1996

Robust Sensor Fusion Algorithms: Calibration and Cost Minimization.

Richard Ree Brooks
Louisiana State University and Agricultural & Mechanical College

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

ROBUST SENSOR FUSION ALGORITHMS: CALIBRATION AND COST MINIMIZATION

A Dissertation

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

in

The Department of Computer Science

by
Richard R. Brooks
B.A., The Johns Hopkins University, 1979
August 1996

ACKNOWLEDGMENTS

The author wishes to acknowledge the support of the following people and personally thank them for their help and support in the preparation of this document. The author's major professor, Dr. Iyengar, has provided guidance and inspiration throughout the author's graduate studies and provided many helpful insights. Dr. Suresh Rai of the department of Electrical Engineering has been very helpful in aiding the author with all portions of this work related to system reliability. Dr. Jianhua Chen of the department of Computer Science has provided guidance and technical advice in the area of Genetic Algorithms and Machine learning. In addition, this dissertation has been supported in part by the Office of Naval Research through grants N 00014-94-1-0343 and N 00014-91-J-1306. Other members of the faculty and staff of the departments of Computer Science and Electrical Engineering of Louisiana State University have been an invaluable resource for the author during his studies, and the author wishes to thank them as a group.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

A system reacting to its environment requires sensor input to model the environment. Unfortunately, sensors are electromechanical devices subject to physical limitations. It is challenging for a system to robustly evaluate sensor data which is of questionable accuracy and dependability. Sensor fusion addresses this problem by taking inputs from several sensors and merging the individual sensor readings into a single logical reading. The use of heterogeneous physical sensors allows a logical sensor to be less sensitive to the limitations of any single sensor technology, and the use of multiple identical sensors allows the system to tolerate failures of some of its component physical sensors. These are examples of fault masking, or *N*-modular redundancy.

This research resolves two problems of fault masking systems: the automatic calibration of systems which return partially redundant image data is problematic, and the cost incurred by installing redundant system components can be prohibitive. Both are presented in mathematical terms as optimization problems.

To combine inputs from multiple independent sensors, readings must be registered to a common coordinate system. This problem is complex when functions equating the readings are not known *a priori*. It is even more difficult in the case of sensor readings, where data contains noise and may have a sizable periodic component. A practical method must find a near optimal answer in the presence of large amounts of noise. The first part of this research derives a computational scheme capable of registering partially overlapping noisy sensor readings.

Another problem with redundant systems is the cost incurred by redundancy. The trade-off between reliability and system cost is most evident in fault-tolerant systems. Given several component types with known dependability statistics, it is possible to determine the combinations of components which fulfill dependability constraints by modeling the system using Markov chains. When unit costs are known, it is desirable to use low cost combinations of components to fulfill the reliability constraints. The second part of this dissertation develops a methodology for designing sensor systems, with redundant components, which satisfy dependability constraints at near minimal cost. Open problems are also listed.

1 INTRODUCTION TO SENSOR FUSION

Technology allowing automated systems to work in hostile and unstructured environments is needed for many military, medical and industrial applications. These applications rely on sensors to provide timely and accurate information regarding their environment. Sensor data is difficult for automated systems to treat because: accuracy is limited, precision is limited, readings are subject to noise, sensors are subject to transient failures, and sensors are subject to permanent mechanical failures [42,29]. Processing of sensor data has therefore become an important subject of basic research which can be directly implemented in many signal and image processing application areas including national defense, environmental monitoring and medical image processing.

Many applications involve the use of multiple heterogeneous sensors grouped together by a network. These sensors may have varying levels of intelligence, accuracy and reliability. This results in sensor systems being a combination of sensors which are physically, logically, spatially and even geographically distinct from each other. As an example, consider a hypothetical autonomous vehicle navigation platform. The vehicle position information can be inferred by combining data from multiple sources: a very accurate but slow *ladar* (**l**aser **d**etection **a**nd **r**anging) sensor provides range information, simple gyroscopes provide heading data, and two video cameras with distinct resolutions provide visual images of the environment. Note the number of issues raised by this system: video data requires an intelligent system to translate video images into a format which can be compared to the ladar data, the heading sensor provides data on a time-scale which differs greatly from the ladar sensor, and video data of varying resolutions must be merged. Combining these kinds

of information into a single robust source of data is the goal of sensor integration and fusion. A wider treatment of this topic can be found in the research presented by Iyengar [44,45,46,47,49].

Sensor integration and fusion has been proposed as a technology which can be used to correct many of the shortcomings of existing sensor systems. Luo and Kay define *sensor integration* as the synergistic use of information from multiple sensors and *sensor fusion* as the portion of sensor integration concerned with merging multiple sensor readings into a common data structure [57,58]. Sensor fusion is therefore a subset of the more general sensor integration research. The problems considered by this dissertation are involved with combining multiple sensor readings into one reliable reading; for this reason our work is best characterized as sensor fusion.

Multiple sensor networks have been divided into *complementary*, *competitive*, and *cooperative* configurations [28]. *Complementary* sensor networks are ones where many sensors provide information which can be combined to provide a more complete picture of the environment, such as multiple radar sites that provide radar coverage of a large geographic region. *Competitive* sensor networks use redundant sensors to improve the dependability of a sensor network. A system with three radar inputs would be able to tolerate the failure of any single radar component. They would thus be in a *competitive* configuration. *Cooperative* sensor networks refer to the coordinated use of multiple sensors to extract information which would be unavailable to a single sensor, an example of this is the use of stereo and trinocular vision systems for 3-dimensional vision [3].

This dissertation is primarily interested in competitive sensor networks which use sensor fusion to increase system dependability. For these networks, the actual

physical sensors are generally referred to as *concrete sensors*. Readings from multiple concrete sensors can be merged into a single *abstract* [61] or *logical* [84] sensor reading.

Marzullo[61] has derived a paradigm for representing multi-dimensional sensor data of limited accuracy. The data can easily be represented by an upper and lower bound in each dimension. If sensor readings from N independent sensors consist of an upper and lower bound in each of D dimensions it is possible to construct an accurate abstract sensor reading whose precision is at least as precise as the least precise concrete sensor as long as F, the number of faulty concrete sensors, is less than $N / 2D$ [61,22]. A number of algorithms have been proposed for performing this type of multidimensional sensor fusion [14,15,16,22,50,61].

It is worth noting that the construction of reliable abstract sensors is often referred to as *fault-masking* [52], or *N-modular redundancy* [72]. Both terms refer to the use of redundant components to increase system dependability. By comparing the readings from multiple inputs, it is possible to infer the correct answer if only a minority of the input sources are faulty. This problem is closely related to the general distributed agreement problem, also known as the *Byzantine Generals Problem* [7,56]. The Byzantine Generals Problem refers to a very pessimistic model which has been used to model fault tolerant systems. The name refers to a metaphor in which a number of generals command armies besieging an enemy city. A subset of the generals are traitors and the loyal generals must retreat or attack in a coordinated manner to avoid defeat. This model allows arbitrary and even malicious errors to occur within the system. Algorithms which solve the problem are truly reliable. The parallels between sensor fusion and the Byzantine Generals Problem are presented in [17,18].

This dissertation resolves two basic problems which have impeded the practical implementation of multi-sensor systems: it is difficult to find a common frame of reference for two sensors if it is not known *a priori*, and the cost of redundant components can make multi-sensor systems prohibitively expensive. In this work we phrase both problems as optimization problems which use a fitness function to rate the quality of possible solutions. A number of heuristic methods are then used to find near optimal answers.

Chapter two describes the two problems in detail and formally defines the concepts necessary for the rest of the dissertation. Existing literature concerning the two problems being considered is discussed in chapter three. Chapter four reviews standard optimization methodologies, explains which methods are used by this research, which methods have not been used, and the reasoning behind these decisions. Details as to the application of the methods described in chapter four to the image registration problem is given in chapter five. Chapter six gives similar details for the redundant sensor system cost minimization problem. Chapter seven presents the quantitative results of our research, and the dissertation concludes with a formal discussion of these results in chapter eight.

2 PROBLEM STATEMENT

Designing robust sensor fusion systems is a challenging endeavor. Computer software is notoriously sensitive to input data inconsistencies, and competitive sensor fusion systems are designed primarily to resolve data inconsistencies among input data sets. Several general problems exist concerning the implementation of systems using sensor integration:

- **Distributed agreement** Brooks and Iyengar [17, 18] have recently shown sensor fusion to be related to the Byzantine Generals Problem posed by Lamport et al [7,56].
- **Proper interpretation of inexact data** Marzullo [61] initially posed this problem based on modeling inexact sensor readings as an upper and lower bound. An improved algorithm for the one dimensional case was developed by Jayasimha [50]. The model was extended to an arbitrary number of dimensions by Chew and Marzullo [22]. Algorithms providing improvements on the multi-dimensional case have been provided by Brooks and Iyengar [14,15,16]. Other approaches taken to this problem include the multi-resolution method proposed by Prasad and Iyengar [47], and Kalman filtering as implemented by Ayache and Faugeras [3] and Maheshkumar et al [60].
- **Architectural issues** A number of these issues, such as bandwidth reduction, network topology and routing requirements, have been explored in publications by Iyengar et al [44,45,46,49].

This dissertation research resolves two problems central to the implementation of robust sensor fusion systems. These are basic and distinct problems that exist when

implementing distributed sensor fusion systems based on fault masking. Both are related to system redundancy and both can be expressed mathematically as optimization problems.

In order to compare readings from independent physical sensors, all the input data must be put into a common frame of reference. This is generally referred to as *registering* the data. It is frequently assumed that this transformation is known a priori, partially because the problem is not trivial. Existing methods, which are described fully in chapter three, are primarily based on methods traditionally used by cartographers. These methods have a number of drawbacks, often making assumptions concerning the input data which may not be true. The first goal of this research is to develop a system capable of automatically registering arbitrary sensor readings that contain non-negligible amounts of noise. We derive a new methodology for the registration problem which solves difficult registration problems effectively.

Another problem which must be addressed is that increasing system dependability by installing redundant sensors also increases the system cost. This is a major problem for designers of dependable systems. While a certain cost increase is inevitable, it is necessary to attempt to minimize this factor. This research will also develop a methodology which, given a set of components applicable to a sensing task and given a dependability requirement for the system, is capable of finding a combination of components that fulfills the dependability constraint at minimal, or near minimal, cost. We will show that methods very close to the ones used for image registration are capable of finding these sensor configurations.

2.1 IMAGE REGISTRATION

Competitive multiple sensor networks consist of a large number of concrete sensors providing readings which are at least partially redundant. The first step in

fusing multiple sensor readings is registering the images to find the correspondence between them [33].

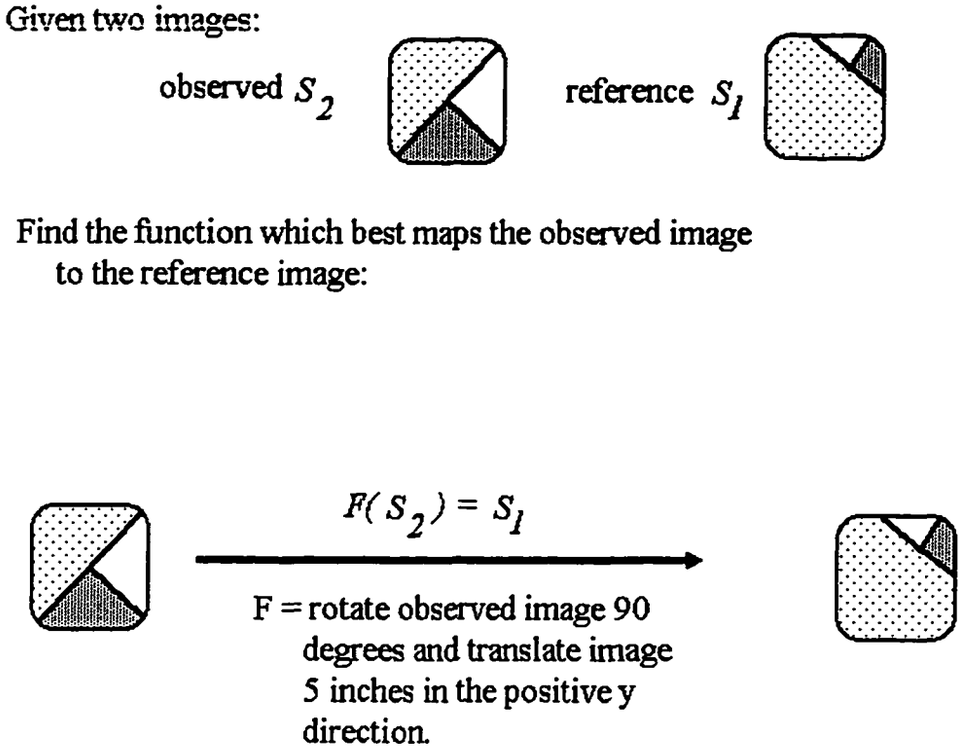

Figure 2.1 Registration is finding the mapping function $F(S_2)$.

This section describes the general image registration problem, and a specific example which will be used experimentally to test the methods used in this dissertation. As shown in **figure 2.1** the general problem is: given two *N*-dimensional sensor readings find the function F which best maps the reading from sensor two $S_2(x_1,...,x_n)$ onto the reading from sensor one $S_1(x_1,...,x_n)$ so that ideally $F(S_2(x_1,...,x_n)) = S_1(x_1,...,x_n)$. In practice all sensor readings contain some amount of measurement error or *noise* so that the ideal case will rarely occur, if ever.

We approach the registration problem as an attempt to automatically find a gruence (translation and rotation) which correctly calibrates two 2-dimensional

sensor readings with identical geometries. This can be done without a loss of generality since:

- A method which works for two readings can be extended to register any number of images.
- The majority of sensors currently work in one or two dimensions. Extensions of calibration methods to more dimensions is desirable but not imperative.
- Calibration of two sensor images presupposes a known sensor geometry. If the geometries are known, a function can be derived which maps the readings as if the geometries were identical when a registration is given.

Our research finds gruences (translations and rotations) since these functions are most representative of the most common class of problems. If desired, the approaches used can be directly extended to include the class of all affine transformations by adding scaling transformations [38]. This dissertation does not consider the class of "rubber sheet" transformations which warp the contents of the image, since these transformations are used mainly to correct local effects after the affine transformation which correctly map the images has been found [82]. It is explicitly assumed either that any rubber sheet deformations of the sensor image are known and corrected before the mapping function is applied, or that their effects over the image intersections are negligible.

The computational examples used deal with two sensors returning 2-dimensional gray scale data from the same environment. The sensors have identical geometric characteristics, return readings covering a circular region, and it is known that the readings overlap. Both sensors' readings contain noise. The amount of noise and the relative positions of the two sensors are not known.

Sensor two is translated and rotated by an unknown amount with relation to sensor one.

If the size, or the contents, of the overlapping areas were known, it would be possible to perform a correlation using the contents of the overlap on the two images and find the point where they overlap directly. Since this information is unavailable, this approach is impossible.

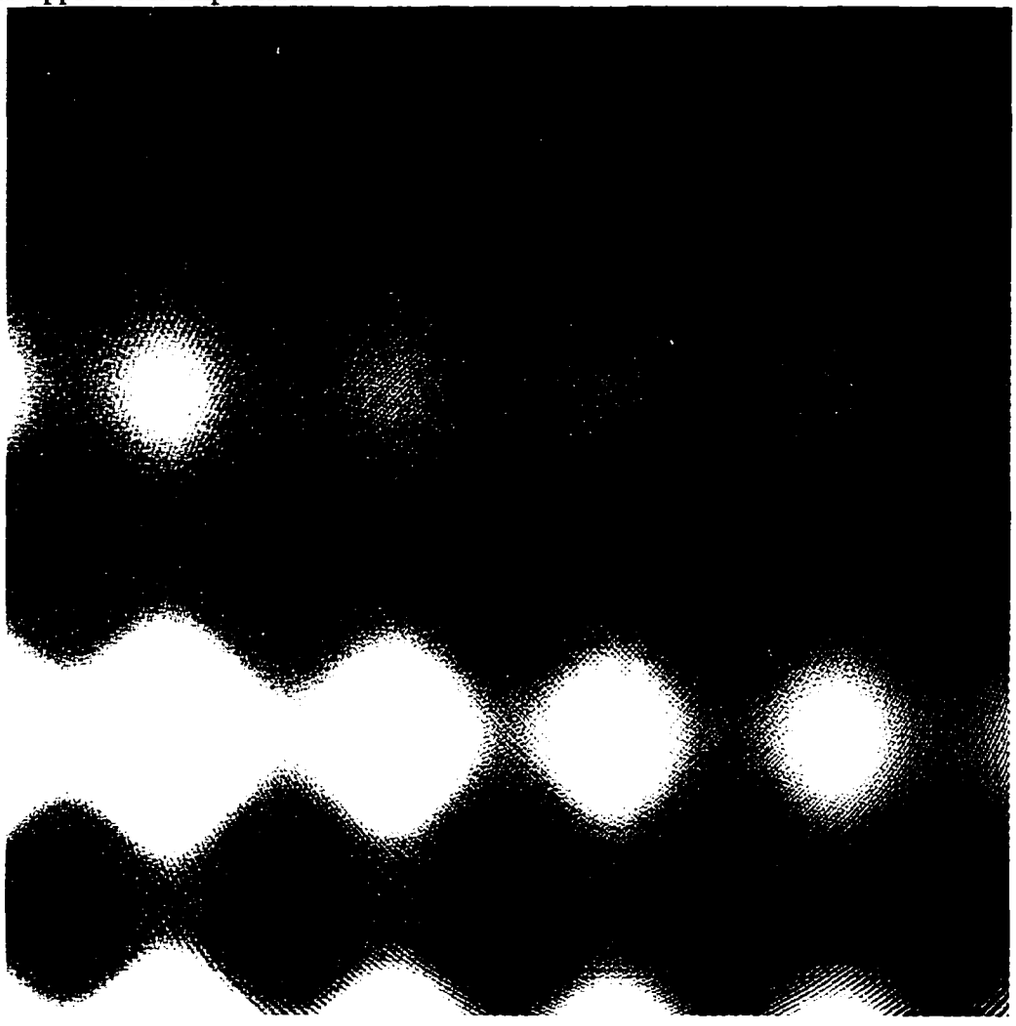

Fig. 2.2 Noise free synthetic terrain image used for image registration

The best way to solve this problem depends on the nature of the terrain being observed. If unique landmarks can be identified in both images then it is possible to

attach the two images at those points. Depending on the number of landmarks available minor adjustments may be needed to fit the readings exactly. It is assumed in this work that landmarks are not readily available in the sensor images.

The model used to represent the terrain in examples here has several periodic components combined with non-periodic elements. The equation used is:

$$\begin{aligned}\text{terrain}(x,y) = 100.0 + (&-40x + 45y - 0.003xy + 0.02x^2 - 0.01y^2 \\ &- 20y*\sin(x/18) + 35y*\cos(y/29) - 35\sin(x/4-y/12) \\ &+ 12x*\cos(xy/100))/100\end{aligned}$$

This equation was found through trial and error, and the result is vaguely reminiscent of mountainous terrain. This terrain model, shown in **Fig 2.2**, consists of a 512 x 512 gray scale array. This synthetic image will be used for testing image registration schemes. Two overlapping regions from the terrain model are taken, corrupted with noise, rotated and translated. The methods proposed then try to find an approximation of the function F mapping the sensor 2 image onto the sensor one image.

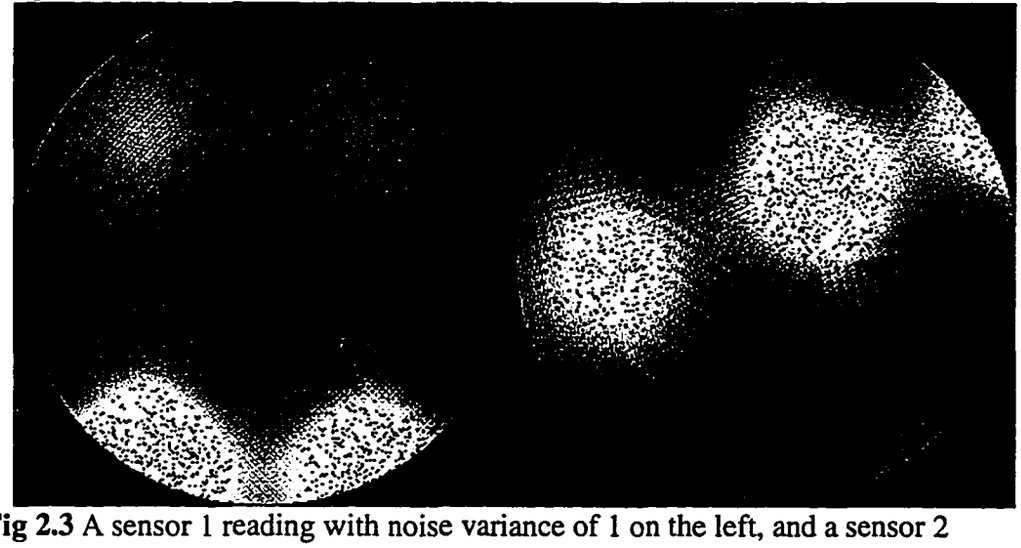

Fig 2.3 A sensor 1 reading with noise variance of 1 on the left, and a sensor 2 reading with noise variance of 1 on the right.

This model has been chosen since it presents two characteristics which are necessary for the problem to be solvable, but not trivially solvable. Since it has non-periodic elements, there will be a unique best match for the two sensors. The periodic elements in the model mean that this match is not obvious: an algorithm which searches for the best match will have to be capable of dealing with local minima in the search space. The effects of the local minima is aggravated by the noisy character of the sensor data.

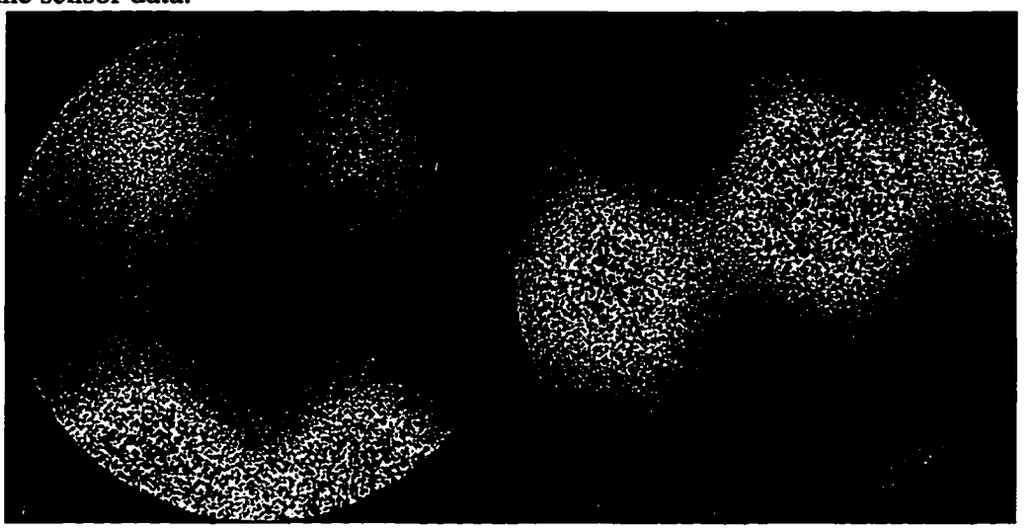

Fig 2.4 A sensor 1 reading with noise variance of 33, and a sensor 2 reading with noise variance of 33.

Noisy sensor readings are shown in **Fig 2.3** and **Fig. 2.4**, which give examples of readings from the terrain returned by sensor 1 and sensor 2. The noise in **Fig 2.3** has a variance of 1, and the noise in **Fig 2.4** has a variance of 33. Since the gray scale used has only 256 levels, a variance of 1 obscures a relatively minor amount of information in the sensor readings. This sensor 1 reading is from the center of the terrain. It has not been rotated, its center is at point (256,256). The corresponding example of a sensor 2 readings have the same noise variance as the sensor 1 readings. Note that the relation between the readings is not intuitively obvious, that several possible correlations exist, and that the noise of variance 33 obscures a large amount

of information in the image. There is, however, one registration which is the best mapping of the sensor 2 data onto sensor 1. This reading is centered at the lower right hand edge of the sensor 1 reading, which is point (347,347). It has been rotated by 2.74889 radians (157.5 degrees).

Thus, the problem to be solved is: given noisy gray scale data readings from sensor one and sensor two, find the optimal set of parameters (x-displacement, y-displacement, and angle of rotation) which defines the center of the sensor two image relative to the center of the sensor one image. These parameters are optimal in that they give the best mapping of sensor 2's readings to the readings from sensor 1. Note that the same terrain is used throughout this dissertation for ease of comprehension, this terrain is one of many terrains which have been used for experimentation. The results given by this terrain are representative.

2.2 COST MINIMIZATION OF REDUNDANT SYSTEMS

Fault masking systems, also referred to as *N*-modular redundancy, achieve fault tolerance by duplication of components. This is a standard method for achieving system dependability, however, it must be noted that module duplication also drastically increases system cost. In order for a system to be reliable and feasible, attention must be paid to both reliability bounds and total system cost.

Assuming that a redundant system is built from existing components, it is reasonable to characterize these components by their dependability statistics and unit costs. For any combination of components a Markov chain can be constructed which accurately models the state of the system [1,68,70,71,72]. The probability of the system being in any given state of the Markov chain can be calculated using one of the following approaches:

- Constructing a set of differential equation directly from the chain and solving the equations using Laplace transforms
- Using statistical distributions for each component and assuming independent failure a direct solution based on probability can be deduced
- Using Kirchoff's law and the frequency of transition between states, the probability of the system being in a set of states can be deduced for the steady state

Given a dependability requirement which the system is required to meet, this may be either a mean time to failure or service availability, it is now possible to classify the possible sensor network configurations according to whether or not they meet the given dependability criteria. The dependability requirement will be met only if the mean time to failure of more than a given percentage (for example 50%) of the components is after the specified mission time for a reliability requirement, or if the system is expected to be in a state with more than a specified percentage of the components functioning for a percentage of the time greater than the given system availability requirement. The problem to be solved is now to find which combination of components fulfills the system's dependability criteria, and has the lowest total system cost.

This dissertation solves a system design problem based on the application of fault masking for dependable systems design. We assume throughout that component failure is statistically independent. Application of our results requires dependability statistics for the individual components, which may be based on several existing statistical models. For sensor systems, statistical dependability models depend on the exact problem being considered; if the design is concerned primarily with mechanical failure then the use of mean-time-to-failure and mean-time-to-repair is usually

adequate. If the design must also consider transient and intermittent errors, statistics must be available to determine the distributions of fault arrival and duration. The numerical examples given, however, use exponential distributions for component failure and repair. This is for tractability and consistency with reliability literature. Equations are derived for systems where over half of all components must be functional, this can be changed to any fraction simply by replacing *N/2* as necessary.

Dependability is used as a generic term for either *reliability*, the probability that the system is working at a given mission time, or *availability*, the percentage of time that the system is functional. When a system with strict dependability requirements is being designed using Byzantine agreement or sensor fusion algorithms, a choice must be made between different component types which could be used redundantly to mask errors. Given a choice between various modules, each with different dependability parameters and per item costs, we attempt to obtain the configuration which meets the dependability requirements with the lowest system cost.

3 REVIEW OF EXISTING RESEARCH

This section discusses the current state of research concerning the two problems under consideration. Image registration is a basic problem in image processing and a large number of methods have been proposed. Current methods are outlined and discussed. Special attention is given to differences between the assumptions made by previous methods and the assumptions used in deriving the methodology used in this dissertation.

Similarly, the use of redundant components is one of the basic strategies used to increase system reliability. A large amount of literature has therefore been written concerning how to implement these systems so as to increase system dependability and methods of evaluating the dependability of existing systems. We review this literature as a basis for our cost minimization methodology. The application of this work to designing low cost dependable systems is, however, a new approach to the problem.

3.1 IMAGE REGISTRATION TECHNIQUES

Several methods exist for registering images, this is the first research which finds gruence transformations to match noisy gray scale images. First we will discuss practical applications of automatic image registration systems. **Table 3.1** summarizes the features of representative image registration methods discussed in this section. The discussion will be followed by a detailed discussion of the established methodologies, and algorithms currently in use. This section concludes by pointing out the drawbacks in these methods and where the methodology proposed here differs from the established ones.

Many processes require that data from one image called the *observed image* be compared with or mapped to another image called the *reference image*. It is not surprising, therefore, that a wide range of critical applications depend on image registration. Perhaps the largest amount of current image registration research is in the field of medical imaging. One application is sensor fusion to combine outputs from several medical imaging technologies, such as PET and MRI, to form a more complete image of internal organs [63]. Registered images are then used for medical diagnosis of illness [59], and automated control of radiation therapy [65]. Similar applications of registered and fused images are referred to as using terrain "footprints" in military applications [74], are used in remote sensing applications [80] and are common in robotics as well. A novel application is registering portions of images to estimate motion. Descriptions of motion can then be used to construct intermediate images in television transmissions. Jain and Jain describe the applications of this to bandwidth reduction in video communications [48]. These are some of the more recent applications relying on accurate image registration, methods of image registration have been studied since the beginning of the field of cartography.

The traditional method of registering two images is an extension of methods used in cartography. A number of control points are found in both images. The control points are matched, and this match is used to deduce equations which interpolate all points in the new image to corresponding points in the reference image [34,82].

Several algorithms exist for each phase of this process. It is necessary to guarantee that the control points can be chosen automatically, are unique and easily identified in both images. In various studies control points have been explicitly placed in the image by the experimenter [65], edges defined by intensity changes [83],

specific points peculiar to a given image [62], line intersections, center of gravity of closed regions, or points of high curvature [34]. The kind of control point which should be used is mainly dependent on the contents of the image.

Table 3.1 Image registration methods

Algorithm	Image Type	Matching Method	Interpolation Function	Transforms Supported	Comments
Andrus [2]	Boundary Maps	Correlation	None	Gruence	Noise intolerant Small rotations
Barnea [8]	No Restriction	Improved Correlation	None	Translation	No rotation, scaling noise, rubber sheet
Barrow [10]	No Restriction	Hill Climbing	Parametric Chamfer	Gruence	Noise intolerant Small displacement
Brooks Iyengar	No Restriction	Elitist Gen. Alg.	None	Gruence	Noise tolerant Tolerates periodicity
Cox [25]	Line Segments	Hill Climbing	None	Gruence	Matches using small number of features
Davis [27]	Specific Shapes	Relaxation	None	Affine	Matches shapes
Goshtasby [34]	Control Points	various	Piecewise Linear	Rubber Sheet	Fits images using mapped points
Goshtasby [35]	Control Points	various	Piecewise Cubic	Rubber Sheet	Fits images using mapped points
Goshtasby [36]	Control Points	various	Least Squares	Rubber Sheet	Fits images using mapped points
Jain [48]	Sub-images	Hill Climbing	None	Translation	Small translations No rotation, no noise
Mandara [59]	Control Points	Classic G.A. S.A.	Bi-linear	Rubber Sheet	Fits 4 fixed points using error fitness
Mitiche [62]	Control Points	Least Squares	None	Affine	Uses control points
Oghabian [63]	Control Points	Sequentiai Search	Least Squares	Rubber Sheet	Assumes small displacement
Pinz [66]	Control Points	Tree Search	None	Affine	Difficulty with local minima
Stockman [74]	Control Points	Cluster	None	Affine	Assumes landmarks, periodicity problem
Wong [83]	Intensity Diffferences	Exhaustive Search	None	Affine	Uses edges, Intense computation

Similarly, a large number of methods have been proposed for matching control points in the observed image to the control points in the reference image. The obvious method is to correlate a template of the observed image against the reference

image [2,8]. Another widely used approach is to calculate the transformation matrix which describes the mapping with the least square error [62,36,80]. Other standard computational methods such as relaxation and hill-climbing have also been used [48,41,10]. Pinz et al use a hill climbing algorithm to match images and note the difficulty posed by local minima in the search space, to overcome this they run a number of attempts in parallel with different initial conditions [66].

Some interesting methods have been implemented which work by considering all possible transformations. Stockman et al. construct vectors between all pairs of control points in an image. For each vector in each image an affine transformation matrix is computed which converts the vector from the observed image to one of the vectors from the reference image. These transformations are then plotted, and the region containing the largest number of correspondences is assumed to contain the correct transformation [74]. This method is computationally expensive since it considers with the power set of control points in each image. Wong and Hall match scenes by extracting edges or intensity differences and constructing a tree of all possible matches which are below a given error threshold [83]. They reduce the amount of computation needed by stopping all computation concerning a potential matching once the error threshold is exceeded, but this method is none the less computationally intensive.

The approach which most closely resembles the work done in this dissertation has been implemented by Mandara and Fitzpatrick [59]. As done here, they use simulated annealing and genetic algorithms heuristics to find good matches between two images. A number of differences exist between their research and ours, in that they find a rubber sheet transformation which fits two images by using linear interpolation around four control points, and assume that the images match

approximately at the beginning. In addition to this they use different fitness functions and reproduction schemes than the ones used in this dissertation.

A number of researchers have used multiresolution methods to prune the search space considered by their algorithms. Mandara and Fitzpatrick[59] use a multiresolution approach to reduce the size of their initial search space for registering medical images using simulated annealing and genetic algorithms. This work influenced Oghabian and Todd-Prokopek[63] who similarly reduced their search space when registering brain images with small displacements. Pinz[66] adjusted both multiresolution scale space and step size in order to reduce the computational complexity of a hill climbing registration method. By starting with low resolution images these researchers believe it is possible to reject large numbers of possible matches, and that the correct match can be found by progressively increasing the resolution. It should be noted that, in images with a strong periodic component, a number of low resolution matches may be feasible. In this case the multiresolution approach will be unable to prune the search space and instead increase the computational load.

Once the appropriate match has been found a number of methods have been proposed for fitting the entire image around the control points. Simple linear interpolation is computationally straightforward [59]. Goshtasby has explored using weighted least-squares approach [36], constructing piecewise linear interpolation functions within triangles defined by the control points [34], and developing piecewise cubic interpolation functions [35]. These methods create non-affine rubber sheet transformation functions to attempt to reduce the image distortion caused by either errors in control point matching, or differences in the sensors which constructed the image.

As we have shown, several algorithms exist for image registration. The algorithms described in this section are more restrictive than the ones constructed in this dissertation and have some common drawbacks. The matching algorithms given assume that a small number of distinct features can be matched [62,74,20,4], that specific shapes are to be matched [27], that no rotation exists[8,48] or that the relative displacement is small [2,10,48,59,62,63]. Refer to **table 3.1** for a summary of many of these points. Choosing a small number of control points is not a trivial problem and has a number of inherent drawbacks. The control point found may be a product of measurement noise. Since the two images intersect only, control points in one image may not exist in the other image. This can only be overcome by considering the power sets of the control points. When an image contains strong periodic components, control points may not define a unique mapping of the observed image to the reference image.

Additional problems exist. The use of multi-resolution can not always trim the search space and if the image is dominated by periodic elements will only increase the computational complexity of an algorithm[63,59,66]. Many algorithms attempt to minimize the square error over the image, this does not consider the influence of noise in the image [63,59]. Most of the existing methods are sensitive to noise [2,62,63]. This dissertation assumes none of those restrictions, all tests were run using gray scale images with many possible approximate matches and a large displacement. Our approach can be used on any arbitrary gray scale image and in the presence of non-negligible amounts of noise .

3.2 REDUNDANT SYSTEM RELIABILITY

As technology advances, automated systems have become increasingly complicated and require the use of an increasingly large number of sub-components.

In particular, distributed computing and multisensor based automation systems consist of a very large number of individual components. Multisensor applications generally involve hazardous work environments and safety critical applications. Example applications include automated manufacturing systems, battlefield monitoring equipment, nuclear power plant regulators and deep-space probes. These applications rely on correct interpretation of a dynamic and unstructured environment. They are also characterized by the potentially catastrophic consequences of misinterpretations of their work environment. In order to better monitor the environment more than one sensor technology is used in parallel. This makes the system less sensitive to the failures of a particular technology, at the cost of increasing system complexity.

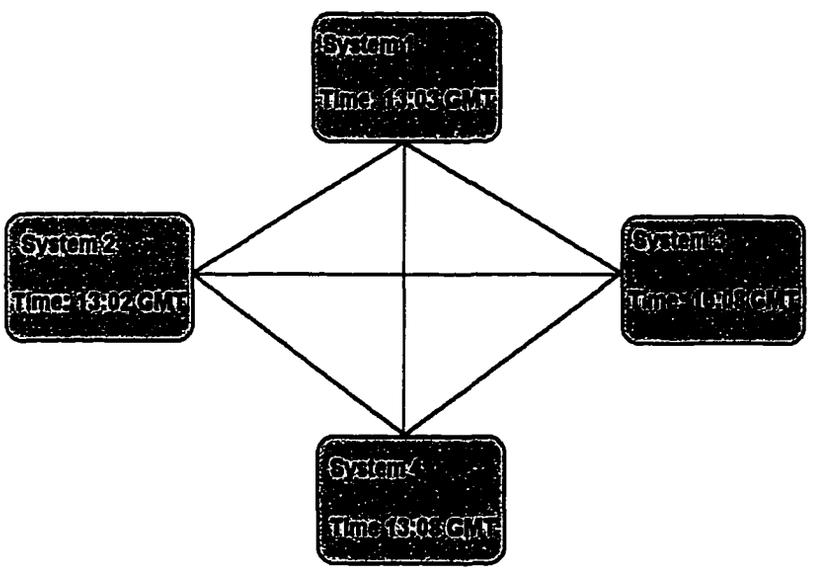

Fig. 3.1 A Byzantine agreement problem

Note that a complex system is sensitive to the failure of its individual components. Several techniques exist for designing a robust system which continues to function correctly in spite of the failure of isolated components. One such method, termed *N*-modular redundancy (*NMR*), involves using *N* independent components in parallel. The processing results from the *N* components can be compared to insure correctness[72]. Using a number of processors working in parallel and spares to tolerate isolated failures has been the subject of ongoing research, such as the dependability analysis of a distributed computing system given in [77], the mathematical study of the consequences of spare failures given in [5], and the algorithm for computing the change in reliability when the number of spares is increased by one in [6].

Data processing and robotics systems must infer correct information even in the presence of failures in a relatively large number of modules, this is known as the Byzantine Generals Problem [55]. Algorithms exist which make a unanimous decision in the presence of arbitrary errors as long as less than one third of the processors are faulty. For a full discussion of the topic refer to [7]. Typical applications for Byzantine Agreement are ones which demand homogeneity among distributed systems; such as commitment of transactions for a distributed database or network clock synchronization (**Fig. 3.1**).

A similar problem exists for multisensor fusion applications that attempt to glean the best possible interpretation from data of limited resolution which has been corrupted with noise. Note that robotics and automation control systems make decisions based on data gathered by sensors. Methods are given in the literature for finding the correct information from a set of sensor readings, some of which are faulty, and some of which are partially contradictory. Refer to [50,61] for algorithms

to solve one dimensional problems of this kind. **Fig. 3.2** illustrates a one dimensional case, for which these algorithms can tolerate failures in up to one half of the sensors. Marzullo and Chew [22] expanded this to cover more than one dimension. Reference [16] presents an efficient algorithm to give approximate results. For multidimensional problems the number of faulty sensors which can be tolerated depends on the number of dimensions used [11,22]. An efficient algorithm finding the optimal answer in an arbitrary number of dimensions is described in [14,15]. The information theoretical basis of fault masking is explored by Krol [52], and an application of this view of fault masking can be found in [79]. The relationship between sensor fusion and the Byzantine Generals problems is explained by Brooks and Iyengar in [17,18]. The sensor fusion and Byzantine agreement extensions to *NMR* are advantageous to the design of high reliability systems and guarantee that a system will continue to function correctly as long as more than half (in the case of one-dimensional sensor fusion) or more than two-thirds (in the case of Byzantine Agreement) of the individual components continue to function correctly.

This dissertation solves a design problem based on the application of fault masking for dependable systems design. We assume throughout that component failure is statistically independent. Application of our results requires dependability statistics for the individual components, which may be based on several existing statistical models. For sensor systems, statistical dependability models depend on the exact problem being considered; if the design is concerned primarily with mechanical failure then the use of mean-time-to-failure and mean-time-to-repair is usually adequate. If the design must also consider transient and intermittent errors, statistics must be available to determine the distributions of fault arrival and duration. The numerical examples given in this dissertation, however, use exponential distributions

for component failure and repair. This is for tractability and consistency with reliability literature. Equations are derived for systems where over half of all components must be functional, this can be changed to any fraction simply by replacing *N/2* as necessary.

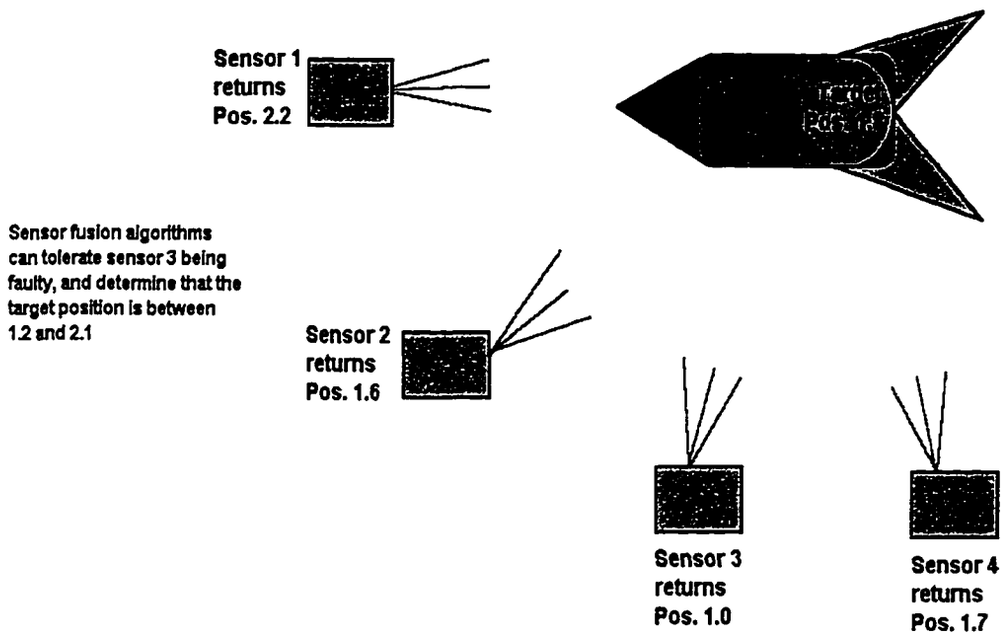

Fig 3.2 These sensors function correctly if the data is within 0.5 of the real value

Dependability is used as a generic term for both reliability and availability. When a system with strict dependability requirements is being designed using Byzantine agreement or sensor fusion algorithms, a choice must be made between different component types which could be used redundantly to mask errors. Given a choice between various modules, each with different dependability parameters and per item costs, we obtain the configuration which meets the dependability requirements with the lowest system cost.

4 CURRENT OPTIMIZATION METHODS

This dissertation views both the image registration and the redundant system cost minimization problems as combinatorial optimization problems. Combinatorial optimization problems are characterized by a set of solution vectors $\{x\}$ which consist of parameters describing possible solutions to the problem, and a fitness function $f(x)$ which associates a numerical value with each vector. The goal is to find the vector x for which $f(x)$ is a minimum (the same methodologies can be used to find maximum values by substituting $-f(x)$ for $f(x)$).

This chapter provides a brief description of the principal methodologies in use for solving these kinds of problems. Chapters five and six discuss which methodologies are applicable to these problems and how they have been applied. Explanations will also be given stating which methodologies are unsuited to solving these problems and why.

4.1 LINEAR PROGRAMMING

Probably the most widely used optimization methodology is linear programming. Given a cost vector $c = (c_1,c_2,...c_n)$ linear programming searches for the solution vector $x = (x_1,x_2,...x_n)$ which minimizes $f(x) = \Sigma c_i x_i$ and also satisfies a number of given constraints $\Sigma a_{ij}x_j \leq b_i$ (or $\Sigma a_{ij}x_j \geq b_i$). This is a large set of problems with a number of practical applications. It should be noted that the linear nature of this problem formulation excludes the existence of local minima of $f(x)$ in the parameter space.

Two major classes of methods exist for solving linear programming problems: the simplex method and interior methods. The simplex method was developed by a

research team led by Dantzig in 1947 [1]. Conceptually it considers the problem as an *N*-dimensional space where each vector x is a point in the space, and each constraint defines a hyper-plane dividing the space into two half-spaces. The constraints therefore define a convex region containing the set of points which are possible answers to the problem, called the *feasible set* [75]. The set of optimal answers must contain a vertex. The simplex algorithm starts from a vertex on the *N*-dimensional surface and moves in the direction of greatest improvement along the surface until the optimal point is found [1,75].

The first interior method was found by Karmarkar[1]. Interior methods start with a point in the feasible set and move through the interior of the feasible set towards the optimal answer without being constrained to remain on the defining surface. Research is active in finding less computationally expensive implementations of both interior and simplex methods.

4.2 TABU SEARCH

Tabu search is one of a class of non-monotonic search methods which have been proposed for solving optimization problems. All methods discussed in sections 4.2-4.4 have been constructed to find globally optimal solutions in problems containing local minima. Local minima in optimization problems can be either the result of the feasible set being defined by a non-convex surface, or the function $f(x)$ being non-linear. Optimization problems containing local minima are more difficult than the linear problems which can be solved by linear programming.

Monotonic methods include simulated annealing (see section 4.4) and threshold acceptance. The majority of search heuristics start at a point in the parameter space and move to neighboring points whose value of $f(x)$ is inferior to the current value. Monotonic methods avoid local minima by occasionally moving to a neighboring

point whose value of $f(x)$ is superior to the current value. The amount of increase that will be accepted by the method can be either probabilistic (simulated annealing) or deterministic (threshold acceptance) in nature and is dependent on a parameter whose value decreases as the search progresses [32,43].

Non-monotonic methods also occasionally move to points in the search space where the value of $f(x)$ is superior to the current value, but the algorithm's ability to make this move is not dependent on a strictly decreasing parameter. A number of methods have been proposed including Old Bachelor Acceptance [43] and many variations of Tabu Search [31]. Old Bachelor Acceptance is non-monotone in that it uses a parameter which is dependent on whether or not previous moves have been accepted, and on the number of moves remaining before the final answer is due [43]. Tabu search is non-monotone in that it disqualifies a number of moves due to the recent history of moves made by the algorithm [32]. We limit our discussion to Tabu search since it is the most widely implemented heuristic of this class.

Tabu search involves modifying an existing heuristic search by keeping a list of the nodes in the search space which were visited most recently by the search algorithm. These points then become "tabu" for the algorithm, where "tabu" means that these points are not revisited as long as they are on the list. This simple modification will allow a search algorithm to eventually climb out of shallow local minima in the search space. It requires less computation than simulated annealing, while providing roughly equivalent or superior results[32]. Several questions are being studied as to how to optimize Tabu searches, such as the optimal length for the tabu list [11], and methods for implementing parallel searches [76]. Our implementation uses a tabu list that is considered infinite.

The example shown in **figure 4.1** illustrates the problem addressed by tabu search. By following a greedy heuristic a search starting at point A, the search will move immediately to point B then C with decreasing values. All neighbors of point C have larger values than C. Point C is therefore a local minima and a greedy heuristic would return the value at point C (3.0) as the fitness function minimum. This is incorrect, the minimum value is 2.6 at point E. Tabu search keeps track of points already visited on a list and forbids movement to points on the tabu list. At point C the tabu list would be {A,B,C} and the lowest cost neighbor not in the tabu list would be point D. From point D, tabu search moves directly to the global minimum point E.

To go from point A to global minimum at point E, the search needs to escape from local minima such as point C.

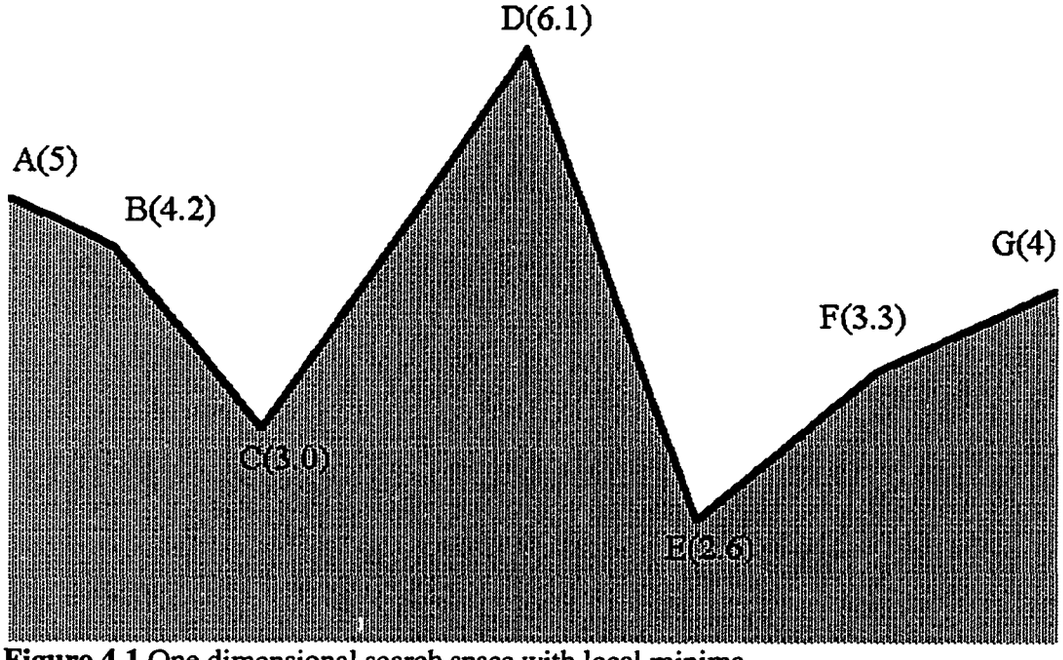

Figure 4.1 One dimensional search space with local minima.

The implementation of the Tabu search used in this dissertation relies on a "greedy" heuristic and starts with all parameters set to zero, i.e. there is neither

translation nor relative rotation between the two readings. The search can move one position in each direction. The algorithm evaluates the fitness function for each of these possibilities. Naturally, the search chooses to visit the next node in the direction with the minimum value for the fitness function. When one of these parameter sets is visited it is placed on the tabu list.

Values on the tabu list are disqualified for consideration in the future. Should the search arrive at a neighboring point later, the fitness function value given to parameter sets on the tabu list is set to a very large value essentially disqualifying it from consideration.

As each parameter set is visited by the search, the value attributed to it by the fitness function is compared to the parameter set already visited with the smallest value for the fitness function up to this point. If the value is smaller the parameter set now becomes the best fit found.

It is impossible to find a clear stopping criteria for this algorithm since the only way to be sure that the global minimum for the fitness function has been found is through an exhaustive search of the search space.

4.3 GENETIC ALGORITHMS

Genetic algorithms, first developed in the mid-1960's, attempt to apply a Darwinian concept of survival of the fittest to optimization problems. The idea being that biological systems adapt themselves to fit into an ecological niche. It was hypothesized that using the same evolutionary process with a parametric description of an answer instead of a DNA code would result in a series of increasingly fit answers. This metaphorical approach has been shown experimentally to be useful for solving many difficult optimization problems.

Possible solutions to a problem are called *chromosomes* and a diverse set of chromosomes are grouped into a *gene pool.* The relative quality of these answers are determined using a *fitness function,* and this quality is used to determine whether or not the chromosomes will be used in producing the next generation of chromosomes. The contents of high quality solutions are more likely to continue into the next generation. The next generation is generally formed via the processes of *crossover*: combining elements of two chromosomes from the gene pool, and *mutation*: randomly altering elements of a chromosome. Refer to [40,73] for details. An in depth comparison of genetic algorithms versus exhaustive search for another reliability design problem can be found in [53].

A large number of strategies exist for determining the contents of a new generation. Two different genetic algorithms were used to contrast their effectiveness. They differed only in their reproduction strategies. The strings used to characterize the problem here consisted of the same parameters used for the tabu search, and the same fitness function was used.

4.3.1 CLASSIC REPRODUCTION SCHEME

The first strategy has been described by Holland [40]. Each string in the gene pool is evaluated by the fitness function. Based on the quality of the answer represented by the string it is assigned a probability of being chosen for the pool of strings used to produce the next generation. Those with better answers being more likely to be chosen. A mating pool is then constructed by choosing strings at random from the gene pool following the probability distribution derived.

The new generation is formed by mixing the elements of two strings in the mating pool chosen at random. This is generally called crossover. In both genetic algorithms used in this study a crossover probability of one was used. Where the string was

split is chosen at random and all points were equally likely. A certain amount of mutation exists in the system, where one element at random is replaced by a random value. In our implementations mutation occurs once with every 700th string processed.

4.3.2 ELITE

The second strategy applied has been described in a recent paper by Bean [12]. This strategy is described as elitist since the 20% of the strings with the best fitness function values are copied directly into the next generation. In addition to this, in our implementation, 3% of the strings for the next generation are the result of random mutations. In this implementation the random mutations were strings where all elements were chosen at random. The rest of the new generation is formed by performing crossover between random strings in the current generation. The choice is done entirely at random, no weighting based on the quality of the string is performed. Bean reports that this strategy has been found to be stable experimentally. Its implementation is straightforward.

Genetic algorithms are not sensitive to the presence of local minima since they work on a large number of points in the problem space simultaneously. By comparing many possible solutions they achieve what Holland has termed *implicit parallelism* which increases the speed of their search for an optimal solution[40]. Discussion of the advantages gained by using genetic algorithms instead of exhaustive search for a different optimization problem based on system reliability can be found in [64].

4.4 SIMULATED ANNEALING

Simulated annealing attempts to find optimal answers to a problem in a manner analogous to the formation of crystals in cooling solids. A material heated beyond a

certain point will become fluid, if the fluid is cooled slowly the material will form crystals and revert to a minimal energy state. Refer to [54] for a full description of simulated annealing and a discussion of its scientific basis.

The strategy of the algorithm is again based on a *fitness function* comparing the relative merit of various points in the problem space. For our applications we use the same parameters to create vectors describing points in the answer space with simulated annealing as with tabu search and genetic search. Likewise the same fitness function is used. The exact functions and parameters used are described in chapters five and six. The algorithm starts at a point in the search space. From the algorithm's current position a neighboring point is chosen at random. The difference in the fitness function value between the new point and the current point is calculated. This difference is used together with the current system temperature to calculate the probability of the new position being accepted. This probability is given by the distribution $e^{-\Delta C/\tau}$. The process continues with the same temperature τ for either a given number of iterations, or until a given number of positions have been occupied, at which time the value τ is decreased. The temperature decreases until no transitions are possible, so the system remains frozen in one position. This occurs only when ΔC is positive for all neighboring points, therefore the position must be a local minimum and may be the global minimum[67].

The simulated annealing method used in our research is based on the algorithm given in [54,67]. The algorithm has been modified so that the parameters being optimized and the fitness function are appropriate for our applications, and a cooling schedule has been found which allows the algorithm to converge to a reasonable solution.

Just as many different reproduction schemes exist for genetic algorithms, several possible *cooling schedules* exist for simulated annealing. A cooling schedule is defined by the initial temperature, the number of iterations performed at each temperature, the number of position modifications allowed at a given temperature and the rate of decrease of the temperature. The answers found by the algorithm are directly dependent on the cooling schedule and no definite rules exist for defining the schedule [54,67]. The cooling schedule is important in that it determines the rate of convergence of the algorithm as well as the quality of the results obtained.

4.5 ARTIFICIAL NEURAL NETWORKS

For completeness the use of artificial neural networks for solving optimization problems is discussed in this section. In spite of their successful application to some problems, neural networks have not been applied to solving the problems addressed by this dissertation. The reasons for this will be explained in chapters five and six.

Artificial neural networks, like simulated annealing and genetic algorithms, originated as a simulation of a natural phenomena. They are also called *connectionist learning schemes* since they store information in weights given to connections between the artificial neurons[39]. The first artificial neural network application was developed in the 1940's by McCulloch and Pitts as a system designed to mimic a theory of how biological nervous systems work [26]. Since then a number of variations of artificial neural networks have been developed some of which bear little resemblance to functioning biological systems [26,37,39,51].

Each neuron is essentially a primitive computational device which accepts a number of inputs and outputs a function of the weighted sum of the inputs. The function is commonly a step function, a sigmoid function, or another similar function [51]. A number of architectures have been proposed for connecting the neurons into

networks, among which are fully connected networks and networks divided into layers. Different architectures are capable of performing different tasks such as pattern recognition[37], classification of inputs [39] and solution of classical optimization problems [26]. A neural net approximates a function which classifies input into regions defined by a number of hyperplanes. Depending on the architecture of the neural net, the threshold function used, and the weights given to the connections between the neurons, this function may become arbitrarily complex [51].

5 A NEW APPROACH TO IMAGE REGISTRATION

This chapter explains the applications of the methods discussed in chapter 4 to the image registration problem posed in chapter two. This problem has been approached using the methodologies explained in chapter four. Some are not appropriate for either problem and this chapter explains why they are inappropriate. Application details and pseudo-code are given concerning the simulations which have been used to determine the relative effectiveness of the various methods. Heuristic strategies and probabilities are explicitly provided as well. Results from sample data sets are given in chapter seven.

Artificial neural networks have not been used for either problem. While neural nets have been used with a certain amount of success for solving optimization problems, such as the traveling salesman problem [26,51], two drawbacks exist for applying a connectionist approach to these problems. The first problem is that encoding the input and output is not obvious. If a straightforward approach were used for the image registration problem and one neuron used per sensor pixel input the number of neurons needed would be prohibitive. If fully connected weight tables were used then the amount of storage required would be $O(N^2)$ with N being the number of pixels which is also prohibitive. The cost minimization problem faces similar difficulties. The problem is phrased so that the number of component types is not limited, and in most cases will be large. A naive coding which inputs one component type per pixel is also excluded. This is not to say that it is impossible to phrase the input and output of these problems in a manner which is appropriate for neural networks. The correspondence is not obvious.

The second drawback to using a connectionist approach is the time required for training a neural network. Both of these problems would require large numbers of neurons. For the registration problem this is due to the number of pixels involved. For the cost minimization problem it will be shown in chapter six that the surface defining the feasible space is very jagged. This surface could be defined by a set of hyperplanes, each hyperplane corresponding to a neuron, but the cardinality of the set would be a very large number compared to the number of component types. Due to the size of the networks required, it is clear that training the networks would be time consuming, and it is doubtful that a connectionist approach would provide better answers in the same amount of time as the other approaches.

Linear programming is also inappropriate for both problems. It cannot be used for image registration since the fitness function $f(x)$ is not a linear combination of the elements of the vector x. In fact, $f(x)$ is directly dependent on the sensor images so it is impossible to make any valid assumptions about the type of function it may be.

Linear programming is also inappropriate for the cost minimization problem. This is in spite of the fact that $f(x)$ is a linear combination of the elements of x for this problem. The constraints defining the feasible set for this problem are non-linear and, as will be demonstrated in chapter seven, the surface defining the feasible set is not convex. Since these assumptions are not met, linear programming and extensions of linear programming cannot be applied to solving these two problems.

The rest of this chapter explains the implementations of tabu search, genetic programming, and simulated annealing used for approaching the image registration problem.

5.1 PARAMETERS AND FITNESS FUNCTION

We have defined the goal of our image matching problem as finding the gruence (translation and rotation) which best matches the two sensor images. This is easily translated into an optimization problem where we try to find a vector x which minimizes the value of a function $f(x)$. In which case, the vector x consists of three parameters: x_1 displacement in the x direction, x_2 the displacement in the y direction and x_3 the angle of rotation.

The algorithms used to register images need to compare the quality of different vectors x. All the approaches used in this research were implemented using the same fitness function $f(x)$.

The noise which is introduced is approximately Gaussian. Gaussian noise, also known as white noise, follows a normal distribution and has an expected value of zero. The noise introduced here follows a normal distribution as much as possible, but the gray scale used has a range limited to 256 discrete values. A pixel with a gray scale of 255 can not have a larger value due to noise since the resulting value will go beyond the gray scale, but noise can reduce the value of that pixel. A similar effect exists for pixels with a gray scale value of 0. In spite of this, the assumption was made that, over the entire intersection, there will be no appreciable bias to the noise.

The fitness function is derived by first computing the intersection between sensor 1 and sensor 2 using the parameter set to be evaluated. The gray levels of every pixel from sensor 1 in the intersection are compared with the gray level of the corresponding sensor 2 pixel. If read1(x,y) is the value returned by sensor 1 at point (x,y) and read2(x',y') is the reading returned by sensor 2 at point (x',y'). Point (x',y') is found by reversing the translation and rotation defined by the parameters being tested. It is possible to present the difference of read1(x,y) and read2(x',y') as:

$$read1(x,y) - read2(x',y') = (v1(x,y) + noise1(x,y)) - (v2(x',y') + noise2(x',y'))$$

where v1(x,y) and v2(x',y') are the actual gray scale values and noise1(x,y) and noise2(x,y) are the noise in the sensor 1 and sensor 2 readings respectively.

This expression can be rewritten as:

$$read1(x,y) - read2(x',y') = (v1(x,y) - v2(x',y'))+ (noise1(x,y) - noise2(x',y'))$$

If we square this value and sum it over the entire intersection this becomes:

$$\Sigma\ (read1(x,y) - read2(x',y'))^2 = \Sigma\ ((v1(x,y) - v2(x',y'))+ (noise1(x,y)\ \ noise2(x',y')))^2$$

Note that when the parameters are correct the gray scale values v1(x,y) and v2(x',y') will be identical, and this expression becomes:

$$\Sigma\ (read1(x,y) - read2(x',y'))^2 = \Sigma\ (noise1(x,y) - noise2(x',y'))^2$$

Since all noise follows the same distribution with the same variance the expected value of this is identical for all intersections of the same area and, as such, the minimum value for the function over all intersections of a given area. Variation in this value thus consists of two parts: the difference in the gray scale values of the noise-free image, and a random factor which is distributed according to a Chi-square distribution of unknown variance. The number of degrees-of-freedom for the Chi-square distribution is the number of pixels in the intersection.

It is possible to have small intersections which match coincidentally. In order to favor intersections of larger area we divide by the number of pixels in the intersection squared. The fitness function *f(x)* thus becomes:

$$\Sigma\ (\text{read1}(x,y) - \text{read2}(x',y'))^2 / (\text{Number of pixels in the intersection})^2$$

The expected value of a Chi-square function is the number of degrees of freedom, and the number of degrees of freedom in this case is equal to the number of pixels in the intersection. In the case of a perfect fit (i.e. $v1(x,y) = v2(x',y')$) the expected value of this function is therefore within a constant factor of:

$$1 / (\text{Number of pixels in the intersection})$$

This function is the summation of the error per pixel squared over the intersection of the sensor 1 and sensor 2 readings. As shown above, the unique global minimum of this function is found when using the parameters which define the largest intersection where the gray scale values of sensor 1 are the same as the gray scale values of the translated and rotated sensor 2 reading.

In practice, this fitness function *f(x)* adequately reflects the quality of the answers represented by a given set of parameters, as is shown by the simulation. Note that all three algorithms used in this dissertation depend on this fitness function being an accurate measure of the quality of a potential answer.

Other fitness functions could exist which adequately represent the quality of potential answers. Since these functions measure the same effects as the fitness function derived in this section, the values returned by these hypothetical functions

must be approximately equal to the values given by the fitness function used here. Two accurate metrics of the same phenomenon must give similar readings. For this reason, replacing the fitness function with another equally valid function would not severely affect the results of the experiments presented in this dissertation. The fitness function derived in this section is suited to measuring the gruence of noisy images since it has a global minimum where the non-stochastic portion of the data provides the optimal answer, and the stochastic portion of the data is represented by a consistent known statistical distribution over the entire answer space.

5.2 TABU SEARCH FOR IMAGE REGISTRATION

As stated in section 4.2 tabu search involves modifying an existing heuristic search by keeping a list of the nodes in the search space which were visited most recently by the search algorithm and prohibiting moves to these points. Our implementation uses a tabu list that is considered infinite.

Algorithm: tabu_search
Inputs: 2 sets of noise corrupted sensor readings S_1, S_2.
Outputs: Vector x with the best gruence matching the sensor readings.
Procedure:

Step 1: Initialize x to no translation and no rotation.
Step 2: $Best = x$
For $I := 1$ to N do /* N is the number of iterations, found experimentally */
  Step 3: Compute $f(x)$ for all six neighbors of x. If any neighbor is on the tabu list set the value of $f(x)$ for that neighbor to a prohibitively large value.
  Step 4: Set x to the neighbor with the smallest $f(x)$ value.
  Step 5. If $f(x) < f(Best)$ then $Best = x$
  Step 6. Append x to tabu list.
  Step 7. Get new instances of S_1 and S_2
end for loop.
return(*Best*)

Figure 5.1 Pseudo-code for image registration using tabu search.

The implementation of the tabu search used here relies on a "greedy" heuristic and starts with all parameters set to zero, i.e. there is neither translation nor relative

rotation between the two readings. The search can move one pixel in each x direction, one pixel in each y direction, or by rotating plus or minus one degree. The algorithm evaluates how well the two sensor readings would match, using the fitness function, for each of these six neighboring x vectors. The search visits the node in the direction with the minimum value for the fitness function $f(x)$. When a vector x is visited it is placed on the tabu list. Should the search return to the neighborhood of x later, the fitness function value given to x is set to a very large value. As each x is visited, the value of $f(x)$ *is c*ompared to the smallest value of $f(x)$ up to that point. If the value $f(x)$ is smaller, x now becomes the best fit found. Each iteration of the tabu search is run on a new instance of the noisy sensor one and two readings. **Fig. 5.1** presents pseudo-code for this implementation of tabu search.

Since no clear stopping criteria exists for tabu search, this study was done by comparing the results from a given number of iterations of the tabu search with the results obtained by performing the same number of iterations with the genetic algorithms, and the results from simulated annealing which ran until reaching its stopping criteria.

5.3 NEW GENETIC ALGORITHM FOR REGISTRATION

A large number of strategies exist for determining the contents of a new generation, the registration simulation used two different strategies to contrast their effectiveness. The strings used to characterize the problem were: the offset in the x direction, the offset in the y direction, and the angle of rotation. The fitness function used has also been described in section 5.1.

The gene pools consist of 150 sets of parameters which were initialized with random values at the start of the program. The two different genetic algorithms which were used differed only in their reproduction strategies.

The first strategy, which we refer to as the classic genetic algorithm, has been proposed by Holland [40]. the function $f(x)$ is computed for each x in the gene pool. A mating pool is chosen by picking 150 elements from the old gene pool. Each member of the gene pool has a normalized weighted probability of being chosen proportional to 1 - $f(x)$ / (sum of $f(x)$ for all x). The new generation is formed by mixing the elements of two strings in the mating pool chosen at random. This is generally called crossover. In both genetic algorithms used in this study a crossover probability of one was used.

Algorithm: classic_genetic_search
Inputs: 2 sets of noise corrupted sensor readings S_1, S_2.
Outputs: Vector x with the best gruence matching the sensor readings.
Procedure:

Step 1: Generate initial gene pool *GP* of 150 vectors x chosen at random.
Step 2: Set *Mutation_counter* to 0.
For I :=1 to N do /* N is the number of iterations, determined experimentally */
  Step 3: Compute $f(x)$ for all x.
  Step 4: *Total* = sum of all $f(x)$
  For J := 1 to 150 do
    Step 5: Copy a vector x chosen from *GP* with a normalized probability of
      1 - $f(x)$/*Total* into *GP_breed[J]*
  end for loop with counter J.
  For J := 1 to 150 do
    Step 6: Increment *Mutation_counter*.
    If *Mutation_counter* = 700 then
      Step 7: *GP[J]* = an x chosen totally at random.
      Step 8: *Mutation_counter* := 0
    Else
      Step 9: *GP[J]* = cross between two x's chosen at random from *GP_breed*
    end if
  end for loop with counter J.
  Step 10: Get new instances of S_1 and S_2.
end for loop with counter I.
Step 11: *Best* = element of *GP* with smallest $f(x)$.
Step 12: return(*Best*)

Figure 5.2 Pseudo-code for image registration using the classic genetic algorithm.

Since x consists of three elements, the result of this mixing always consisted of two elements from one parent and one from the other. The element switched was chosen at random and all three were equally likely. A certain amount of mutation exists in the system, where one element at random is replaced by a random value. In our implementation mutation occurs once with every 700th x processed consisting of random elements. The pseudo-code for the classic reproduction genetic algorithm is shown in **Fig. 5.2.**

Algorithm: elite_genetic_search
Inputs: 2 sets of noise corrupted sensor readings S_1, S_2.
Outputs: Vector x with the best gruence matching the sensor readings.
Procedure:

Step 1: Generate initial gene pool *GP* of 150 vectors x chosen at random.
For *I*:= 1 to N do /* *N* is the number of iterations, determined experimentally */
Step 2: for *h*:= 1 to 150 compute *f(GP[h])*.
Step 3: for *h* = 1 to 30 begin /* keep the elite */
GP_next[*h*] = the h'th least expensive configuration in *GP*.
end
Step 4: for *h* = 31 to 145 begin /*new combinations from crossover*/
GP_next[*h*] = randomly combine 2 configurations chosen at random from *GP*.
end
Step 5: for *h* = 146 to 150 begin/*mutants */
GP_next[*h*] = randomly create configurations.
end
Step 6: *GP* = *GP_next*
Step 7: Get new instances of S_1 and S_2.
end for loop with counter *I*.
Step 8: *Best* = element of *GP* with smallest *f(x)*.
Step 9: return(*Best*)

Figure 5.3 Pseudo-code for image registration using the elitist genetic algorithm.

The second strategy which we applied has been described by Bean [12]. This strategy is described as elitist since the 20% of the x vectors with the best fitness function values are copied directly into the next generation. In our implementation another 3% of the gene pool for the next generation are random mutations where all

three elements are chosen at random. The remaining 77% of each new generation is formed by performing crossover between random x vectors in the current generation. The choice is done entirely at random, no weighting based on the value of $f(x)$ is performed. Bean reports that this strategy has been found to be stable experimentally. Its implementation is straightforward. **Fig. 5.3** shows the pseudo-code for the elitist genetic algorithm used for image registration.

5.4 SIMULATED ANNEALING FOR IMAGE REGISTRATION

As explained in section 4.4, simulated annealing attempts to find optimal answers to a problem in a manner analogous to the formation of crystals in cooling solids. For the image registration problem we applied a simulated annealing algorithm using the same coding for the solution vector x and the fitness function $f(x)$ as used for the tabu search, and genetic algorithms implementations. Since simulated annealing has a clear stopping criteria, this algorithm ran until that criteria was reached and not for an arbitrary number of iterations as was the case with tabu search and the genetic algorithms.

As with tabu search, the algorithm starts its search with the initial position in the search space corresponding to no translation and no rotation. From the algorithm's current position one of the six neighboring points (±1 pixel in the x direction, ±1 pixel in the y direction, ± degree of rotation) is chosen at random. The cost difference between the new point and the current point is calculated. This difference is used together with the current system temperature to calculate the probability of the new position being accepted. This probability is given by a Boltzmann distribution $e^{-\Delta C/\tau}$. The initial value of τ is set at 1.1. The process continues with the same temperature τ for either 5120 iterations, or until 512 positions have been occupied, at which time the value τ is decreased by 10%. The temperature decreases

until no transitions are possible, so the system remains frozen in one position. Since this occurs only when ΔC is positive for all neighboring points, the position must be a local minimum and may be the global minimum[67]. The simulated annealing method used in our research is based on the algorithm given in [54,67]. Pseudo-code of the simulated annealing algorithm is summarized in **Fig. 5.4**.

```
Algorithm: simulated_annealing
Inputs: 2 sets of noise corrupted sensor readings S1, S2.
Outputs: Vector x with the best gruence matching the sensor readings.
Procedure:

Step 1: CC = ( 0,0,0)      /* Initial position in search space          */
Step 2: τ = 1.0            /* Initial temperature                       */
Step 4: CC_mod = 1 and iterations = 0
While (CC_mod not = 0) and (iterations < 100) do
 begin
   Step 5: CC_mod = 0 and inner_loop_iter = 0
    While(CC_mod < 512) and (inner_loop_iter < 5120) do
    begin
      Step 6: new_CC = random modification of CC
      Step 7: ΔC = f(CC) - f(new_CC)
      if(ΔC < 0) or (according to Boltzmann distribution using ΔC and τ) then
       begin
         Step 8: CC = new_CC
         Step 9: CC_mod = CC_mod + 1
       end
     Step 10: inner_loop_iter = inner_loop_iter + 1
    end
   Step 11: τ = 0.9 * τ
   Step 12: iterations = iterations + 1
 end
Step 13: return(CC).
```

Figure 5.4 Pseudo-code for image registration using simulated annealing.

5.5 ANALYSIS OF ALGORITHMS

The image registration problem which is being considered can also be stated as finding a function which best maps a subset of an arbitrary array of gray scale values to a subset of another arbitrary array. Since no assumptions are made regarding the

contents of the two images, it is possible to construct arrays where the search space defined by the fitness function has an arbitrary configuration.

Given a problem with an arbitrary search space, the only method guaranteed to always find the globally optimum answer is an exhaustive search. Since an exhaustive search is prohibitively expensive, this dissertation proposes the application of general optimization methods which often find reasonable approximations of globally optimal answers of problems whose search space contains local optima.

There is no guarantee that these algorithms will find the most desirable answer. The only algorithm with a concrete stopping criteria is simulated annealing. The answer found by simulated annealing will be locally optimal. Tabu search and genetic algorithms have no concrete stopping criteria and in our tests have been run for an arbitrary number of iterations. Simulated annealing and genetic algorithms also depend on the use of nondeterminism for making decisions. Since a number of arbitrary choices are made by the algorithm it is unreasonable to deduce convergence bounds for the algorithms.

For these reasons, the success of these approaches are measured experimentally instead of analytically. Our experimental results are presented and discussed in chapter seven. These results show the relative merits of the approaches proposed here, and indicate the value of the elitist genetic algorithm for the image registration application.

6 A NEW APPROACH TO COST MINIMIZATION

This chapter explains the applications of the methods discussed in chapter four to the cost minimization problem posed in chapter two. This problem has been approached using many of the methodologies explained in chapter four. Some are not appropriate for these problems and have not been applied. The explanations for this are given at the start of chapter five. Application details are given here concerning the simulations which have been used to determine the relative effectiveness of the various methods. Pseudo-code is given to provide a precise description of each method. Heuristic strategies and probabilities are explicitly provided as well.

First we discuss the dependability constraints on solutions and how they can be calculated. The solution vector x and fitness function $f(x)$ are derived after that. The next section discusses an exhaustive search approach to the problem. Exhaustive search was not used for image registration since its application for that problem is trivial. Finally, implementations of tabu search, an elitist genetic algorithm, and simulated annealing for this problem are discussed. Results from these implementations are given in chapter seven.

6.1 DEPENDABILITY CONSTRAINTS

To compute system reliability, we may either consider a Markov model or perform a combinatorial analysis. **Figure 6.1** shows a Markov chain to model the process (for reliability evaluation, we assume the repair rate μ is zero). Let there be N identical components which fail independently. Assuming that the rates shown in

Figure 6.1 are constant and that the system starts at time $t = 0$ with all components working, we get the following differential equation:

$$\mathrm{d}P_N(t)/dt = -N\lambda P_N(t) \qquad (1)$$

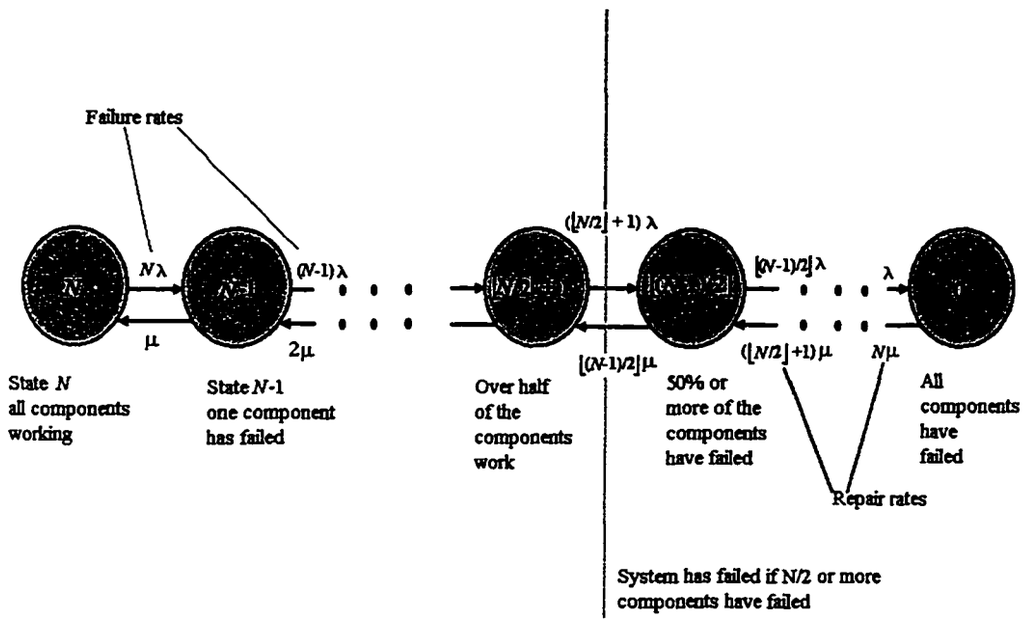

Figure 6.1 Markov chain availability model for a system which can tolerate failure of up to half of its components.

Equation (1) is solved directly using Laplace transform or indirectly using an appropriate numerical technique. The probability of system being in state i, where i components work, is given in (2).

$$\binom{N}{i} [e^{-\lambda t}]^{i} [1-e^{-\lambda t}]^{N-i} \qquad (2)$$

The reliability of the system is the sum of the probabilities given by equation 2 for the states with i equal to N to $\lfloor N/2 \rfloor + 1$ because the sensor fusion algorithm defines the system as functional only when more than $N/2$ components are operational.

Alternatively, a combinatorial approach can be used to compute system reliability. Assume each component has an identical probability of success $r(t)$ (if a component has a constant failure rate λ, $r(t) = e^{-\lambda t}$). Let $q(t) = 1 - r(t)$ be the component failure probability. The assumption of statistical independence allows us to use Bernoulli's law, which finds the probability of i out of N components working at time t as:

$$\binom{N}{i} [\, r(t)]^{\,i} [\, q(t)]^{\,N-i} \qquad (3)$$

Note the similarity between (2) and (3). Again the reliability for the system is the summation of the terms with i varying from N to $\lfloor N/2 \rfloor + 1$. The combinatorial approach offers two advantages: First, it is independent of the distribution defined by the reliability function $r(t)$. Second, it is easier to apply to the case where more than one type of component is used (a situation encountered in sensor fusion). For example, consider a simple instance where two different types of components exist in a system. If $N_1(N_2)$ is the number of components of type 1(2) such that $N_1+N_2=N$, the Markov chain model would require order of $3*(N-2)$ states. Thus, solving the problem using this approach becomes computationally expensive as the number of types of modules increases. The derivation using Bernoulli's law can be extended to cover the new configuration with little additional computation. The following analysis explains the procedure: first we describe a model with two types of components, then we provide a generalized version.

Consider the cases where no components of type 1 have failed, one component of type 1 has failed, *etc.*, up to the case where all N_1 components of type 1 have failed. These cases are disjoint and the sum of the probabilities of these cases is one [23]. It, thus, partitions the sample space giving an expression for system reliability at time t in terms of $r_1(t)$ the reliability of component type 1, and $r_2(t)$ the reliability of component type 2.

$$R(t) = \sum_{k=0}^{N_1} \left[\binom{N_1}{k} [r_1(t)]^k [1 - r_1(t)]^{N_1 - k} \sum_{m=\lfloor N/2 \rfloor + 1 - k}^{N_2} \left[\binom{N_2}{m} [r_2(t)]^m [1 - r_2(t)]^{N_2 - m} \right] \right] \quad (4)$$

The concept can easily be extended to more than two types of components. Evaluating a combination of J different types of components will require J levels of summations in the format of (4).

Figure 6.1 shows a Markov model for availability computation if the repair rate μ is non-zero. Since the chain is finite and strongly connected, the system represented by the model will eventually reach a steady state. The formula, thus derived, is the same as the reliability equation (3), except that component reliability $r(t)$ is replaced by component availability $a(t)$ [70]. Note that, for components with a constant failure rate λ and a constant repair rate μ, the steady state component availability a will be $\mu/(\mu+\lambda)$. Also, steady state component unavailability is $1-a$ or $\lambda/(\mu+\lambda)$. Following this logic and assuming each component failure and repair is statistically independent, we obtain the probability of exactly i components out of N total being available for a system of one component type with component availability a when the system has reached a steady state. As long as the components reach a steady state availability, this equation applies to non-exponential distributions types for component failure and repair rates. The steady state system availability is the

summation of the probabilities of exactly N to $\lfloor N/2 \rfloor+1$ components being available in the steady state. If the system consists of two different component types: N_1 components of type 1 with component availability a_1 and N_2 components of type 2 with component availability a_2, we obtain the system availability by substituting a_i for r_i in equation (4). As with reliability, the extension of (4) to more than two types of components is straight forward and can be obtained using recursion.

The following algorithm, given in [71] but reproduced here from [69], is equivalent to (4) and helps reduce the number of multiplications drastically. For sensor fusion applications the constant m would be equal to $\lfloor (N-1)/2 \rfloor$.

Input: component failure probabilities $q_1, q_2, \ldots, q_N$
Output: reliability of system composed of the N components Pr(*f*)
Procedure:
Step 1a. Let $s[1,1] = q_1$
b. For $k=(2,\ldots,N)$ and $m=(2,\ldots,k-1)$, obtain
$s[1,k] = s[1,k-1] + q_k$
$s[k,k] = s[k-1,k-1] * q_k$
$s[m,k] = s[m,k-1] + s[m-1,k-1] * q_k$
Step 2. Solve:

$$Pr(f) = \sum_{k=m}^{N} (-1)^{k-m} \binom{k-1}{m-1} s[k,N]$$

The algorithm calculates the nested summations efficiently by creating a tableau of intermediate terms. Should the number of components of a single type be large, say N_1, the algorithm may be made more efficient by making the first N_1 elements of the tableau all elements of type T_1. In this way column N_1 can either be approximated using Stirling's approximation or calculated directly as:

$$s[i,N_1] = \binom{N_1}{i} q_1^{\,i} \qquad (5)$$

Figure 6.2 shows the elements of the tableau which must be computed in this special case. If Stirling's approximation is used approximately N_1 + 8(N-m) multiplications and 3(N-m) logs must be calculated to compute all elements $s[i,N_1]$. If $N_1/2 < m$, as is the case for the sensor fusion problem, in the worst case calculating $s[i,N_1]$ directly requires approximately N_1 + $m(4N_1-m)/2$ + N-m multiplications. The original algorithm takes $N_1*(N-m)/2$ multiplications to compute the column. O((N-N_1)(N-m)) multiplications are required to compute the rest of the tableau.

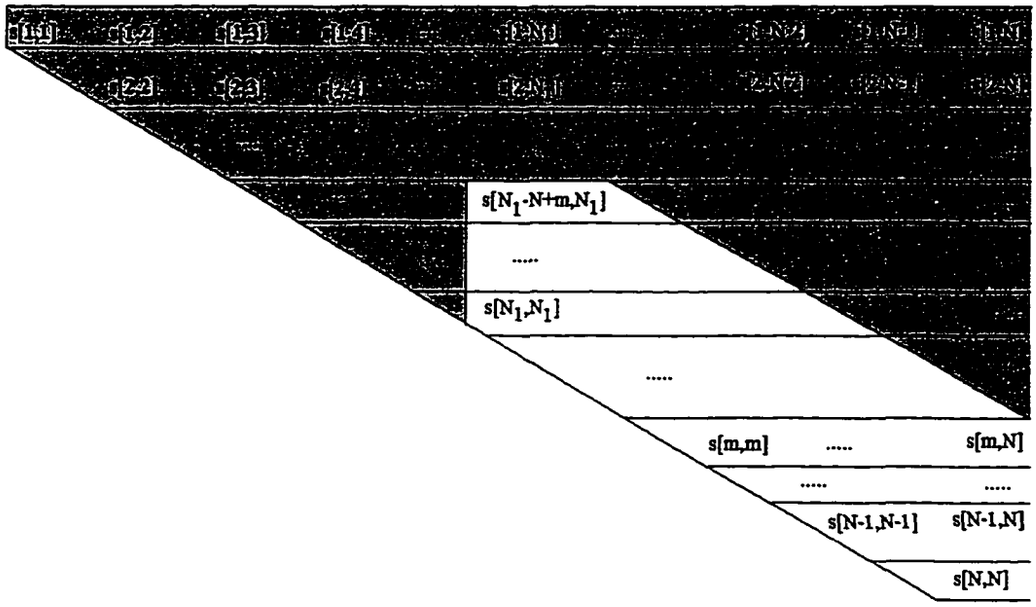

Figure 6.2 If there are a large number of components of one type, only the non-shaded elements need be computed.

6.2 FITNESS FUNCTION AND SOLUTION VECTOR

Section 6.1 provides a metric to determine which system configurations meet the given dependability requirements. Here, we find a model for the configuration that both fulfills the dependability criteria and has the minimum cost. Components have

either a known component reliability function (for systems with a fixed mission time) or a known component availability (for systems where the percentage of the time the system is functioning is most important.) For experimental results we assume constant failure and repair rates. Each component type is characterized by its dependability statistics and positive per item cost.

To find the combination of components meeting dependability requirements with the lowest total cost, it is necessary to consider each combination of J types of components as a point in a discrete J-dimensional space. This point is described by a J-dimensional vector $(Q_1,Q_2,...,Q_J)$ where each position in the vector corresponds to the number of components of a given type included in the system. For example, if the choice is to be made among three types of components, the combination of 2 components of type 1, 25 components of type 2, and none of type 3 corresponds to point (2, 25, 0). Equation (4) determines which points in the J-dimensional space correspond to systems that fulfill the reliability requirements, and which do not. Since each type of component has a given per item cost it is also possible to determine the total cost of each combination. If component type i has cost c_i, the cost of combination (2,25,0) is $2*c_i + 25*c_2 + 0*c_3$. The question remains as to how to find the combination which minimizes:

$$f(x) = \sum_{i=0}^{J} c_i * Q_i \qquad (6)$$

This is a combinatorial optimization problem. Unfortunately it can not be solved by established mathematical programming techniques such as the Simplex algorithm for linear programming, or integer programming. These established techniques are inappropriate since equation (4) which defines whether or not a given combination

fulfills the necessary dependability requirements is non-linear [70]. Another approach is necessary.

In spite of the nonlinear dependability equations, the problem resembles linear programming to a certain extent. It is necessary to find the optimal point in a J-dimensional solution space where J is the number of component types under consideration. The region with valid solutions is known as the feasible set in the J-dimensional space [75]. It is possible to determine a surface which divides the points in the feasible set from the rest of the J-dimensional space. We will use this surface to restrict our search for the optimal configuration to a small portion of all configurations in the feasible set.

Lemma 1. An optimal answer must lie on the surface dividing the J-dimensional problem space into two regions: one region containing points which satisfy the dependability requirements, and one region containing points which do not satisfy the dependability requirements.

Proof. Three distinct types of points exist: points beneath the surface in the region which does not satisfy reliability constraints, points on the surface dividing points which satisfy reliability constraints from those which do not, and points above the surface in the region which satisfies reliability constraints. Points below the surface can be dismissed trivially since they do not satisfy the dependability constraints. Any point K above the surface satisfies dependability constraints, but can be dismissed since there is at least one other point L which satisfies the dependability constraints with at least one component of at least one type fewer. Point L has a lower cost than point K, since all costs are positive. Thus the minimum cost point being sought must be located on the surface defining the set of points which satisfy the reliability requirements.

6.3 EXHAUSTIVE SEARCH

First, we present a method based on an exhaustive search, it finds the minimal cost solution by considering all points on the surface defined in section 6.2. Since many decisions involve a choice between a relatively small number of alternative components this search will often be reasonable. If the choice is among a large number of component types, the time required will be prohibitive, and the heuristics presented in sections 6.4, 6.5, and 6.6 should be considered.

In order to search the entire surface for the minimum cost point it is necessary to consider the surface's exact shape. To explain the surface shape it is easiest to start with the one dimensional case which can then be extended to cover higher dimensions.

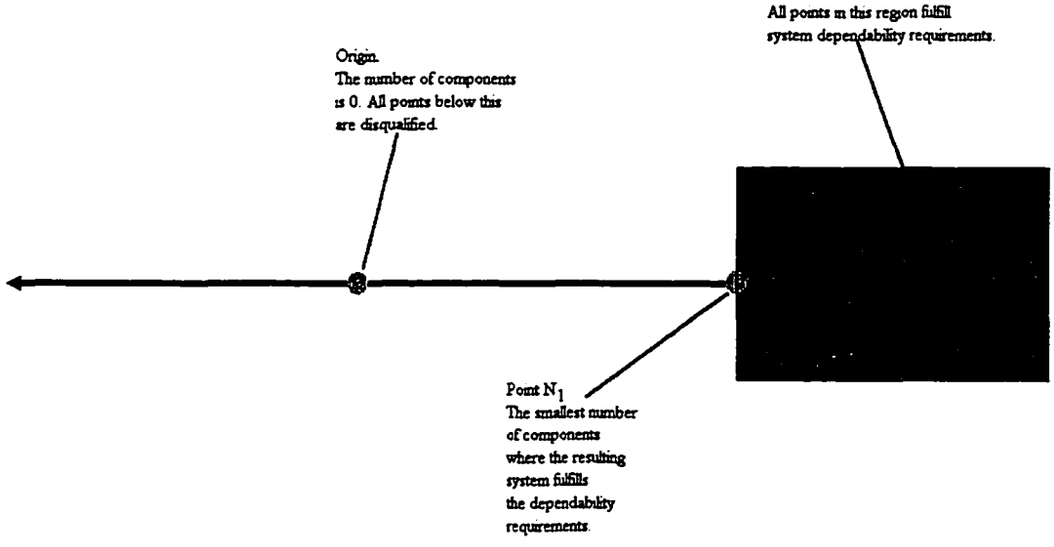

Figure 6.3 Half-space satisfying constraints for one dimension.

Component dependability is constrained to a range of values between 0 and 1. A value of 1(0) indicates a component never fails(functions). The dependability value of

a system tends towards zero, remains about 1/2, or asymptotically approaches 1 with the increase in the number of components comprising the system, provided the dependability measure of the components is less than 1/2, exactly 1/2, or greater than 1/2, respectively. It, thus, depicts the "S-Shaped property" described in [9]. If the components are perfect, system reliability will be 1 no matter how many components are used. Fault masking systems are feasible for component types whose dependability values are between 50% and 100%. Only components whose dependability is within this range should be considered and the rest of this dissertation implicitly assumes components fit this requirement. **Figure 6.3** shows the problem space, when only one type of component is under consideration. The feasible set which fulfills the dependability requirements is a half line and the bounding surface is a point N_1. As long as the system dependability constraint is less than 1 and individual component dependability is greater than 1/2, the fact that system dependability asymptotically approaches 1 as the number of redundant components increases proves that some number of components, N_1, must exist which fulfills system dependability constraints. In the one dimensional case, point N_1 is all the exhaustive search algorithm needs to find.

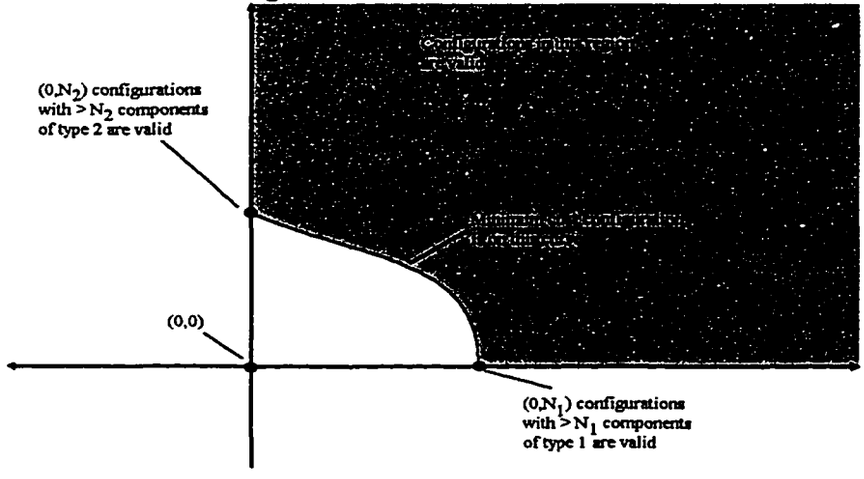

Figure 6.4 Bounding surface for two dimensional case.

Figure 6.4 shows the feasible set and bounding surface for the two dimensional case. Note that this diagram shows general relationships and not necessarily the actual curve. Example curves will be given in chapter seven. The x(y) axis is the number of components of type 1(2) in the system configuration. Ignoring the possible exception of points N_1+1(N_2+1) due to border effects, the feasible set includes all points greater than N_1(N_2) on the x(y) axis. Although these points are part of the bounding surface, they do not have to be considered by the exhaustive search since they are of higher cost than configurations (N_1,0) and (0,N_2) respectively. The portion of the surface the algorithm needs to search is the curve between (N_1,0) and (0,N_2). The shape of the bounding curve is defined by equations (4) and (7). In general increasing the value of the second dimension variable will tend to decrease the value needed for the first dimension variable, but the curve may not be smooth since adding one module to the second dimension may lower the reliability of the system and increase the number of first dimension components required to meet dependability requirements. The experimental results presented in Chapter 7 confirm this, as do the results presented in [78].

Note that lemma 1 proves the minimum cost configuration must lie on the bounding curve inclusive of points (N_1,0) and (0,N_2). Extending the same logic from two to three to J dimensions is straightforward; J represents the number of types of components being considered. Pseudo-code for the exhaustive search algorithm is given in **Figure 6.5**.

As an example, **Figure 6.6** considers a three dimensional case and the arrows show the path taken by the search. The search starts with configuration (N_1,0,0) which consists only of components of type 1. The procedure then finds how many components of type 1 are needed if one component of type 2 is present giving

configuration (x_1,1,0). Components of type 2 are added one at a time until configuration (0,N_2,0) is reached. After (0,N_2,0) the procedure finds out how many components of type 1 are needed if no components of type 2 and one component of type 3 are present, i.e. configuration (x_1,0,1). As before, components of type 2 are added one at a time until configuration (0,x_2,1) is reached. The diagram shows how this procedure continues adding components of type 3 one at a time until the final configuration (0,0,N_3) is reached.

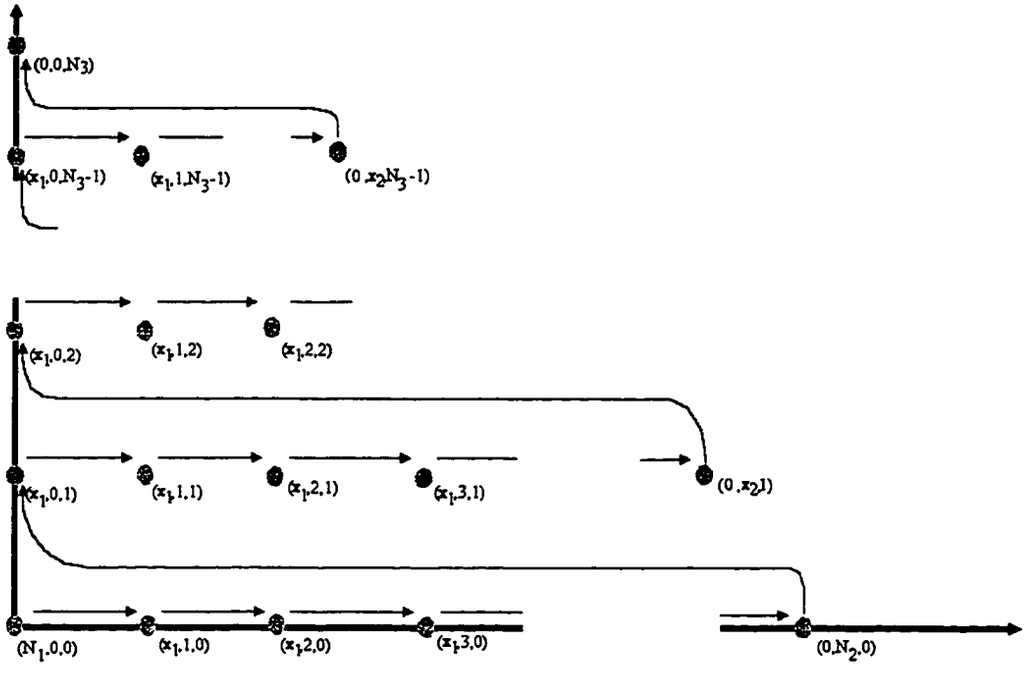

Figure 6.6 Path taken by the exhaustive search in three dimensions

When J is greater than three, the planes defining a 4 dimensional subspace can be traversed in order, similarly the 4 dimensional subspaces defining a 5 dimensional subspace will be traversed, etc. until the entire J dimensional space has been traversed.

Algorithm: ex_sea /* exhaustive search */
Inputs: Number of components J, d_i dependability measure of each, c_i cost of each, and required system dependability D.
Outputs: A vector L of length J with the number of components of each type needed for the minimum cost dependable configuration.
Procedure:
Step 1: Compute N_i, $1 \le i \le J$. Note, N_i is the number of components of type i needed to meet requirement D when no components of another type are used.
Step 2: Sort component types in increasing order of the value $N_i * c_i$.
$x_1 = N_1$ $x_i = 0$, $2 \le i \le J$
Low_Conf $= (x_1, x_2, \ldots, x_J)$ *Current_Conf* $= (x_1, x_2, \ldots, x_J)$
Current_Dim = 1
Step 3: $x_2 = x_2 + 1$
x_1 = number of components of type 1 needed for *Current_Conf* with new x_2 to fulfill the dependability constraint
Current_Conf $= (x_1, x_2, \ldots\ldots)$
if cost(*Current_Conf*) < cost(*Low_Conf*) then *Low_Conf* = *Current_Conf*
Step 4: Iterate step 3 until $x_1 = 0$
Step 5: $i = 2$ /* start with dimension 2 */
while $i <$ *Current_Dim* and $x_1 = 0$ do
begin
$x_i = x_i + 1$ /* one more component of type i */
for($k = 2$ to i) $x_k = 0$ /* no components of type less than i */
x_1 = components of type 1 needed for configuration of x's to fulfill constraint
Current_Conf $= (x_1, x_2, \ldots, x_J)$ /* set new current configuration */
if cost(*Current_Conf*) < cost(*Low_Conf*) then *Low_Conf* = *Current_Conf*
if $x_1 = 0$ then $i = i + 1$
end
if all $x_i = 0$, $1 < i <$ *Current_Dim* then go to step 6
else go to step 3
Step 6: if *Current_Dim* < J then
begin
Current_Dim = *Current_Dim* + 1 $x_{Current\_Dim} = 1$
for($k = 2$ to *Current_Dim*) $x_k = 0$
x_1 = components of type 1 needed for configuration of x's to fulfill constraint
Current_Conf $= (x_1, x_2, \ldots, x_J)$
if cost(*Current_Conf*) < cost(*Low_Conf*) then *Low_conf* = *Current_Conf*
go to step 3
end
else go to step 7
Step 7: Output *Low_Conf* as the minimum cost configuration.

Figure 6.5 Exhaustive search algorithm

6.4 TABU SEARCH PARADIGM FOR COST MINIMIZATION

As stated in section 4.2, tabu search modifies an existing heuristic by prohibiting moves to the nodes in the search space which were visited most recently. Our implementation is based on a greedy heuristic and uses a tabu list that is infinite for our purposes.

Algorithm: tabu_search
Inputs: J, d_i for $1 \le i \le J$, c_i for $1 \le i \le J$, and D.
Outputs: Vector L of length J with the minimum cost dependable configuration.
Procedure:

Step 1: Compute N_i, $1 \le i \le J$. Note, N_i is the number of components of type i needed to meet requirement D when no components of another type are used.
Step 2: Sort component types in increasing order of the value $N_i * c_i$.
$x_1 = N_1 \quad x_i = 0, \ 2 \le i \le J$
Low_Conf $= (x_1, x_2, \ldots, x_J)$ *Current_Conf* $= (x_1, x_2, \ldots, x_J)$
For $I := 1$ to N do /* N is the number of iterations, determined experimentally */
Step 3: Compute $f(x)$ for all $2J$ neighbors of *Current_Conf*. A neighbor is a configuration which is made by adding or subtracting 1 component from any x_i ($2 \le i \le J$). A negative number of components is not allowed. If any neighbor is on the tabu list set the value of $f(x)$ for that neighbor to a prohibitively large value.
Step 4: Set *Current_Conf* to the neighbor with the smallest $f(x)$ value.
Step 5. If f(*Current_Conf*) $< f$(*Low_Conf*) then *Low_Conf* $= x$
Step 6. Append *Current_Conf* to tabu list.
end for loop.
Step 7. return(*Low_Conf*)

Figure 6.7 Pseudo-code for cost minimization using tabu search.

Throughout sections 6.4, 6.5 and 6.6 x_1 refers to the number of components of type one in the configuration. Type 1 is determined at the start of the algorithm. It always refers to the component which can be used by itself to create the lowest cost configuration, we call this the *lowest cost solo configuration*. The value of x_1 is the smallest number of components of type 1 needed with the values of x_2, x_3, ..., x_J to create a configuration which fulfills the dependability constraints. The fitness function

$f(x)$ refers to equation 6 given in section 6.2 with the value of x_1 being used as a dependent variable.

The implementation of the tabu search used here relies on a "greedy" heuristic and starts with the lowest cost solo configuration. The search can move by adding or subtracting one to each x_i. Configurations on the tabu list and configurations with negative numbers of components are disqualified. The algorithm evaluates the cost of each configuration using the fitness function described in this section. The search moves to the configuration with the minimum value for the fitness function $f(x)$. When a configuration is visited it is placed on the tabu list. Should the search return to the neighborhood of a configuration later, the fitness function of a point on the tabu list is set to a very large value. As each configuration is reached, the value of the fitness function is compared to the smallest value up to that point. If the value is smaller, the configuration becomes the best fit found. **Fig. 6.7** presents pseudo-code for this implementation of tabu search.

Since no clear stopping criteria exists for tabu search, this study was done by comparing the results from a given number of iterations of the tabu search with the results obtained by performing the same number of iterations with the genetic algorithms, and the results from simulated annealing which ran until reaching its stopping criteria.

6.5 GENETIC ALGORITHM MINIMIZATION PARADIGM

Several different reproduction strategies for genetic algorithms have been attempted in the literature. Based on the results of our experiments on sensor calibration with genetic algorithms, which will be presented in chapter seven, we apply only an *elitist strategy*, given in [12], to the configuration problem.

The solution space to this optimization problem consists of component configurations. For this reason the chromosomes used by the genetic algorithm consist of a vector which describes a possible system configuration. Position i of the vector, where i is between 1 and J, is an integer ranging from 0 to N_i giving the number of components of type i in the system. The number of elements of type 1 is calculated by the program and set to the smallest number of components of type 1 needed to fulfill the system dependability requirements. The fitness function used in determining the relative quality of chromosomes is the cost of the configuration described by the chromosome. The elitist reproduction strategy used consists of three major steps. First, the best 20% of the gene pool is copied intact into the gene pool for the next generation. Second, 75% of the next generation is determined by randomly mixing elements from two chromosomes chosen at random from the gene pool of the current generation. The remaining 5% of the next generation's gene pool consist of random mutations. This strategy has been found to be stable since the quality of the best answers will be monotonically increasing. The large number of mutations is useful since it provides a steady stream of new input to the algorithm, which guards against the algorithm converging prematurely to a sub-optimal answer.

The algorithm starts by initializing the gene pool using a set of reasonable answers. These answers include all single component type configurations, and many configurations containing a small number of components. The reproduction scheme determines the following generations. The algorithm was performed for 500 iterations and the best solution present in the gene pool at that point was taken to be the configuration proposed by the genetic algorithm. There are no deterministic means of finding the values for many parameters of a genetic algorithm: such as gene pool size, crossover rate, mutation rate, and stopping criteria, these values must

generally be determined experimentally. We used a genetic algorithm as proposed by Holland [40] and modified it to suit our application. The algorithm is summarized as follows:

Algorithm: genetic_search
Inputs: J, d_i for $1 \le i \le J$, c_i for $1 \le i \le J$, and D.
Outputs: Vector L of length J with the minimum cost dependable configuration.
Procedure:
Step 1: Compute N_i, $1 \le i \le J$. Note, N_i is the number of components of type i needed to meet requirement D when no components of another type are used.
Step 2: Sort component types in increasing order of the value $N_i * c_i$.
Step 3: Generate initial gene pool GP of 150 integer vectors of length J. The size, 150, of GP was determined by experimentally. GP contains all solo configurations, and combinations of small numbers of component types.
Step 4: for $k = 1$ to 500 begin /* 500 determined experimentally, as are */
/* percents for elite, crossover and mutation */
for $h = 1$ to 150 begin
x_1 = number of components of type 1 needed for configuration $GP[h]$ to fulfill dependability requirement D.
$cost[h]$ = cost of configuration $GP[h]$ modified so that position 1 is x_1.
end
for $h = 1$ to 30 begin /* keep the elite */
$GP\_next[h]$ = the h'th least expensive configuration in GP.
end
for $h = 31$ to 142 begin /*new combinations from crossover*/
$GP\_next[h]$ = randomly combine 2 configurations chosen at random from GP.
end
for $h = 143$ to 150 begin /*mutants */
$GP\_next[h]$ = randomly create configurations.
end
$GP = GP\_next$
end
Step 5: for $h = 1$ to 150 begin
x_1 = number of components of type 1 needed for configuration $GP[h]$ to fulfill dependability requirement D.
$cost[h]$ = cost of configuration $GP[h]$ modified so that position 1 is x_1.
end
L = configuration in GP with the lowest cost
output L

Figure 6.8 Pseudo-code for cost minimization with genetic algorithms

Genetic algorithms are not sensitive to the presence of local minima since they work on a large number of points in the problem space simultaneously. By comparing many possible solutions they achieve what Holland[40] has termed *implicit parallelism* which increases the speed of their search for an optimal solution. Discussion of the advantages gained by using genetic algorithms instead of exhaustive search for a different optimization problem based on system reliability can be found in [64]. The elitist reproduction strategy used here is useful in that it guarantees the quality of the answers found will increase monotonically.

6.6 SIMULATED ANNEALING MINIMIZATION PARADIGM

The simulated annealing method used in our research is based on the algorithm given in [54,67]. The algorithm has been modified so that the parameters being optimized and the fitness function are appropriate for our application, and a cooling schedule has been found which allows the algorithm to converge to a reasonable solution.

Just as many different reproduction schemes exist for genetic algorithms, several possible *cooling schedules* exist for simulated annealing. A cooling schedule is defined by the initial temperature, the number of iterations performed at each temperature, the number of position modifications allowed at a given temperature and the rate of decrease of the temperature. The answers found by the algorithm are directly dependent on the cooling schedule and no definite rules exist for defining the schedule [54,67]. The cooling schedule used in this application started with a temperature of 1.0 which decreased at a rate of 10%. The total number of iterations at a given temperature was limited to 100 * J, and the maximum number of positions visited at a given temperature was limited to 10 * J. The cooling schedule is

important in that it determines the rate of convergence of the algorithm as well as the quality of the results obtained.

Algorithm: simulated_annealing
Inputs: J, d_i for $1 \le i \le J$, c_i for $1 \le i \le J$, and D.
Outputs: Vector L of length J with the minimum cost dependable configuration.
Procedure:
Step 1: Compute N_i, $1 \le i \le J$. Note, N_i is the number of components of type i needed to meet requirement D when no components of another type are used.
Step 2: Sort component types in increasing order of the value $N_i * c_i$.
Step 3: $CC = (N_1, 0, ...,0)$ /* Initial position in search space */
$\tau = 1.0$ /* Initial temperature */
Step 4: $CC\_mod = 1$ $step4\_iter = 0$
While (CC_mod not = 0) and ($step4\_iter$ < maximum number for step 4) do
begin
$CC\_mod = 0$ $inner\_loop\_iter = 0$
While(CC_mod < maximum number of transitions) and
($inner\_loop\_iter$ < maximum number for inner loop) do
begin
x_1 = number of components of type 1 needed to fulfill the dependability constraint for CC
$CC = CC$ with first position x_1 new_CC = random modification of CC
x_1 = number of components of type 1 needed to fulfill the dependability constraint for new_CC
$new\_CC = new\_CC$ with first position x_1
ΔC = cost(CC) - cost(new_CC)
if($\Delta C < 0$) then
begin
$CC = new\_CC$ $CC\_mod = CC\_mod + 1$
end
else with probability of Boltzmann distribution using ΔC and τ do
begin
$CC = new\_CC$ $CC\_mod = CC\_mod + 1$
end
$inner\_loop\_iter = inner\_loop\_iter + 1$
end
$\tau = 0.9 * \tau$ $step4\_iter = step4\_iter + 1$
end
Step 5: Output CC as the minimal cost configuration .

Figure 6.9 Pseudo-code for cost minimization with simulated annealing

The starting configuration is taken as the lowest cost solo configuration. New configurations are generated randomly from the current configuration. In choosing a new configuration each position in the vector had a 40% chance of being modified. Those positions chosen for modification had a 25% chance of being incremented, a 25% chance of being decremented, and a 50% chance of staying the same.

7 DISCUSSION OF RESULTS

This chapter presents the experimental results of simulations which have been implemented to test the relative merits of the heuristics developed in chapters five and six. These simulations have provided various sample problems for the heuristics to process. Many of our results will be presented graphically. It is worth noting that the results are encouraging and tend to verify the merit of some of the approaches taken. It is also interesting to note that the relative merits of the heuristics used vary greatly between the two problems considered.

7.1 SUCCESS RATES FOR MATCHING IMAGES

All the algorithms were applied to the same sensor readings. Each iteration of the algorithm was performed against a new sensor reading. Results given here are all from the same sample terrain which has been presented in section 2.1. Similar results have been obtained using different terrains and different starting conditions. All sensor one and two readings covered the same region, but new noise values were introduced each time. This resembles the situation which would be found in a dynamic environment. Using sensor readings which covered regions that change over time should make the problem easier to solve since local minima in the search space would tend to be transient.

The results which follow were obtained by using increasing values for the variance of the Gaussian noise. Comparison between the various methods was based on the value of the fitness function for the best reading found.

7.1.1 TABU SEARCH

Figure 7.1 shows the path taken by the tabu search algorithm when searching for an optimal match between the two sensor readings with varying amounts of noise.

The search started at the middle of the sensor 1 reading. Note that the correct answer would have been at the bottom right hand edge of the sensor 2 reading. It is interesting to note, that even in the presence of noise with a variance of 90 which is strong enough to obscure most of the information contained in the picture, the search took approximately the same path as with very little noise.

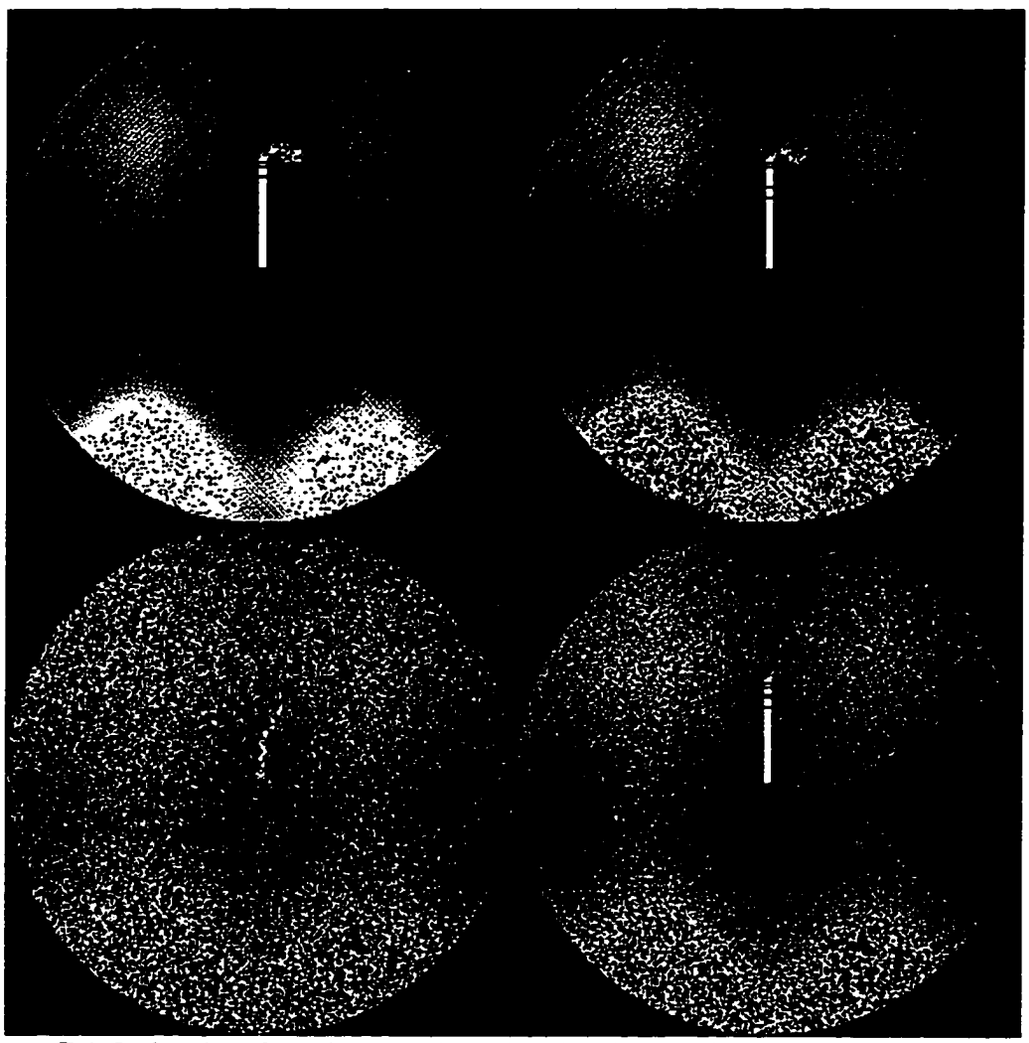

Figure 7.1 Path taken by tabu search registering images in the presence of noise. Noise variance (clockwise from top left) 1, 20, 50 and 90. Number of iterations 500, 325, 125 and 125.

The charts in **figures 7.2 and 7.3**, which show the value of the best parameter set found by the various search methodologies, confirm this observation. The tabu

search algorithm tended to move towards locally optimal values and performed in a stable manner. Unfortunately the answer found was not close to the globally optimal values for the parameters.

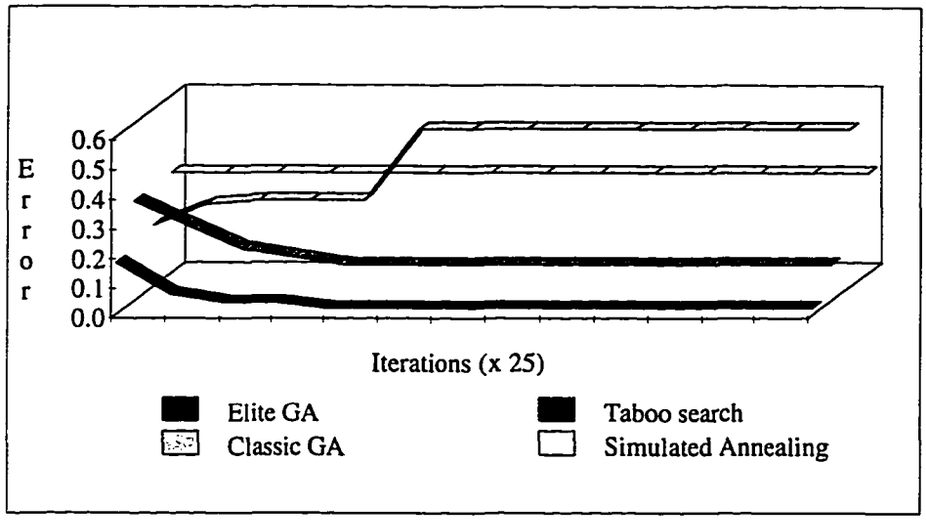

Fig. 7.2 Fitness Function Results Variance 1

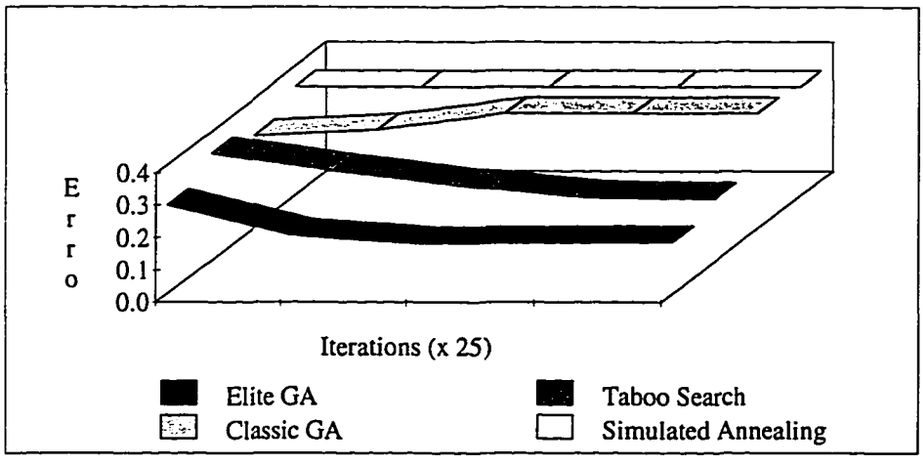

Fig. 7.3 Fitness Function Results Variance 50

7.1.2 GENETIC ALGORITHM-CLASSIC REPRODUCTION SCHEME

Figure 7.4 shows the convergence of values in the gene pool after several iterations of the Genetic Algorithm using the classic reproduction scheme. **Figure 7.5** contains the same scenario after 100 iterations. Note that the parameter set at the

lower right hand corner which is closest to the correct answer is no longer present after 50 iterations.

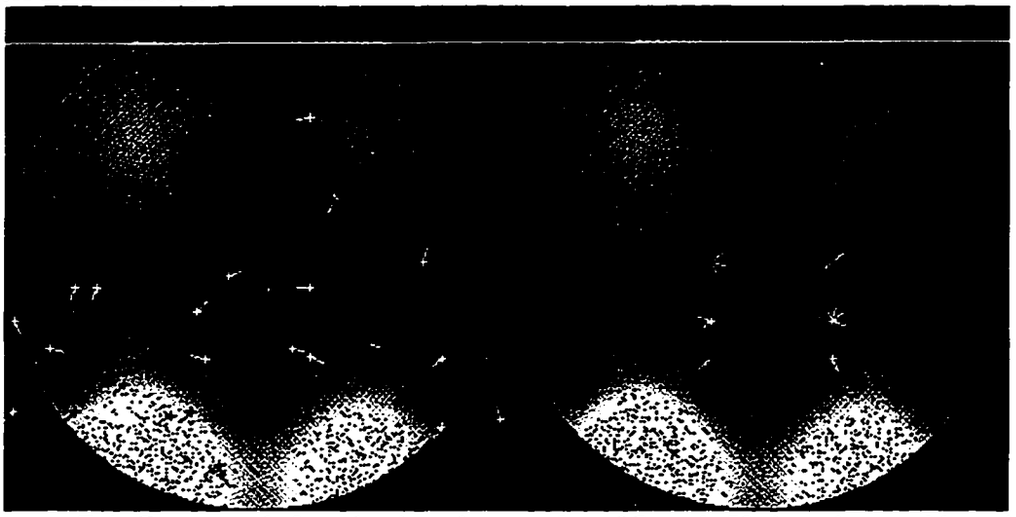

Figure 7.4 Gene pool for classic scheme after 25 and 50 iterations with noise variance of 1.

The gene pool values found after a number of iterations with variance values of 50 and 90, show that even in the presence of noise the values contained in the gene pool tend to converge. Unfortunately the convergence is not towards the globally optimal values.

Figures 7.2 and 7.3 show the relationship of the fitness function value of the best parameter set to the number of iterations used by the algorithm. Oddly enough, the value tends to increase instead of decrease as would be expected. The algorithm tended to remove the parameter sets with extreme values for the fitness function. This converged towards values which were stable but far from the global optimum.

7.1.3 GENETIC ALGORITHM-ELITIST REPRODUCTION SCHEME

Figure 7.6 shows the gene pool found by the elite reproduction scheme after 100 iterations for several different levels of noise. Notice that most of these samples contain values very close to the globally optimal value. The genetic algorithm with

the elite reproduction scheme tended to converge towards the globally optimal value even in the presence of moderate noise.

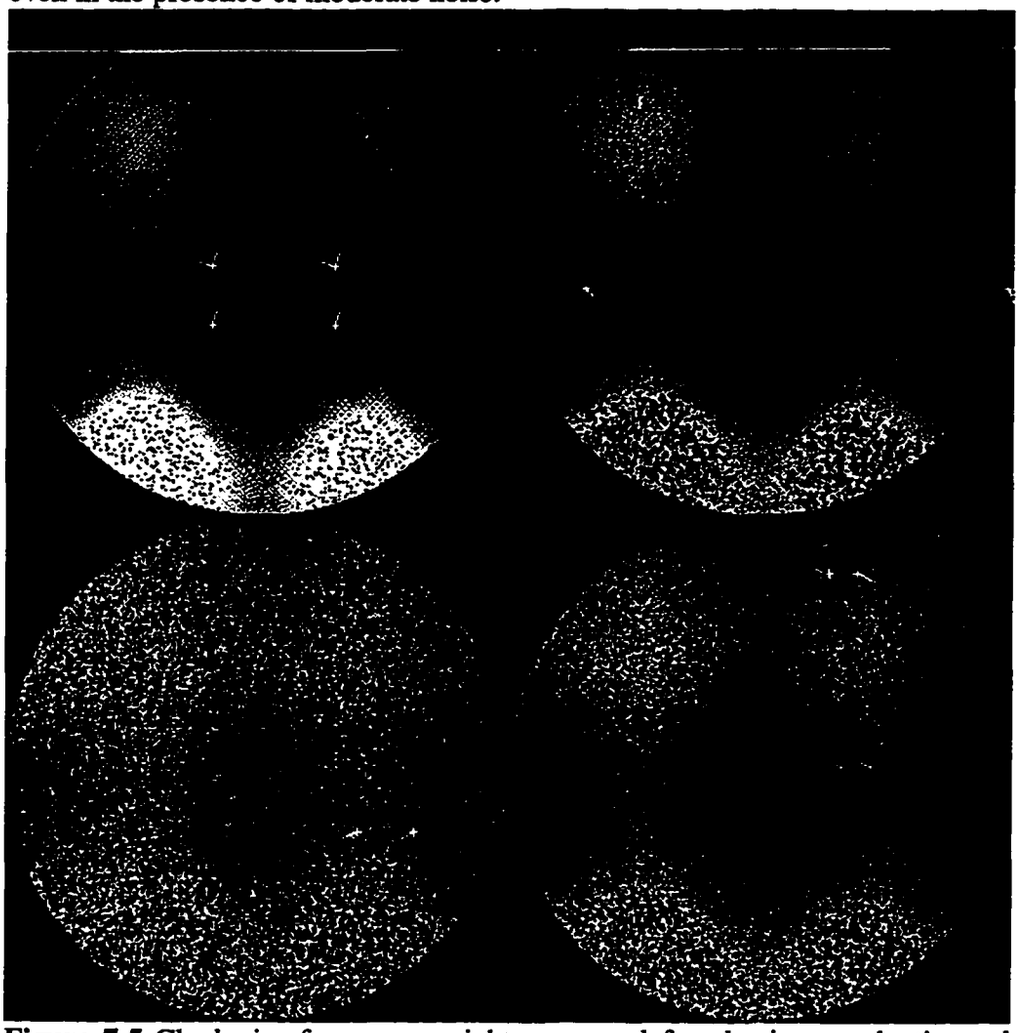

Figure 7.5 Clockwise from upper right gene pool for classic reproduction scheme with noise variance of 1 after 100 iterations, with noise variance of 20 after 100 iterations, with noise variance of 50 after 100 iterations, and with noise variance of 90 after 75 iterations.

The results after 100 iterations with the variance of the error set to 90 show that the amount of information in the image has been severely compromised, and that no parameter sets are found near the globally optimal answer.

The graphs in **figures 7.2 and 7.3** verify that this algorithm tended to converge rapidly towards very good solutions to the problem. In fact, the shapes of the graphs are surprisingly similar considering the differences in the images they are treating.

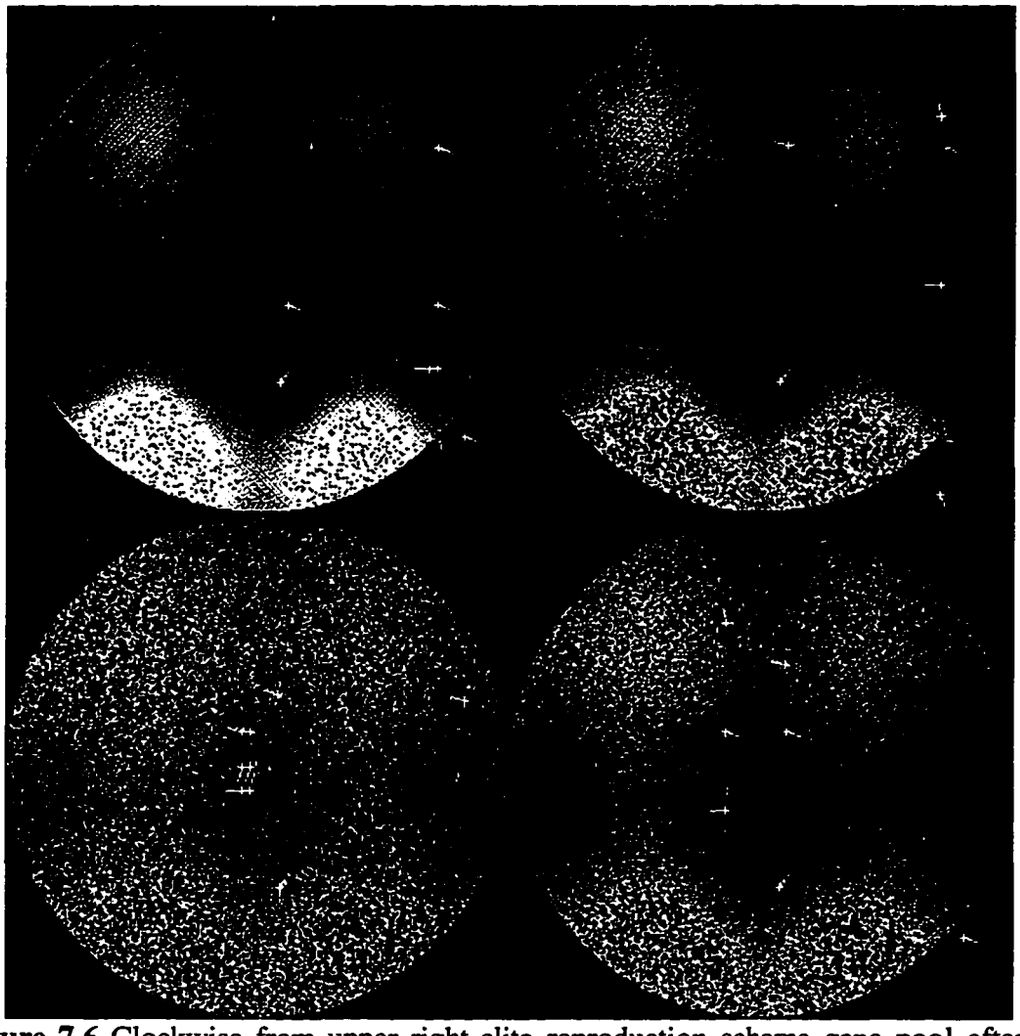

Figure 7.6 Clockwise from upper right elite reproduction scheme gene pool after 100 iterations with noise variance of 1, 20, 50 and 90.

In spite of the fact that the algorithm converged towards good solutions even in the presence of overwhelming amounts of noise, there was a limit to its ability to find the globally optimal solution. The values found by the algorithm are given in **table 7.1**. Note that the globally optimal parameter values are: x displacement = 91, y displacement = 91, rotation = 2.74889 radians.

Table 7.1 Values found by elitist genetic algorithm.

Variance	X-displacement	Y-Displacement	Rotation
1	89	91	2.74744
10	92	92	0
20	91	91	2.74744
30	89	89	2.74744
50	86	-18	2.79768
70	-48	6	6.02138
90	0	5	1.23297

These values show that the algorithm does not always find the globally optimal values, but it tends to do a good job even in the presence of relatively large amounts of noise. However, once the noise reaches a point where it obscures too much of the information present in the image, it no longer locates the optimal values.

It is also worth noting that the quality of the answers found is not strictly a function of the noise variance. This is a consequence of the stochastic nature of genetic algorithms. Since the original gene pool and mutations are chosen at random, the quality of the answers found by the algorithm is also partially random in nature. Genetic algorithms are based on comparing the results of a number of random choices. The answers found by the elitist reproduction scheme show this strategy can be effective in finding reasonable answers to combinatorial optimization problems. On the other hand, the results found are non-deterministic in nature and the quality of the algorithm's results depend partially on the random nature of this method.

7.1.4 SIMULATED ANNEALING

Simulated annealing is the only method used with a clear termination criteria. This makes a direct comparison with tabu search and genetic algorithms difficult. The final answers found by simulated annealing were roughly comparable to the answers found by tabu search. The number of iterations used to find these answers were much larger than the number of iterations taken by tabu search.

Figure 7.7 displays the paths taken by simulated annealing when searching for the correct registration in the presence of varying levels of noise. Since the correct answer is in the lower right hand corner of the sensor reading, it is obvious that simulated annealing did not converge to the global optimum. The simulated annealing approach searched within any of a number of local optima. It did not remain trapped in the region defined by the first local optima, but did not find the global optimum either.

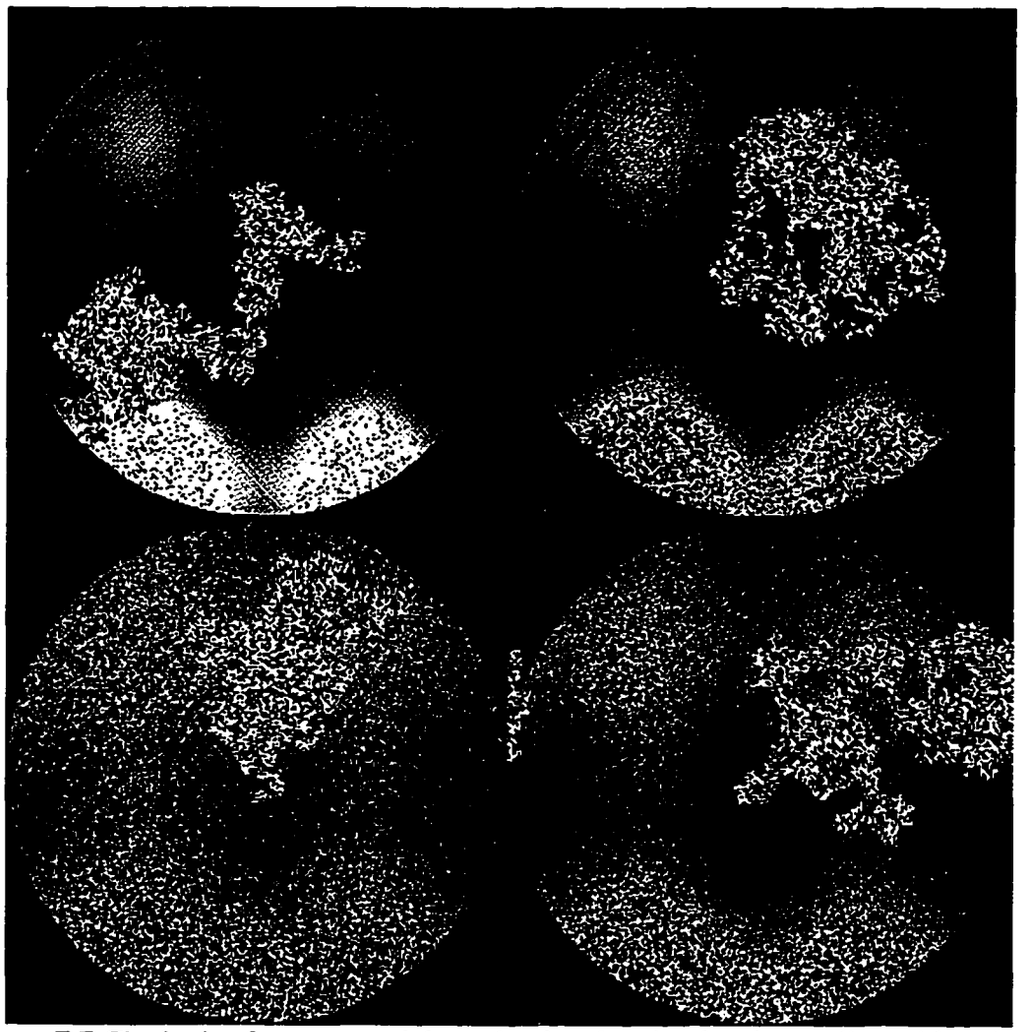

Figure 7.7 Clockwise from upper right path taken by simulated annealing with noise variance of 1, 20, 50 and 90.

In **figures 7.2 and 7.3** the best answer found by simulated annealing is represented by a straight line. This is not an entirely fair representation. The first several iterations of the algorithm are when the temperature parameter is at its highest point. In which case the algorithm amounts to a random walk. This is intentional, convergence is delayed until a later point in the algorithm when the system starts to cool.

7.2 MINIMAL COST SYSTEM CONFIGURATIONS

This section discusses the results gathered from experiments using implementations of the methods discussed in chapter six. The first method applied was exhaustive search. Exhaustive search is the best method for solving small scale problems since it is the only method guaranteed to find the optimal answer to this problem. Exhaustive search has then been used as a baseline for comparison with the other methods: tabu search, elitist genetic algorithm and simulated annealing.

7.2.1 EXHAUSTIVE SEARCH RESULTS

The exhaustive search algorithm has been implemented and tested on several sample data sets of fewer than five dimensions. This section discusses a few representative examples and observations. We have arbitrarily selected an availability constraint of 0.9999999995. Similar results are also obtained with different availability constraints such as 0.995. Failure and repair rates are given in terms of FITs.

Table 7.2 Parameters and costs for example 1.

Parameters	Component 1	Component 2	Component 3
Failure rate	0.0001	0.01	0.005
Repair rate	0.5	0.1	0.3
Unit cost	36.57	5.25	12.35
Solo Config.	5	35	15
Solo Cost	182.85	183.75	185.25
Min. Config.	0	2	13
Min. Cost	171.05		

Example 1 consists of three components (refer to table 7.2). The solo configuration describes the number of components of one single type needed to fulfill the availability constraint. Solo cost refers to the cost of that configuration. All components are comparable in that the solo configurations do not vary greatly in cost. It is also interesting to note that the minimum cost solution does not contain any components of type 1 although its solo configuration is the least expensive. The minimum cost configuration represents a savings of 6.5% compared to the lowest cost solo configuration.

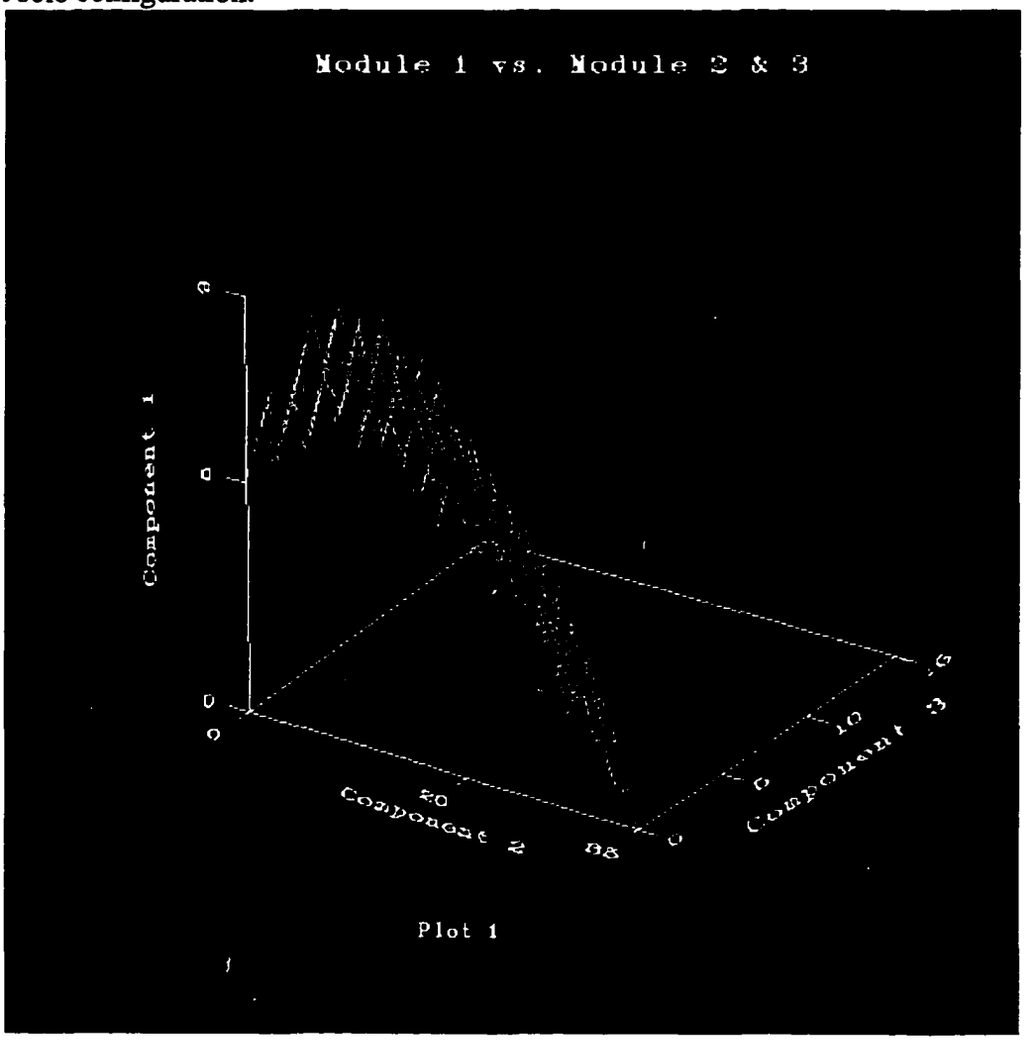

Figure 7.8 Components of type 1 needed versus components of types 2 and 3.

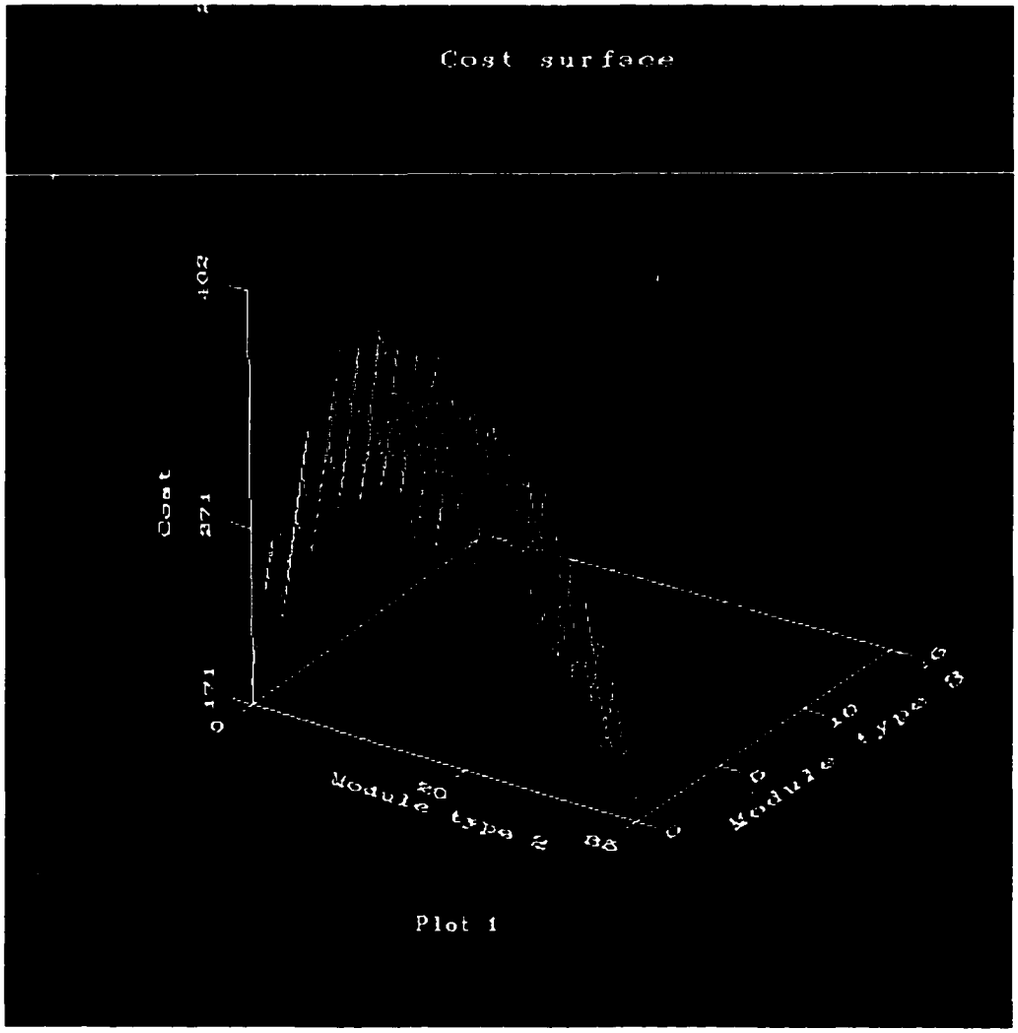

Figure 7.9 3-dimensional surface searched by exhaustive search

Figures 7.8 and 7.9 represent the surface on which the minimal solution is located. **Figure 7.8** depicts the number of components of type 1 needed for various combinations of components of type 2 and type 3. **Figure 7.9** shows the cost of each configuration, again in terms of the number of components of type 2 and type 3. Note that the relative shape of the surface is basically the same. Macroscopically the surface is represented by a series of jagged lines. The jaggedness of the lines are caused by two factors: the problem space is discrete, and tolerating failures of less than 50% of the components means it is necessary to add two components to

increase system reliability. Adding one component increases the number of items which may fail without increasing the number of failures which may be tolerated, this effect has also been noted in [78]. This jaggedness means that the search space has many local minima, and search methods which depend only on information in the neighborhood of a point will be unsuited to solving this problem.

Table 7.3 Parameters and costs for examples 2 and 3.

Parameters	Component 1	Component 2	Component 3	Component 4
Failure rate	0.003	0.002	0.005	0.0001
Repair rate	0.45	0.995	0.3	0.05
Unit cost (ex. 1)	11.00	11.86	10.25	10.00
Unit cost (ex. 2)	11.06	11.86	10.25	10.00
Solo Config.	11	9	15	15
Solo Cost (ex. 1)	121.00	106.74	153.75	150.00
Solo Cost (ex. 2)	121.66	106.74	153.75	150.00
Min. Config. (ex. 1)	4	4	0	1
Min. Config. (ex. 2)	0	6	2	1
Min. Cost(ex. 1)	101.44			
Min. Cost(ex. 2)	101.65			

Table 7.3 illustrates the parameters and results for two examples. Examples 2 and 3 are identical except that the unit price of component 1 has been increased from $11.00 to $11.06. Note that this minor change has radically altered the minimum cost configuration. This further illustrates the jagged nature of the surface.

7.2.2 RESULTS FOR HEURISTIC APPROACHES

Examples 4 and 5 are test cases of multiple dimensions which have been used to test the exhaustive search, simulated annealing, and genetic algorithms approaches to this problem. They consist of eight and eleven dimensions respectively. The number of dimensions was kept relatively small to allow the use of the exhaustive search algorithm for verifying the global minimum. For comparison purposes the same availability constraints were used as in section 7.2.1.

Table 7.4 Parameters and results for example 4

	Comp. 1	Comp. 2	Comp. 3	Comp. 4	Comp. 5	Comp. 6	Comp. 7	Comp. 8
Failure rate	0.003	0.002	0.005	0.001	0.00775	0.00195	0.0025	0.01
Repair rate	0.45	0.995	0.3	0.05	0.45	0.997	0.3	0.1
Unit Cost	11.06	11.86	10.86	10.00	9.50	13.70	14.20	4.10
Solo Config.	11	9	15	15	15	9	11	35
Solo Cost	121.66	106.74	153.75	150.00	142.5	123.30	156.20	143.50
Min. Config.	0	7	0	0	1	0	0	1
Min. Cost	96.62							
GA Config.	0	7	1	0	0	0	0	1
GA Cost	97.37							
SA Config.	0	7	0	0	1	0	0	1
SA Cost	96.62							
Tabu Config	0	7	0	0	1	0	0	1
Tabu Cost	96.62							

The genetic algorithms approach found the global minimum only for example 5, however its solution for example 4 is very close to the global minimum. This algorithm was more computationally intensive than the simulated annealing algorithm, and there is no clear criteria for stopping the genetic algorithm. This approach does have two positive characteristics: it is less computationally expensive than the exhaustive search method, and the elite reproduction scheme guarantees that the cost of the best answers will be monotonically decreasing.

For both cases simulated annealing succeeded in finding the global minimum. In the tests performed here, simulated annealing was also less computationally intensive than genetic algorithms and exhaustive search, providing its answer within minutes instead of hours. While these results are positive, it should be noted that this algorithm is not guaranteed to provide the globally optimal answer.

Tabu search has also succeeded in finding the global minimum in all tests which have been performed. These tests show that this implementation has about the same computational complexity as simulated annealing. For this application both approaches appear to be roughly comparable, except that simulated annealing has the advantage of having a clear stopping criteria which tabu search lacks.

Table 7.5 Parameters and costs for example 5

	C. 1	C. 2	C. 3	C. 4	C. 5	C. 6	C. 7	C. 8	C. 9	C. 10	C. 11
Failure rate	0.001	0.01	0.001	0.05	5E-05	0.002	0.003	0.018	1E-04	0.009	0.018
Repair rate	0.3	0.3	0.9	0.995	0.995	0.997	0.15	0.585	0.07	0.3	0.622
Unit Cost	20.00	10.00	20.00	5.00	25.00	15.00	7.00	8.00	20.70	6.80	7.00
Solo Config	9	19	7	23	5	9	15	17	7	17	17
Solo Cost	180.00	190.00	140.00	115.00	125.00	135.00	105.00	136.00	144.90	115.60	119.00
Min. Config	0	0	0	2	0	0	12	0	0	1	0
Min. Cost	100.8										
GA Config.	0	0	0	2	0	0	12	0	0	1	0
GA Cost	100.80										
SA Config.	0	0	0	2	0	0	12	0	0	1	0
SA Cost	100.80										
Tabu Config	0	0	0	2	0	0	12	0	0	1	0
Tabu Cost	100.80										

8 CONCLUSION AND OPEN PROBLEMS

This chapter presents the conclusions which have been drawn from the preceding research. Both problems which have been approached are practical problems with immediate applications in sensor fusion, and are both computationally challenging. This dissertation has evaluated the applicability of general optimization methods to resolving these problems. The results are very promising and indicate that elitist genetic algorithms are very useful for solving both problems. Simulated annealing and tabu search are useful for cost minimization but their applicability to image registration is limited.

8.1 PARADIGM FOR REGISTRATION OF NOISY IMAGES

The problem posed was to find ways of automatically calibrating two noisy sensors using optimization methods. This has been shown to be possible as long as the noise is held within certain limits.

Several attempts have been made to solve this type of problem, this is the first attempt which matches noisy gray scale images. Several algorithms exist for roughly equivalent problems, the other algorithms are more specific in that they assume that the image consists of a small number of distinct features which can be matched [25,3], that specific shapes are to be matched [27], or that the relative displacement is small [10]. This dissertation assumes none of those restrictions, the tests were run using gray scale images with many possible approximate matches and a large displacement. The approach can be used on any arbitrary gray scale image.

It is necessary to register the readings from sensor two with the readings from sensor one. This can be done by finding three parameters: the offset of the two

sensors in the x direction, the offset in the y direction, and the angle of rotation between them.

Four methods have been attempted: tabu search, simulated annealing, genetic algorithms using a classical reproduction scheme, and genetic algorithms using an elitist reproduction scheme. Of the four, the genetic algorithm using the elitist reproduction scheme has tended to produce the best results. It often found even close to globally optimal results.

All approaches are able to deal with data containing reasonable amounts of noise. The Genetic Algorithm using the elitist reproduction scheme was able to continue to find answers with two out of three parameters very close to the optimum even when the noise variance was set to 50. With possible values ranging only from 0 to 255 noise at that level severely distorts the actual image. It is therefore safe to say that the Genetic Algorithm paradigm when implemented with an elitist reproduction scheme is very tolerant to noise.

Tabu search is more sensitive to local minima than the genetic algorithms, since its searching mechanism only considers points in the search space which are in its immediate vicinity. It quickly converges to the local minima. It may take a prohibitively long time, however, for the search to climb its way out of the local minima if the attraction basin for the local minima is large. Simulated annealing has the same drawbacks as tabu search. While the results here show that simulated annealing does not necessarily remain in its original local minima, it may fall into the basin of attraction of another local minima. Genetic Algorithms have the advantage that they can simultaneously process many different possible answers and are not constrained to looking for local minima.

The elitist reproduction scheme preserves the members of the gene pool which have the best performance. This contrasts with the classical scheme where the quality of the best answers may degrade as the algorithm progresses. The elitist scheme keeps the best answers from preceding generations forcing the quality of the best answer to be monotone increasing, as shown in **figures 7.1 and 7.2.** This is why the elitist scheme is most suited to this application where a single best answer is sought. **Figure 7.6** contains examples of elitist gene pools with a few near optimal answers and several far from optimal answers.

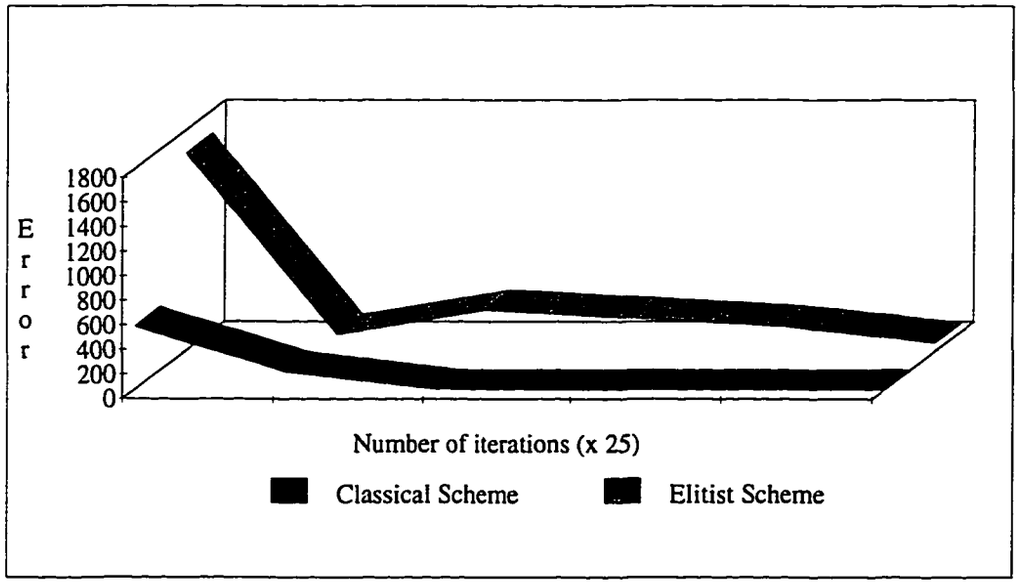

Fig. 8.1 Average answer fitness of classical and elitist genetic algo. v. 20

The classical reproduction scheme produces a new generation by mixing genes between members of the current gene pool. Higher quality members of the gene pool are more likely to be chosen for reproduction. Genetic algorithms that use the classical scheme produce a new generation of higher average quality than the previous generation. The average quality of answers using the elitist reproduction scheme described in this dissertation may decrease from one generation to the next. **Figures 8.1 and 8.2** illustrate this point by comparing the average error of gene

pools derived by this application. As the process continues, the classical gene pool will contain a diverse number of answers, and the average quality of answers in the gene pool tends to increase. **Figure 7.5** shows some gene pools containing a number of good possible answers which are unfortunately far from the optimal answer. When using the classical scheme a near optimal answer is unlikely to be carried over unchanged into the next generation. For this reason, the quality of the best answer found by the classical scheme may decrease from one generation to the next, as shown in **figures 7.2 and 7.3**. For these reasons the elitist reproduction scheme provides better results than the classical reproduction scheme for this application.

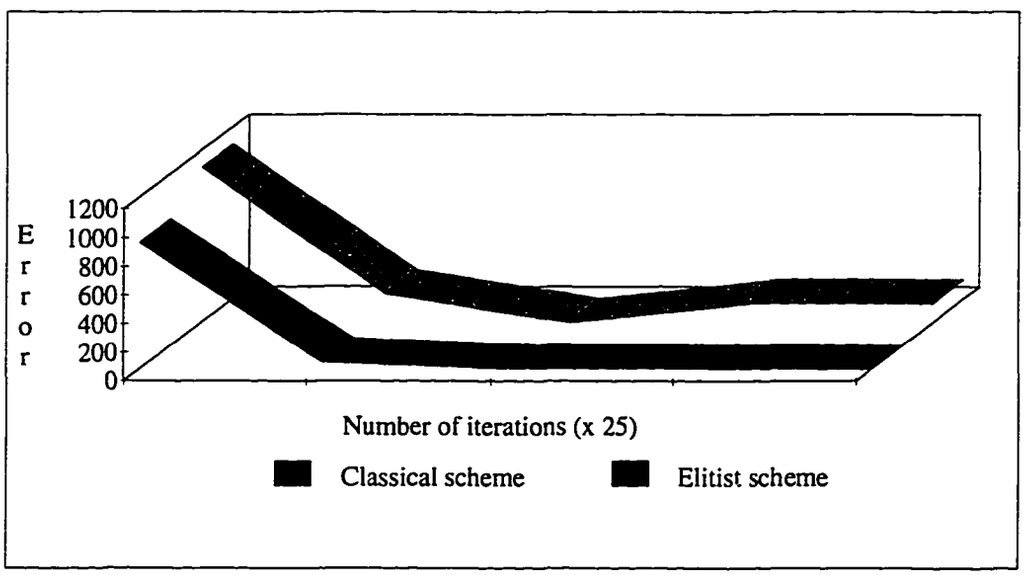

Fig. 8.2 Average answer fitness of claclassical and elitist genetic algo. v.70

Tests have been made using terrain models other than the equation used for **figure 2.1**. The results presented in this dissertation are typical for all terrain models tested. The terrain models tested were all qualitatively similar in that they contained both periodic and non-periodic elements. Removing the periodic components of the terrain modifies the original problem, changing the nature of the problem space being

searched. Gruences for a terrain model with little or no periodicity are much easier to find and deterministic search algorithms could then be used to solve the problem posed in a straightforward manner.

The problem posed has many possible applications. It is also worth repeating that the problem would be easier to solve in a dynamic environment. If the image varies as well as the noise, local minima would tend to be transient. In which case, this approach would be especially relevant to *active vision* research.

8.2 A NEW SCHEME FOR COST MINIMIZATION

The methods described in this dissertation are useful for finding system configurations for high reliability systems made up of individual components and relying on fault masking. They use the formulas presented to verify that the system will fulfill dependability constraints as long as the component failures are statistically independent. Examples were given to illustrate how this methodology can be put into practice and result in cost savings.

For small scale problems the exhaustive search method is preferable, since it is guaranteed to find the lowest cost combination of systems components. This dissertation has shown that genetic algorithms, tabu search and simulated annealing are viable approaches to finding optimal, or near optimal configurations for large scale problems. In these tests, simulated annealing and tabu search performed remarkably well and appear to be the methodologies best suited to solving the problem. Simulated annealing has the advantage of having a well defined halting criteria. A reasonable approach would be to use genetic algorithms as a secondary approach to verify the results found by the simulated annealing algorithm.

It should be noted, however, that genetic algorithms, tabu search and simulated annealing all have drawbacks. Genetic algorithms and simulated annealing are

relatively insensitive to the presence of local minima in the search space, but this insensitivity is partially obtained by the creative application of nondeterminism. This nondeterminism also means that the quality of the answers found by the algorithms will vary from case to case. It is also impossible to know how long the algorithms will need to find the global minima, or if they will ever find the global minima.

Genetic algorithms are sensitive to the reproduction strategy chosen, including mutation rates and how elements are chosen for crossover. Simulated annealing is sensitive to the cooling schedule which includes the initial temperature and the rate of decrease of temperature. The quality of the answers found and the amount of time needed to find reasonable answers are directly dependent on the reproduction strategy of a genetic algorithm, and the cooling schedule of a simulated annealing approach. Both the reproduction strategy and cooling schedule must be found through a process of trial and error. For neither one is there any guarantee that one particular strategy or schedule will be appropriate for all cases which may be encountered. In spite of this, our research has shown that both can be applied effectively as heuristics for solving this problem.

Tabu search is deterministic but relies on a number of factors, such as the length of the tabu list. Research is being done on finding ways to optimize tabu search, such as finding the optimal list length, and finding methods to avoid cycling in the search space [11,32]. A particular drawback is the lack of a concrete halting criteria which indicates when and under which conditions the search terminates [43].

8.3 COMMENTS ON HEURISTICS PARADIGM

The results of this dissertation show there is no single strategy which is optimal for all problems. Simulated annealing and tabu search work well for the sensor configuration cost minimization problem which has a search space covered with a

large number of very shallow local minima. Another reason for their success is that an initial position is available which places these algorithms in a position sufficiently close to the global minima to find the global minima in many cases.

On the other hand both strategies fail to find optimal registrations for the images used in this dissertation. While the strategies are designed to avoid local minima, they are still sensitive to local minima which have large basins of attraction. This makes them also sensitive to the initial starting position of the search.

Genetic algorithms have been shown to be a search strategy which is not constrained by using only local information. For our problems this has made them more resistant to the presence of local minima. It also means that the answers provided may be close to the global minimum but may not be locally minimal. The classic genetic algorithm effectively produces a set of reasonable answers by minimizing the average cost of members of the gene pool. This appears to occur at the cost of destroying the truly exceptional results.

Elitist genetic algorithms are effective for finding near optimal answers even in the presence of a large number of local minima, even with large basins of attraction. They seek to find a few exceptional results and not to benefit the entire gene pool. A further drawback to genetic algorithms is that they are computationally expensive. Each iteration of the algorithm requires the evaluation of the fitness function for all members of the gene pool.

8.4 OPEN PROBLEMS

The registration approach explored here could be refined further. In addition to the gruences explored here, a scaling factor could be added. This should be possible without any additional effort and would increase the set of possible applications to include all affine mappings. Rubber sheet transformations are also possible, but care

should be taken since extreme modifications of the images could result in false registrations.

1 - In order to reduce the amount of computation required a number of researchers have proposed multi-resolution methods. A similar approach could be applied here by not matching every pixel in the images but only every *n*th pixel. This would allow rough matches to be made while reducing the computational load. Like the multi-resolution methods this approach discards available information and the danger exists that sub-optimal matchings will be made.

2 - Another extension that deserves attention is to combine the elitist genetic algorithm with a greedy heuristic. Once the genetic algorithm has converged to a small number of matchings, a greedy heuristic could be used to find the local minima in the neighborhood of the most promising solution found by the genetic algorithm. This approach would have the advantages of the genetic algorithm approach and be guaranteed to find a matching which is at least locally optimal.

3 - For sensor applications, further research could be based on loosening the assumptions used in defining the algorithm. One assumption taken is that the sensors all share the same coverage of the area being sensed. This will most often not be the case. It would also be of interest to find configurations which cover a given region but where individual sensors can only cover a small portion of the region.

4 - It has been assumed that component failures are statistically independent. Sensors of the same type will be sensitive to the same type of interference, where sensors of another type may not be sensitive to that particular noise. This would tend to favor building systems using many different types of components. Another factor to be considered is that real time systems have specific time constraints. For this reason, it would often be preferable to have systems using a smaller number of components to

reduce the amount of bandwidth needed for reading and comparing the inputs from the individual components. For many applications these considerations are desirable additions to the framework developed in this dissertation.

REFERENCES

[1] A. Alj, and R. Faure *Guide de la Recherche Operationelle vol. 1-2*, Masson, Paris, 1990.

[2] J. Andrus and C. Campbell "Digital Image Registration Using Boundary Maps", *IEEE Transactions on Computers*, vol. 19, pp. 935-940, 1975.

[3] N. Ayache, and O. Faugeras, "Building, Registrating, and Fusing Noisy Visual Maps", *International Journal of Robotics Research*, vol. 7, No. 6(Dec), pp 45-65, 1988.

[4]H. S. Baird *Model-based Image Matching Using Location*, MIT Press, 1985.

[5] M. Balakrishnan and C. Raghavendra, "On Reliability Modeling of Closed Fault-Tolerant Computer Systems", *IEEE Transactions on Computers*, 39, 4(Apr), pp. 571-575, 1990.

[6] M. Balakrishnan and C. Raghavendra, "An Analysis of a Reliability Model for Repairable Fault Tolerant Systems", *IEEE Transactions on Computers*, 42, 3(Mar), pp. 327-339, 1993.

[7] M. Barborak, M. Malek and A. Dahbura, "The Consensus Problem in Fault Tolerant Computing" *ACM Computing Surveys* 25, 2(June), pp. 171-220, 1993.

[8] D. Barnea and H. Silverman "A Class of Algorithms for Fast Digital Image Registration", *IEEE Transactions on Computers*, vol. C-21 no. 2, pp. 179-186, 1972.

[9] R. Barlow and F. Proschan, *Mathematical Theory of Reliability*, Wiley, New York, 1965.

[10] H. G. Barrow, J. M. Tennenbaum, R. C. Bolles, and H. C. Wolf "Parametric Correspondence and Chamfer Matching: Two New Techniques for Image Matching" in *Proc. Int. Joint Conf. Artificial Intelligence*, pp. 659-663, 1977.

[11] R. Battiti and G. Tecchioli "The Reactive Tabu Search" in *ORSA Journal on Computing* 6(2), pp. 126-140, Spring 1994.

[12] J. C. Bean "Genetic Algorithms and Random Keys for Sequencing and Optimization" in *ORSA Journal on Computing* 6(2), pp. 154-160, Spring 1994.

[13] J. Birrien, *Histoire de l'Informatique* in the series Que Sais Je, Presse Universitaires de France, Paris, 1990.

[14]R. R. Brooks, N. S. V. Rao and S. S. Iyengar, "Resolution of Contradictory Sensor Data", *Journal of Intelligent Automation and Soft Computing,* accepted for publication, 1995.

[15] R. R. Brooks and S. S. Iyengar, "Optimal Matching Algorithm for Multi-Dimensional Sensor Readings", *Sensor Fusion and Networked Robotics*, eds. P. Schenker and G. McKee, SPIE, Bellingham, Wa., vol. 2589, Proceedings of SPIE International Symposium on Intelligent Systems and Advanced Manufacturing, Philadelphia, Pa. October 1995, pp. 91-99, 1995.

[16] R. R. Brooks and S. S. Iyengar, "Algorithm for Resolving Inter-Dimensional Inconsistencies in Redundant Sensor Arrays", in *Parallel and Distributed Signal and Image Integration Problems*, Proceedings of Indo-US Workshop December 1993 Pune, India, World Scientific, Singapore, pp. 254-262, 1995.

[17] R. R. Brooks and S. S. Iyengar, "Methods of Approximate Agreement for Multisensor Fusion", *Signal Processing, Sensor Fusion and Target Recognition IV*, eds. I. Kadar and V. Libby, SPIE, Bellingham, Wa., vol. 2484, Proceedings of SPIE International Symposium on Aerospace /Defense Sensing and Dual-use Photonics, Orlando, Fla. April 1995, pp. 37-44, 1995

[18] R. R. Brooks and S. S. Iyengar, "Approximate Agreement for Distributed Computing by Masking Data Conflicts", *IEEE Computer*, pp. 53-60, June 1996.

[19] R. R. Brooks and S. S. Iyengar, "Self-Calibration of A Noisy Multiple Sensor System with Genetic Algorithms", *Self-Calibrated Intelligent Optical Sensors and Systems*, SPIE, Bellingham, Wa., Proceedings of SPIE International Symposium on Intelligent Systems and Advanced Manufacturing, Philadelphia, Pa. October 1995.

[20]R. R. Brooks, S. S. Iyengar, and J. Chen, "Automatic Correlation and Calibration of Noisy Sensor Readings using Elite Genetic Algorithms", *Artificial Intelligence*, scheduled for publication 1996.

[21] R. R. Brooks, S. S. Iyengar, and Suresh Rai, "Minimizing cost of redundant sensor-systems with non-monotone and monotone search algorithms" *Proceedings 1997 IEEE Annual Reliability and Maintainability Symposium*, Philadelphia, Pa. January 1997.

[22] P. Chew and K. Marzullo, "Masking failures of multidimensional sensors." in *Proc. of the Tenth Symposium on Reliable Distributed Systems*, pp. 32-41, 1991.

[23] K. L. Chung, *Elementary Probability Theory with Stochastic Processes*, Springer Verlag, Heidelberg, 1974.

[24] P. Coiffet, and M. Chirouze *An Introduction to Robot Technology*, Mc-Graw Hill, London, p. 22, 1982.

[25] I. J. Cox and J. B. Kruskal "On the Congruence of Noisy Images to Line Segment Models" in *2nd Conference on Computer Vision*, IEEE, pp. 252-258, 1988.

[26] E. Davalo and P. Daïm, *Des Reseaux de Neurones*, Editions Eyrolles, Paris, 1989.

[27] L. S. Davis "Shape Matching Using Relaxation Techniques", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 1, No. 1, pp. 60-72, 1979.

[28] H. F. Durrant-Whyte, "Sensor Models and Multisensor Integration", *International Journal of Robotics Research*, vol. 7, No. 6(Dec), pp 97-113, 1988.

[29] L. Feng, J. Borenstein, and H. R. Everett, *Sensors and Methods for Autonomous Mobile Robot Positioning*, ed. J. Borenstein, University of Michigan Technical Report UM-MEAM-94-21, 1994.

[30] H.-J. Fletchner, *Grundbegriffe der Kybernetik*, dtv Wissenschaft, 1984.

[31] F. Glover, "Tabu Search - Part 1", *ORSA Journal on Computing* 1, pp. 190-206, 1989.

[32] F. Glover, "Tabu Thresholding: Improved Search by Nonmonotonic Techniques", *ORSA Journal on Computing*, vol. 7, No. 4(Fall), pp. 426-442, 1995.

[33] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison-Wesley, Reading, p. 302, 1993.

[34] A. Goshtasby, "Piecewise linear mapping functions for image registration", *Pattern Recognition*, Vol. 19 No. 6, pp. 459-466, 1986.

[35] A. Goshtasby, "Piecewise cubic mapping functions for image registration", *Pattern Recognition*, Vol. 20 No. 5, pp. 525-535, 1987.

[36] A. Goshtasby, "Image Registration by Local Approximation Methods", *Image and Vision Computing*, Vol. 6 No. 4(Nov), pp. 255-261, 1988.

[37] S. Gulati, J. Barhen and S. S. Iyengar "Neurocomputing Formalisms for Computational Learning and Machine Intelligence" in Advances in Computers Academic Press, 1991.

[38] F. S. Hill, Jr., *Computer Graphics*, Prentice Hall, Englewood Cliffs, 1990.

[39] G. E. Hinton "Connectionist Learning Procedures" in *Machine Learning* ed. J. Carbonnell, pp. 185-234, MIT/Elsevier, 1992.

[40] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, 1975.

[41] B. R. Horn and B. L. Bachman "Using Synthetic Images with Surface Models" *Communications of the ACM*, vol. 21, pp. 914-924, 1977]

[42] Hovanessian, S.A. *Introduction to Sensor Systems*, Artech House, Norwood MA, 1988.

[43] T. C. Hu, A. B. Kahng and C. A. Tsao, "Old Bachelor Acceptance: A New Class of Non-Monotone Threshold Accepting Methods", *ORSA Journal on Computing*, vol. 7, No. 4(Fall), pp. 417-425, 1995.

[44] S. S. Iyengar, L. Prasad, and H. Min, *Advances in Distributed Sensor Integration: Applications and Theory*, Prentice Hall, 1995.

[45] S. S. Iyengar, and D. Thomas "A Distributed Sensor Network structure with Fault-Tolerant Facilities" in *Intelligent Control and Adaptive Systems* proceedings of SPIE, Nov. 1989.

[46] S. S. Iyengar, D. N. Jayasimha, and D. Nadig "A Versatile Architecture for Distributed Sensor Integration Problem," *IEEE Trans. on Computers* C-43, 2(Feb.), pp. 175-185, 1994.

[47] S. S. Iyengar and L. Prasad, "A General Computational Work for Distributed Sensing and Fault Tolerant Sensor Integration," *IEEE Trans. on SMC*, March 1995.

[48] J. R. Jain and A. K. Jain "Displacement mmeasurement and its application in interface image coding", *IEEE Transactions on Communications*, vol. COM-29 No. 12 (Dec), pp. 1799-1808, 1981.

[49] D. N. Jayasimha, S. S. Iyengar, and R. L. Kashyap, "Information Integration and Synchronization in Distributed Sensor Networks" *IEEE Transactions on Systems, Man, and Cybernetics* 21 5(Sept-Oct), pp. 1-13, 1991.

[50] D. N. Jayasimha, "Fault Tolerance in a Multisensor Environment." Technical Report, Department of Computer Science Ohio State University, Fall 1993.

[51] S. Kak *Neural Networks, Iterative Maps, and Chaos*, Course Notes Louisiana State University, 1994.

[52] T. Krol, "(N,K) Concept Fault Tolerance" *IEEE transactions on Computers* Vol. C-35, No 4(April), 339-349, 1986.

[53] A. Kumar, R. Pathak, and Y. Gupta, "Genetic-Algorithm-Based Optimization for Computer Network Expansion", *IEEE Transactions on Reliability*, 44, 1(Mar), pp. 63-72, 1995.

[54] P. J. M. van Laarhoven and E. H. L. Aarts, *Simulated Annealing: Theory and Applications*, D. Reidel Publishing Co. , Dordrecht, 1987.

[55] L. Lamport, R. Shostak, and M. Pease "The Byzantine Generals Problem", *ACM Trans. Program. Lang. Syst.*, 4, 3(July), pp. 382-401, 1982.

[56] L. Lamport and P. Mellar-Smith, "Byzantine Clock Synchronization" in *3rd ACM Symposium on Principles of Distributed Computing* ACM, New York, 68-74, 1984.

[57] R. Luo and M. Kay, "Multisensor Integration and Fusion in Intelligent Systems" *IEEE transactions on Systems, Man, and Cybernetics*, vol. 19 No. 5(Sept-Oct), pp. 901-930, 1989.

[58] R. Luo and M. Kay "Data Fusion and Sensor Integration: State-of-the-art 1990s" *Data Fusion in Robotics and Machine Intelligence* Abidi and Gonzales ed.s, Academic Press, Boston, pp. 7-136, 1992.

[59] V.R. Mandara and J.M. Fitzpatrick, "Adaptive search space scaling in digital image registration" *IEEE Transactions on Medical Imaging*, vol. 8 No. 3(Sept.), pp. 251-262, 1989.

[60] J. R. Maheshkumar, V. Veeranna, S. S. Iyengar, and R. R. Brooks, "A New Computational Technique for Complementary Sensor Integration in Detection-Localization Systems", *Journal of Optical Engineering* SPIE, vol. 35 No. 3(March), 1996.

[61] K. Marzullo, "Tolerating failures of continuous-valued sensors." *ACM Transactions on Computer Systems*, 8 (4):284-304, 1990.

[62] A. Mitiche and K. Aggarwal "Contour registration by shape specific points for shape matching comparison", *Vision, Graphics and Image Processing*, vol. 22 pp 396-408, 1983.

[63] M. A. Oghabian and A. Todd-Pokropek, "Registration of brain images bay a multi-resolution sequential method" *Information Processing in Medical Imaging*, pp. 165-174, Springer, 1991.

[64] L. Painton and J. Campbell, "Genetic algorithms in optimization of system reliability", *IEEE Transactions on Reliability*, 14, 2(Jun), pp. 172-178, 1995.

[65] C. A. Palazzini, K. K. Tan and D. N. Levin, "Interactive 3D patient-image registration" *Information Processing in Medical Imaging*, pp. 132-141, Springer, 1991.

[66] A. Pinz, M. Prontl, and H. Ganster "Affine Matching of Intermediate Symbolic Presentations", *CAIP '95 Proceedings LNCS 970*, ed. Hlavac and Sara, Springer Verlag, pp. 359-367, 1995.

[67] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes in Fortran*, pp. 436-448, Cambridge University Press, 1986.

[68] S. Rai and D. P. Agrawal, *Distributed Computing Network Reliability*, IEEE Computer Society Press, Los Alamitos, 1990.

[69] S. Rai, "Evaluating FTRE's for Dependability Measures in Fault Tolerant Systems", *IEEE Transactions on Computers*, 44, 2(Feb), pp. 275-285, 1995.

[70] Roseaux, *Exercises et Problemes Resolus de Recherches Operationelle tomes 1-3*, Masson, Paris, 1987.

[71] R. Sahner and K. Trivedi, "Performance and Reliability Analysis Using Directed Acyclic Graphs", *IEEE Transactions on Software Engineering*, SE-13, 10(Oct), pp.1105-1114, 1987.

[72] D. P. Siewiorek and R. S. Swarz, *The Theory and Practice of Reliable System Design*, Digital Press, 1982

[73] M. Srinivas and L.M. Patnaik "Genetic Algorithms: A Survey" in *IEEE Computer*, 27(6) pp. 17-26, June 1994.

[74] G. Stockman, S. Kopstein and S. Benett "Matching images to models for registration and object detection via clustering", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 4 No. 3, pp 229-241, 1982.

[75] G. Strang, *Linear Algebra and Its Applications*, Academic Press, New York, 1976.

[76] E. Taillard "Parallel Taboo Search Techniques for the Job Shop Scheduling Problem" in *ORSA Journal on Computing* 6(2), pp. 108-117, Spring 1994.

[77] D. Tang and R. Iyer "Dependability Measurement and Modelling of a Multi-computer System", *IEEE Transactions on Computers*, 42, 1(Jan.), pp. 62-75, 1993.

[78] N. H. Vaidya and D. K. Pradhan "Fault-Tolerant Design Strategies for High Reliability and safety", *IEEE Transactions on Computers*, 42, 11(Oct), pp. 1195-1206, 1993.

[79] B. Vinnakota and N. K. Jha "Diagnosability and Diagnosis of Algorithm-Based Fault-Tolerant Systems", *IEEE Transactions on Computers*, 42, 8(Aug), pp. 924-937, 1993.

[80] P. Van Wie and M. Stein, "A LANDSAT digital image rectification system" *IEEE Transactions on Geosci. Electron.*, vol. GE-15, July 1977.

[81] Wiener, N. *Kybernetik*, Rohwolt, 1963.

[82] G. Wolberg, *Digital Image Warping*, IEEE Computer Society Press, 1964.

[83] R. Y. Wong and E. L. Hall "Performance Comparison of Scene Matching Techniques", *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. PAMI-1 No. 3(Jul), pp 325-330, 1979.

[84] Y. Zheng, "Integration of Multiple Sensors into a Robotic System and its Performance Evaluation", *IEEE Trans. on Robotics and Automation* 5 (5):658-669, 1989.

VITA

Richard Brooks is a Computer Science doctoral candidate specializing in robotics and image processing at Louisiana State University. He also has a bachelors degree in mathematical sciences (1979) from the Johns Hopkins University in Baltimore, and studied operations research and computer science at the Conservatoire National des Arts et Metiers (CNAM) in Paris. His work experience includes projects with NASA Goddard Spaceflight Center, Radio Free Europe/ Radio Liberty Munich and the French stock exchange authority. As a consultant for the World Bank he helped implement their network in Africa, Eastern Europe and Central Asia.

Mr. Brooks speaks French and German fluently, and is a member of Association of Computing Machinery and the Institute for Operations Research and Management Sciences.

DOCTORAL EXAMINATION AND DISSERTATION REPORT

Candidate: Richard R. Brooks

Major Field: Computer Science

Title of Dissertation: Robust Sensor Fusion Algorithms: Calibration and Cost Minimization

Approved:

Major Professor and Chairman

Dean of the Graduate School

EXAMINING COMMITTEE:

Date of Examination:

6/24/96