14, Number 1, p141-150, January 1978.

80. Pratt, W. K.: "Digital Image Processing", Second Edition, Wiley Interscience Publication, 1991.

81. De Castro, E., Morandi, C.: "Registration of Translated and Rotated Images Using Finite Fourier Transforms", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 9, Number 5, p700-703, September 1987.

82. Barnea, D. I., Silverman, H. F.: "A class of Algorithms for Fast Digital Image Registration", IEEE Transactions on Computers, Volume 21, Number.2, p179-186, February 1972.

83. Anuta, P. E.: "Digital Registration of Multspectral Video Imagery", SPIE Journal, Volume 7, p168-175, September 1969.

84. Cooley, J. W., Tukey, J. W.: "An Algorithm for the Machine Calculation of Complex Fourier Series", Mathematics of Computation, Volume 19, p297-301, 1965.

85. Keane, R. D., Adrian, R. J.: "Theory of cross-correlation analysis of PIV images", Aplied Scientific Research, Volume 49, Number 3, p191-215, 1992.

86. Duda, R. O., Hart, E.: "Pattern classification and scene analysis" (John Wiley and Sons, 1973).

87. Pratt, W. K.: "Correlation Techniques of Image Registration", IEEE Transactions on Aerospace and Electronic Systems, Volume 10, Number 3, p353-358, May 1974.

88. Mostafavi, H., Smith F. W.: "Image Correlation with Geometric Distortion Part I: Acquisition Performance", IEEE Transactions on Aerospace and Electronic Systems, Volume 14, Number 3, p487-493, May 1978.

89. Mostafavi, H.: "Optimal Window Functions for Image Correlation in the Presence of Geometric Distortion", IEEE Transactions on Acoustics, Speech and Signal ProcessingVolume 27, Number 2, p163-169, April 1979.

90. Aggarwal, J. K., Davis, L. S., Martin, W. N.: "Correspondence processes in dynamic scene analysis", Proceedings of the IEEE, Volume 69 Number 5 p562-572, May 1981.

91. Bell, W. B., Devarajan, V.: "Image Correlation Under Full-Perspective Distortion", Proceedings of the SPIE, Volume 2753, p38-49, 1996.

92. Peters, W. H., Ranson, W. F.: "Digital imaging techniques in experimental stress analysis", Optical Engineering, Volume 21, Number 3, p427-431, May/June 1982.

93. Peters, W. H., Ranson, W. F., Sutton, M. A., Chu, T. C., Anderson, J.: "Application of digital correlation methods to rigid body mechanics", Optical Engineering, Volume 22, Number 6, p738-742, November/December 1983.

94. Sutton, M. A., Wolters, W. J., Peters, W. H., Ranson, W. F., McNeill, S. R.: "Determination of displacements using an improved digital correlation method", Image and Vision Computing, Volume 1, Number 3, p133-139, August 1983.

95. Ranson, W. F., Sutton, M. A., Peters, W. H.: "Digital image correlation of white light speckle Including the effects of image distortion", SPIE International conference on Speckle, Volume 556, p160-167, 1985.

96. Chu, T. C., Ranson, W. F., Sutton, M. A., Peters, W. H.: "Applications of Digital-Image-Correlation Techniques to Experimental Mechanics", Experimental Mechanics, Volume 25, Number 3, p232-244, September 1985.

97. McNeill, S. R., Peters, W. H., Sutton, M. A.: "Estimation of Stress Intensity Factor by Digital Image Correlation", Engineering Fracture Mechanics, Volume 28, Number 1, p101-112, 1987.

98. Sutton, M. A., Cheng, M., Peters, W. H., Chao, Y. J., McNeill S. R.: "Application of an optimized digital correlation method to planear deformation analysis", Image and Vision Computing, Volume 4, Number 3, p143-150, August 1986.

99. Sutton, M. A., McNeill, S. R., Jang. J, Babai, M.: "Effects of sub-pixel image restoration on digital correlation error estimates", Optical Engineering, Volume 27, Number 10, p870-877, October 1988.

100. Bruck, H. A., McNeill, S. A., Sutton, M. A., Peters III, W. H.: "Digital Image Correlation Using Newton-Raphson Method of Partial Differential Correlation", Experimental Mechanics, Volume 28, p261-267, September 1989.

101. Sutton, M. A., Turner, J. L., Bruck, H. A., Chae, T. A.: "Full-field Representation of Discretely Sampled Surface Deformation and Strain Analysis", Experimental Mechanics, Volume 30, p168-177, June 1991.

102. Sutton, M. A., Chao, Y. J.: "Measurement of Strains in a paper tensile specimen using computer vision and digital image correlation, Part 1: Data acquisition and image analysis system", Tappi Journal, Volume 71, Number 3, p173-175, March 1988.

103. Russell, S. S., Sutton, M. A., Chen, H. S.: "Image Correlation Quantitative Nondestructive Evaluation of Image and Fabrication Damage in a Glass Fiber-Reinforced Composite System", Materials Evaluation, Volume 47, p550-557, May 1989.

104. Lyons, J. S., Liu, J., Sutton, M. A.: "High-temperature Deformation Measurements Using Digital-image Correlation", Experimental Mechanics, p64-70, March 1996.

105. Sun, Z., Lyons, J. S., McNeill, S. R.: "Measuring Microscopic Deformations with Digital Image Correlation", Optics and Lasers in Engineering, Volume 27, p409-428, 1997.

106. Mguil-Touchal, S., Morestin, F., Brunet, M., "Various Experimental Applications of Digital Image Correlation Method", Computer Methods and Experimental Measurements, p45-58, 1997.

107. Yamaguchi, I., Fujita, T.: "Linear and rotary encoders using electronic speckle correlation", Optical Engineering, Volume 30, Number 12, p1862-1868, December 1991.

108. During, C.: "Real Time Optical Measurement of Position", Mechatronics, Volume 4, Number 2 p125-138, 1994.

109. During, C.: "Recursive versus Nonrecursive Correlation for Real-Time Peak Detection and Tracking", IEEE Transactions on Signal Processing, Volume 45, Number 3, p781-785, March 1997.

110. Szajnowski, W. J.: "Efficient estimation of a normalised correlation function from discrete-time samples", Electronics Letters, Volume 29, Number 16, p1399-1401, 5th August 1993.

111. Lu, H., Vendroux, G., Knauss, W. G.: "Surface Deformation Measurements of a Cylindrical Specimen by Digital Image Correlation", Experimental Mechanics, Volume 37, Number 4, p433-439, December 1997.

112. Vendroux, G., Knauss, W. G., "Submicron Deformation Field Measurements: Part 2. Improved Digital Image Correlation", Experimental Mechanics, Volume 38, Number 2, p86-92, June 1998.

113. Bar-Gill, A., Ben-Ezra, P., Bar-Itzhack, I. Y., "Improvement of Terrain-Aided Navigation via Trajectory Optimization", IEEE Transactions on Control Systems Technology, Volume 2, Number 4, p336-342, December 1994.

114. Scambos, T. A., Dutkiewicz, M. J., Wilson, J. C., Bindschadler, R. A.: "Application of Image Cross-Correlation to the Measurement of Glacier Velocity Using Satellite Image Data", Remote Sensing of the Environment, Volume 42, Number 3, p177-186, 1992.

115. Ching, W. S.: "Normalized cross-correlation: the contrast-dependent problem and its solution", Journal of Electronic Imaging, 1995, Volume 4, Number 3, p278-282, July 1995.

116. Franke, E. A., Wenzel, D. J., Davidson, D. L.: "Measurement of microdisplacements by machine vision photogrammetry (DISMAP)", Review of Scientific Instruments, Volume 62, Number5, p1270-1279, 1991.

117. Choi, S., Shah, S. P.: "Measurement of Deformations on Concrete Subjected to Compression Using Image Correlation", Experimental Mechanics, Volume 37, Number 3, p307-313, September 1997.

118. Hata, S., Hanafusa, H.: "Precise Pattern Inspection System for Printed Sheets", 5th UK Mechatronics Forum International Conference/3rd International Conference on

Mechatronics and Machine Vision in Practice, Guimarães, Portugal, Volume 2, p2-405 to p2-410, 18-20th September 1996.

119. Willert, C. E., Gharib, M.: "Digital particle image velocimetry", Experiments in Fluids, Volume 10, p181-193, 1991.

120. Gui, L., Merzkirch, W.: "Generating arbitrarily sized integration windows for correlation-based analysis of particle image velocimetry recordings", Experiments in Fluids, Volume 24, p66-69, 1998.

121. Mizuno, Y., Kawasaki, A., Watanabe, R.: "In situ measurement of sintering shrinkage by digital image correlation", Powder Metrology, Volume 38, Number 3, p191-195, 1995.

122. Septanika, E. G., Ernst, L. J., van den Hooff, L. -A. C. M.: "An Automatic and Interactive Large-deformation Measurement System Based on Image Processing", Experimental Mechanics, Volume 38, Number 3, p181-188, September 1998.

123. Vincent, A.: "X-Ray Imaging: A Feasibility Study", Final Year Project, Department of Mechanical Engineering, Loughborough University, 1993.

124. "BS EN 462-1: 1994, Non-destructive testing- Image quality of radiographs, Part 1. Image quality indicators (wire type) - Determination of image quality value" ", British Standard Institute, 1994.

125. "BS EN 462-2: 1994, Non-destructive testing- Image quality of radiographs, Part 1. Image quality indicators (step/hole type) - Determination of image quality value", British Standard Institute, 1994.

126. Bajpai, A. C., Mustoe, L. R., Walker, D.: "Engineering Mathematics", John Wiley & Sons, 1974 (p711).

127. Cooling, J. E.: "Real-Time Interfacing", Van Nostrand Reinhold, 1986 (p52).

128. Dow Corning Data Sheet 61-0045G-01, 1994.

129. Dow Corning Data Sheet 61-0113G-01, 1994.

130. Dow Corning Data Sheet 61-0114D-01, 1994.

131. Web page: "http://www.csrri.iit.edu/periodic-table.html" - based on data from W. H. McMaster et. al.

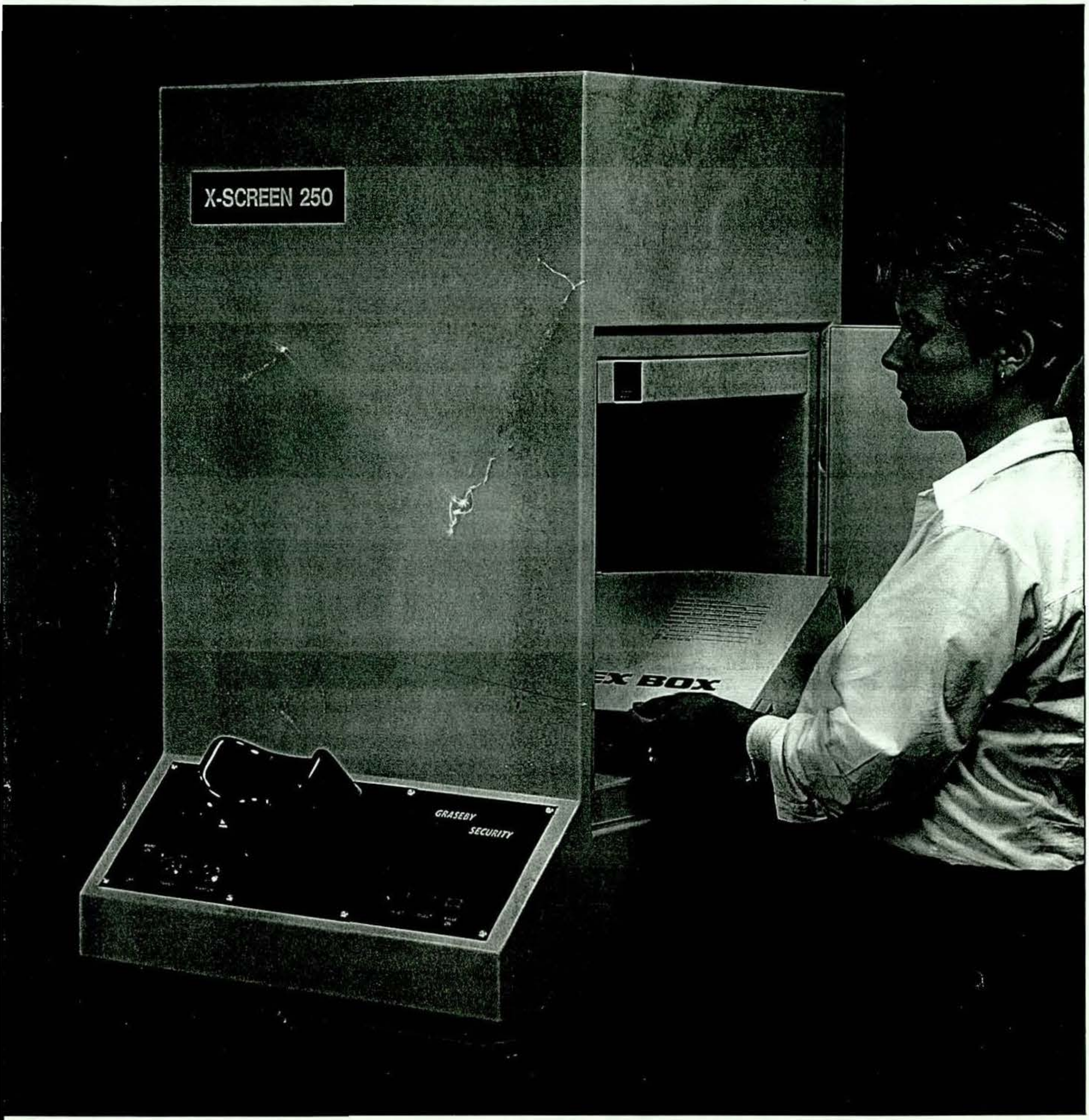
132. Ouchi, S. C. U., Accorsi, M. L., Jordan, E. H., "Influence of Embedded Markers on the Response of the Host Material", Journal of Testing and Evaluation, Volume 23, Part 3, p196-202, 1995.

Appendix I - Hardware Details

Appendix I.1 - Graseby X-Screen 250 System Data Sheet

RASEBY
SECURITY

- Screen 250 System



- Microprocessor controlled
- Constantly monitored systems and interlocks
- Integral backlit 16 character 2 line LCD display
- Larger chamber than equipment of comparable size
- Designed and manufactured in the UK by Graseby

Resulting from continuous development and the introduction of the latest techniques, Graseby Security offer a desk top, microprocessor controlled X-Ray inspection system. The equipment is designed and manufactured in the UK and fully complies with the National Radiological Boards stringent requirements.

The X-Screen 250 is designed for the rapid detection of contraband materials in embassies, prisons, court rooms, post offices, in fact anywhere that suspect material must be detected and isolated. The equipment, which operates with the simplicity of a small office copier has one of the largest inspection chambers against equipment of comparable size, easily accepting a large sized briefcase.

ntrol System

e unit is controlled by an embedded Z80 processor. All the or functions are constantly monitored and displayed on an gral backlit 16 character 2 line display.

e functions monitored include

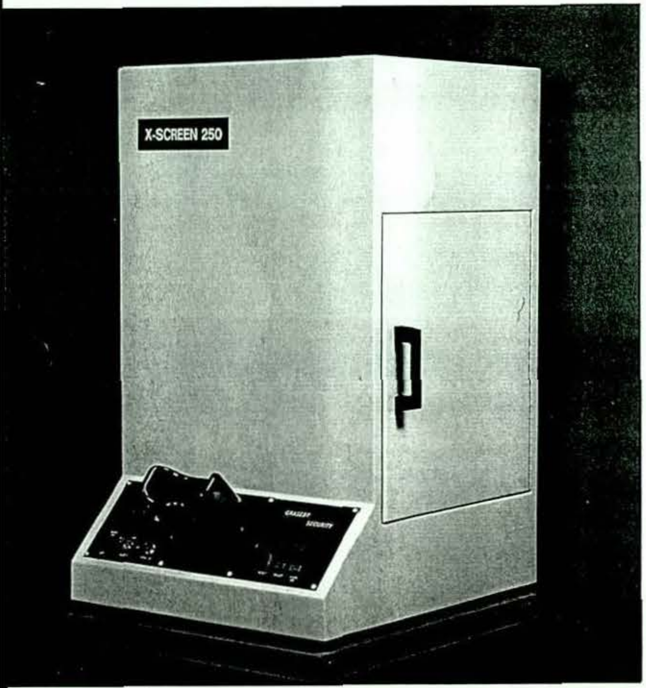
erlocks	X-Ray Set
or Safety Interlock	X-Ray tube temperature
ernal Panel Safety Interlock	X-Ray duty cycle
Interlock	X-Ray tube overcurrent

ewing Methods

e X-Screen 250 Scanner is available in two options:

. with fluorescent screen for direct viewing
. with video monitor – (for applications where more than one individual needs to view the image.

oth cases, the system can be operated in an automatic mode ch provides an exposure sequence of 1 - 10 minutes, allowing a based radiograph to be taken of the subject under inspection. intensity control permits the image brightness to be varied.



General Data*

Input Voltage Requirements
110/130 or 230/250 volts 50-60Hz (Please specify)

Current
110V 6A
250V 3A

X-Ray Tube Capability
Variable 50 to 90KV 4mA

Viewing Methods
Fluoroscopic (direct viewing) or video monitor

Dimensions
1000mm x 610mm x 690mm

Chamber Size
424mm × 560mm × 440mm

Part Number	Description
0488-0200	X-Screen 250 X-Ray System
0488-0201	X-Screen 250 X-Ray System with Video Monitor

* Detailed specification supplied on request.

Graseby Security products are the subject of continuous development and improvement and consequently may differ in detail from this specification.

GRASEBY SECURITY

GRASEBY SECURITY, ODHAMS TRADING ESTATE, ST ALBANS ROAD, WATFORD, HERTS, UK, WD2 5JX TEL: 0923 816766 FAX: 0923 816283

A DIVISION OF GRASEBY DYNAMICS LIMITED

Appendix I.2 - Image Intensifier Data Sheet

THOMSON TUBES ELECTRONIQUES

TH 9428 HP2

X-ray image intensifier tube triple field, 9"

□ **Hi-Res input phosphor screen** with improved resolution and DQE (significant reduction in x-ray dose possible).

□ **Ultra-compact output phosphor screen** with high luminous efficiency, unsurpassed resolution and virtually no structural noise.

□ **All metal technology** with excellent x-ray transparency and negligible x-ray scatter.

Applications: For medical radiology, these improvements provide outstanding performances in DQE and resolution and make the TH 9428 HP2 particularly well adapted to **digitalized x-ray imaging**:

- wide dynamic operating range with incident dose rate ;
- pulsed operation ;
- fast time characteristics and excellent linearity.

Use of the TH 9428 HP2 in **industrial radiology** provides a better detection efficiency with the high-energy x-ray sources commonly used in non-destructive testing. Its technology improves detection of small details by substantially reducing noise.

MAIN CHARACTERISTICS

Nominal entrance field size	230	mm
Useful entrance field size	215	mm
Output image field size	20	mm
Center resolution	48	pl/cm
Small detail contrast ratio (10 mm)	16:1	
Large area contrast ratio (10 %)	23:1	
DQE (IEC standard in preparation)	65	%
DQE (17 Hz rms bandwidth)	95	%
Conversion factor	240	cd.m^{-2}/mR.s^{-1}

This data sheet cannot be considered to be a contractual specification. The information given herein may be modified without notice due to product improvement or further development. Consult Thomson Tubes Electroniques before making use of this information for equipment design.

GENERAL CHARACTERISTICS

Mechanical

TH 9428 HP2 can be supplied in the following versions:

With magnetic shielding only the tube is lined with a high permeability mu-metal alloy for efficient screening against the earth's magnetic field. An aluminium output flange provides the mechanical interface with the rest of the equipment and allows the optical system to be accurately positioned with respect to the image plane.

- Approximate weight: 9 kg

With magnetic and lead shielding in a painted housing (with or without power supply), the lead shielding provides effective protection against x-rays in compliance with DHHS regulations. The outer part of the housing consists of a ruggedized envelope such that the tube may be fastened either at the input flange or the output flange.

- Approximate weight: 15 kg
- Maximum permissible load: 20 kg on the output flange ;
  2 kg on the input flange

With additional side flange, identical to the version above, but includes an additional side flange for lateral fastening of the tube.

- Approximate weight: 16 kg
- Maximum permissible load: 18 kg on the output flange ;
  2 kg on the input flange

With reinforced side flange, to be used when very heavy optical systems are to be mounted directly on the output flange.

- Approximate weight: 18 kg
- Maximum permissible load: 30 kg on the output flange ;
  2 kg on the input flange

Power supply

To take full advantage of the high performance capabilities of the TH 9428 HP2, the electrode voltages must be very stable and well smoothed. In particular, the ripple voltages must not exceed 0.3 %.

Thomson has developed a range of highly reliable and compact high-voltage supplies for optimum tube performances in the continuous and pulsed modes of operation. The input voltage options for these high-voltage power supplies are 24 Vdc, 110 Vac or 220 Vac.

Dynamic ion-getter pump

All models have a built-in, high-capacity dynamic ion-getter pump. Unlike static getter devices, a dynamic ion-getter pump keeps its full pumping capacity and ensures the highest vacuum throughout tube life. The operation of the ion-getter pump is fully automatic.

TYPICAL ELECTRO-OPTICAL CHARACTERISTICS

Nominal entrance field size (1)	230	mm
Output image field size	20	mm
Output window thickness	3.6	mm
Output window refractive index	1.5	
Input phosphor:	Hi-Res CsI layer	
Output phosphor:	P20 (peak wavelength between 520 and 540 nm)	
Fluorescence and phosphorescence	yellow/green	

Operating mode	**Normal**	**Zoom 1**	**Zoom 2**	
Useful entrance field size (2)	215	160	120	mm
DQE at 59.5 keV (3)			see figure 3	
Conversion factor (4)	240	120	60	cd.m^{-2}/mR.s^{-1}
	930	460	230	cd.m^{-2}/μC.kg^{-1}s^{-1}
Limiting resolution (5)				
center	48	56	64	lp/cm
70 % radius	44	52	58	lp/cm
93 % radius	42	50	54	lp/cm
MTF (6)			see figure 2	
Low-frequency drop (7)	5	4	3	%
Contrast ratio (8)				
large area (10 %)	23:1	25:1	30:1	
small detail (10 mm)	16:1	18:1	20:1	
Brightness non-uniformity (9)	20	10	5	%
Integral distortion (10)	4	2	1	%
Differential distortion (10)	15	6	3	%
Average background luminance (11)	0.02	-	-	cd.m^{-2}
Decay time (12)	3	-	-	ms

NOTES

Radiation quality A: 22 mm Al total filtration; 7 mm Al 1st HVL (≈75 kVp)

Radiation quality B: 2.5 mm Al total filtration; 50 kVp

(1) The nominal entrance field size is defined as the diameter of the input field when the image intensifier is irradiated with a parallel x-ray beam; i.e. when the x-ray source is at infinite distance.

(2) The useful entrance field size is defined as the diameter of the input field when the x-ray source is at distance of 100 cm from the input plane.

(3) The DQE is defined as SNR$^2_{out}$ / SNR$^2_{in}$, where SNR$_{out}$ and SNR$_{in}$ are respectively the signal-to-noise ratio in the output image and the input plane. The values given are obtained with conventional rms bandpass filter methods for two frequently used cut-off frequencies (1.7 Hz and 17 Hz), as well as the zero frequency values according to the new IEC standard (in preparation).

(4) The conversion factor is defined as the ratio of the luminance in the center of the output image to the x-ray exposure rate at the center of the input plane. Radiation quality: A

(5) The limiting resolution is measured with a rectangular lead bar pattern. Radiation quality: B

(6) The measurement of the Modulation Transfer Function (MTF) takes into account the veiling glare in the whole output image and therefore contains the full low-frequency drop. Radiation quality: B

(7) The low-frequency drop is defined as the difference between the MTF value at zero frequency (= 1.00 by definition) and at 1 lp/cm.

(8) The contrast ratio is defined as the ratio between the luminance in the center of the output image if no object is present in the input plane, *and the residual luminance at th same point if the central zone of the input plane is covered with an x-ray opaque lead disk under exactly* the same exposure rate.

- the large area contrast ratio is the ratio when the area of the lead disk is equal to 10 % of the useful input field area;
- the small detail contrast ratio is the ratio when the diameter of the lead disk is 10 mm.

Radiation quality: B.

(9) The brightness non-uniformity is defined as the drop in the luminance of the output image at 90 % radius with respect to the value at the center of the image. Radiation quality: A

(10) The integral distorsion D$_i$ and the differential distortion D$_d$ are defined as:

D$_i$ = [M/m$_0$ -1] x 100 %

D$_d$ = [m/m$_0$ -1] x 100 %

m$_0$ and m are respectively the local magnifications (of a 1 cm object) at the center and at 90 % radius and M is the total (integrated) magnification of a centered disk with a diameter of 90 % of the useful input field diameter. Radiation quality: A

(11) The average background luminance is the average light level in the output image when the x-ray beam is off.

(12) The decay time is the time necessary to decay to 10 % of the initial luminance level after stopping an exposure rate of 5 mR.s^{-1} in the normal mode.

Decay time depends strongly on the exposure rate. Radiation quality: A

CHARACTERISTICS CURVES

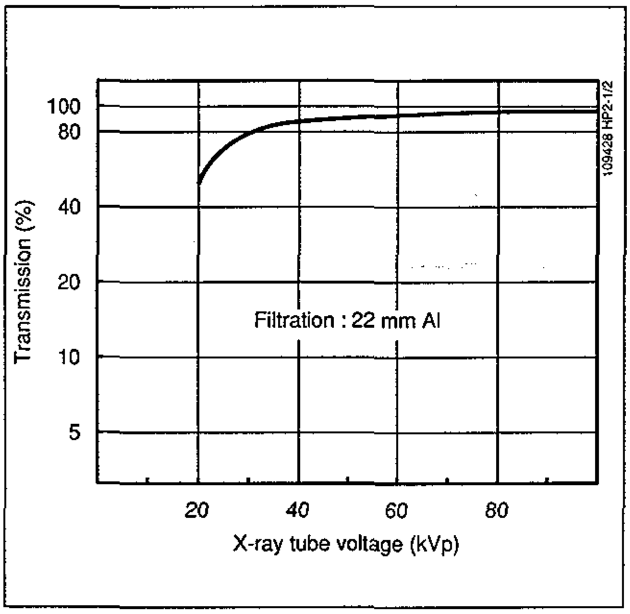


Figure 1 - X-ray transmission of the aluminium input window as a function of the x-ray tube voltage.

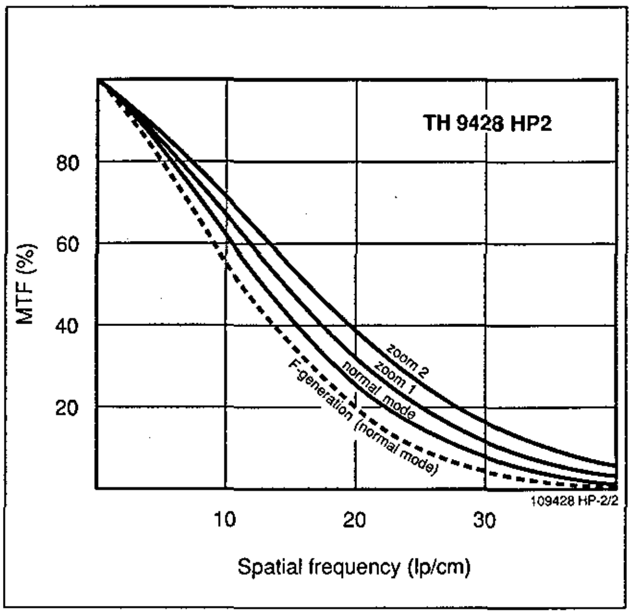


Figure 2 - MTF curves including the low-frequency drop. The MTF of F-generation is shown for comparison.

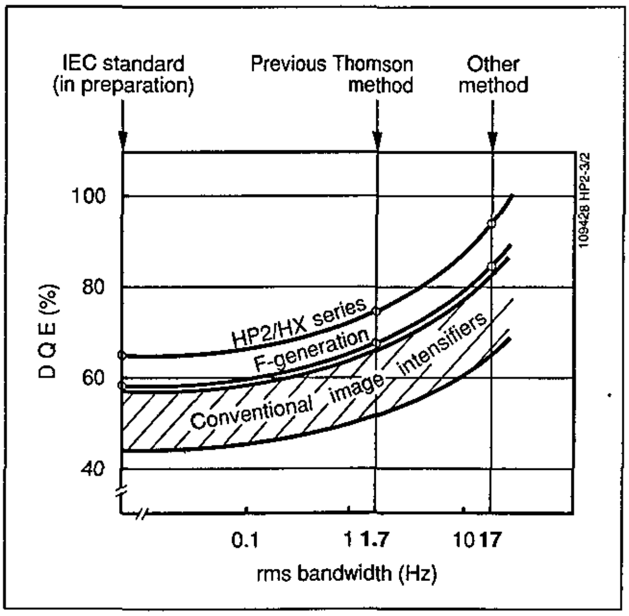


Figure 3 - DQE of the HP2/HX series as a function of the measuring method.

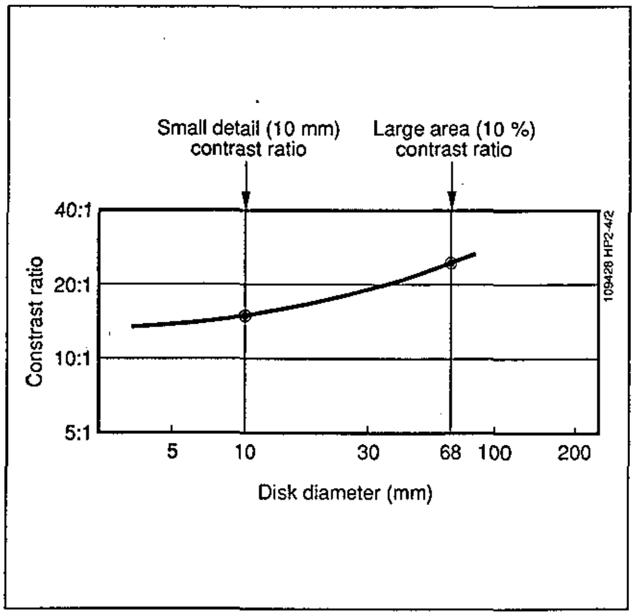


Figure 4 - Variation of the contrast ratio as a function of the disk size.

OPERATING VOLTAGES

Operating mode	Normal	Zoom 1	Zoom 2	
Photocathode PC	0	0	0	V
Electrode G1*	100 to 300	100 to 300	100 to 300	V
Electrode G2*	400 to 900	400 to 900	400 to 900	V
Electrode G3	2.5 to 4.5	5.0 to 8.0	9.0 to 12.0	kV
Screen G5	30.0	30.0	30.0	kV
Getter cathode E	0	0	0	kV
Getter anode G4	2.5 to 3.0	2.5 to 3.0	2.5 to 3.0	kV
G2 blanking	-700 to -1000	-700 to -1000	-700 to -1000	V

* For optimum performances G1 and G2 should be adjusted for each mode.

MAXIMUM RATINGS

Electrical

Maximum input voltages:

electrode G1	350	V
electrode G2	1.5	kV
	(-1.3 kV blanking)	
electrode G3	13	kV
electrode G4	3.5	kV
electrode G5	31	kV
Photocathode maximum continuous current	5	μA
Photocathode maximum pulsed current	50	μA

Environmental data

Climatic

Operating temperature range	+5 to +40	°C
Storage temperature range	-10 to +55	°C
Recommended long term storage temperature range	0 to +40	°C

Mechanical

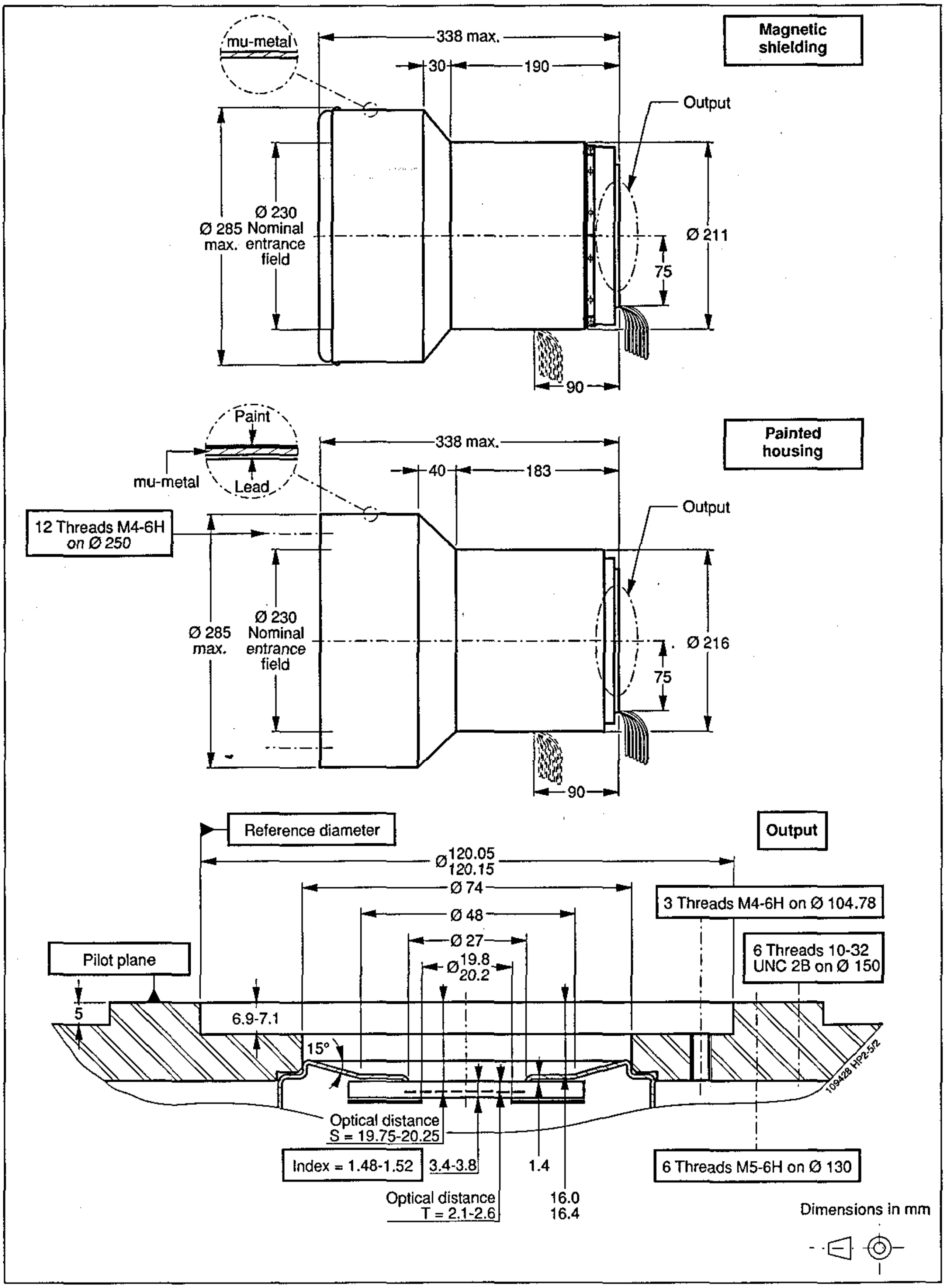
- Without any load on the output flange, the x-ray image intensifier should not be submitted to mechanical shocks *exceeding 10 g (6 ms duration).*
- With the maximum allowed static load on the output flange the mechanical shocks should not exceed 1 g (6 ms duration).

Operating instructions are included with the tube

NOTE

Read carefully the SAFETY WARNING for electron tubes (document TTE 807 enclosed with the product)

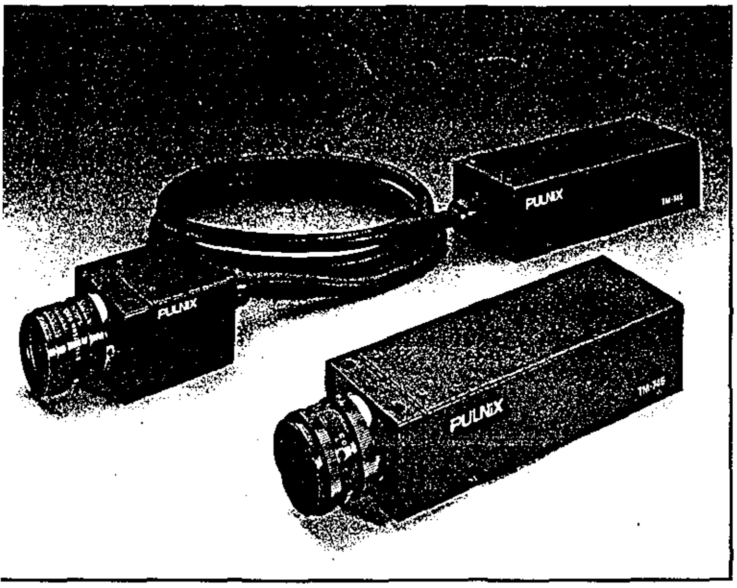
TH 9428 HP2
OUTLINE DRAWING



Dimensions in mm

Appendix I.3 - Pulnix TM-765E Camera Data Sheet

PULNiX TM-765E/TM-745E HIGH RESOLUTION CCD SHUTTER CAMERA WITH EDGE ENHANCER

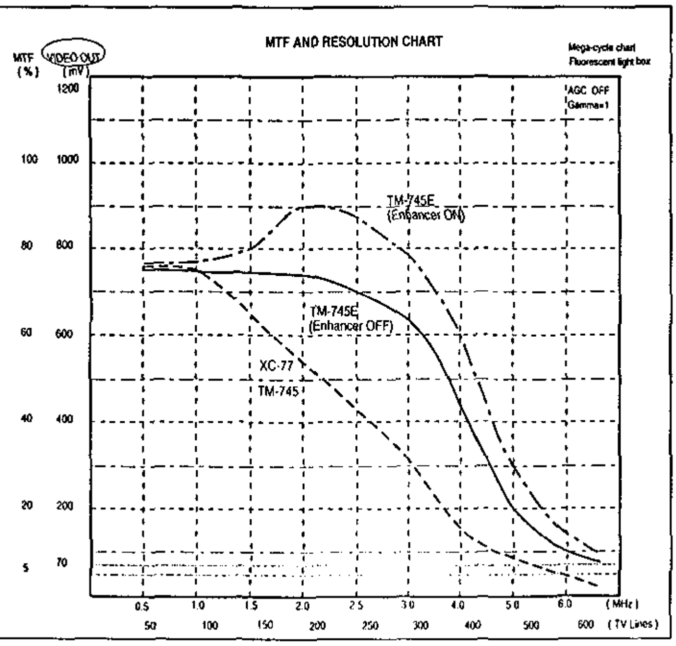


Features

- Very high resolution
  756 (H) x 581 (V) ... TM-765E
  768 (H) x 493 (V) ...TM-745E
- Low light sensitivity (0.5 lux)
- Smearless shutter 1/60 to 1/10,000 (or 1/31,500).
- Full frame integration and asynchronous reset
- Ext. sync, interlace, non-interlace auto select
- Built-in edge enhancement
- Excellent S/N (50 dB)
- AGC on/off, gamma 1 or 0.45
- Small, light weight

General Description

he PULNiX TM-765E is a high resolution 756 (H) x 581 (V) black nd white shutter camera with a built-in edge enhancing capability nd uniform MTF (Modulation Transfer Function) characteristics. hese cameras are excellent in applications such as bar code eading, gauging, character reading, high definition graphics, ntensified CCD cameras and detailed serveillance. Added to the vide versatility of this camera is an excellent low light sensitivity of .5 lux at F1.4. The capability to do asynchronous reset and full rame integration are standard features of these cameras. AGC nable, blemish free imagers, internal IR cut filter, gamma = 0.45 nd the popular remote imagers (standard 48", other lengths are ossible by special order) are all optional features that PULNiX ffers for these cameras.



Integration

The CCD imager of the TM-765E can be exposed longer than normal TV timing (16.7 msec.). This feature provides high sensitivity for dark environment applications. Integration is achieved by controlling the #11 pin of the 12-pin connector to Low (TTL). Because PULNiX uses an interline transfer chip in the TM-765E, a full frame of resolution is obtainable. (A full frame is not available in the shutter mode. Use PULNiX TM-9700 if full frame shuttering is required.) In integration, the signal process keeps optical black levels as reference black video to clamp video levels and this results in the cancelling out of thermal noise during the integration period.

Electronic Shutter

The TM-765E has a substrate drain type shutter mechanism which provides a superb picture at various speeds without smearing. Normal operation requires the CCD chip to construct an individual potential well at each image cell. These potential wells are seprated from each other by a barrier. The barrier is sequentially removed to transfer the charge from one CCD to another by the pixel clock. This is the basic principle of CCD operation for interline transfer.The substrate drain vertically moves the charges. When excess potential is applied to the substrate underneath each cell, the potential barrier is pulled down to release the charge into the drain. This can occur with all the cells simultaneously, whereas normal CCD shuttering is done with a horizontal charge shift to the drain by interline transferring or reverse transferring of the frame transfer chip.

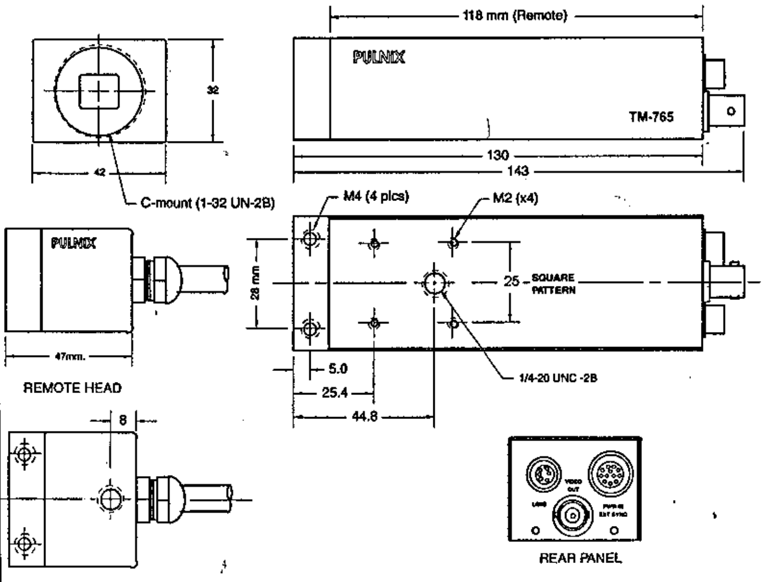
The capability to externally vary the electronic shutter rate to 1/60, 1/125, 1/250, 1/500, 1/1000, 1/2000, 1/4000, and 1/10,000 sec. is a standard feature of this camera. A special option is available for variable shutter speeds up to 1/31,000. Shuttering eliminates the need for costly and distracting strobe lights on a high speed assembly or inspection line.

Specifications

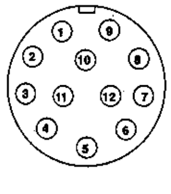
Model	TM-765E (CCIR)	TM-745E (EIA)
Imager	2/3" interline transfer CCD, HAD type	
Pixel	756 (H) x 581 (V)	768 (H) x 493 (V)
Cell size	11 µm x 11 µm	11 µm x 13 µm
Scanning	625 lines 50 Hz, 2:1 interlace	525 lines 60 Hz, 2:1 interlace
Sync	Internal/external auto switch HD/VD, 4.0 Vp-p impedance 4.7 KΩ VD = interlace/non-interlace	
	HD = 15.625 kHz ± 5%	HD = 15.734 kHz ± 5%
TV resolution	560 (H) x 420 (V)	570 (H) x 350 (V)
S/N ratio	50 dB min.	
Min. illumination	0.5 lux f = 1.4 without IR cut filter	
Video output	1.0 Vp-p composite video, 75Ω	
AGC	ON/OFF (default)	
Gamma	0.45/1.0 (defalt)	
Lens mount	C-mount std., mini-bayonet optional	
Power req.	12V DC, 350 mA	
Operating temp.	-10°C to 50°C	
Vibration & shock	Vibration: 7G (200 Hz to 2000 Hz) Shock: 70G	
Size (W x H x L)	42 mm x 32 mm x 130 mm 1.65" x 1.76" x 5.12"	
Weight	200 grams	

Due to ongoing product improvements, specifications may change without notice.

Physical Dimensions



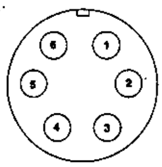
Pin Configuration



12-Pin connector

1	GND
2	+12V
3	GND
4	Video
5	GND
6	Vinit
7	VD in
8	GND
9	HD in
10	IDLE*
11	Int. cont.
12	GND

*IDLE: 1/1000 sec. shutter when #10 is low.



6-Pin connector

1	D2
2	GND
3	Video
4	+12V (or 5V)
5	D0
6	D1

D0, D1, D2 Shutter control

the UK
ILNiX Europe Ltd,
ILNiX House, Aviary Court,
ade Road, Basingstoke,
nts RG24 8PE
: 01256 475555
x: 01256 466268

PULNiX
Industrial Products Division

In Germany
PULNiX Europe Ltd.
Industriepark, Steinbruch 5,
63755 Alzenau, Germany
Tel: (49) 06023 4666
Fax: (49) 06023 4667

Appendix II - "C" Source Code

Appendix II.1 - xray1.c

```
/*  MACRO TO GRAB AND SAVE IMAGES ON FG100 SYSTEM  */
/*  WRITTEN 02 FEBRUARY 1995 BY D. KERR FOR IAN BROWBANK  */
/*  Modified 3/11/95 R. Drew to include integration  */
/*  Modified 11/3/96 R. Drew to include examine and MMAX  */
/*  Modified  2/9/96 R. Drew to get gain/offset  */
/*  Modified 7/11/96 R. Drew for use with VS100 board  */
/*  to compile type goi xray1  */
#include <dos.h>
#include <stdio.h>
#include <stdlib.h>
#include <stddef.h>
#include <itex100.h>
/* columns */
#define NMAX 512 /*768*/
/* rows */
#define MMAX 512
void getim(void);
void savim(void);
void snapim(void);
void integrat(void);
void examine(void);
void gain(void);
void offset(void);
char ch, argv[20];
unsigned int i, j, k;
unsigned int zout;
unsigned char  nc[NMAX];
unsigned base = 0x300;
long mem=0xd0000L;
int flag = 1, block=1;
FILE *fp1;
```

```
main()
{
        sethdw( base, mem, flag, block );
        setdim(NMAX,MMAX,12);
        initialize();
        linlut(INPUT,0);
        setvmask(0xF00);
        setinmux( 0 );
        zout=64;
labl:   printf("FG100: PLEASE SELECT ONE OF THE FOLLOWING: \n");
        printf("      get  image......... 1\n");
        printf("      save image......... 2\n");
        printf("      snap image......... 3\n");
        printf("      clear frame........ 4\n");
        printf("        integration grab... 5\n");
        printf("      exit to DOS........ 6\n");
        printf("      examine............ 7\n");
/*      printf("      gain/offset........ 8\n");
*/      printf("  \n");
        printf("      selection ?........ ");
        ch = getch();
        printf("  \n");
        if (ch == '1')
                getim();
        if (ch == '2')
                savim();
        if (ch == '3')
                snapim();
        if (ch == '4')
                sclear(0);
        if (ch == '5')
                integrat();
        if (ch == '7')
                examine();
/*      if (ch == '8')
                offset();
*/      if (ch == '6')
                {outp(0x323,0x83);
                outp(0x320,0x00);
                exit(-1);}
        goto labl;

}       /* END OF MAIN MACRO ROUTINE */
```

```
void getim()
/* get active 512x512 image from hard disc */
{
        gets(argv);
        printf("enter file name: \n");
        gets(argv);
        fp1 = fopen(argv, "rb");
        if( fp1 == NULL ){printf("file does not exist \n"); exit(1); }
        printf(" reading file: %s\n",argv);
        for (i = 0 ; i < MMAX ; i++)
        {
                fread (nc, 1, NMAX, fp1);
                bwhline( 0,i,NMAX,nc);
        }
        fclose(fp1);
}

void savim()
/* save active 512x512 image to hard disc */
{
        gets(argv);
        printf("enter file name: \n");
        gets(argv);
        fp1 = fopen(argv, "wb");
        if( fp1 == NULL ){printf("cannot open file \n"); exit(1); }
        printf(" writing file: %s\n",argv);

        for (i = 0 ; i < MMAX ; i++)
        {
                brhline( 0,i,NMAX,nc);
                fwrite (nc, 1, NMAX, fp1);
        }
        fclose(fp1);
}

void snapim()
/* freeze 512x512 active image */
{
        int old, new;
        initialize();
/*      offset commands
        old=inpw(0x31C);
        new=(old & 0xCFFF);
        outpw(0x31c,new);
        outpw(0x316,zout);
        outpw(0x31c,old);
*/      linlut(INPUT,0);
        setvmask(0xF00);
        setinmux( 0 );
        grab(-1);
        printf("\n............ hit <RETURN> to freeze .........\n");
        getchar();
        stopgrab(1);
}

void examine()
/* gives a few details of image */
{
        int v;
        int max,min;
        max=0;min=255;
        for (i=0;i<NMAX;i++)
        {
                for (j=0;j<MMAX;j++)
                {
                        v=(brpixel(i,j)&255);
                        if(v<min) min=v;
                        if(v>max) max=v;
                }
        }
        printf("maximum %d, minimum %d \n",max,min);
}

void offset()
{
        unsigned int scr,old,new;
        unsigned char zgain, zoffset;
        old=inpw(0x31c);
        new=(old & 0xCFFF);
        outpw(0x31c,new);
        scr=inpw(0x316);
        printf("The gain is %d\n",(scr-(0x100*(scr/0x100))));
        printf("The offset is %d\n",(scr/0x100));
        printf("New gain?\n");
        fscanf(stdin,"%d",&zgain);
        printf("New offset?\n");
        fscanf(stdin,"%d",&zoffset);
        zout=zgain+zoffset*0x100;
        outpw(0x316,zout);
        outpw(0x31c,old);
}
```

```
void integrat()
/* use camera integration facility*/
{
        double tme;
        int old,new,scr,frames;
        printf("How many 1/25ths of a second to integrate for?\n");
        fscanf(stdin,"%d",&frames);
        tme = frames;
        tme = tme/25;
        printf("The shutter is open for %.3f seconds.\n",tme);
        /* read for sync */
        initialize();
        linlut(INPUT,0);
        setvmask(0xF00);
        setinmux( 0 );
/*      offset commands
        old=inpw(0x31C);
        new=(old & 0xCFFF);
        outpw(0x31c,new);
        outpw(0x316,zout);
        outpw(0x31c,old);
*/      do{scr=inp(0x31b);}
        while ((scr & 5) && 5);
        /* switch on integration */
        OUTP(0x323,0x83);
        OUTP(0x320,0xFF);
        /* count frame syncs until = frames */
        frames=frames*2;
        while (frames-3 >= 0)
        {
                do{scr=inp(0x31b);/*printf("a%d ",(~scr & 4) && 4);*/}
                while ((scr & 4) && 4);
                do{scr=inp(0x31b);/*printf("b%d ",(~scr & 4) && 4);*/}
                while ((~scr & 4) && 4);
                frames—;
        }
        /* switch off integration */
        OUTP(0x323,0x83);
        OUTP(0x320,0x00);
        do{scr=inp(0x31b);printf("%d ",scr);}
        while((~scr & 1) && 1);
        snap(1);
        printf("\n%d \n",scr);
}
```

Appendix II.2 - findgrid.c

```
// Image is processed to remove spatial distortion.
#include <math.h>
#include <dos.h>
#include <stdio.h>
#include <stdlib.h>
#include <stddef.h>
#include <conio.h>
#include <graph.h>
#include <string.h>
#define NMAX 512                        // line length
#define MMAX 512                        // no of lines
#define FALSE (int)0
#define TRUE (int)1
#define pi 2*asin(1)                    // 3.14159
#define RGB(r,g,b) (((long) ((b) << 8 | (g)) << 8) | (r))
#define polar {angle2=atan2(y2,x2);if((angle1>pi/2)&&(angle2<-
        pi/2))angle2=angle2+2*pi;if((angle2>pi/2)&&(angle1<-
        pi/2))angle1=angle1+2*pi;rdist2=x2*x2+y2*y2;}
#define test (((angle2-angle1)<.15)&&((angle1-
        angle2)<.15)&&((rdist1/rdist2)>.85)&&((rdist2/rdist1)>.85))
unsigned char bimage[MMAX][NMAX];
unsigned char mimage[MMAX][NMAX];
struct pxl
{
        int ival,jval;
        struct pxl *link;
};

struct posit
{
        struct posit *link;
        struct pxl *plink;
};

struct fpxl
{
        int    real;
        int size;
        float ival,jval;
        struct fpxl *link;
};
```

```
struct fposit
{
        signed int fhole,lhole;
        struct fposit *link;
        struct fpxl *plink;
};


// // // // // // // // // // // // // // // // // // // // // // // // // // // //


void Findholes(unsigned char image[MMAX][NMAX],struct posit *tmp,char aga[60])
{
        struct pxl *hole;
        struct posit *current;
        signed int i,j;
        FILE *fp1;
        fp1 = fopen(aga, "rb");                     // grid file aga
        if( fp1 == NULL ){printf("file does not exist \n"); exit(1);}
        for (j = 0 ; j < MMAX ; j++) fread(image[j],1,NMAX,fp1);
        fclose(fp1);
        printf("File of %dx%d Required. Looking for grid holes",NMAX,MMAX);
        for (j=0;j<MMAX;j++)
        {       // each line
                for (i=0;i<NMAX;i++)
                {       // each pixel
                        mimage[j][i]=0;
                        if((image[j][i]>0))
                        {       // pixel above threshold: part of a hole.
                                mimage[j][i]=32;
                                if (i>0)
                                if (image[j][i-1]==0)
                                if (j>0)
                                if (image[j-1][i]==0)
                                {       // probably new hole
                                        mimage[j][i]=64;
                                        hole=malloc(sizeof(struct pxl));
                                        if(hole == NULL) printf("Out of memory");
                                        current=tmp;
                                        current->plink=hole;
                                        hole->ival=i;
                                        hole->jval=j;
                                        tmp=malloc(sizeof(struct posit));
                                        if(tmp == NULL) printf("Out of memory");
                                        current->link=tmp;
                                        }
                                }
                        }
                }
                current->link=NULL;
                free(tmp);
                printf(" done.\n");
        }       // List of holes made in posit.


        void Finddirection(struct pxl *line,int *newdir,unsigned char image[MMAX][NMAX])
        {       // direction of next pixel in perimeter of hole found.
                int direction[5];
                if (line->jval==0) direction[1]=FALSE;
                else
                {       // looking above
                        if (image[line->jval-1][line->ival]!=0) direction[1]=TRUE;
                        else direction[1]=FALSE;
                }
                if (line->ival==NMAX-1) direction[2]=FALSE;
                else
                {       // looking to right
                        if (image[line->jval][line->ival+1]!=0) direction[2]=TRUE;
                        else direction[2]=FALSE;
                }
                if (line->jval==MMAX-1) direction[3]=FALSE;
                else
                {       // looking below
                        if (image[line->jval+1][line->ival]!=0) direction[3]=TRUE;
                        else direction[3]=FALSE;
                }
                if (line->ival==0) direction[4]=FALSE;
                else
                {       // looking to left
                        if (image[line->jval][line->ival-1]!=0) direction[4]=TRUE;
                        else direction[4]=FALSE;
                }
                (*newdir)--;  // (old direction, turn anti-clockwise 90'
                if(*newdir==0) *newdir=4;
                if(direction[*newdir]==FALSE)
                {
                        (*newdir)++;                    // not left turn
                        if (*newdir==5) *newdir=1;
                        if (direction[*newdir]==FALSE)
                        {
                                (*newdir)++;            // not straight on
                        if (*newdir==5) *newdir=1;
                        if (direction[*newdir]==FALSE)
                        {
                                (*newdir)++;            // not right turn
                                if (*newdir==5) *newdir=1;
                                if (direction[*newdir]==FALSE) *newdir=0;
                        }                               // else reverse
                }
        }
}


void Mapholes(unsigned char image[MMAX][NMAX],struct posit *initial)
        // ,struct) fposit *finitial)
{
        struct pxl *hole, *line;
        struct posit *current, *tmp, *prev;
        int newdir;
        signed int i,j,ioff,joff;
        printf("finding centres of grid nodes");
        current=initial;
        i=0;
        while(current!=NULL)
        {       // look at each potential hole
                i++;
                line=current->plink;
                line->link=NULL;
                newdir=2;
                j=0;
                mimage[line->jval][line->ival]=255;
                while(line->link!=current->plink)
                {       // draw a line right round the hole.
                        j++;
                        if (j==1) line=current->plink;
                        Finddirection(line,&newdir,image);
                        if (newdir==1) {ioff= 0;joff=-1;}
                        if (newdir==2) {ioff= 1;joff= 0;}
                        if (newdir==3) {ioff= 0;joff= 1;}
                        if (newdir==4) {ioff=-1;joff= 0;}
                        if (((line->ival+ioff==current->plink->ival)
                                &&(line->jval+joff==current->plink->jval)
                                &&((newdir==1)||((newdir==4)
                                &&((current->plink->jval==MMAX-1)
                                ||(image[current->plink->jval+1][current->plink-
                                        >ival]==0)))))
                                ||(newdir==0))
                        {       // hole mapped
                                line->link=current->plink;
                        }
                        else
                        {       // add another pixel to line
                                hole=(struct pxl *)malloc(sizeof(struct pxl));
                                if(hole==NULL) printf("Out of Memory");
                                hole->ival=line->ival+ioff;
                                hole->jval=line->jval+joff;
                                if ((mimage[line->jval+joff][line->ival+ioff]==64)
                                  ||(mimage[line->jval+joff][line->ival+ioff]==192))
                                {
                                        mimage[line->jval+joff][line->ival+ioff]=192;
                                }
                                else
                                {
                                        mimage[line->jval+joff][line->ival+ioff]=128;
                                }

                                line->link=hole;
                                line=hole;
                        }
                        tmp=current->link;
                        prev=current;
                        while (tmp != NULL)
                        {       // look at all holes not yet processed.
                                if ((line->ival==tmp->plink->ival)&&(line->jval==tmp-
                                        >plink->jval))
                                {       // remove surplus grid node
                                        prev->link=tmp->link;
                                        free(tmp->plink);
                                        free(tmp);
                                        tmp=current;
                                }
                                prev=tmp;
                                tmp=tmp->link;
                        }
                }
                current=current->link;
        }
        printf(" done\n");
}
```

```
void Findcentre(        unsigned char image[MMAX][NMAX],
                        struct posit *current,
                        struct fposit *fint)
{
        int imin,imax,jmin,jmax,count,i,j;
        float den,isum,jsum,icog,jcog,cv;
        struct fpxl *next, *fhole;
        struct pxl *pixel;
        printf("Finding Centres of Holes");
        while (current!=NULL)
        {       // go through all mapped holes
                next=malloc(sizeof(struct fpxl));
                if (next==NULL) printf("Out of memory\n");
                if (fint->plink!=NULL)
                {       // go to end of list fint
                        fhole->link=next;
                }
                else
                {       // at start of list fint
                        fint->plink=next;
                }
                imin=NMAX-1;jmin=MMAX-1;imax=0;jmax=0;count=0;
                pixel=NULL;
                while (current->plink!=pixel)
                {       // look at each pixel on edge of hole
                        count++;
                        if (count==1) pixel=current->plink;
                        if (pixel->ival>imax) imax=pixel->ival;
                        if (pixel->ival<imin) imin=pixel->ival;
                        if (pixel->jval>jmax) jmax=pixel->jval;
                        if (pixel->jval<jmin) jmin=pixel->jval;
                        pixel=pixel->link;
                }       // found outer limits of hole
                isum=0;jsum=0;den=0;
                for(i=imin;i<=imax;i++)
                {       // look at each line in hole
                        for(j=jmin;j<=jmax;j++)
                        {       // look at each pixel in hole area
                                cv=image[j][i];
                                if (cv>0)
                                {       // if pixel is in hole
                                        den=den+cv;
                                        isum=isum+cv*(i-imin);
                                        jsum=jsum+cv*(j-jmin);
                                }
                        }
        }
        icog=imin+(isum/den);
        jcog=jmin+(jsum/den);
        next->ival=icog;
        next->jval=jcog;
        next->real=TRUE;
        next->size=(int)den;
        if((imin<=0)||(imax>=NMAX-1)||(jmin<=0)||(jmax>=MMAX-1)
            |((den<200)&&((imax-imin<2)||(jmax-jmin<2))))
        {   // hole not up to specification - MAY NEED TWEAK HERE
            if(fint->plink==next) fint->plink=NULL; else fhole-
                >link=NULL;
            free(next);
        }
        else
        {
            fhole=next;
            mimage[(int)jcog][(int)icog]=200;
        }
        current=current->link;
    }
    next=malloc(sizeof(struct fpxl));
    // Add two dummy holes as padding
    if (next==NULL) printf("Out of memory\n");
    fhole->link=next;
    next->ival=NMAX*4;
    next->jval=MMAX*4;
    next->real=FALSE;
    next->size=1;
    fhole=next;
    next=malloc(sizeof(struct fpxl));
    if (next==NULL) printf("Out of memory\n");
    fhole->link=next;
    next->ival=NMAX*4+1;
    next->jval=MMAX*4+1;
    next->real=FALSE;
    next->size=1;
    fhole=next;
    fhole->link=fint->plink;
    printf(" done\n");
}
```

```
void Findorigin(struct fposit *finitial, int ct)
{
        int i;
        struct fpxl *hole;
        hole=finitial->plink;
        if (ct!=1)
        {
                for (i=1;i<ct;i++)
                {       // find next hole not yet tested
                        hole=hole->link;
                }
        }
        finitial->link->plink=hole;
}


void Findaxes(struct fposit *finitial)
{
        double angle1,angle2;
        float p1,px,p2,x1,x2,y1,y2;
        struct fpxl *nearest[2], *hole;
        hole=finitial->plink;
        p1=(float)NMAX*(float)NMAX+(float)MMAX*(float)MMAX;
        p2=(float)NMAX*(float)NMAX+(float)MMAX*(float)MMAX;
        do
        {       // look at each hole in un-mapped list
                px=(hole->ival-finitial->link->plink->ival)*(hole->ival-finitial-
                        >link->plink->ival)+(hole->jval-finitial->link->plink-
                        >jval)*(hole->jval-finitial->link->plink->jval);
                if (px!=0)
                {       // i.e. not looking at the same hole
                        if (px<p1)
                        {       // hole is nearer than previous nearest
                                p1=px;
                                nearest[0]=hole;
                        }
                }
                hole=hole->link;
        }       while (hole!=finitial->plink);
        x1=nearest[0]->ival-finitial->link->plink->ival;
        y1=nearest[0]->jval-finitial->link->plink->jval;
        angle1=atan2(x1,y1);
        hole=finitial->plink;
        nearest[1]=finitial->link->plink;
        do
        {       // look at each hole in un-mapped list
        x2=hole->ival-finitial->link->plink->ival;
        y2=hole->jval-finitial->link->plink->jval;
        px=(x2*x2+y2*y2);
        if ((px!=0)&&(hole!=nearest[0]))
        {   // i.e. not looking at the same hole
            angle2=atan2(x2,y2);
            if ((angle1>pi/2)&&(angle2<-pi/2)) angle2=angle2+2*pi;
            // correct for 180`
            if ((angle2>pi/2)&&(angle1<-pi/2)) angle1=angle1+2*pi;
            //    "
            if ((px<p2)&&(pow(cos(angle2-angle1),2)<.3))
            {   // hole is nearer than previous nearest
                p2=px;
                nearest[1]=hole;
            }
        }
        hole=hole->link;
    }   while (hole!=finitial->plink);
    angle1=atan2(nearest[0]->jval-finitial->link->plink->jval,nearest[0]->ival-
        finitial->link->plink->ival);
    angle2=atan2(nearest[1]->jval-finitial->link->plink->jval,nearest[1]->ival-
        finitial->link->plink->ival);
    if ((angle1>pi/2)&&(angle2<-pi/2)) angle2=angle2+2*pi;   // correct for 180`
    if ((angle2>pi/2)&&(angle1<-pi/2)) angle1=angle1+2*pi;   //      "
    if (pow(sin((double)angle1),2)>pow(sin((double)angle2),2))
    {   // [1] is nearer horizontal than [0]
        finitial->link->link->plink=nearest[0];
        nearest[0]=nearest[1];
    }
    else finitial->link->link->plink=nearest[1];
    finitial->link->link->link->plink=nearest[0];
}


void Do_a_line(     struct fposit *finitial,
                    struct fposit *line1,
                    double angle3,
                    float rdist3)
{
    int done, forget;
    double angle1,angle2;
    float x2,y2,rdist1,rdist2;
    struct fpxl *node, *node2, *hole, *tmp, *start;
    rdist1=rdist3;
    angle1=angle3;
    tmp=finitial->plink;
```

```
hole=tmp->link;
start=hole;
node=line1->plink;
while (node->link!=NULL) node=node->link;
do
{
        done=TRUE;
        do
        {       // section finds all the holes in line after first hole.
                forget=FALSE;
                x2=hole->ival-node->ival;
                y2=hole->jval-node->jval;
                polar
                if test
                {       // hole is next in line
                        tmp->link=hole->link;
                        if (hole==start)
                        {
                                forget=TRUE;
                                start=hole->link;
                        }
                        if (hole==finitial->plink)
                        {
                                finitial->plink=tmp;
                        }
                        if (done==TRUE)
                        {
                                node->link=hole;
                                node2=hole;
                                node2->link=NULL;
                                (line1->lhole)++; // line ends one hole later
                                done=FALSE;
                        }
                        else
                        {
                                node2->ival=(node2->ival*node2->size+hole-
                                   >ival*hole->size)/(node2->size+hole->size);
                                node2->jval=(node2->jval*node2->size+hole-
                                   >jval*hole->size)/(node2->size+hole->size);
                                node2->size=node2->size+hole->size;
                                free(hole);
                        }
                }
                else
                {       // hole not in spec, go to next.
                        tmp=hole;
                }
                hole=tmp->link;
        }       while ((hole!=start)||(forget!=FALSE));
        if (done==FALSE)
        {
                x2=node2->ival-node->ival;
                y2=node2->jval-node->jval;
                polar
                rdist1=rdist2;                          // update spec
                angle1=angle2;
                node=node2;
        }
}       while (done==FALSE);
node=line1->plink;
rdist1=rdist3;
if (angle3>0) angle1=(angle3-pi); else angle1=(angle3+pi);
do
{
        done=TRUE;
        do
        { // this section finds all the holes in line before first hole.
                forget=FALSE;
                x2=hole->ival-node->ival;
                y2=hole->jval-node->jval;
                polar
                if test
                {       // hole is next in line
                        tmp->link=hole->link;
                        if (hole==start)
                        {
                                forget=TRUE;
                                start=hole->link;
                        }
                        if (hole==finitial->plink)
                        {
                                finitial->plink=tmp;
                        }
                        if (done==TRUE)
                        {
                                line1->plink=hole;
                                node2=hole;
                                node2->link=node;
                                (line1->fhole)—; // line ends one hole early
                                done=FALSE;
                        }
                        else
                {
                    node2->ival=(node2->ival*node2->size+hole-
                       >ival*hole->size)/(node2->size+hole->size);
                    node2->jval=(node2->jval*node2->size+hole-
                       >jval*hole->size)/(node2->size+hole->size);
                    node2->size=node2->size+hole->size;
                    free(hole);
                }
            }
            else
            {   // hole not in spec, go to next.
                tmp=hole;
            }
            hole=tmp->link;
        }   while ((hole!=start)||(forget!=FALSE));
        if (done==FALSE)
        {
            x2=node2->ival-node->ival;
            y2=node2->jval-node->jval;
            polar
            rdist1=rdist2;                          // update spec
            angle1=angle2;
            node=node2;
        }
    }   while (done==FALSE);
    finitial->plink=tmp;
}


void Look_for_odd_lines(    struct fposit *finitial,
                            struct fpxl *node,
                            double angle1,
                            float rdist1,
                            double angle3,
                            float rdist3,
                            int i)
{
    int done, forget;
    double angle2;
    float x2,y2,rdist2;
    struct fposit *line1, *line2;
    struct fpxl *hole, *tmp, *start, *node2;
    tmp=finitial->plink;
    hole=tmp->link;
    start=hole;
    done=TRUE;
```

```
do
{       // this section finds first hole.
        forget=FALSE;
        x2=hole->ival-node->ival;
        y2=hole->jval-node->jval;
        polar
        if test
        {       // hole is next in line
                tmp->link=hole->link;
                if (hole==start)
                {
                        forget=TRUE;
                        start=hole->link;
                }
                if (hole==finitial->plink)
                {
                        finitial->plink=tmp;
                }
                if (done==TRUE)
                {
                        line1=finitial->link;
                        line2=malloc(sizeof(struct fposit));
                        if (line2==NULL) printf("Out of memory");
                        finitial->link=line2;
                        line2->link=line1;
                        line2->plink=hole;
                        line2->fhole=i;
                        line2->lhole=i;
                        node2=hole;
                        node2->link=NULL;
                        done=FALSE;
                }
                else
                {
                        node2->ival=(node2->ival*node2->size+hole->ival*hole-
                                >size)/(node2->size+hole->size);
                        node2->jval=(node2->jval*node2->size+hole->jval*hole-
                                >size)/(node2->size+hole->size);
                        node2->size=node2->size+hole->size;
                        free(hole);
                }
        }
        else
        {       // hole not in spec, go to next.
                tmp=hole;
        }
```

```
			hole=tmp->link;
		}	while ((hole!=start)||(forget!=FALSE));
		if (done==FALSE)
		{
			finitial->plink=tmp;
			Do_a_line(finitial,line2,angle3,rdist3);
		}
}


void Check_to_fside(	struct fposit *finitial,
				float rdist3,
				double angle3,
				struct fposit *line1,
				struct fposit *line2)
{	// looks for holes at start of line
	int iq,done,a;
	double angle1;
	float x1,y1,rdist1,x3,y3,frac;
	struct fposit *line3;
	struct fpxl *tmp, *tmp2, *node, *hole1, *hole2, *hole3;
	hole1=line1->plink;
	hole2=line2->plink;
	line3=finitial;
	// find where to start (a)
	if (line1->link==line2)
	{
		if (line2->link!=NULL)
		{	// side
			a=line2->link->fhole-1;
			line3=line2->link;
		}
		else
		{	// at end
			a=line2->lhole;
			line3=line2;	// line3 shouldn't be used
		}
	}
	else
	{
		while (line3->link!=line2)
		{
			line3=line3->link;
		}
		if (finitial->link!=line2)
		{
```

```
                a=line3->fhole-1;
        }
        else
        {
                a=line2->lhole;
        }
}
// line up holes to get rdist1, angle1.
if ((line1->fhole<=a)&&(line1->lhole>=a))
{       // can get good match
        iq=line1->fhole;
        while (iq<a)
        {
                iq++;
                hole1=hole1->link;
        }
        iq=line2->fhole;
        while (iq<a)
        {
                iq++;
                hole2=hole2->link;
        }
        x1=hole2->ival-hole1->ival;
        y1=hole2->jval-hole1->jval;
        rdist1=x1*x1+y1*y1;
        angle1=atan2(y1,x1);
        node=hole2;
}
else
{
        if (line1->fhole>line1->lhole)
        {       // bottom line, nline in use.
                hole3=line3->plink;
                hole2=line2->plink;
                iq=line2->fhole;
                while (iq<a+1)
                {
                        iq++;
                        hole2=hole2->link;
                }
                x1=hole3->ival-hole2->ival;
                y1=hole3->jval-hole2->jval;
        }
        else
        {
                if (line1->fhole>a)
                {       // move hole2
                        iq=line2->fhole;
                        while (iq<line1->fhole)
                        {
                                iq++;
                                hole2=hole2->link;
                        }
                }
                else
                {
                        iq=line2->fhole;
                        while (iq<line1->lhole)
                        {
                                iq++;
                                hole2=hole2->link;
                        }
                        while (hole1->link!=NULL)
                        {
                                hole1=hole1->link;
                        }
                }
                x1=hole2->ival-hole1->ival;
                y1=hole2->jval-hole1->jval;
        }
        rdist1=x1*x1+y1*y1;
        angle1=atan2(y1,x1);
        node=line2->plink;
        iq=line2->fhole;
        while (iq<a)
        {
                iq++;
                node=node->link;
        }
}
line3=finitial->link;
done=FALSE;
iq=a;
while ((line2->fhole<=iq)&&(done==FALSE))
{       // check for holes above line1
        if (node->link!=NULL)
        {       // update spec
                x3=node->link->ival-node->ival;  // find offset to next node
                y3=node->link->jval-node->jval;   //                 "
                rdist3=x3*x3+y3*y3;
                angle3=atan2(y3,x3); // Possible problems with pi, -pi
        }
```

```
Look_for_odd_lines(finitial,node,angle1,rdist1,angle3,rdist3,iq);
if (line3==finitial->link)
{       // no holes found
        hole2=node;
        node=line2->plink;
        while ((node->link!=hole2)&&(node!=hole2))
        {
                node=node->link;
        }
        iq--;
}
else
{ // add line3 at start of line2 and makeup any that are missing
        done=TRUE;
        line3=finitial->link;
        if (line3->link!=line2)
        {       // i.e. line 3 should not be on top row
                finitial->link=line3->link;
                if (line2->link==NULL)
                {       // at bottom
                        line2->link=line3;
                        line3->link=NULL;
                }
                else
                {       // somewhere in between
                        hole3=line3->plink;
                        if (line1->link==line2)
                        {
                                line1=line2->link;
                        }
                        else
                        {
                                line1=finitial;
                                while (line1->link!=line2)
                                {
                                        line1=line1->link;
                                }
                        }
                        if (line3->lhole>=line1->fhole-1)
                        {       // overlapping/match
                                iq=line3->fhole;
                                while(iq<line1->fhole-1)
                                {       // find new end of line3
                                        hole3=hole3->link;
                                        iq++;
                                }       // should have kept rest of line
```

```
            node=hole3;
            while (node->link!=NULL)
            {
                node=node->link;
            }
            node->link=finitial->plink->link;
            finitial->plink->link=hole3->link;
        }
        else
            {       // missing holes
            while(hole3->link!=NULL)
            {       // find end of line3
                hole3=hole3->link;
            }
            tmp=hole3;
            if (line3->lhole<line1->fhole-1)
            {
            for (iq=line3->lhole+1;iq<line1-
                >fhole;iq++)
            {       // make up missing holes
                tmp2=(struct fpxl
                  *)malloc(sizeof(struct fpxl));
                if(tmp2==NULL) printf("Out of
                  Memory");
                frac=(float)((float)(iq-line3-
                  >lhole)/(float)(line1->fhole-
                  line3->lhole));
                tmp2->ival=(tmp->ival)*(1-
                  frac)+(line1->plink-
                  >ival)*frac;
                tmp2->jval=(tmp->jval)*(1-
                  frac)+(line1->plink-
                  >jval)*frac;
                tmp2->real=FALSE;
                tmp2->size=1;
                hole3->link=tmp2;
                hole3=tmp2;
            }
            }
        }
        hole3->link=line1->plink;
        line1->fhole=line3->fhole;
        line1->plink=line3->plink;
        free(line3);
        line3=finitial->link;
    }
```

```
                        }
                }
        }       // that's the upper first found. :-)
}


void Check_to_lside(    struct fposit *finitial,
                        float rdist3,
                        double angle3,
                        struct fposit *line1,
                        struct fposit *line2)
{
        int iq,done,a,iqq;
        double angle1;
        float x1,y1,rdist1,x3,y3,frac;
        struct fposit *line3;
        struct fpxl *tmp, *tmp2, *node, *hole1, *hole2, *hole3;
        done=FALSE;
        hole1=line1->plink;
        hole2=line2->plink;
        // find where to start (a)
        line3=finitial;
        if (line1->link==line2)
        {
                if (line2->link!=NULL)
                {       // side
                        a=line2->link->lhole+1;
                        line3=line2->link;
                }
                else
                {       // at end
                        a=line2->fhole;
                        line3=line2;                    // line3 shouldn't be used
                }
        }
        else
        {
                while (line3->link!=line2)
                {
                        line3=line3->link;
                }
                if (finitial->link!=line2)
                {
                        a=line3->lhole+1;
                }
                else
        {
                a=line2->fhole;
        }
}
// line up holes to get rdist1, angle1.
if ((line1->fhole<=a)&&(line1->lhole>=a))
{       // can get good match
        iq=line1->fhole;
        while (iq<a)
        {
                iq++;
                hole1=hole1->link;
        }
        iq=line2->fhole;
        while (iq<a)
        {
                iq++;
                hole2=hole2->link;
        }
        iqq=a;
        x1=hole2->ival-hole1->ival;
        y1=hole2->jval-hole1->jval;
        rdist1=x1*x1+y1*y1;
        angle1=atan2(y1,x1);
        node=hole2;
}
else
{
        if (line1->fhole>line1->lhole)
        {       // bottom line, nline in use.
                hole3=line3->plink;
                while (hole3->link!=NULL)
                {
                        hole3=hole3->link;
                }
                hole2=line2->plink;
                iq=line2->fhole;
                while (iq<a-1)
                {
                        iq++;
                        hole2=hole2->link;
                }
                iqq=line1->fhole;
                x1=hole3->ival-hole2->ival;
                y1=hole3->jval-hole2->jval;
        }
        else
        {
                if (line1->fhole>a)
                {       // move hole2
                        iq=line2->fhole;
                        while (iq<line1->fhole)
                        {
                                iq++;
                                hole2=hole2->link;
                        }
                        iqq=line1->fhole;
                }
                else
                {
                        iq=line2->fhole;
                        while (iq<line1->lhole)
                        {
                                iq++;
                                hole2=hole2->link;
                        }
                        while (hole1->link!=NULL)
                        {
                                hole1=hole1->link;
                        }
                        iqq=line1->lhole;
                }
                x1=hole2->ival-hole1->ival;
                y1=hole2->jval-hole1->jval;
        }
        rdist1=x1*x1+y1*y1;
        angle1=atan2(y1,x1);
        node=line2->plink;
        iq=line2->fhole;
        while (iq<a)
        {
                iq++;
                node=node->link;
        }
}
line3=finitial->link;
while ((node!=NULL)&&(done==FALSE))
{       // check for holes in line line2->link
        if (node->link!=NULL)
        {       // update spec
                x3=node->link->ival-node->ival;
                y3=node->link->jval-node->jval;
        rdist3=x3*x3+y3*y3;
        angle3=atan2(y3,x3);
}
Look_for_odd_lines(finitial,node,angle1,rdist1,angle3,rdist3,a);
if (line3==finitial->link)
{       // no holes found
        if ((hole2->link!=NULL)&&(hole1->link!=NULL)&&(iqq==a))
        {
                hole2=hole2->link;
                hole1=hole1->link;
                x1=hole2->ival-hole1->ival;
                y1=hole2->jval-hole1->jval;
                rdist1=x1*x1+y1*y1;
                angle1=atan2(y1,x1);
        }
        node=node->link;
        a++;
}
else
{       // add line3 at end of line1 and makeup any that are missing
        done=TRUE;
        line3=finitial->link;
        if (line3->link!=line2)
        {       // i.e. line 3 should not be on top row
                finitial->link=line3->link;
                if (line2->link==NULL)
                {       // at bottom
                        line2->link=line3;
                        line3->link=NULL;
                }
                else
                {       // somewhere in between
                        hole3=line3->plink;
                        if (line1->link==line2)
                        {
                                line1=line2->link;
                        }
                        else
                        {
                                line1=finitial;
                                while (line1->link!=line2)
                                {
                                        line1=line1->link;
                                }
                        }
                        hole1=line1->plink;
        while (hole1->link!=NULL)
        {
                hole1=hole1->link;
        }
        if (line1->lhole>=line3->fhole)
        {       // overlapping/matching
                iq=line3->fhole;
                while(iq<=line1->lhole)
                {       // find new start of line3
                        tmp2=hole3;
                        hole3=hole3->link;
                        tmp2->link=finitial->plink-
                                >link;
                        finitial->plink->link=tmp2;
                        iq++;
                }
        }
        else
        {       // missing holes
                tmp=hole1;
                if (line1->lhole<line3->fhole-1)
                for (iq=line1->lhole+1;iq<line3-
                        >fhole;iq++)
                {       // make up missing holes
                        tmp2=(struct fpxl
                          *)malloc(sizeof(struct fpxl));
                        if(tmp2==NULL) printf("Out of
                          Memory");
                        frac=(float)((float)(iq-line1-
                          >lhole)/(float)(line3->fhole-
                          line1->lhole));
                        tmp2->ival=(tmp->ival)*(1-
                          frac)+(hole3->ival)*frac;
                        tmp2->jval=(tmp->jval)*(1-
                          frac)+(hole3->jval)*frac;
                        tmp2->real=FALSE;
                        tmp2->size=1;
                        hole1->link=tmp2;
                        hole1=tmp2;
                }
        }
        line1->lhole=line3->lhole;
        hole1->link=hole3;
        free(line3);
    }
}
```

```
                }
        }       // that's the upper first found. :-)
}


void Add_fhole(struct fposit *short_line, struct fposit *long_line)
{       // used to add holes to square up grid
        int t;
        struct fpxl *tmp, *hole;
        tmp=(struct fpxl *)malloc(sizeof(struct fpxl));
        if(tmp==NULL) printf("Out of Memory");
        tmp->link=short_line->plink;
        short_line->plink=tmp;
        (short_line->fhole)—;
        hole=long_line->plink;
        t=long_line->fhole;
        while (t<(short_line->fhole-1))
        {       // get to right place in long line
                hole=hole->link;
                t++;
        }
        tmp->ival=tmp->link->ival-hole->link->ival+hole->ival;
        tmp->jval=tmp->link->jval-hole->link->jval+hole->jval;
        tmp->real=FALSE;
        tmp->size=1;
}

void Add_lhole(struct fposit *short_line, struct fposit *long_line)
{       // used to add holes to square up grid
        int i;
        struct fpxl *tmp, *lhole, *shole;
        shole=short_line->plink;
        lhole=long_line->plink;
        if (short_line->fhole<long_line->fhole)
        {
                i=short_line->fhole;
                while (i<long_line->fhole)
                {
                        shole=shole->link;
                        i++;
                }
        }
        if (long_line->fhole<short_line->fhole)
        {
                i=long_line->fhole;
                while (i<short_line->fhole)
                {
                        lhole=lhole->link;
                        i++;
                }
        }
        while(shole->link!=NULL)
        {
                shole=shole->link;
                lhole=lhole->link;
        }
        tmp=(struct fpxl *)malloc(sizeof(struct fpxl));
        if(tmp==NULL) printf("Out of Memory");
        tmp->link=NULL;
        shole->link=tmp;
        (short_line->lhole)++;
        tmp->ival=shole->ival+lhole->link->ival-lhole->ival;
        tmp->jval=shole->jval+lhole->link->jval-lhole->jval;
        tmp->real=FALSE;
        tmp->size=1;
}


void Store_Size(struct fposit *finitial)
{       // output only. use to analyse grid
        struct fposit *line;
        int val;
        char a1,a2;
        FILE *fp1;
        fp1 = fopen("gridata.m", "wt");
        line=finitial->link;
        fwrite("v=[",1,3,fp1);
        while (line!=NULL)
        {       // print true grid data for a line
                val=line->fhole;
                if (val<0) {a1=45;val=-val;fwrite(&a1,sizeof(char),1,fp1);}
                a1=(char)(48+(val/10));
                val=val-10*(int)(val/10);
                a2=(char)(48+val);
                val=(val-10*(int)(val/10));
                if (a1!=48)    fwrite(&a1,sizeof(char),1,fp1);
                fwrite(&a2,sizeof(char),1,fp1);
                fwrite(",",1,1,fp1);
                val=line->lhole;
                if (val<0) {a1=45;val=-val;fwrite(&a1,sizeof(char),1,fp1);}
                a1=(char)(48+(val/10));
                val=val-10*(int)(val/10);
                a2=(char)(48+val);
                val=(val-10*(int)(val/10));
                if (a1!=48)    fwrite(&a1,sizeof(char),1,fp1);
                fwrite(&a2,sizeof(char),1,fp1);
                fwrite(";",1,1,fp1);
                line=line->link;
        }
        fwrite("];",1,2,fp1);
        a1=10;
        fwrite(&a1,1,1,fp1);
        fclose(fp1);
}


void Square_Grid(struct fposit *finitial)
{       // makes grid into a square
        int done,fhol,lhol;
        struct fposit *line1, *line2, *line3;
        done=FALSE;
        do
        {       // repeat until grid is square
                line2=finitial->link;
                while (line2->link!=NULL)
                {       // look for missing holes in grid
                        line1=line2;
                        line2=line2->link;
                        if (line2->fhole<line1->fhole) Add_fhole(line1,line2);
                        // line2 longer
                        if (line1->fhole<line2->fhole) Add_fhole(line2,line1);
                        if (line2->lhole>line1->lhole) Add_lhole(line1,line2);
                        // line2 longer
                        if (line1->lhole>line2->lhole) Add_lhole(line2,line1);
                }
                line3=finitial->link;
                fhol=line3->fhole;
                lhol=line3->lhole;
                done=TRUE;
                while(line3->link!=NULL)
                {       // see if grid is square
                        line3=line3->link;
                        if (line3->fhole!=fhol) done=FALSE;
                        if (line3->lhole!=lhol) done=FALSE;
                }
        }       while (done==FALSE);
}
```

```
void Store_Grid(struct fposit *finitial)
{       // output only. use to analyse grid
        struct fpxl *hole;// *dhole;
        struct fposit *line;
        float val;
        char a1,a2,a3,a4,a5,a6,a7,a0;
        FILE *fp1;
        fp1 = fopen("gridata.m", "at");
        line=finitial->link;
        fwrite("u=[",1,3,fp1);
        a1=10;
        fwrite(&a1,1,1,fp1);
        while (line!=NULL)
        {       // each line in grid
                hole=line->plink;
                while (hole!=NULL)
                {       // each hole in line (print ival)
                        val=hole->real;
                        if (val==0)
                        {
                                fwrite("0,",1,2,fp1);
                        }
                        else
                        {
                                fwrite("1,",1,2,fp1);
                        }
                        hole=hole->link;
                }
                fwrite(";",1,1,fp1);
                a1=10;
                fwrite(&a1,1,1,fp1);
                line=line->link;
        }
        fwrite("];",1,2,fp1);
        fwrite(&a1,1,1,fp1);
        line=finitial->link;
        fwrite("x=[",1,3,fp1);
        a1=10;
        fwrite(&a1,1,1,fp1);
        while (line!=NULL)
        {       // each line in grid
                hole=line->plink;
                while (hole!=NULL)
                {       // each hole in line (print ival)
                        val=hole->ival;
                if (val<0) {a1=45;val=-val;fwrite(&a1,sizeof(char),1,fp1);}
                a0=(char)(48+(val/1000));
                val=val-1000*(int)(val/1000);
                a1=(char)(48+(val/100));
                val=val-100*(int)(val/100);
                a2=(char)(48+(val/10));
                val=(val-10*(int)(val/10));
                a3=(char)(48+val);
                val=10*(val-(int)val);
                a4=(char)(48+val);
                val=10*(val-(int)val);
                a5=(char)(48+val);
                val=10*(val-(int)val);
                a6=(char)(48+val);
                val=10*(val-(int)val);
                a7=(char)(48+val);
                if (a0==48){a0=32; if (a1==48){a1=32; if (a2==48) a2=32;}}
                fwrite(&a0,sizeof(char),1,fp1);
                fwrite(&a1,sizeof(char),1,fp1);
                fwrite(&a2,sizeof(char),1,fp1);
                fwrite(&a3,sizeof(char),1,fp1);
                fwrite(".",1,1,fp1);
                fwrite(&a4,sizeof(char),1,fp1);
                fwrite(&a5,sizeof(char),1,fp1);
                fwrite(&a6,sizeof(char),1,fp1);
                fwrite(&a7,sizeof(char),1,fp1);
                fwrite(",",1,1,fp1);
                hole=hole->link;
        }
        fwrite(";",1,1,fp1);
        a1=10;
        fwrite(&a1,1,1,fp1);
        line=line->link;
}
fwrite("];",1,2,fp1);
fwrite(&a1,1,1,fp1);
line=finitial->link;
fwrite("y=[",1,3,fp1);
a1=10;
fwrite(&a1,1,1,fp1);
while (line!=NULL)
{       // each line in grid
        hole=line->plink;
        while (hole!=NULL)
        {       // each hole in line (print jval)
                val=hole->jval;
```

```
            if (val<0) {a1=45;val=-val;fwrite(&a1,sizeof(char),1,fp1);}
            a0=(char)(48+(val/1000));
            val=val-1000*(int)(val/1000);
            a1=(char)(48+(val/100));
            val=val-100*(int)(val/100);
            a2=(char)(48+(val/10));
            val=(val-10*(int)(val/10));
            a3=(char)(48+val);
            val=10*(val-(int)val);
            a4=(char)(48+val);
            val=10*(val-(int)val);
            a5=(char)(48+val);
            val=10*(val-(int)val);
            a6=(char)(48+val);
            val=10*(val-(int)val);
            a7=(char)(48+val);
            if (a0==48){a0=32; if (a1==48){a1=32; if (a2==48) a2=32;}}
            fwrite(&a0,sizeof(char),1,fp1);
            fwrite(&a1,sizeof(char),1,fp1);
            fwrite(&a2,sizeof(char),1,fp1);
            fwrite(&a3,sizeof(char),1,fp1);
            fwrite(".",1,1,fp1);
            fwrite(&a4,sizeof(char),1,fp1);
            fwrite(&a5,sizeof(char),1,fp1);
            fwrite(&a6,sizeof(char),1,fp1);
            fwrite(&a7,sizeof(char),1,fp1);
            fwrite(",",1,1,fp1);
            hole=hole->link;
        }
        fwrite(";",1,1,fp1);
        a1=10;
        fwrite(&a1,1,1,fp1);
        line=line->link;
    }
    fwrite("];",1,2,fp1);
    fwrite(&a1,1,1,fp1);
    fclose(fp1);
}


void Definegrid(struct fposit *finitial)
{
    int ct,done,holecount,newcount;
    double angle1,angle3;
    float x1,y1,rdist1,rdist3,x3,y3;
    struct fposit *line1, *line2, *line3, *nline;
struct fpxl *node, *tmp2;
printf("working out structure of grid");
// this version to find three holes in top left corner
ct=1;
line1=malloc(sizeof(struct fposit));          // create new list
if (line1==NULL) printf("Out of memory");
line1->link=NULL;
finitial->link=line1;
tmp2=finitial->plink;
Findorigin(finitial,ct);
line2=malloc(sizeof(struct fposit));
if (line2==NULL) printf("Out of memory");
line1->link=line2;
line3=malloc(sizeof(struct fposit));
if (line3==NULL) printf("Out of memory");
line2->link=line3;
do
{       // find axes.
        done=TRUE;
        Findaxes(finitial);
        printf("axes found");
        x1=line3->plink->ival-line1->plink->ival;  // horizontal
        y1=line3->plink->jval-line1->plink->jval;
        rdist1=x1*x1+y1*y1;
        x3=line2->plink->ival-line1->plink->ival;  // vertical
        y3=line2->plink->jval-line1->plink->jval;
        rdist3=x3*x3+y3*y3;
        angle1=atan2(x1,y1);
        angle3=atan2(x3,y3);
        if ((angle1>pi/2)&&(angle3<-pi/2)) angle3=angle3+2*pi;
        // correct for 180`
        if ((angle3>pi/2)&&(angle1<-pi/2)) angle1=angle1+2*pi;
        //     "
        printf("\nThe points found are origin: %g %g; row: %g %g; column %g
                %g\n",line1->plink->ival,
        line1->plink->jval,line3->plink->ival,line3->plink->jval,line2-
                >plink->ival,line2->plink->jval);
        printf("Giving distances^2 of %g %g, and angles of %g %g\n",rdist1,
                rdist3,angle1,angle3);
        if((pow(cos(angle3-angle1),2)>.1)
                ||(rdist1*9/rdist3/4<.9)||(rdist3*4/rdist1/9<.9))
                // limits assume grid is square to axes.
        {       // didn't work, get new origin.
                // NB pixels are not square - thus rdist*2
                printf(" not accepted \n");
                ct++;
                Findorigin(finitial,ct);
                done=FALSE;
        }       // NB no get out clause if no axes found.
        if((finitial->plink==finitial->link->plink)&&(ct>1))
        {
                done=TRUE;
                printf("\n......Failed to find Origin.....\n");
        }
}while(done==FALSE);
node=finitial->plink;
while (node->link!=line1->plink)
        node=node->link;
node->link=node->link->link;
finitial->plink=node;
while (node->link!=line2->plink)
        node=node->link;
node->link=node->link->link;
finitial->plink=node;
line1->plink->link=NULL;
line2->plink->link=NULL;
line2->link=NULL;
tmp2=line1->plink;
node=line3->plink;
free(line3);
line2->fhole=1;
line2->lhole=1;
line1->fhole=1;
line1->lhole=1;
x1=node->ival-tmp2->ival;
y1=node->jval-tmp2->jval;
angle1=atan2(y1,x1);
rdist1=x1*x1+y1*y1;
Do_a_line(finitial,line1,angle1,rdist1);
Do_a_line(finitial,line2,angle1,rdist1);
newcount=3;
do
{
        printf("y");
        holecount=newcount;
        line1=finitial->link;
        if (line1->plink->link!=NULL)
        {       // update spec
                x3=line1->plink->link->ival-line1->plink->ival;
                y3=line1->plink->link->jval-line1->plink->jval;
                angle3=atan2(y3,x3);
                rdist3=x3*x3+y3*y3;
```

```
}
nline=malloc(sizeof(struct fposit));
if (nline==NULL) printf("Out of memory");
nline->link=finitial->link;
finitial->link=nline;
nline->fhole=nline->link->fhole-1;
nline->lhole=nline->link->fhole-2;
tmp2=(struct fpxl *)malloc(sizeof(struct fpxl));
if(tmp2==NULL) printf("Out of Memory");
tmp2->link=NULL;
tmp2->ival=2049;
tmp2->jval=2050;
tmp2->real=FALSE;
tmp2->size=1;
nline->plink=tmp2;
line1=finitial->link;
line2=line1->link;
while(line2->link!=NULL)
{       // check below.
        if (line2->fhole<line2->link->fhole)
        {       // does line2 start first?
                Check_to_fside(finitial,rdist3,angle3,line1,line2);
        }
        if (line2->lhole>line2->link->lhole)
        {       // does line2 end last
                Check_to_lside(finitial,rdist3,angle3,line1,line2);
        }
        line1=line2;
        line2=line2->link;
        if (line2->plink->link!=NULL)
        {       // update spec
                x3=line2->plink->link->ival-line2->plink->ival;
                y3=line2->plink->link->jval-line2->plink->jval;
                angle3=atan2(y3,x3);
                rdist3=x3*x3+y3*y3;
        }
}
Check_to_lside(finitial,rdist3,angle3,line1,line2);
// find third line here
while (line2->link!=NULL)
{
        line1=line2;
        line2=line2->link;
        if (line2->plink->link!=NULL)
        {       // update spec
                x3=line2->plink->link->ival-line2->plink->ival;
                y3=line2->plink->link->jval-line2->plink->jval;
                angle3=atan2(y3,x3);
                rdist3=x3*x3+y3*y3;
        }
        Check_to_lside(finitial,rdist3,angle3,line1,line2);
        // find new lines here
}
finitial->link=nline->link;
free(tmp2);
free(nline);
printf("x");
line2=finitial;
line1=finitial->link;
while (line1!=NULL)
{       // check above
        line3=line2;
        line2=line1;
        line1=line1->link;
        if (line2->plink->link!=NULL)
        {       // update spec
                x3=line2->plink->link->ival-line2->plink->ival;
                y3=line2->plink->link->jval-line2->plink->jval;
        }
}       // line1 is bottom row.
angle3=atan2(y3,x3);
rdist3=x3*x3+y3*y3;
nline=malloc(sizeof(struct fposit));
if (nline==NULL) printf("Out of memory");
line1=nline;
line2->link=nline;
nline->link=NULL;
nline->fhole=line2->fhole-1;
nline->lhole=line2->fhole-2;
tmp2=(struct fpxl *)malloc(sizeof(struct fpxl));
if(tmp2==NULL) printf("Out of Memory");
tmp2->link=NULL;
tmp2->ival=2049;
tmp2->jval=2050;
tmp2->real=FALSE;
tmp2->size=1;
nline->plink=tmp2;
while(line2!=finitial)
{
        if ((line2->fhole<line3->fhole)||(line3==finitial))
        {       // does line1 start first
                Check_to_fside(finitial,rdist3,angle3,line1,line2);
                }
                if (line3->link!=line2) line3=line3->link;
                if ((line2->lhole>line3->lhole)||(line3==finitial))
                {   // does line1 end last
                    Check_to_lside(finitial,rdist3,angle3,line1,line2);
                }
                line1=line2;
                line2=finitial;
                while (line2->link!=line1)
                {   // move to next line to test
                    line3=line2;
                    line2=line2->link;
                }
                if (line2->plink->link!=NULL)
                {   // update spec
                    x3=line2->plink->link->ival-line2->plink->ival;
                    y3=line2->plink->link->jval-line2->plink->jval;
                    angle3=atan2(y3,x3);
                    rdist3=x3*x3+y3*y3;
                }
            }
            newcount=0;
            line1=finitial;
            while (line1->link!=nline)
            {   // check above
                line1=line1->link;
            }
            line1->link=NULL;
            free(tmp2);
            free(nline);
            line1=finitial->link;
            while(line1!=NULL)
            {
                newcount=newcount+line1->lhole-line1->fhole+1;
                line1=line1->link;
            }
        }
        while (holecount!=newcount);
    }
```

```
void savim(unsigned char image[MMAX][NMAX],char outg[60])
{       // used to check holes are found correctly.
        FILE *fp1;
        char a;
        int i,j;
        printf("saving image");
        fp1 = fopen(outg, "wb");
        for (j=0;j<MMAX;j++) for(i=0;i<NMAX;i++)
        {       // save each pixel
                a=(char)image[j][i];
                fwrite(&a,1,1,fp1);
        }
        fclose(fp1);
        printf(" done.\n");
}


void main (void)
{
        char argg[60]="input.raw";
        char outg[60]="modg.raw";
        struct posit *initial,*del;
        struct fposit *finitial;
        struct pxl *pixel, *tmp;
        initial=malloc(sizeof(struct posit));
        if(initial==NULL) printf("Out of memory\n");
        finitial=malloc(sizeof(struct fposit));
        if(finitial==NULL) printf("Out of memory\n");
        finitial->plink=NULL;
        finitial->link=NULL;

        Findholes(bimage,initial,argg);      // Also produces mimage
        Mapholes(bimage,initial);
        Findcentre(bimage,initial,finitial);
        savim(mimage,outg);
        while (initial!=NULL)
        {       // delete initial linked list - to free up memory
                if (initial->plink!=NULL)
                {       // if there are pixels in list
                        pixel=initial->plink->link;
                        while (pixel!=initial->plink)
                        {       // delete pixels in list
                                tmp=pixel->link;
                                free(pixel);
                                pixel=tmp;
                        }
                        free(pixel);
                }
                if (initial->link!=NULL)
                {       // delete initial
                        del=initial->link;
                        free(initial);
                        initial=del;
                }
                else
                {       // delete last initial
                        free(initial);
                        initial=NULL;
                }
        }
        Definegrid(finitial);
        Store_Size(finitial);
        Square_Grid(finitial);
        Store_Grid(finitial);
        printf(" done\n");
}
```

Appendix III - Matlab Code

Appendix III.1 - mmodkx.m

```
function [kx,x,y]=mmodkx(x0,y0,u,k,kx);
%       MMODKX.M - Alters grids to remove manufacturing errors.
%       used by MMOD, uses ZMOD

%       R. J. Drew 25/11/1998

'shifting'
ta=toc;
[n1,n2]=size(x0);
x(1:2:n1,1:2:n2)=x0(1:2:n1,1:2:n2)+kx(2);
y(1:2:n1,1:2:n2)=y0(1:2:n1,1:2:n2)+kx(1);
x(2:2:n1,1:2:n2)=x0(2:2:n1,1:2:n2)+kx(4);
y(2:2:n1,1:2:n2)=y0(2:2:n1,1:2:n2)+kx(3);
x(1:2:n1,2:2:n2)=x0(1:2:n1,2:2:n2)+kx(6);
y(1:2:n1,2:2:n2)=y0(1:2:n1,2:2:n2)+kx(5);
x(2:2:n1,2:2:n2)=x0(2:2:n1,2:2:n2)+kx(8);
y(2:2:n1,2:2:n2)=y0(2:2:n1,2:2:n2)+kx(7);
%       initial displacements
sb=[.05 .05 .05 .05 .05 .05 .05 .05];
zs=5;
for lps=1:15
 kx1=kx;
 for f=1:4
  kp1=kx1(f*2);
  kp2=kx1(f*2-1);
  for l=1:zs*2+1;
   kx1(f*2)=kp1+(l-zs-1)*sb(f*2);
   for m=1:zs*2+1;
    kx1(f*2-1)=kp2+(m-zs-1)*sb(f*2-1);
    x(2-rem(f,2):2:n1,ceil(f/2):2:n2)=x0(2-rem(f,2):2:n1,ceil(f/2):2:n2)+kx1(f*2);
    y(2-rem(f,2):2:n1,ceil(f/2):2:n2)=y0(2-rem(f,2):2:n1,ceil(f/2):2:n2)+kx1(f*2-1);
    best(1,m)=zmod(k,u,x,y);
   end;
  end;
  [t,q1]=min(min(best'));
  kx1(f*2)=kp1+(q1-zs-1)*sb(f*2);
  [t,q2]=min(min(best));
  kx1(f*2-1)=kp2+(q2-zs-1)*sb(f*2-1);
  x(2-rem(f,2):2:n1,ceil(f/2):2:n2)=x0(2-rem(f,2):2:n1,ceil(f/2):2:n2)+kx1(f*2);
  y(2-rem(f,2):2:n1,ceil(f/2):2:n2)=y0(2-rem(f,2):2:n1,ceil(f/2):2:n2)+kx1(f*2-1);
  if q1==1
   sb(f*2)=sb(f*2)*1.1;
  elseif q1==zs*2+1
   sb(f*2)=sb(f*2)*1.1;
  else
   sb(f*2)=sb(f*2)*.4;
  end;
  if q2==1
   sb(f*2-1)=sb(f*2-1)*1.1;
  elseif q2==zs*2+1
   sb(f*2-1)=sb(f*2-1)*1.1;
  else
   sb(f*2-1)=sb(f*2-1)*.4;
  end;
 end;
 kx=kx1;
end;
x(1:2:n1,1:2:n2)=x0(1:2:n1,1:2:n2)+kx(2);
y(1:2:n1,1:2:n2)=y0(1:2:n1,1:2:n2)+kx(1);
x(2:2:n1,1:2:n2)=x0(2:2:n1,1:2:n2)+kx(4);
y(2:2:n1,1:2:n2)=y0(2:2:n1,1:2:n2)+kx(3);
x(1:2:n1,2:2:n2)=x0(1:2:n1,2:2:n2)+kx(6);
y(1:2:n1,2:2:n2)=y0(1:2:n1,2:2:n2)+kx(5);
x(2:2:n1,2:2:n2)=x0(2:2:n1,2:2:n2)+kx(8);
y(2:2:n1,2:2:n2)=y0(2:2:n1,2:2:n2)+kx(7);
return
```

Appendix III.2 - dewarp.m

```
%       DEWARP.M - loads grid images, and pre-processes.
%       FINDHOLE.C is run next, and finds the grid.
%       Includes a number of variables that may need altering.
%       MMOD.M uses results from FINDHOLE.C to model
%       distortion.

%       R. J. Drew 25/1/1999

clear
%       load images:
FID=fopen('ggb3.raw','r');          % grid image (b3)
[a]=fread(FID,[512,512]);
fclose(FID);
FID=fopen('gbb.raw','r');           % blank image
[b]=fread(FID,[512,512]);
fclose(FID);

%       Filter to detect holes.
if min(b(:))==0                     % often caused by board
 u=1-ceil(floor(b/30)./(floor(b/30)+1));
 b=b+u.*mean(b(:));
end
%       Use when only target visible.
a=a./b*mean(b(:));
%       Use to deal with images larger than target area.
%a=a./(b-40)*(mean(b(:))-40);       % method changed here
a=a-min(a(:));
k=30;                               % user variable b:30, m:30, s:15
am=(a-k+abs(a-k))/2;
bimage(am')
colormap(gray);
title('Grid in grey')
figure
bimage(ceil(am./(am+1))')
colormap(gray);
title('Grid in two tone')
drawnow
FID=fopen('c:\matlab\bin\input.raw','w');
fwrite(FID,am,'char');
fclose(FID);
clear
```

```
tic
['close C window when it has finished']
!findgrid
clear
t=toc
%       Normalise results. (Do every time loading gridata).
gridata                             % loads results
x=x+1;y=y+1;                        % range from 0-511 to 1-512
[n1,n2]=size(x);
if x(1,1)>x(1,n2)
 x=x(:,n2:-1:1);
 y=y(:,n2:-1:1);
 u=u(:,n2:-1:1);
end;
if y(1,1)>y(n1,1)
 x=x(n1:-1:1,:);
 y=y(n1:-1:1,:);
 u=u(n1:-1:1,:);
end;
x=x';
y=y';
u=u';
bmesh([x y],'y')                    % see result
```

Appendix III.3 - bimage.m

```
function bimage(x,y,z)
%       BIMAGE.M - Produces normalised image to x,y co-ordinates.
%       Wants COLORMAP(GRAY) for binary images.

%       R. J. Drew 25/11/98

if x~=real(x)
 x=abs(x);
end
if nargin==3
 maxx=max(max(z));
 minn=min(min(z));
 diff=maxx-minn;
 image(x,y,(z-minn)*max(size(colormap))/diff);
else
 maxx=max(max(x));
 minn=min(min(x));
 diff=maxx-minn;
 image((x-minn)*max(size(colormap))/diff);
end
drawnow
```

Appendix III.4 - zmod.m

```
function [merit,rx,sx]=zmod(k,u,x,y,vx)
%       ZMOD.M - iterative function for modelling grid
%       If changing maths, also alter IMOD

%       R. J. Drew 25/11/1998

if nargin==4 | nargin==2
 vx=0;
end
if nargin<4
 x=real(u);
 y=imag(u);
 u=ones(size(x));
end
vx=vx(1,1);
[m,n]=size(x);
qx=(ones(m,1)*[1:n]).*u;
qy=([1:m]'*ones(1,n)).*u;
sx=qx+i*qy;

px=k(6)*(x-k(2));
py=k(5)*(y-k(1));
r=sqrt(px.^2+py.^2);
an=atan2(py,px);
an=an+k(7)+k(11)*r+k(12)*r.^2+k(8)*r.^3;
rd=r+k(9)*r.^2+k(10)*r.^3;
dx=(rd.*cos(an)+k(4)).*u;
dy=(rd.*sin(an)+k(3)).*u;
rx=dx+i*dy;
merit=sum(sum((qx-dx).^2+(qy-dy).^2));
if vx==3
 mmesh(dx+dy*i,qx+qy*i,10);
 drawnow
end
if vx==4
 bmesh([qy qx],'y');hold;
 bmesh([dy dx],'c');hold off;
 drawnow
end
return
```

Appendix III.5 - mmesh.m

```
function mmesh(a,b,k)
%       MMESH.M - draws lines indicating the difference between a and b
%       A and B are grids (x+yi). Origin at A. Includes 0,0.

%       R. J. Drew 25/11/1998

if nargin==2
 k=1;
end
if size(a)~=size(b)
 ['error']
 return
end

clf
hold on
a=a(:);
b=b(:);
for x=1:size(a)
 if a(x)~=0 | b(x)~=0
  plot([real(a(x)),real(a(x))+k*(real(b(x))-real(a(x)))],...
       [imag(a(x)),imag(a(x))+k*(imag(b(x))-imag(a(x)))])
 end
end
plot(real(a),imag(a),'r.')
plot(0,0,'k.')
hold off
```

Appendix III.6 - bmesh.m

```
function bmesh(a,c)
%      BMESH - draws a mesh. A can be either x+yi, or [x y]. C is colour.

%      R. J. Drew 16/2/1999

if nargin==1
 c='';
end
if ((real(a)==a)&(sum(abs(a(:)))~=0))
 [t,q]=size(a);
 q=(q/2);
 a2=a(1:t,1:q);                     % x - swap here
 a1=a(1:t,q+1:q*2);                 % y
else
 a1=real(a);
 a2=imag(a);
 [t,q]=size(a);
end
rows=size(a1,1);
columns=size(a1,2);
if ishold==0
 newplot;
 hold on;
 h=0;
else
 h=1;
end
for i=1:rows-1
 for j=1:columns
  if a1(i,j)~=0 | a2(i,j)~=0
   if a1(i+1,j)~=0 | a2(i+1,j)~=0
    plot([a1(i,j) a1(i+1,j)],[a2(i,j) a2(i+1,j)],c);
   end
  end
 end
end
for i=1:rows
 for j=1:columns-1
  if a1(i,j)~=0 | a2(i,j)~=0
   if a1(i,j+1)~=0 | a2(i,j+1)~=0
    plot([a1(i,j) a1(i,j+1)],[a2(i,j) a2(i,j+1)],c);
   end
  end
 end
end
for i=1:rows
 for j=1:columns
  if a1(i,j)~=0 | a2(i,j)~=0
   plot(a1(i,j),a2(i,j),'.k')
  end
 end
end
if h==0
 hold off;
end
```

Appendix III.7 - mmodall.m

```
%	MMODALL.M - Main body of distortion correction.
%	Used by Mmod files.
%	inputs k fno zs fq s u x y ta  outputs k s t
%	holding ta  notes lpx ct  version with sliding
%	used by MMOD,  uses ZMOD

%	R. J. Drew 25/11/1998

k1=k;kz=k;ct=0;
for f=1:fno;
 clear best
 kp1=kz(fq(f*2-1));
 kp2=kz(fq(f*2));
 for l=1:zs*2+1;
  if fq(f*2-1)~=fq(f*2)  | l==1
   kz(fq(f*2-1))=kp1+(l-zs-1)*s(fq(f*2-1));
   for m=1:zs*2+1;
    kz(fq(f*2))=kp2+(m-zs-1)*s(fq(f*2));
    best(l,m)=zmod(kz,u,x,y);
   end;
  end;
 end;
 if fq(f*2-1)~=fq(f*2)
  [t,q2]=min(min(best));
  kz(fq(f*2))=kp2+(q2-zs-1)*s(fq(f*2));
  [t,q1]=min(min(best'));
  kz(fq(f*2-1))=kp1+(q1-zs-1)*s(fq(f*2-1));
 else
  [t,q2]=min(best);
  kz(fq(f*2))=kp2+(q2-zs-1)*s(fq(f*2));
  q1=1;
 end;
 if fq(f*2-1)~=fq(f*2)
  if q1==1
   t=best(1,q2);
   dn=0;
   while (dn==0)
    ct=ct+1;
    tp=t;
    kz(fq(f*2-1))=kz(fq(f*2-1))-s(fq(f*2-1));
    t=zmod(kz,u,x,y);
   if tp<=t | ct>998
    dn=1;
   end;
  end;
  kz(fq(f*2-1))=kz(fq(f*2-1))+s(fq(f*2-1));
  s(fq(f*2-1))=s(fq(f*2-1))*1.5;
 elseif q1==zs*2+1
  t=best(zs*2+1,q2);
  dn=0;
  while (dn==0)
   ct=ct+1;
   tp=t;
   kz(fq(f*2-1))=kz(fq(f*2-1))+s(fq(f*2-1));
   t=zmod(kz,u,x,y);
   if tp<=t | ct>998
    dn=1;
   end;
  end;
  kz(fq(f*2-1))=kz(fq(f*2-1))-s(fq(f*2-1));
  s(fq(f*2-1))=s(fq(f*2-1))*1.5;
 else
  s(fq(f*2-1))=s(fq(f*2-1))*.3;
 end;
end;
if q2==1
 t=best(q1,1);
 dn=0;
 while (dn==0)
  ct=ct+1;
  tp=t;
  kz(fq(f*2))=kz(fq(f*2))-s(fq(f*2));
  t=zmod(kz,u,x,y);
  if tp<=t | ct>998
   dn=1;
  end;
 end;
 kz(fq(f*2))=kz(fq(f*2))+s(fq(f*2));
 s(fq(f*2))=s(fq(f*2))*1.5;
elseif q2==zs*2+1
 t=best(q1,zs*2+1);
 dn=0;
 while (dn==0)
  ct=ct+1;
  tp=t;
  kz(fq(f*2))=kz(fq(f*2))+s(fq(f*2));
  t=zmod(kz,u,x,y);
```

```
   if tp<=t | ct>998
    dn=1;
   end;
  end;
  kz(fq(f*2))=kz(fq(f*2))-s(fq(f*2));
  s(fq(f*2))=s(fq(f*2))*1.5;
 else
  s(fq(f*2))=s(fq(f*2))*.3;
 end;
 clear best
 for lpx=1:5
  kp1=kz(fq(f*2-1));
  kp2=kz(fq(f*2));
  for l=-1:1;
   kz(fq(f*2-1))=kp1+l*s(fq(f*2-1))/(lpx*2);
   for m=-1:1;
    kz(fq(f*2))=kp2+m*s(fq(f*2))/(lpx*2);
    best(l+2,m+2)=zmod(kz,u,x,y);
   end;
  end;
  [t,q2]=min(min(best));
  kz(fq(f*2))=kp2+(q2-2)*s(fq(f*2))/(lpx*2);
  if fq(f*2-1)~=fq(f*2)
   [t,q1]=min(min(best'));
   kz(fq(f*2-1))=kp1+(q1-2)*s(fq(f*2-1))/(lpx*2);
  end;
 end;
end;
k=kz;
lpx=1;
dn=0;
t=zmod(k,u,x,y);
while(dn==0)
 ka=k+lpx*(kz-k1)/20;
 tp=t;
 t=zmod(ka,u,x,y);
 if tp<=t | lpx>999
  dn=1;
 end
 lpx=lpx+1;
end
k=k+(lpx-2)*(kz-k1)/20;
t=zmod(k,u,x,y);
%[lpx t ct toc-ta],%ta=toc;
clear kz kp1 kp2 q1 q2 f l m dn tp k1 %ct lpx
```

Appendix III. 8 - mmod.m

```
%      MMOD.M - Models the distortion data previously
%      saved in GRIDATA.M by FINDHOLE.EXE. 2D version.
%      input is GRIDATA, intermediate: GRIXZ.MAT, GRIX.MAT.
%      output model: GRIXK.MAT
%      uses MMODALL, ZMOD, MMODKX

%      R. J. Drew 25/11/98

ta=0;
tb=0;
tf=0;
flops(0);
tic
if 1                  % Bypass initial value section to run more iterative loops
 gridata;
 [n1,n2]=size(x);
 x=x+1;y=y+1;         % Range from 0-511 to 1-512
 if x(1,1)>x(1,n2)
  x=x(:,n2:-1:1);
  y=y(:,n2:-1:1);
  u=u(:,n2:-1:1);
 end;
 if y(1,1)>y(n1,1)
  x=x(n1:-1:1,:);
  y=y(n1:-1:1,:);
  u=u(n1:-1:1,:);
 end;
 p6=x(:,2:n2)-x(:,1:n2-1);
 p6=1/mean(p6(:));
 p5=y(2:n1,:)-y(1:n1-1,:);
 p5=1/mean(p5(:));
 p1=256;
 p2=256;
 p3=1+(n1-1)*(p1-y(1,1))/(y(n1,n2)-y(1,1));
 p4=1+(n2-1)*(p2-x(1,1))/(x(n1,n2)-x(1,1));
 k=[p1,p2,p3,p4,p5,p6,0,0,0,0,0,0];
 ky=k;
 mr2=zmod(k,u,x,y,2)
%      Find centre of distortion
 zs=4;
 wa=250;
```

```
k1=k;
zz=1;
for lps=1:5
 clear mr r1 r2 r3 r4 r7 r8 r9 rA rB rC
 for qx1=1:zz*2+1
  for qx2=1:zz*2+1
   p1=k1(1)+wa*(qx1-zz-1);
   p2=k1(2)+wa*(qx2-zz-1);
   p3=1+(n1-1)*(p1-y(1,1))/(y(n1,n2)-y(1,1));
   p4=1+(n2-1)*(p2-x(1,1))/(x(n1,n2)-x(1,1));
   k=[p1,p2,p3,p4,p5,p6,0,0,0,0,0,0];
   s=[4 4 .1 .1 .001 .001 .02 .00001 .002 .0002 .002 .0002]*.15;
   fq=[3 4 9 10 7 11 8 12 5 6 0 0];
   fno=4;
   mmodall
   s(3:4)=s(3:4)/200;
   mmodall
   mmodall
   [t lpx ct toc-ta qx2 qx1]
   ta=toc;
   mr(qx1,qx2)=t;
   r1(qx1,qx2)=k(1);
   r2(qx1,qx2)=k(2);
   r3(qx1,qx2)=k(3);
   r4(qx1,qx2)=k(4);
   r7(qx1,qx2)=k(7);
   r8(qx1,qx2)=k(8);
   r9(qx1,qx2)=k(9);
   rA(qx1,qx2)=k(10);
   rB(qx1,qx2)=k(11);
   rC(qx1,qx2)=k(12);
  end
 end
 [t,q1]=min(min(mr'));
 [t,q2]=min(min(mr));[q2 q1 t]
 k=[r1(q1,q2),r2(q1,q2),r3(q1,q2),r4(q1,q2),p5,p6,r7(q1,q2),r8(q1,q2),...
    r9(q1,q2),rA(q1,q2),rB(q1,q2),rC(q1,q2)]
 mr2=[mr2;t];
 tb=[tb;toc];
 tf=[tf;flops];
 ky=[ky;k];
 k1=k;
 zmod(k,u,x,y,2)
 if lps==1 wa=200;end
 if lps==2 wa=100;end
 if lps==3 wa=60;end
```

```
  if lps==4 wa=30;end
 end
 save grixz
 %return
else
 load grixz
end;
ta=0;
tic
tc=tb(size(tb,1));
tg=tf(size(tf,1));
zs=1;
%steps=zs*2+1
fq=[2 4 1 3 5 6 9 10 7 12 8 11];
fno=6;
zz=0;
dd=0;
while dd==0
 zz=zz+1;
 k2=k;
 s=[4 4 .1 .1 .001 .001 .02 .00001 .002 .0002 .002 .0002]*.15;
 mmodall;
 [t lpx ct toc-ta zz]
 ta=toc;
 ky=[ky;k];
 mr2=[mr2;t];
 tb=[tb;toc+tc];
 tf=[tf;flops];
 clear best
 for lps=1:size(ky)
  best(lps)=zmod(2*k-ky(lps,:),u,x,y);
 end;
 [t,q1]=min(best);
 k=2*k-ky(q1,:);
 ky=[ky;k];
 mr2=[mr2;t];
 tb=[tb;toc+tc];
 tf=[tf;flops];
 t=zmod(k,u,x,y);
 clear best
 if zz==1
  x0=x;y0=y;
  [kx,x,y]=mmodkx(x,y,u,k,zeros(1,8));
 end
 save grix
 if(((sum(abs(k-ky(size(ky,1)-2,:)))~=0)...
      &(mr2(max([1 size(mr2,1)-10]))-t>.0000001))...
      |(size(mr2,1)<10))
  dd=0;
 else
  dd=1;
 end
 if (dd==1)&(zs==1)
  zs=15;
  dd=0;
 else
  zs=1;
 end
 [t q1 zs dd]
end
save grixk k
return
```

Appendix III.9 - imod.m

```
%       IMOD.M - pixel mapping
%       Finds mapping from undistorted to distorted image.
%       Saves mapping of each pixel in IMM.MAT.
%       Uses ZMOD. Also uses model for reverse mapping, so alterations
%       in ZMOD must be repeated here.

%       R. J. Drew 25/11/1998

clear
%       load k (save starting values)
load grixkb                                              % input model
zz=ones(512,1)*[1:512]+[1:512]'*ones(1,512)*i;
[w,zz]=zmod(k,zz);                                       % distorted grid
za=(min(real(zz(:,512)))-max(real(zz(:,1))))/511;        % scale
zb=(min(imag(zz(512,:)))-max(imag(zz(1,:))))/511;
ya=max(real(zz(:,1)));                                   % only need scale
yb=max(imag(zz(1,:)));
clear zz
tic                          % have origin (ya,yb), scale (za,zb) and model (h).
qx=ones(512,1)*[0:511]*za+ya;                      % undistorted grid locations
qy=[0:511]'*ones(1,512)*zb+yb;
th=atan2(qy-k(3),qx-k(4));                               % polar co-ordinates
rn=((qx-k(4)).^2+(qy-k(3)).^2).^.5;
clear qx qy
for l1=1:512
 for l2=1:512
  r=roots([k(10) k(9) 1 -rn(l1,l2)]);
  rz(l1,l2)=min(abs(r)+(floor(1-
sign(r))/2+ceil(abs(imag(r))./(abs(imag(r))+1)))*max(abs(r)));
end
 toc
 l1
end
clear rn
%       (find results with all roots)
x=k(2)+rz.*cos(th-(k(7)+k(11)*rz+k(12)*rz.^2+k(8)*rz.^3))/k(6);
y=k(1)+rz.*sin(th-(k(7)+k(11)*rz+k(12)*rz.^2+k(8)*rz.^3))/k(5);
clear r3 th
save imm x y
toc
return
```

Appendix III.10 - cout1.m

```
%       COUT1.M - Function produces undistorted versions of the image.

%       R. J. Drew 25/11/1998

clear
tic
tvv='ggb.raw';                    % input
tvn='ggbd.raw';                   % output
load imm                          % x - distortion grid from IMOD (immX)
[ms,ns]=size(x);
x=x+y*i;
%for n=1:24
 za=zeros(ms,ns);
 FID=fopen(tvv,'r');
 [a]=fread(FID,[512,512]);
 fclose(FID);
 a=a';
 for l1=1:ms
  for l2=1:ns
   if real(x(l1,l2))>0
    if real(x(l1,l2))<513
     if imag(x(l1,l2))>0
      if imag(x(l1,l2))<513
       n2=max([1 min([511 floor(real(x(l1,l2)))])]);
       n1=max([1 min([511 floor(imag(x(l1,l2)))])]);
       m2=real(x(l1,l2))-n2;
       m1=imag(x(l1,l2))-n1;
       za(l1,l2)=(1-m2)*((1-m1)*a(n1,n2)+m1*a(n1+1,n2))+...
                 m2*((1-m1)*a(n1,n2+1)+m1*a(n1+1,n2+1));
      end
     end
    end
   end
  end
  [l1 toc]
 end
 bimage(za);
 colormap(gray)
 FID=fopen(tvn,'w');
 fwrite(FID,za','char');
 fclose(FID);
```

```
%end
return

aa=[real(ba) imag(ba)];
ab=[real(bb) imag(bb)];
z1=(max(real(zz(:)))-min(real(zz(:))))/511;
z2=(max(imag(zz(:)))-min(imag(zz(:))))/511;
%cmesh(aa,'k',za);
%figure
%cmesh(ab,'k',zb);
```

Appendix III.11 - improc.m

```
%       IMPROC.M - Test to find ideal seed density
%       - inc. seed density and correlation match

%       R. J. Drew 25/11/1998

f=[ 0  0 -1 -2 -1  0  0;...
    0 -1  0  0  0 -1  0;...
   -1  0  1  2  1  0 -1;...
   -2  0  2  8  2  0 -2;...
   -1  0  1  2  1  0 -1;...
    0 -1  0  0  0 -1  0;...
    0  0 -1 -2 -1  0  0];
%       iron filings k=-8
% b: k=-8
% m: k=-12
% s: k=-18

for n=1:21
 tth=['i01.raw ';'i02.raw ';'i03.raw ';'i04.raw ';'i05.raw ';...
      'i06.raw ';'i07.raw ';'i08.raw ';'in1.raw ';'in2.raw ';...
      'in3.raw ';'in4.raw ';'in5.raw ';'in6.raw ';'in7.raw ';...
      'in8.raw ';'in9.raw ';'in10.raw';'in11.raw';'in12.raw';'in13.raw'];
 tti=['i11.raw ';'i12.raw ';'i13.raw ';'i14.raw ';'i15.raw ';...
      'i16.raw ';'i17.raw ';'i18.raw ';'in21.raw';'in22.raw';...
      'in23.raw';'in24.raw';'in25.raw';'in26.raw';'in27.raw';...
      'in28.raw';'in29.raw';'in30.raw';'in31.raw';'in32.raw';'in33.raw'];
 k=-8;
 FID=fopen(tth(n,:),'r');
 [a]=fread(FID,[512,512]);
 fclose(FID);
 a=a';
 c=1-ceil((a-3)./(abs(a-3)+1));
 b=ones(size(a)+size(f)-1)*round(mean(a(:)));
 b(floor(size(f,1)/2)+1:floor(size(f,1)/2)+size(a,1),...
   floor(size(f,2)/2)+1:floor(size(f,2)/2)+size(a,2))=(a-mean(a(:)))/std(a(:));
 b=conv2(b,f,'valid');
 c=ceil((c+1-ceil((-k+b)./(abs(-k+b)+1)))/2);
 d=[zeros(1,512);c(1:511,:)];
 d=ceil((c-d)/2);
 if 0
  subplot(2,2,1)
 bimage(a(101:200,101:200));
 subplot(2,2,2)
 bimage(-c(101:200,101:200));
 subplot(2,2,3)
 bimage(a(101:200,101:200).*(1-c(101:200,101:200))+max(a(:))*(c(101:200,101:200)));
 subplot(2,2,4)
 bimage(b(101:200,101:200));
end
sm(n)=sum(sum(c(17:496,17:496)))/480/480;
sd(n)=sum(sum(d(17:496,17:496)))/480/480;
[mean(b(:)) std(b(:)) max(b(:)) min(b(:))]
for l1=1:8
 for l2=1:8
  ms(l1,l2)=sum(sum(c(97+l1*32:128+l1*32,97+l2*32:128+l2*32)))/32/32;
  ds(l1,l2)=sum(sum(d(97+l1*32:128+l1*32,97+l2*32:128+l2*32)))/32/32;
 end
end
if n==1
 m2=ms(:);n2=ds(:);
else
 m2=[m2,ms(:)];
 n2=[n2,ds(:)];
end
clear b c d k tth
FID=fopen(tti(n,:),'r');
[b]=fread(FID,[512,512]);
fclose(FID);
b=b';
for l1=1:8
 for l2=1:8
  tic
  [aa,ab,bb]=corfuncx(a(97+l1*32:128+l1*32,97+l2*32:128+l2*32),...
                       b(1+l1*32:225+l1*32,1+l2*32:225+l2*32));
  dx=(1-2*ab./(aa+bb));    [d1 p1(2)]=min(min(dx));[d1 p1(1)]=min(min(dx'));
  ts(l1,l2)=(mean(min(dx'))-d1)/mean(min(dx'));
  dx(p1(1)-5:p1(1)+5,p1(2)-5:p1(2)+5)=ones(11);
  vs(l1,l2)=(min(min(dx))-d1)/min(min(dx));
  dq=(1-ab./(aa.*bb).^.5);[d2 p2(2)]=min(min(dq));[d2 p2(1)]=min(min(dq'));
  tq(l1,l2)=(mean(min(dq'))-d2)/mean(min(dq'));
  dq(p2(1)-5:p2(1)+5,p2(2)-5:p2(2)+5)=ones(11);
  vq(l1,l2)=(min(min(dq))-d2)/min(min(dq));
  dv=(aa+bb-2*ab);         [d3 p3(2)]=min(min(dv));[d3 p3(1)]=min(min(dv'));
  tv(l1,l2)=(mean(min(dv'))-d3)/mean(min(dv'));
  dv(p3(1)-5:p3(1)+5,p3(2)-5:p3(2)+5)=ones(11)*1000000;
  vv(l1,l2)=(min(min(dv))-d3)/min(min(dv));[p1 l2 l1 n toc]
  subplot(3,1,1)
```

```
   plot(min(dx'))
   subplot(3,1,2)
   plot(min(dq'))
   subplot(3,1,3)
   plot(min(dv'))
   drawnow;
  end
 end
 s1(n)=mean(vs(:));
 q1(n)=mean(vq(:));
 v1(n)=mean(vv(:));
 if n==1
  s2=vs(:);q2=vq(:);v2=vv(:);
  s3=ts(:);q3=tq(:);v3=tv(:);
 else
  s2=[s2,vs(:)];
  q2=[q2,vq(:)];
  v2=[v2,vv(:)];
  s3=[s3,ts(:)];
  q3=[q3,tq(:)];
  v3=[v3,tv(:)];
 end
 save crow
end
clear vv vs vq tti p3 p2 p1 f dv dq d3 d2 d1 bb dx tq tv ts
clear FID TICTOC a aa ab ans b l1 l2 n
save crow
plot(m2*100,q3,'+')
title('Ideal Seed Density')
plot(mean(m2),(m2),'.')
title('seed density variation across images')
return
```

Appendix III.12 - corl1.m

```
%	CORL1 - main correlation function.
%	most stuff is now in CORREL. This bit
%	loads images and saves results.
%	this version tests manufactured images
%	test set jewmxXa has noise of 2 rather than 10.
%	(spacing changes as well)

%	R. J. Drew 17/2/1999

clear

if 0
%	Make new image - minus noise
 load noise
 FID=fopen('gbb3.raw','r');		% (gbb3) undistorted
 [c]=fread(FID,[512,512]);
 fclose(FID);
 c=c';
 if 1
  %	how to get noise mask
  z=ones(512,1)*(-255:256);
  z=fftshift(z.^2+z'.^2).^.5;
  z=ceil((z-30)./(abs(z-30)+1));
  d=abs(fft2(c));
  d=sparse(ceil((d-15000)./(abs(d-15000)+1)));
  noise=z.*d;
  %	save noise noise
 end
 c=abs(ifft2((1-noise*.999).*fft2(c)));
 e=abs(ifft2((1-noise2*.999).*fft2(c)));
 if min(c==0) c=c+1;end;
 FID=fopen('g01b.raw','r');		% undistorted
 [a]=fread(FID,[512,512]);
 a=a';
 subplot(2,1,1)
 semilogy(fftshift(abs(fft2(a))))
 a=abs(ifft2((1-noise*.999).*fft2(a)));
 fclose(FID);
 subplot(2,1,2)
 semilogy(fftshift(abs(fft2(a))))
 an=randn(512)*2;
 a=a./c;
 an=an./c;
 ca=round((255*(an'-min(an(:)))./(max(a(:))-min(a(:)))));
 c=round((255*(a'-min(a(:)))./(max(a(:))-min(a(:)))));
 'noise level is '
 std(ca(:))                             %10
 FID=fopen('o01b.raw','w');
 fwrite(FID,c,'char');
 fclose(FID);
 clear c FID
end

%       stretch etc using PSP. - save as Oxxb.raw
%       now can correlate images.

%       variables that can be altered:
u1=32;                                  % window size (y)
u2=32;                                  % window size (x)
nlev=2;                                 % noise level (measured noise in images=10)
for nc=1:6
ncx=nc;                                 % different pre-processing method
%       alter variable changed for each run here
f1=[0  0 -1 -2 -1  0  0;...
    0 -1  0  0  0 -1  0;...
   -1  0  1  2  1  0 -1;...
   -2  0  2  8  2  0 -2;...
   -1  0  1  2  1  0 -1;...
    0 -1  0  0  0 -1  0;...
    0  0 -1 -2 -1  0  0];
f12=[1 -2 1;-2 4 -2;1 -2 1]/4;
for nb=1:1                              % image used for 'b' (large window)
 randn('seed',0);
 if nb<10
  tvv=['00',int2str(nb),'b.raw'];
 else
  tvv=['0',int2str(nb),'b.raw'];
 end
 FID=fopen(tvv,'r');
 [b]=fread(FID,[512,512]);
 fclose(FID);
 b=round(b'+randn(512)*nlev);
%       limit range
 z=ceil((b-255)./(abs(b-255)+1));
 b=b.*(1-z)+z*255;
 z=ceil((b)./(abs(b)+1));
 b=b.*(z);
```

```
if (nb>1 & nb<17)                   % correct PSP displacement error
 b=[b(:,512),b(:,1:511)];
end
note=[' '];
if ncx==1 b=b;note=['normal'];end;
if ncx==2 b=(b/std(b(:)));note=('std=1');end;
if ncx==3 b=(b-mean(b(:)));note=('mean=0');end;
if ncx==4 b=((b-mean(b(:)))/std(b(:)));note=('std=1, mean=0');end;
if ncx==5 b=conv2(b,fl,'valid');note=('edge filtered');end;
if ncx==6 b=conv2(b,fl2,'valid');note=('Pratt filtered');end;

clear z
for n=1:26                          % image used for 'a' (small window)
 if n<10
  tvv=['00',int2str(n),'b.raw'];
 else
  tvv=['0',int2str(n),'b.raw'];
 end
 FID=fopen(tvv,'r');
 [a]=fread(FID,[512,512]);
 fclose(FID);
 a=round(a'+randn(512)*nlev);
 z=ceil((a-255)./(abs(a-255)+1));
 a=a.*(1-z)+z*255;
 z=ceil((a)./(abs(a)+1));
 a=a.*(z);
 if (n>1 & n<17)                    % correct PSP displacement error
  a=[a(:,512),a(:,1:511)];
 end
 if ncx==1 a=a;end;
 if ncx==2 a=(a/std(a(:)));end;
 if ncx==3 a=(a-mean(a(:)));end;
 if ncx==4 a=((a-mean(a(:)))/std(a(:)));end;
 if ncx==5 a=conv2(a,fl,'valid');end;
 if ncx==6 a=conv2(a,fl2,'valid');end;
 clear z
 tic
 correl1
 [n toc]
 save jewmx qo qq qs qv qd
 clear da db
 end
end
z=([sum(abs(qd'));sum(abs(qo'));sum(abs(qv'));sum(abs(qs'));sum(abs(qq'))]');
f=1;
qd1=(ceil((abs(qd)-f)./(abs(qd-f)+1)));     % 1- good match. 0 - bad match
```

```
qo1=(ceil((abs(qo)-f)./(abs(qo-f)+1)));
qv1=(ceil((abs(qv)-f)./(abs(qv-f)+1)));
qs1=(ceil((abs(qs)-f)./(abs(qs-f)+1)));
qq1=(ceil((abs(qq)-f)./(abs(qq-f)+1)));
z1=([sum(abs(qd1'));sum(abs(qo1'));sum(abs(qv1'));sum(abs(qs1'));sum(abs(qq1'))]');
subplot(2,1,1)
plot([0:15]'*ones(1,5),z1(1:16,:))
subplot(2,1,2)
plot([0:10]'*ones(1,5),z1([1,17:26],:))
drawnow
[sum(z1(1:16,:))/16;sum(z1([1,17:26],:)/11)]
%       save here
if nc==1 save jewmx1a qo qq qs qv qd z z1 u1 u2 nlev note;end
if nc==2 save jewmx2a qo qq qs qv qd z z1 u1 u2 nlev note;end
if nc==3 save jewmx3a qo qq qs qv qd z z1 u1 u2 nlev note;end
if nc==4 save jewmx4a qo qq qs qv qd z z1 u1 u2 nlev note;end
if nc==5 save jewmx5a qo qq qs qv qd z z1 u1 u2 nlev note;end
if nc==6 save jewmx6a qo qq qs qv qd z z1 u1 u2 nlev note;end
end
return
```

Appendix III.13 - correl1.m

```
%       CORREL1.M - main correlation procedure.
%       Given two images (A,B), the node points in a grid
%       in image A are correlated to find their equivalent
%       locations in image B.
%       Processing in SBCOR.

%       This version is to test manufactured images.

%       R. J. Drew 31/12/1998

[7 16 32 64 80 128 135 141 144 160 162 256 270 288 300 324 512];

t=0;
sa=[u1,u2];                             % size of small
sb=[64,64];                             % size of large
for y=35.5:33:332.5
 for x=111.5:33:408.5
  t=t+1;
  if n<17
   py=y+y*(n-1)*16/512;
   px=x;
  else
   py=y;
   px=x+(254.5-y)*tan((n-16)/90*pi);
  end
  ae=[py,px];
  be=[y,x];
  sbcor1
  qs(n,t)=ps(1)+i*ps(2);
  qq(n,t)=pq(1)+i*pq(2);
  qv(n,t)=pv(1)+i*pv(2);
  qo(n,t)=po(1)+i*po(2);
  qd(n,t)=pd(1)+i*pd(2);
 end
end
return
```

Appendix III.14 - sbcor1.m

```
%       SBCOR1.M - the heart of calculations in CORREL1
%       inputs ae be sa sb a b
%       outputs pps ppq ppv ppo
%       includes absolute difference test.

%       R. J. Drew 5/1/1999

pa=round(ae+.5-sa/2);
pb=round(be+.5-sb/2);
a1x=a(pa(1):pa(1)+sa(1)-1,pa(2):pa(2)+sa(2)-1);
b1x=b(round(pb(1)):round(pb(1))+sb(1)-1,round(pb(2)):round(pb(2))+sb(2)-1);
if 0
 colormap(gray)
 subplot(2,2.7,1)
 bimage(a1x)
 subplot(1,1.4,1.4)
 r=ones(64);
 r(17:48,17:48)=r(17:48,17:48)*1.5;
 bimage(b1x.*r)
 pause
end
 [aa,ab,bb,a1,b1]=corfunc1(a1x,b1x);
ds=(1-2*ab./(aa+bb));
dq=(1-ab./(aa.*bb).^.5);
dv=(aa+bb-2*ab);
sz=sa(1)*sa(2);
do=1-(sz*ab-a1.*b1)./((sz*aa-a1.^2).*(sz*bb-b1.^2)).^.5;
for l1=1:sb(1)-sa(1)+1
 for l2=1:sb(2)-sa(2)+1
  dd(l1,l2)=sum(sum(abs(a1x-b1x(l1:l1+sa(1)-1,l2:l2+sa(2)-1))));
 end
end
 [d pps(2)]=min(min(ds));[d pps(1)]=min(min(ds'));
 [d ppq(2)]=min(min(dq));[d ppq(1)]=min(min(dq'));
 [d ppv(2)]=min(min(dv));[d ppv(1)]=min(min(dv'));
 [d ppo(2)]=min(min(do));[d ppo(1)]=min(min(do'));
%[d ppd(2)]=min(min(dd));[d ppd(1)]=min(min(dd'));
%clear aa ab bb a1 b1 pa d pb
%clear dq ds dv do
ps=pps-ceil((sb-sa)/2+1);
pq=ppq-ceil((sb-sa)/2+1);
```

```
pv=ppv-ceil((sb-sa)/2+1);
po=ppo-ceil((sb-sa)/2+1);
pd=ppd-ceil((sb-sa)/2+1);
%clear pps ppo ppv ppq

return
```

Appendix III.15 - corfunc1.m

```
function [aa,ab,bb,a1,b1]=corfunc1(a,b)
%       CORFUNC1 - The three terms (aa, ab, bb) needed by the correlation
%       algorithms are determined. The inputs 'a' and 'b' are the
%       images that need to be correlated. 'a' is the small, image
%       from the first image, and 'b' is the larger window from the
%       second window. The size of 'b' should be padded so that the
%       fft2 function is efficient.
%       version with A1, B1

%       R. J. Drew 21/12/1998

[ay,ax]=size(a);
[by,bx]=size(b);

aa=sum(a(:).^2);
a1=sum(a(:));

ffb=fft2(b,by,bx);
ffa=fft2(a,by,bx);
ffa=ffb.*conj(ffa);
clear ffb
ab=ifft2(ffa);
clear ffa
ab=real(ab(1:by-ay+1,1:bx-ax+1));

ffb=fft2(b.^2,by,bx);
ffa=fft2(ones(ay,ax),by,bx);
ffb=ffb.*conj(ffa);
bb=ifft2(ffb);
bb=real(bb(1:by-ay+1,1:bx-ax+1));

ffb=fft2(b,by,bx);
ffb=ffb.*conj(ffa);
clear ffa
b1=ifft2(ffb);
clear ffb
b1=real(b1(1:by-ay+1,1:bx-ax+1));

return

%       alternative method - sometimes faster
```

```
cx=ax+bx-1;
cy=ay+by-1;
d=zeros(cy,bx);
for r=1:ay
 d(r:r+by-1,1:bx)=b+d(r:r+by-1,1:bx);
end
e=zeros(cy,cx);
for q=1:ax
 e(1:cy,q:q+bx-1)=d+e(1:cy,q:q+bx-1);
end
b1=e(ay:by,ax:bx);
clear d e r q
d=zeros(cy,bx);
b=b.^2;
for r=1:ay
 d(r:r+by-1,1:bx)=b+d(r:r+by-1,1:bx);
end
e=zeros(cy,cx);
for q=1:ax
 e(1:cy,q:q+bx-1)=d+e(1:cy,q:q+bx-1);
end
bb=e(ay:by,ax:bx);
```

Appendix III.16 - tbcor.m

```
%       TBCOR.M - processing time for different methods:
%       use results to optimise CORFUNCX

%       R. J. Drew 13/12/1998
clear
bbx=round(256*rand(256)+.5)-2;
a=round(256*rand(6,32)+.5)-2;

for z=1:224
b=bbx(1:32+z,1:32+z);

flops(0)
tic
aa=sum(a(:).^2);
tt(1,z)=toc;
ff(1,z)=flops;
clear aa

flops(0)
tic
aa=sum(a(:));
tt(8,z)=toc;
ff(8,z)=flops;
clear aa

[ay,ax]=size(a);
[by,bx]=size(b);
%       Sum of difference method
flops(0)
tic
for pl1=1:by-ay+1
 for pl2=1:bx-ax+1
  p(pl1,pl2)=sum(sum(abs(a-b(pl1:ay+pl1-1,pl2:ax+pl2-1))));
 end
end
tt(5,z)=toc;
ff(5,z)=flops;
clear p pl1 pl2

[ay,ax]=size(a);
[by,bx]=size(b);
```

```
%       Sum of difference sign change
flops(0)
tic
for pl1=1:by-ay+1
 for pl2=1:bx-ax+1
  m=a-b(pl1:ay+pl1-1,pl2:ax+pl2-1);
  m1=abs(sign(m(2:size(m,1),:))-sign(m(1:size(m,1)-1,:)));
  m2=abs(sign(m(:,2:size(m,2)))-sign(m(:,1:size(m,2)-1)));
  p(pl1,pl2)=sum(m1(:))+sum(m2(:));
 end
end
tt(7,z)=toc;
ff(7,z)=flops;
clear p pl1 pl2 m m1 m2

%       SSDA method
[ay,ax]=size(a);
[by,bx]=size(b);
cy=by-ay+1;
cx=bx-ax+1;
flops(0)
tic
p=zeros(cy,cx);
for pl=1:(by-ay+1)*(bx-ax+1)
 pl1=0;
 while  (pl1<ay)&(p(rem(pl-1,cy)+1,ceil(pl/cy))<500)
  pl2=0;
  while  (pl2<ax)&(p(rem(pl-1,cy)+1,ceil(pl/cy))<500)
   p(rem(pl-1,cy)+1,ceil(pl/cy))=p(rem(pl-1,cy)+1,ceil(pl/cy))...
   +abs(a(pl1+1,pl2+1)-b(pl1+rem(pl-1,cy)+1,pl2+ceil(pl/cy)));
   pl2=pl2+1;
  end
  pl1=pl1+1;
 end
end
tt(6,z)=toc;
ff(6,z)=flops;
clear p pl1 pl2

flops(0)
tic
ffb=fft2(b,by,bx);
ffa=fft2(a,by,bx);
ffa=ffb.*conj(ffa);
clear ffb
ab=ifft2(ffa);
```

```
clear ffa
ab=real(ab(1:by-ay+1,1:bx-ax+1));
tt(2,z)=toc;
ff(2,z)=flops;
clear ab

%       Slow method
tic
flops(0)
ffb=fft2(b.^2,by,bx);
ffa=fft2(ones(ay,ax),by,bx);
ffa=ffb.*conj(ffa);
clear ffb
bb=ifft2(ffa);
clear ffa
bb=real(bb(1:by-ay+1,1:bx-ax+1));
ff(4,z)=flops;
tt(4,z)=toc;
clear bb

cx=ax+bx-1;
cy=ay+by-1;
flops(0)
tic
d=zeros(cy,bx);
for r=1:ay
 d(r:r+by-1,1:bx)=b+d(r:r+by-1,1:bx);
end
e=zeros(cy,cx);
for q=1:ax
 e(1:cy,q:q+bx-1)=d+e(1:cy,q:q+bx-1);
end
bb=e(ay:by,ax:bx);
tt(9,z)=toc;
ff(9,z)=flops;
clear d r q e bb

flops(0)
tic
d=zeros(cy,bx);
b=b.^2;
for r=1:ay
 d(r:r+by-1,1:bx)=b+d(r:r+by-1,1:bx);
end
e=zeros(cy,cx);
for q=1:ax
```

```
 e(1:cy,q:q+bx-1)=d+e(1:cy,q:q+bx-1);
end
bb=e(ay:by,ax:bx);
tt(3,z)=toc;
ff(3,z)=flops;
clear d r q e bb


[z,tt(2:6,z)']
save tmxx4 ff tt
end
notes2=['ff- flops, tt-time ct-SSDA count. 1:a2, 2:ab, ', ...
        ' 3:b2 4:b2(fft) 5:a-b 6:SSDA 7:sign 8:a 9:b']
notes=['times for correlation functions - see tbcor.m']
notes1=['average additions in SSDA is 6, 333 processor']
save tmxx4 ff tt notes notes2 notes1
```

Appendix III.17 - gridmet.m

```
function [a]=gridmet(qoz,mk)
%      GRIDMET.M - used to find a grid from a complete mess.....


%      R. J. Drew 7/2/1999


%qoz=[qo(n,1:10);qo(n,11:20);qo(n,21:30);qo(n,31:40);qo(n,41:50);...
%     qo(n,51:60);qo(n,61:70);qo(n,71:80);qo(n,81:90);qo(n,91:100)];
%      this lot is metric for grid quality.
if nargin==1
 mk=0;
end
ss=[1,   10,   18,   25,31,36,40,43,45;...
    1, 2,   11,   19,   26,32,37,41,44;...
    2, 3,10,12,   20,   27,33,38,42;...
    3, 4,11,13,18,21,   28,34,39;...
    4, 5,12,14,19,22,25,29,35;...
    5, 6,13,15,20,23,26,30,31;...
    6, 7,14,16,21,24,27,   32,36,
    7, 8,15,17,22,   28,   33,37,40;...
    8, 9,16,   23,   29,   34,38,41,43;...
    9,   17,   24,   30,   35,39,42,44,45];
%      NB use i* as close to -pi
%      Find best angle, an
goy(1:9,:)=angle(i*(qoz(1:9,:)-qoz(2:10,:)));
goy(10:17,:)=angle(i*(qoz(1:8,:)-qoz(3:10,:)));
goy(18:24,:)=angle(i*(qoz(1:7,:)-qoz(4:10,:)));
goy(25:30,:)=angle(i*(qoz(1:6,:)-qoz(5:10,:)));
goy(31:35,:)=angle(i*(qoz(1:5,:)-qoz(6:10,:)));
goy(36:39,:)=angle(i*(qoz(1:4,:)-qoz(7:10,:)));
goy(40:42,:)=angle(i*(qoz(1:3,:)-qoz(8:10,:)));
goy(43:44,:)=angle(i*(qoz(1:2,:)-qoz(9:10,:)));
goy(45:45,:)=angle(i*(qoz(1,:)-qoz(10,:)));
[g,h]=histogra(goy(:)*50/pi,0);
[koy,l]=max(g);
an=h(l);
f=.15;
hy=1-ceil((abs(goy-an*pi/50)-f)./(abs(abs(goy-an*pi/50)-f)+1));           % good-> 1
thy=hy(ss(:,1),:)+hy(ss(:,2),:)+hy(ss(:,3),:)+hy(ss(:,4),:)+...
    hy(ss(:,5),:)+hy(ss(:,6),:)+hy(ss(:,7),:)+hy(ss(:,8),:)+hy(ss(:,9),:);
tiy=ceil((thy-3)./(abs(thy-3)+1));
%      Find L - best length
```

```
roy(1:9,:)=abs(qoz(1:9,:)-qoz(2:10,:));
roy(10:17,:)=abs(qoz(1:8,:)-qoz(3:10,:))/2;
roy(18:24,:)=abs(qoz(1:7,:)-qoz(4:10,:))/3;
roy(25:30,:)=abs(qoz(1:6,:)-qoz(5:10,:))/4;
roy(31:35,:)=abs(qoz(1:5,:)-qoz(6:10,:))/5;
roy(36:39,:)=abs(qoz(1:4,:)-qoz(7:10,:))/6;
roy(40:42,:)=abs(qoz(1:3,:)-qoz(8:10,:))/7;
roy(43:44,:)=abs(qoz(1:2,:)-qoz(9:10,:))/8;
roy(45:45,:)=abs(qoz(1,:)-qoz(10,:))/9;
[g,h]=histogra(roy(:),0);
[koy,l]=max(g);
toy(1:9,:)=abs(abs(qoz(1:9,:)-qoz(2:10,:))-l);
toy(10:17,:)=abs(abs(qoz(1:8,:)-qoz(3:10,:))-l*2);
toy(18:24,:)=abs(abs(qoz(1:7,:)-qoz(4:10,:))-l*3);
toy(25:30,:)=abs(abs(qoz(1:6,:)-qoz(5:10,:))-l*4);
toy(31:35,:)=abs(abs(qoz(1:5,:)-qoz(6:10,:))-l*5);
toy(36:39,:)=abs(abs(qoz(1:4,:)-qoz(7:10,:))-l*6);
toy(40:42,:)=abs(abs(qoz(1:3,:)-qoz(8:10,:))-l*7);
toy(43:44,:)=abs(abs(qoz(1:2,:)-qoz(9:10,:))-l*8);
toy(45:45,:)=abs(abs(qoz(1,:)-qoz(10,:))-l*9);
f=5;
soy=sum(sum(ceil((abs(toy)-f)./(abs(abs(toy)-f)+1))));
sy=1-ceil((abs(toy)-f)./(abs(abs(toy)-f)+1));                    % good-> 1
tsy=sy(ss(:,1),:)+sy(ss(:,2),:)+sy(ss(:,3),:)+sy(ss(:,4),:)+...
    sy(ss(:,5),:)+sy(ss(:,6),:)+sy(ss(:,7),:)+sy(ss(:,8),:)+sy(ss(:,9),:);
tty=ceil((tsy-3)./(abs(tsy-3)+1));
tny=tiy.*tty;
gox(:,1:9)=angle((qoz(:,1:9)-qoz(:,2:10)));
gox(:,10:17)=angle((qoz(:,1:8)-qoz(:,3:10)));
gox(:,18:24)=angle((qoz(:,1:7)-qoz(:,4:10)));
gox(:,25:30)=angle((qoz(:,1:6)-qoz(:,5:10)));
gox(:,31:35)=angle((qoz(:,1:5)-qoz(:,6:10)));
gox(:,36:39)=angle((qoz(:,1:4)-qoz(:,7:10)));
gox(:,40:42)=angle((qoz(:,1:3)-qoz(:,8:10)));
gox(:,43:44)=angle((qoz(:,1:2)-qoz(:,9:10)));
gox(:,45:45)=angle((qoz(:,1)-qoz(:,10)));
[g,h]=histogra(gox(:)*50/pi,0);
[kox,l]=max(g);
an=h(l);
f=.15;
hx=1-ceil((abs(gox-an*pi/50)-f)./(abs(abs(gox-an*pi/50)-f)+1));  % good-> 1
thx=hx(:,ss(:,1))+hx(:,ss(:,2))+hx(:,ss(:,3))+hx(:,ss(:,4))+...
    hx(:,ss(:,5))+hx(:,ss(:,6))+hx(:,ss(:,7))+hx(:,ss(:,8))+hx(:,ss(:,9));
tix=ceil((thx-3)./(abs(thx-3)+1));
rox(:,1:9)=abs(qoz(:,1:9)-qoz(:,2:10));
rox(:,10:17)=abs(qoz(:,1:8)-qoz(:,3:10))/2;
rox(:,18:24)=abs(qoz(:,1:7)-qoz(:,4:10))/3;
rox(:,25:30)=abs(qoz(:,1:6)-qoz(:,5:10))/4;
rox(:,31:35)=abs(qoz(:,1:5)-qoz(:,6:10))/5;
rox(:,36:39)=abs(qoz(:,1:4)-qoz(:,7:10))/6;
rox(:,40:42)=abs(qoz(:,1:3)-qoz(:,8:10))/7;
rox(:,43:44)=abs(qoz(:,1:2)-qoz(:,9:10))/8;
rox(:,45:45)=abs(qoz(:,1)-qoz(:,10))/9;
[g,h]=histogra(rox(:),0);
[kox,l]=max(g);
tox(:,1:9)=abs(abs(qoz(:,1:9)-qoz(:,2:10))-l);
tox(:,10:17)=abs(abs(qoz(:,1:8)-qoz(:,3:10))-l*2);
tox(:,18:24)=abs(abs(qoz(:,1:7)-qoz(:,4:10))-l*3);
tox(:,25:30)=abs(abs(qoz(:,1:6)-qoz(:,5:10))-l*4);
tox(:,31:35)=abs(abs(qoz(:,1:5)-qoz(:,6:10))-l*5);
tox(:,36:39)=abs(abs(qoz(:,1:4)-qoz(:,7:10))-l*6);
tox(:,40:42)=abs(abs(qoz(:,1:3)-qoz(:,8:10))-l*7);
tox(:,43:44)=abs(abs(qoz(:,1:2)-qoz(:,9:10))-l*8);
tox(:,45:45)=abs(abs(qoz(:,1)-qoz(:,10))-l*9);
f=5;
sox=sum(sum(ceil((abs(tox)-f)./(abs(abs(tox)-f)+1))));
sx=1-ceil((abs(tox)-f)./(abs(abs(tox)-f)+1));                           % good-> 1
tsx=sx(:,ss(:,1))+sx(:,ss(:,2))+sx(:,ss(:,3))+sx(:,ss(:,4))+...
    sx(:,ss(:,5))+sx(:,ss(:,6))+sx(:,ss(:,7))+sx(:,ss(:,8))+sx(:,ss(:,9));
tnx=ceil((tsx-2)./(abs(tsx-2)+1));
ttx=ceil((tsx-3)./(abs(tsx-3)+1));
tnx=tix.*ttx;
tn=tnx|tny;
a=sum(tn(:));
if mk
 bmesh(qoz,'c')
 hold on
 bmesh((qoz.*tnx),'y')
 bmesh((qoz.*tny),'m')
 bmesh((qoz.*tnx.*tny),'k')
 plot(real(qoz.*tn),imag(qoz.*tn),'ro')
 hold off
 drawnow
 %[sum(tnx(:)) sum(tny(:)) sum(tnx(:).*tny(:)) sum(tnx(:).*(1-tny(:))) ...
 %sum((1-tnx(:)).*tny(:))]
end
return
```

Appendix III.18 - histogra.m

```
function [f,g]=histogra(ipt,c)
%       HISTOGRA.M - histgram of contents of array.
%       Done to nearest integer.

%       R. J. Drew 19/1/1999

ipt=ipt(:);
ipt=floor(ipt+.5);
mmin=min(min(ipt));
mmax=max(max(ipt));
ipt=ipt-mmin;
[m,n]=size(ipt);
f=zeros(mmax-mmin+1,1);
for i=1:m
 for j=1:n
  f(ipt(i,j)+1)=f(ipt(i,j)+1)+1;
 end;
end;
g=(mmin:mmax);
if nargin==1
 plot(g,f);
else
 if isstr(c)
  plot(g,f,c);
 end
 if c==0
  f=[zeros(mmin-1,1);f];
 end
end
```

Appendix III.19 - correl.m

```
%       CORREL.M - main correlation procedure.
%       Given two images (A,B), the node points in a grid
%       in image A are correlated to find their equivalent
%       locations in image B.
%       Processing in SBCOR. (variable 'be' correction)

%       R. J. Drew 12/2/1999

sa=[32,64];                     % size of small
sb=[256,256];                   % size of large
t=[32 32];                      % distance between windows

ao=ceil(size(a)/2);             % centre of image - constant.
ae=ao;                          % location of interest in 'a'
be=ao;                          % estimate location in 'b'
sbcor;
bo=be+p;                        % equivalent location - constant.

%       find node to left
ae=ao-[0 t(2)];
be=ao-[0 t(2)];
sbcor;
ah=ae;                          % constant
bh=be+p;                        % constant

%       find node above
ae=ao-[t(1) 0];
be=ao-[t(1) 0];
sbcor;
av=ae;                          % constant
bv=be+p;                        % constant
da=[0 av*[i;1];ah*[i;1] ao*[i;1]];
db=[0 bv*[i;1];bh*[i;1] bo*[i;1]];
pt=[2,2];

%       first vertical line
sb=[64,128];                    % other size of large
ae=av-[t(1) 0];
be=2*bv-bo;
be=bv-[t(1) 0];
while ae>=ceil(sa/2) & ae<=size(a)-ceil(sa/2)...
```

```
    & be>=ceil(sb/2) & be<=size(a)-ceil(sb/2)
 sbcor;
 da=[0 ae*[i;1];da];
 db=[0 (be+p)*[i;1];db];
 pt(1)=pt(1)+1;
 ae=ae-[t(1) 0];
 be=2*(be+p)-[imag(db(2,2)) real(db(2,2))];
end
ae=ao+[t(1) 0];
be=bo+[t(1) 0];
while ae>=ceil(sa/2) & ae<=size(a)-ceil(sa/2)...
    & be>=ceil(sb/2) & be<=size(a)-ceil(sb/2)
 sbcor;
 da=[da;0 ae*[i;1]];
 db=[db;0 (be+p)*[i;1]];
 ae=ae+[t(1) 0];
 be=2*(be+p)-[imag(db(size(db,1)-1,2))...
             real(db(size(db,1)-1,2))];
end

%      correlate to left
ae=ah;
be=bh;
fn=0;
while fn==0
 ae=ae-[t(1) 0];
 if db(pt(1),2)~=0
  be=be+[imag(db(pt(1)-1,2)-db(pt(1),2))...
         real(db(pt(1)-1,2)-db(pt(1),2))];
 else
  be=[0,0];
 end
 %      line top left
 for ct=pt(1)-1:-1:1
  if ae>=ceil(sa/2) & ae<=size(a)-ceil(sa/2)...
   & be>=ceil(sb/2) & be<=size(a)-ceil(sb/2)
   sbcor;
   da(ct,1)=ae*[i;1];
   db(ct,1)=(be+p)*[i;1];
   ae=ae-[t(1) 0];
   be=2*(be+p)-[imag(db(ct+1,1)) real(db(ct+1,1))];
  end
 end
 ct=pt(1);
 ae=[imag(da(pt(1),1)) real(da(pt(1),1))]+[t(1) 0];
 if db(pt(1)-1,1)~=0
```

```
  be=[imag(2*db(pt(1),1)-db(pt(1)-1,1))...
      real(2*db(pt(1),1)-db(pt(1)-1,1))];
 else
  be=[0,0];
 end
 %     line bottom left
 while ae>=ceil(sa/2) & ae<=size(a)-ceil(sa/2)...
     & be>=ceil(sb/2) & be<=size(a)-ceil(sb/2)
  sbcor;
  ct=ct+1;
  da(ct,1)=ae*[i;1];
  db(ct,1)=(be+p)*[i;1];
  ae=ae+[t(1) 0];
  be=2*(be+p)-[imag(db(ct-1,1))...
               real(db(ct-1,1))];
 end
 ae=[imag(da(pt(1),1)) real(da(pt(1),1))]-[0 t(2)];
 if db(pt(1),2)~=0
  be=[imag(2*db(pt(1),1)-db(pt(1),2))...
      real(2*db(pt(1),1)-db(pt(1),2))];
 else
  be=[0,0];
 end
 %     new vertical line
 if ae>=ceil(sa/2) & ae<=size(a)-ceil(sa/2)...
  & be>=ceil(sb/2) & be<=size(a)-ceil(sb/2)
  sbcor;
  da=[zeros(size(da,1),1) da];
  db=[zeros(size(db,1),1) db];
  da(pt(1),1)=ae*[i;1];
  db(pt(1),1)=(be+p)*[i;1];
  pt(2)=pt(2)+1;
 else
  fn=1;
 end
end

%      correlate to right
ae=ao+[0 t(2)];
be=2*bo-bh;
if ae>=ceil(sa/2) & ae<=size(a)-ceil(sa/2)...
 & be>=ceil(sb/2) & be<=size(a)-ceil(sb/2)
 sbcor;
 da(pt(1),pt(2)+1)=ae*[i;1];
 db(pt(1),pt(2)+1)=(be+p)*[i;1];
 fn=0;
```

```
while fn==0;
 ae=ae-[t(1) 0];
 if db(pt(1),size(db,2)-1)~=0
  be=be+[imag(db(pt(1)-1,size(db,2)-1)-db(pt(1),size(db,2)-1))...
         real(db(pt(1)-1,size(db,2)-1)-db(pt(1),size(db,2)-1))];
 else
  be=[0,0];
 end
 %    top right
 for ct=pt(1)-1:-1:1
  if ae>=ceil(sa/2) & ae<=size(a)-ceil(sa/2)...
   & be>=ceil(sb/2) & be<=size(a)-ceil(sb/2)
   sbcor;
   da(ct,size(db,2))=ae*[i;1];
   db(ct,size(db,2))=(be+p)*[i;1];
   ae=ae-[t(1) 0];
   be=2*(be+p)-[imag(db(ct+1,size(db,2)))...
                real(db(ct+1,size(db,2)))];
  end
 end
 ct=pt(1);
 ae=[imag(da(pt(1),size(db,2)))...
     real(da(pt(1),size(db,2)))]+[t(1) 0];
 if db(pt(1)-1,size(db,2))~=0
  be=[imag(2*db(pt(1),size(db,2))-db(pt(1)-1,size(db,2)))...
      real(2*db(pt(1),size(db,2))-db(pt(1)-1,size(db,2)))];
 else
  be=[0,0];
 end
 %    bottom right
 while ae>=ceil(sa/2) & ae<=size(a)-ceil(sa/2)...
     & be>=ceil(sb/2) & be<=size(a)-ceil(sb/2)
  sbcor;
  ct=ct+1;
  da(ct,size(db,2))=ae*[i;1];
  db(ct,size(db,2))=(be+p)*[i;1];
  ae=ae+[t(1) 0];
  be=2*(be+p)-[imag(db(ct-1,size(db,2)))...
               real(db(ct-1,size(db,2)))];
 end
 ae=[imag(da(pt(1),size(db,2)))...
     real(da(pt(1),size(db,2)))]+[0 t(2)];
 if db(pt(1),size(db,2)-1)~=0
  be=[imag(2*db(pt(1),size(db,2))-db(pt(1),size(db,2)-1))...
      real(2*db(pt(1),size(db,2))-db(pt(1),size(db,2)-1))];
 else
```

```
    be=[0,0];
   end
   %    new line to right
   if ae>=ceil(sa/2) & ae<=size(a)-ceil(sa/2)...
    & be>=ceil(sb/2) & be<=size(a)-ceil(sb/2)
    sbcor;
    da(pt(1),size(db,2)+1)=ae*[i;1];
    db(pt(1),size(db,2)+1)=(be+p)*[i;1];
   else
    fn=1;
   end
  end
 end
```

Appendix III.20 - mest.m

```
%      MEST.M - finds a model of the (undistorted) movements.

%      R. J. Drew 17/02/1999

clf
hold off
load grixkb
tm=[0  1  2  5 10 15 18 19 20 10  0  0  0  0  0  0  0  0  0  0 ...
    0  0  5 10 15 20 10 10 10 10 10 10 20 20 20 20 20];
km=[0  0  0  0  0  0  0  0  0  0  0  1  2  5 10 15 18 19 20 10 ...
    0 10 10 10 10 10 10  0  5 10 15 20 20 15 10  5  0];
for nb=1:37
 for n=1:37
  mkw(nb,n)=(km(nb)-km(n))/(100+km(n));
  mtr(nb,n)=(tm(nb)-tm(n))/(100+tm(n));
 end
end
for nb=1:37
 if nb==1 load g01b;end
 if nb==2 load g02b;end
 if nb==3 load g03b;end
 if nb==4 load g04b;end
 if nb==5 load g05b;end
 if nb==6 load g06b;end
 if nb==7 load g07b;end
 if nb==8 load g08b;end
 if nb==9 load g09b;end
 if nb==10 load g10b;end
 if nb==11 load g11b;end
 if nb==12 load g12b;end
 if nb==13 load g13b;end
 if nb==14 load g14b;end
 if nb==15 load g15b;end
 if nb==16 load g16b;end
 if nb==17 load g17b;end
 if nb==18 load g18b;end
 if nb==19 load g19b;end
 if nb==20 load g20b;end
 if nb==21 load g21b;end
 if nb==22 load g22b;end
 if nb==23 load g23b;end
if nb==24 load g24b;end
if nb==25 load g25b;end
if nb==26 load g26b;end
if nb==27 load g27b;end
if nb==28 load g28b;end
if nb==29 load g29b;end
if nb==30 load g30b;end
if nb==31 load g31b;end
if nb==32 load g32b;end
if nb==33 load g33b;end
if nb==34 load g34b;end
if nb==35 load g35b;end
if nb==36 load g36b;end
if nb==37 load g37b;end
nb
for n=1:37
 if n==1 a=d0101a;b=d0101b;end
 if n==2 a=d0102a;b=d0102b;end
 if n==3 a=d0103a;b=d0103b;end
 if n==4 a=d0104a;b=d0104b;end
 if n==5 a=d0105a;b=d0105b;end
 if n==6 a=d0106a;b=d0106b;end
 if n==7 a=d0107a;b=d0107b;end
 if n==8 a=d0108a;b=d0108b;end
 if n==9 a=d0109a;b=d0109b;end
 if n==10 a=d0110a;b=d0110b;end
 if n==11 a=d0111a;b=d0111b;end
 if n==12 a=d0112a;b=d0112b;end
 if n==13 a=d0113a;b=d0113b;end
 if n==14 a=d0114a;b=d0114b;end
 if n==15 a=d0115a;b=d0115b;end
 if n==16 a=d0116a;b=d0116b;end
 if n==17 a=d0117a;b=d0117b;end
 if n==18 a=d0118a;b=d0118b;end
 if n==19 a=d0119a;b=d0119b;end
 if n==20 a=d0120a;b=d0120b;end
 if n==21 a=d0121a;b=d0121b;end
 if n==22 a=d0122a;b=d0122b;end
 if n==23 a=d0123a;b=d0123b;end
 if n==24 a=d0124a;b=d0124b;end
 if n==25 a=d0125a;b=d0125b;end
 if n==26 a=d0126a;b=d0126b;end
 if n==27 a=d0127a;b=d0127b;end
 if n==28 a=d0128a;b=d0128b;end
 if n==29 a=d0129a;b=d0129b;end
 if n==30 a=d0130a;b=d0130b;end
```

```
   if n==31 a=d0131a;b=d0131b;end
   if n==32 a=d0132a;b=d0132b;end
   if n==33 a=d0133a;b=d0133b;end
   if n==34 a=d0134a;b=d0134b;end
   if n==35 a=d0135a;b=d0135b;end
   if n==36 a=d0136a;b=d0136b;end
   if n==37 a=d0137a;b=d0137b;end
   ks=k;
   usz(nb,n)=size(a,1)*size(a,2);
   %k(7)=0;                         % adjust angle between images
   [uq,mp]=nevu(a,b);
   uuu(nb,n)=sum(uq(:));
   mpu(nb,n)=sum(mp(:));
   u=uq.*mp;
   uu0(nb,n)=sum(u(:));
   ux=ceil(abs(a)./(abs(a)+1));
   [t,ak]=zmod(k,a);
   ak=ak.*ux;
   ux=ceil(abs(b)./(abs(b)+1));
   [t,bk]=zmod(k,b);
   bk=bk.*ux;
   [tt(n),sk(n),st(n)]=mdomptx(u,ak,bk);
   skw(nb,n)=sk(n);
   str(nb,n)=-st(n);
   ttt(nb,n)=tt(n);
   %    if (sum(uq(:))~=sum(mp(:))|tt(n)>1)
   if sum(u(:))./sum(uq(:))>.85 & sum(uq(:))==1+sum(mp(:))  %sum(u(:))./sum(uq(:))<.85
    bmesh(b,'r')
    hold on
    bmesh((b.*mp),'k')
    hold off
    axis ij
    drawnow
    [nb,n tt(n)]
    %pause
  end
 end
end
%save ress skw str km tm k uuu uu0 mpu ttt mkw mtr usz
return
```

Appendix III.21 - neuv.m

```
function [u,mp]=nevu(a,b)
%       NEVU.M - this function masks out problem areas....

%       R. J. Drew 25/1/99

%       remove out of limit nodes
fa=.15;
fl=8;
u=ceil(abs(a)./(abs(a)+1))+ceil(abs(b)./(abs(b)+1));
u=ceil(u./(u+1));                         % null nodes
[zm,zn]=size(a);
%       find preferred lengths and angles
mu=floor((u(2:zm,:)+u(1:zm-1,:))/2);
nu=floor((u(:,2:zn)+u(:,1:zn-1))/2);
lm=abs(diff(b));
[g,h]=histogra(lm(:),0);
g(1)=0;
[abc,l]=max(g);
if size(g')==size(h)
 lmx=h(l);
else
 lmx=1;
end
lmy=1-ceil((abs(lm-lmx)-fl)./(abs(abs(lm-lmx)-fl)+1));
ln=abs(diff(b')');
[g,h]=histogra(ln(:),0);
g(1)=0;
[abc,l]=max(g);
if size(g')==size(h)
 lnx=h(l);
else
 lnx=1;
end
lny=1-ceil((abs(ln-lnx)-fl)./(abs(abs(ln-lnx)-fl)+1));
am=angle(b(2:zm,:)-b(1:zm-1,:));
[g,h]=histogra(am(:)*50/pi,0);
[abc,l]=max(g);
if size(g')==size(h)
 amx=h(l);
else
 amx=1;
```

```
end
amy=1-ceil((abs(am-amx*pi/50)-fa)./(abs(abs(am-amx*pi/50)-fa)+1));
an=angle(b(:,2:zn)-b(:,1:zn-1));
 [g,h]=histogra(an(:)*50/pi,0);
 [abc,l]=max(g);
if size(g')==size(h)
 anx=h(l);
else
 anx=1;
end
any=1-ceil((abs(an-anx*pi/50)-fa)./(abs(abs(an-anx*pi/50)-fa)+1));
n=any & lny;
m=amy & lmy;
nn=zeros(zm,zn);
nn(:,1:zn-1)=n;
nn(:,2:zn)=nn(:,2:zn) | n;
mm=zeros(zm,zn);
mm(1:zm-1,:)=m;
mm(2:zm,:)=mm(2:zm,:) | m;
mp=mm | nn;
return
```

Appendix III.22 - mdomptx.m

```
function [merit,skew,strn]=mdomptx(u,a,b)
%       MDOMPTX.M - finds a model of the (undistorted) movements.

%       R. J. Drew 17/2/1999


[zm,zn]=size(a);
w=b-a;
a9=imag(a(1:zm-1,:)-a(2:zm,:));
u9=floor((u(2:zm,:)+u(1:zm-1,:))/2);
if sum(u9(:))==0
 u9=1;
end
a8=(1-ceil(abs(a9)./(abs(a9)+1)));          % mask
ke=(((real(w(2:zm,:)-w(1:zm-1,:)))./(a9+a8)));
tr=(imag(w(2:zm,:)-w(1:zm-1,:)))./(a9+a8);
skew=sum(ke(:).*u9(:))./sum(u9(:));
strn=sum(tr(:).*u9(:))./sum(u9(:));
mr0=abs(skew-ke).*u9*20;
mr1=abs(strn-tr).*u9*20;
merit=sum(mr0(:).^2)/sum(u9(:)).*sum(mr1(:).^2)/sum(u9(:));
return

mrx=[mr1(1,:)*2;mr1(1:zm-2,:)+mr1(2:zm-1,:);mr1(zm-1*2,:)];
mry=[mr0(1,:)*2;mr0(1:zm-2,:)+mr0(2:zm-1,:);mr0(zm-1*2,:)];
```