SHRP-ID/UFR-92-616

Investigation of a Pavement Crack-Filling Robot

Department of Civil Engineering
Carnegie Mellon University
Pittsburgh, PA

Strategic Highway Research Program
National Research Council
Washington, DC 1992

SHRP-ID/UFR-92-616
Contract ID-017

Program Manager: *Don M. Harriott*
Project Manager: *Shashikant Shah*
Production Editor: *Marsha Barrett*
Program Area Secretary: *Ann Saccomano*

November 1992

key words:
automated crack filling system
command language interpreter
crack filling
crack sealing
image processing board
laser range sensor
pavement cracks
pixel image
quadtree model
robotic crackfilling system
video imaging

Strategic Highway Research Program
2101 Constitution Avenue N.W.
Washington, DC 20418

(202) 334-3774

The publication of this report does not necessarily indicate approval or endorsement of the findings, opinions, conclusions, or recommendations either inferred or specifically expressed herein by the National Academy of Sciences, the United States Government, or the American Association of State Highway and Transportation Officials or its member states.

Acknowledgments

The research described herein was supported by the Strategic Highway Research Program (SHRP). SHRP is a unit of the National Research Council that was authorized by section 128 of the Surface Transportation and Uniform Relocation Assistance Act of 1987.

Contents

Appendixes

List of Figures

List of Tables

Abstract

This report describes a laboratory prototype designed to identify, map, track and fill pavement cracks. Video imaging and electronic scanning equipment installed in a van is followed by a rack carrying a hot-air lance, a sealant wand, and a supply trailer.

The video sensor identifies potential cracks while the depth sensor verifies actual depth. A map plotter defines the crack length, size and depth. Using the combination of visual and depth sensing equipment, the cracks located and mapped were filled with an accuracy of better than 1 cm. (0.5 in.).

Approximately, $125 million per year is spent by states on crack-filling with labor representing 50 percent of the costs. Assuming a four-year system life at a cost of $100,000 plus annual maintenance and operating costs of $25,000 per year, potential savings could be as much as $60,000 per unit, per year or 19 percent of costs on a national basis equal to $24 million per year.

Executive Summary

1 Introduction

Automation of roadway maintenance presents substantial opportunities to reduce labor costs, improve work quality, and decrease worker exposure to roadway hazards. This report describes a research project investigating an automated method for sealing pavement cracks. A prototype laboratory system was initially constructed and demonstrated as part of the project. A second phase field prototype system was designed, built and demonstrated. Also, the economic feasibility of automation for this application was analyzed. The major result of the research was demonstrating the feasibility of identifying, mapping and tracking pavement cracks. Using a combination of video imaging and electronic range scanning, routed and unrouted cracks of 1 cm. (0.5 in.) could be located, mapped and traversed.

In the recommended field system for an automated crack filling system, a lead vehicle would tow an xy-table assembly or robot arm (Figure 1). The lead vehicle would transport generators, computers, sensor processing hardware, and controllers. Mounted on the rear of the vehicle would be a video camera used for identifying cracks. The xy table is similar to a pen plotter with a cart moving within a rectangular frame. Tools mounted on the xy table would be: a depth sensor for verifying cracks, a hot air lance for cleaning cracks, and a sealant wand for filling cracks. Bulk supplies of sealant material and propane trail behind the xy table.

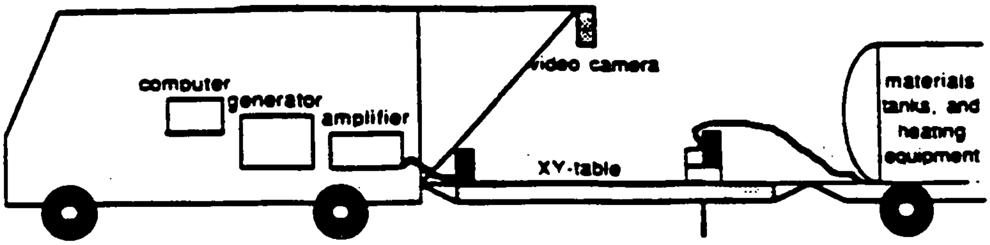


Figure 1: Illustration of a Possible Field System for Filling Cracks

The prototype systems emulated the proposed system and demonstrated the following steps:

- The vision system identified potential cracks and generated a traversal plan for the depth sensor to collect range data.
- The range data was fused with the video data into a unified map. Potential cracks identified by the vision system, but not corroborated by range data, were dismissed as filled cracks, shadows or oil spots.
- The center lines for the corroborated cracks were identified and a trajectory minimizing the motion of the tools was generated. The cracks were traversed first with a tool cleaning the crack with compressed air. After the cracks were prepared, the laboratory system again traversed the crack and filled the crack volume.

A photograph of the second generation apparatus appears in Figure 2. Appendix III lists the various descriptive documents available in addition to this report.

2 Sensing Pavement Cracks

Sensing hardware used in the prototype system included:

- A commercial VHS camera was used to generate the video signal required by the image processing board.
- An image processing board, commonly referred to as a frame grabber, was used to convert an analog video image signal to a digital matrix of numbers representing grey scale. In the prototype, individual raster cells in the digital image were approximately 4 mm by 4 mm (0.2 in x 0.2 in).
- An infra-red laser range sensor was used to develop three dimensional profiles of the roadway surface. The range sensor had a "footprint" of 5 mm (0.2 in) in the configuration used in the prototype and range readings were taken every 2 cm. (0.9 in.) across the pavement surface. The range sensor used cost approximately $ 2,000.

Other sensors were considered for the system, particularly sonar as an alternative for range sensing. However, sonar systems were susceptible to environmental noise from wind and precipitation.

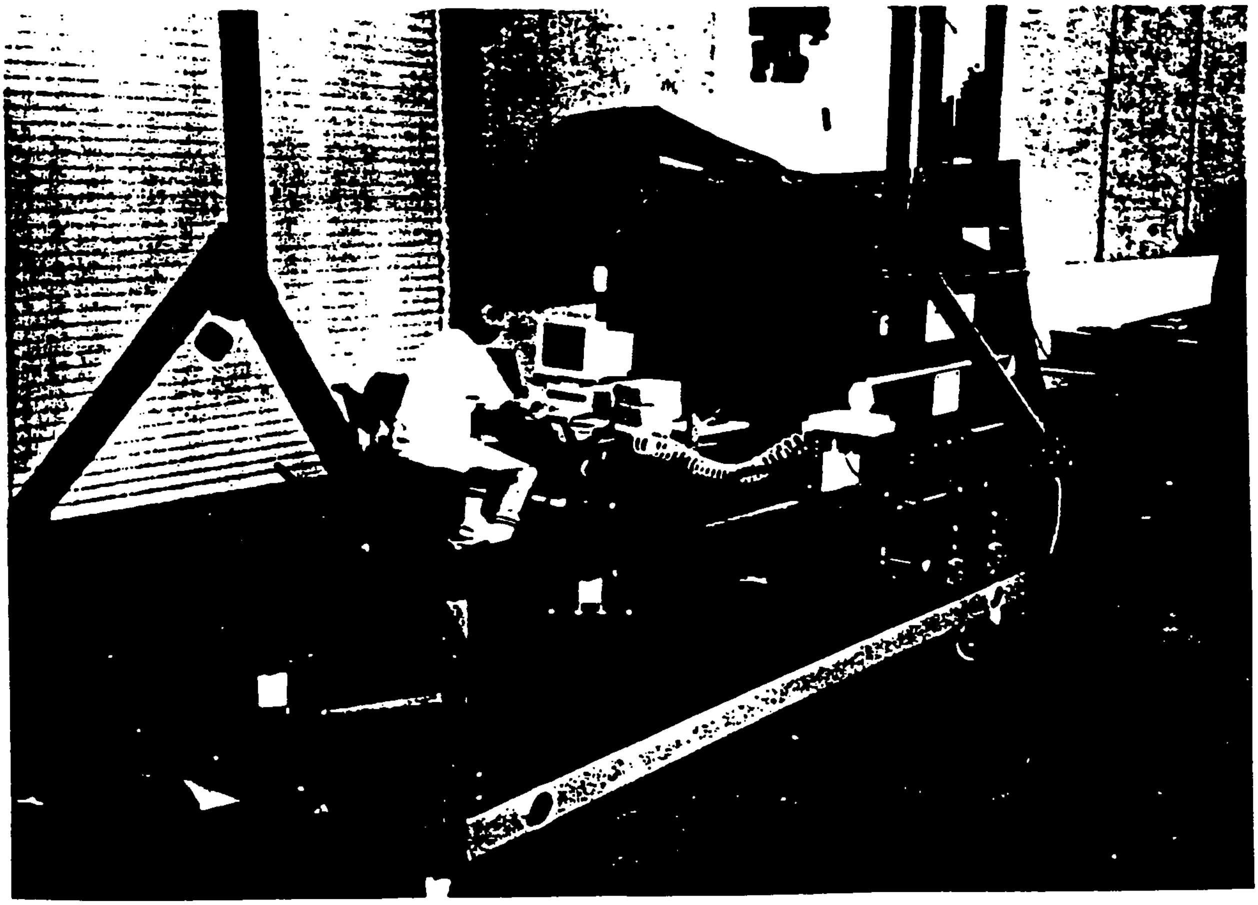


Figure 2: Photograph of the Field Demonstration System

A dual sensor system was used because either video or range sensing was not adequate. Surface dark spots or lines could be easily mistaken for cracks in the video image. Range sensing was relatively slow since the sensor had to be moved above each spot being mapped. Range sensing from a single point would be expensive to achieve the required accuracy, and a linear array of multiple sensors would likewise be expensive. Thus, video imaging was used to identify area of potential cracks, and range sensing was used to confirm the location and identity of pavement cracks.

3 Representing Pavement Cracks

A flexible method of representing pavement surfaces was applied in this research based on a two-and-a-half dimensional "quadtree" computer model. The "half" dimension refers to elevation or range data. This representation model accommodated numerous operations for surface perception and modeling such as image filtering, registration of multiple sensor data and generation of crack traversal patterns. The C++ object oriented language was used for software development. The model has been applied to general condition assessment data derived from several commercial sources in addition to the automated maintenance system [Haas 90a].

4 Filling Pavement Cracks

Once pavement cracks were identified and mapped, specialized control software was used to traverse cracks on the pavement surface for cleaning and filling. The motor controls and related software in the system proved to be quite accurate for single and repeated movements, with an accuracy over a 3m by 3m (10 ft. x 10 ft.) area of 5 mm. (0.2 in.)

5 Alternative System Designs

Several alternative system designs were considered for commercial and field systems. These options included:

- Continuous movement systems with a set of linear nozzles. Unfortunately, control of sealant application would be difficult since instant on-off is required, and the cost of crack perception would be high with such systems.

- Small, mobile, highly maneuverable robots tethered to a field vehicle were considered as an alternative to the xy table. These effectors could provide continuous filling but might increase costs and control problems.

- Alternative sensing systems such as single point laser range sensors were examined. These type of sensors have the potential of dramatically reducing the time required for range scanning, but the existing technology could not provide a fine enough resolution.

- Alternative effectors such as robot arms could replace the xy-table system. However, these pose difficulty in terms of load bearing capacity, controlling contact forces, spatial constraints and computational intensity.

6 Economic Analysis

A robotic crackfilling system can have significant and substantial economic benefits. These benefits include reduced labor costs, and improved quality and safety. A review of literature and reports related to crackfilling practice and survey of the 50 states and various turnpike, towns, counties and provinces provided data on crack filling practice and expenditures for this project. Although crack filling practices vary significantly between states, it is a widely used maintenance procedure. It is also relatively labor intensive with labor costs representing over 61% of costs on a per lane mile basis. It is estimated that about $53.3 million per year is spent by states on crack filling. This is considered to be conservative as it ignores expenditures by counties and cities as well as airports, turnpikes, other authorities and private organizations. Complete automation of this process would require about 450 crack filling units nationwide.

Benefits would be realized in the form of reduced labor costs, improved quality and improved worker safety. Assuming a 6 year system life, based on 6 months of operation per year, system acquisition costs of $100,000 and annual operating and maintenance cost of $10,000 per year, a 5% discount rate, productivity rates comparable to existing approaches and the elimination of three laborers at $12 per hour, the net savings realized using the automated crack filler is approximately $6,300 per unit per year or 19% of costs on a national basis. This is equal to $2-8 million per year in national savings due to labor costs alone.

1. Introduction

While many roadway construction and maintenance tasks have been mechanized, automation has not generally been applied to construction and maintenance activities. Maintenance procedures remain labor intensive and expensive. Recent advances in robotic technology and the related experience in manufacturing facilities suggest that greater automation may be extremely beneficial [Skibniewski 90]. Relevant technological improvements include new sensors, microchip computer processors, sensor interpretation algorithms, vehicle control hardware and manipulator hardware. This report describes a prototype automated pavement crack filling system that has been implemented in a laboratory setting for concept demonstration and testing on routed crack scenes. It also suggests a design for an actual system and provides an economic assessment of automated pavement crack filling.

1.1 Current Crack Filling Practice

Crack filling is normally conducted by a five or six person road crew [AASHTO 87]. The equipment used includes pylons, a heavy truck, a sealant tank, a heated air torch, a sealant wand, and a routing machine if the cracks are being routed prior to being filled. One or two crew members may be necessary to direct traffic and place pylons. If the cracks are being routed, the router precedes the truck. An operator walking behind the truck blows out the cracks with the torch and another in turn fills them in with sealant material. A sand covering may be applied to permit immediate use by traffic. The procedure varies significantly from region to region [FHWA 87]. For example, the Ontario Ministry of Transportation uses an equipment train with two routers to balance workloads [Chong 88].

Many states and provinces routinely fill cracks as a maintenance activity on an "as required" basis. The expenditures on this activity often represent a significant portion of pavement maintenance budgets as shown in Table 1-1. Unit costs vary from approximately $0.44 per linear foot in North Dakota [FHWA 87] to $1.55 per linear foot on Air Force bases [Brown 88] in 1990 dollars.

Table 1-1: Selected Crack Sealing Expenditures by States [FHWA 87] (1984 Dollars)

	California (flexible)	Iowa	North Carolina	North Dakota	Pennsylvania
Annual Crack Sealing Program (1,000's $)	9,546	3,642	1,144	1,500	6,658
Annual Maintenance Budget (1,000's $)	102,000	75,000	298,208	35,000	637,000
Maintenance Budget	9%	5%	0.4%	4%	1%

1.2 Justification For Automated Crack Filling Research

Automation of crack filling operations is of considerable interest for several reasons. First, cracfilling is a widespread and common operation in the United States. If even modest cost savings could be achieved in crack filling operations, the total savings would be substantial. Second, automated crack filling may achieve improved quality over existing field operations, so that the need for maintenance operations may be reduced over time. Finally, automated crack filling would reduce the exposure of maintenance workers to injury and accident.

Automated pavement crack filling is a technically challenging operation for several reasons. Since pavement cracks are irregular in nature and extent, simple numerically controlled devices cannot be used directly. Some means of perceiving crack location and controlling maintenance equipment is required. Moreover, crack filling is undertaken under field conditions which may involve extremes of temperature, precipitation and debris. Maintenance of the equipment used in roadway work varies considerably in quality, therefore robust and reliable equipment is required. Finally, introduction of automated equipment in this domain must be justified by cost savings and quality improvements, so inexpensive and effective equipment is imperative.

1.3 Project Activities

The following specific work tasks were performed in Phase I of this research:

1. **Functional Specification of a Crack Filling Robot**
   An overall specification of equipment and functional software capabilities was an important initial step in the project. This specification constituted the primary design document for the system.
2. **Purchase of Required Equipment and Software for a Laboratory Prototype**
   The laboratory system relied on commercially available parts and either commercial or public domain software to the maximum extent possible. This approach minimized development costs and also insured that further development and deployment of the system could be undertaken.
3. **Assembly of the Prototype Crack Filling Robot**
   Construction of apparatus and integration of software in the laboratory was performed.
4. **Software Development for Vision, Evaluation, Planning and Control**
   Considerable programming for the system was required, particularly for evaluation, planning and control. The language used in this work was C++, an object oriented C dialect available with numerous operating systems [Stroustrup 87]. The programs were written in a modular fashion to permit extension to later system designs.
5. **Demonstration of the Laboratory Prototype Crack Filling Robot**
   Laboratory demonstration occupied the final stages of the research project. Several test specimens were assembled and used to test the system. For simplicity, laboratory tests were done using a powder substitute for actual sealant.
6. **Evaluation of Economic Impact and Overall Technical Feasibility**
   In the process of developing and using the system, more refined estimates of the costs of a field level crack filling robot were developed. Possible benefits were also assessed by interviewing experts and examining state records of crack filling costs.

The following tasks were performed in Phase II of this research:

1. **Functional specification of Modifications Required for Un-routed Crack Perception**
   Included in this specification are both sensor and software modifications.
2. **Purchase of Required Sensors, Field Computer and Table Frame**
   The need for a more sensitive range sensor than the infra red laser sensor currently used to identify routed cracks was reviewed. The existing sensor was still used. To produced a more robust system for field testing, the stronger, mobile system frame was produced.

3. **Modification of Perception Software**
   Existing software coded in C++ was modified to permit perception and modeling of un-routed cracks. No modification of table control software should be required.

4. **Laboratory Testing of Un-routed Cracks**
   Laboratory testing of the refined sensors and software was conducted.

5. **Field Demonstration of the Crack Filling Robot**
   Field demonstration was conducted at the Caltrans Research Laboratories in Sacramento.

6. **Evaluation of Economic Impact and Overall Technical Feasibility**
   In the process of developing and using the system, more refined estimates of the costs of a commercial crack filling robot will be developed. Possible benefits were assessed by interviewing experts surveying possible users and examining field records of crack filling costs.

7. **Documentation and Reporting**
   An important component of the research effort was documentation of the system, including the design, operation and performance. Documentation included written reports. Professional papers were presented at conferences or in refereed publications describing the research work. The system was also demonstrated at the AASHTO Technology Transfer Fair in Milwaukee.

Phase II was conducted with support from Caltrans. Meetings were also conducted with the University of Davis and Caltrans researchers working on Contract H-107, "Fabrication and Testing of Maintenance Equipment for Pavement Surface Repairs".

2. System Architecture

2.1 Introduction

Identifying cracks in the road surface automatically is not an easy problem [Haas 84, Butler 89, Fukuhara 90, Bomar 88, Maser 87, Mendelsohn 87, Wigan 87]. Mapping the layout of the cracks in detail and selecting those to be filled increases the difficulty. In the case of routed cracks, the problem is simplified by distinct visual patterns and by consistent groove dimensions. To identify the cracks, characteristic surface data is required. Applicable sensing technology includes vision, range, and heat sensitive devices. In practice, all these sensors experience noise because of the varied topological and color conditions of the pavement surface, and because of environmental factors such as wind and sunlight. Even with good data, a crack identification system can be fooled. Analysis of a video image alone shows that it is almost impossible to automatically detect the difference between a routed crack, a filled crack, and a strip of dark oil. With the corroboration of range information, however, routed cracks can be distinguished from impostors. Conversely, range data alone is too noisy to build a good crack representation by itself, its sensitivity range is narrow, and it is time consuming to collect. Alternatively, combining information from both range and vision sources into a common surface representation can increase the overall accuracy and speed of crack perception. The capabilities required to do this exist in the pavement surface model presented in Chapter 4.

A schematic illustration of the prototype system design appears in Figure 2-1. A lead vehicle would tow an xy-table assembly, the sealant supply and propane tanks. The lead vehicle would carry the power source for the manipulator and the necessary computing equipment. A camera would be mounted on an extendable boom which could be suspended above the work surface during operation. A video monitor would be located next to the driver to allow monitoring and manual intervention. The monitor could also be used in an interactive configuration in which the driver would enter of rough directions to the system via a joy stick in the form of graphics overlayed on the monitored scene. Three tools would be mounted on the xy-manipulator. They include a hot air lance, a sealant wand, and an infrared laser range sensor.

The laboratory prototype system pictured in Figure 2-2 performs the same crack filling process that would be used in a field hardened version. Assuming a stop-start strategy, the following procedure would be followed as the system moves down the road:

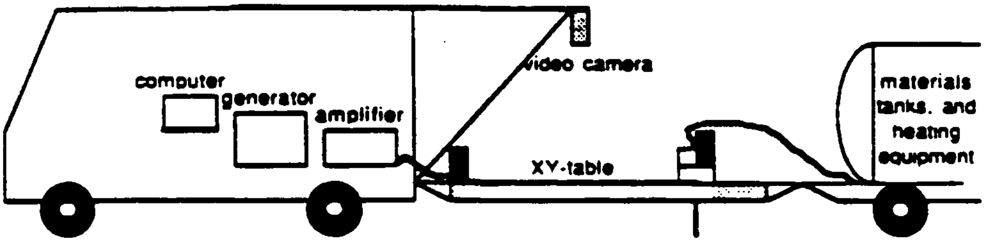


Figure 2-1: A Schematic Illustration of the Crack Filling Robot System

1. Sensor data is acquired and used to develop a representation of the pavement surface.
2. A map of the cracks to be filled is generated.
3. An efficient traversal plan for the surface preparation and filling operations is developed.
4. The Crack Filling Robot performs the blowing and filling operation.
5. The Crack Filling Robot advances and repeats the four steps described above.

In practice, the above steps involve a number of substeps or components (Figure 2-3). First a video image of the site is acquired. The information is processed to extract regions requiring range scanning to corroborate cracks. Since range scanning is relatively slow, this approach minimizes the total area to be range scanned and lowers the time required for the whole process. Range data acquired by the table is returned to the perception system and merged with the video data. Marks and objects that would have misled a single video sensor based system are rejected with the use of the range data. The model's resulting crack representation is used to generate an ordered lists of cartesian coordinates describing points along the cracks' medial axes. This derived vector of coordinates is interpreted to yield a graph representation of the crack network in the area to be repaired. The most efficient traversal through this network can be derived by a variety of means [Peters 90a]. The resulting traversal plan is then compiled into a list of commands to the manipulator and effector actuators for cleaning and filling the routed cracks.

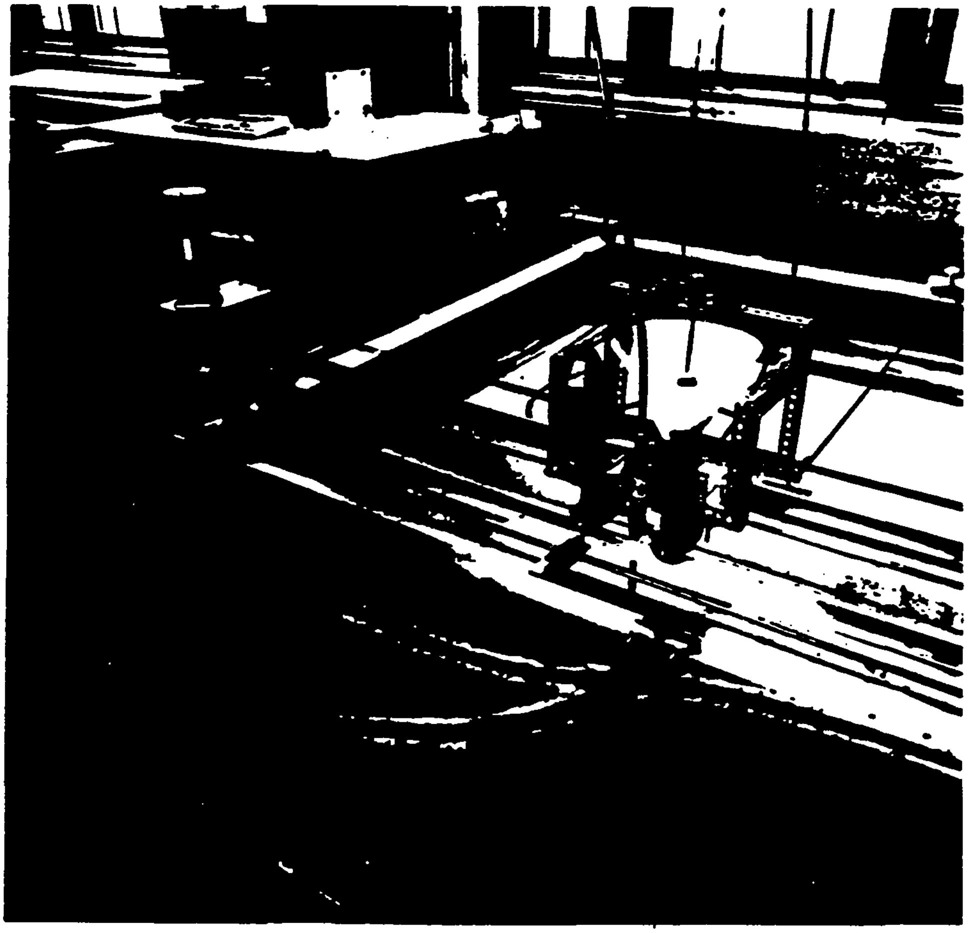


Figure 2-2: Crack Filling Prototype System

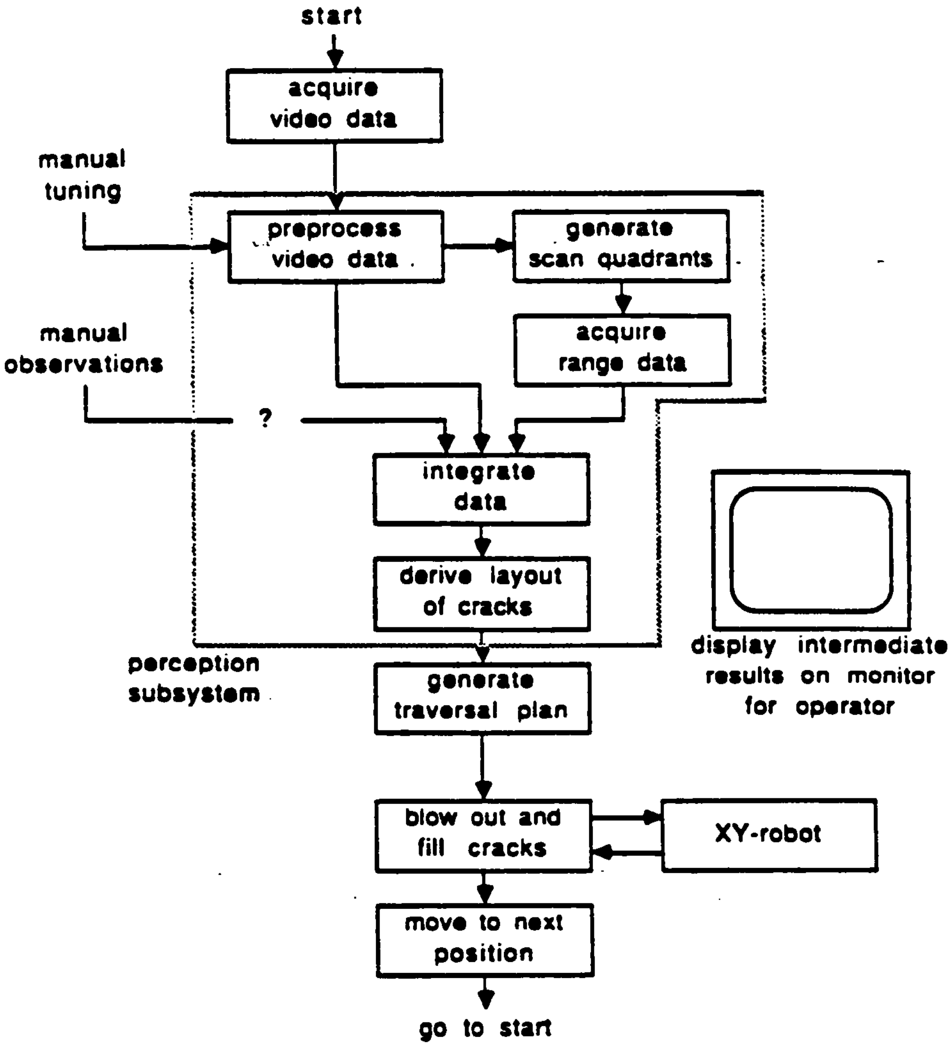


Figure 2-3: Crack Filling Process

2.2 XY-Table Assembly

Before a description of the process begins, it would be beneficial to illustrate how the entire operation fits in with the actual work environment. The manipulator used for the entire crack filling process was an xy-table. It offered durable construction which provided sufficient accuracy for acquiring range data and moving the tools for cleaning and filling cracks. The laboratory model consisted of a gantry and a cart together providing the planar motion. The entire assembly is pictured in Figure 2-4, and the dimensions and coordinate systems in Figure 2-5.

2.3 Crack Sensing

Vision Data Acquisition

The objective of visual data acquisition was to provide an accurate model of the surface of the road which contains the crack. Any aberrations such as oil markings or gravel, would also be captured. Vision data was collected as a square-pixel image (512x512) measuring 2x2 meters (6.6x6.6 ft.). It was also necessary to carefully align the reference frames used by the vision system and the range data system.

Range Data Acquisition

Because the vision algorithms could potentially identify pavement anomalies such as oil spots or shadows as cracks it is necessary to corroborate areas of cracking identified by the vision algorithms. This is accomplished by collecting range data on the *areas of interest* and fusing this data with the vision data. Areas with intersecting details were corroborated as cracks, while those areas not corroborated were dismissed.

Range data acquisition was collected using the xy-table to position the range sensor located on the cart. Range scans for the initial testing scanned the entire scene. This of course would not be the case in the actual process because only *areas of interest*, determined by the vision algorithms, would be range scanned.

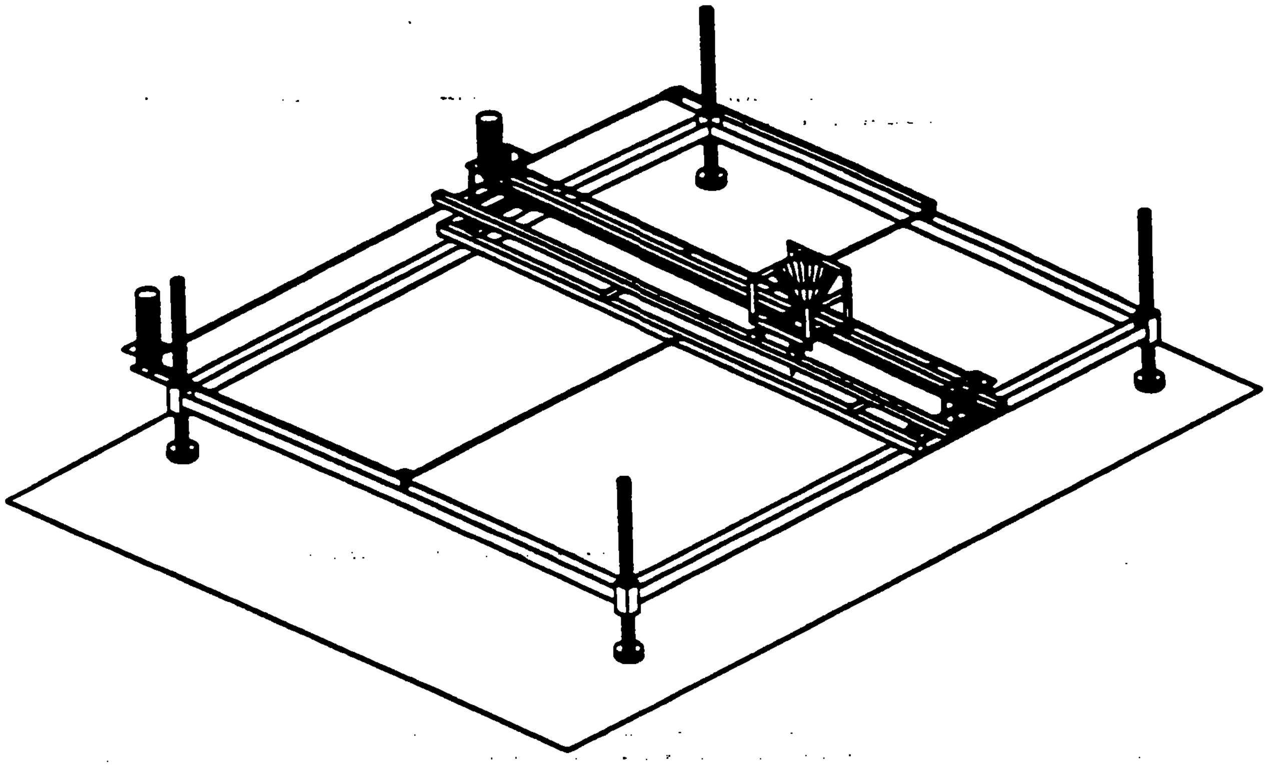


Figure 2-4: An Illustration of the XY Table Mechanism

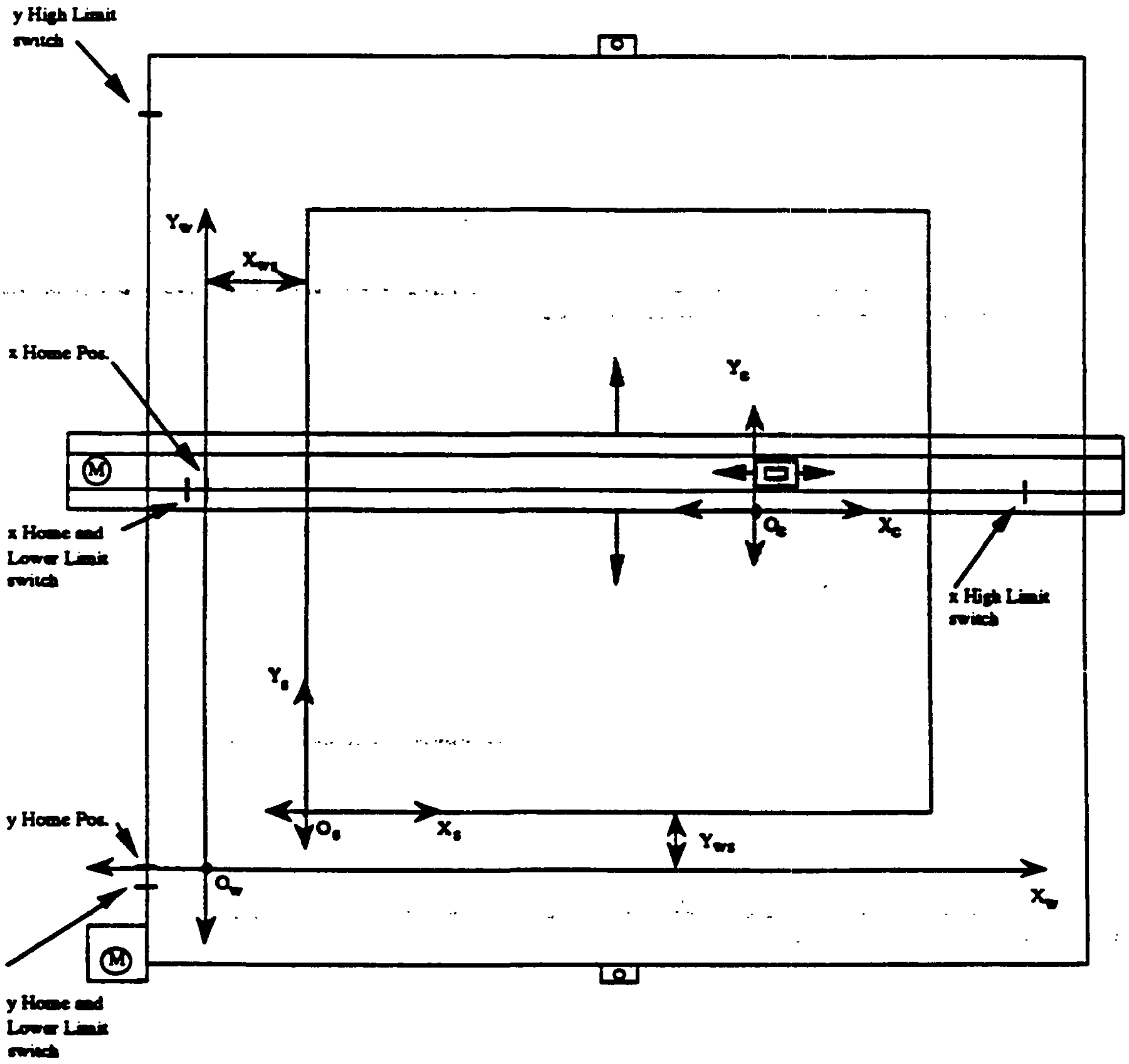


Figure 2-5: Coordinate Systems

2.4 Processing Sensor Data for Surface Characterization

Once the scenes were digitized and adjusted to be of equal size, they were processed to enhance details that would precisely locate cracks. A Command Language Interpreter (CLI) image processing utility was developed that offered a wide variety of useful tools for interactively extracting features from the video and range data. Briefly, there were a set of primary operations which altered the image (vision or data) to enhance the outline of the crack. Those functions included:-

- Binarization: set the threshold for which all grey values were converted to black or white. Processing was simplified by working with only black or white, rather than 255 shades of grey. The crack was represented as black and the background as white.
- Filtering: was used to remove much of the noise and to fill in small holes in the image. It was useful for reducing spurious noise and for smoothing jagged edges of the raw data. This operation is often referred to as *softening* or *blurring* the image.
- Skeletonization: reduced the remaining details to their basic skeleton structure. This skeleton structure was one pixel wide centered along the medial axis of detected cracks.
- Connecting: a process necessary for quadtree conversion (explained in Section 5-9 that removed lone diagonals.

The preprocessing software (CLI) is versatile, and supported many other ancillary features, described in [Grove 90].

2.5 Pavement Representation

In addition to preprocessing, the perception software, CLI, also put the images into a quadtree representation. The use of the quadtree analysis was to fuse the vision and range data into one multi-layer quadtree that could precisely locate and corroborate cracks. With this accomplished, a quadtree encoded file could be interfaced to the path planning and table control software. Those steps are described below:

- Video image loaded, processed and converted to the quadtree representation.
- Components of the quadtree (cracks) were labeled and sized.
- Range data was grafted into the quadtree model containing the data collected by the vision system.
- Components not corroborated by the vision system were rejected.
- The quadtree representation of the corroborated cracks was written to a file for use by the path planning and table control modules.

2.6 Path Planning and Table Control

Once the crack had been identified and placed in the quadtree structure, the representation was passed to the path planning software package. This consisted of several files. First, the crack was extracted from the quadtree so as to allow for readable output. This initial output was composed of several vectors of coordinates describing the locations of cracks detected. The output of this crack description module was a vector of short, linear-crack approximations described by start and end coordinates.

Path optimization software then planned a next-nearest neighbor traversal scheme. Crack start and end coordinates, which included crack intersections called nodes, were traversed in a sequence such that after one short segment had been traversed, the next one closest would then be traversed. The program for planning an efficient route was implemented using a recursive formulation.

The output of the path optimization module was compiled into a trajectory description file, written in a Table Control Language (TCL). The TCL had its own working coordinate systems and primitives that execute crack blowing and filling.

Before implementing any control on the table, a set of coordinate systems was defined. Those included a world coordinate system (Ow), a test scene coordinate system (Os), a cart coordinate system (Oc), a tool coordinate systems for the blower (Ob), filler (On), and range sensor (Or), and a motor coordinate system.

The TCL defined a set of absolute and incremental motion commands for each of the effectors, relative to the scene coordinate system. Those commands controlled displacement, velocity, and tool actuation (on/off/percent flow). Velocities were adjusted by the control algorithms to provide piecewise linear trajectory tracking. Velocity adjustments were necessary to follow a path similar to A, instead of paths B or C shown in Figure 2-6.

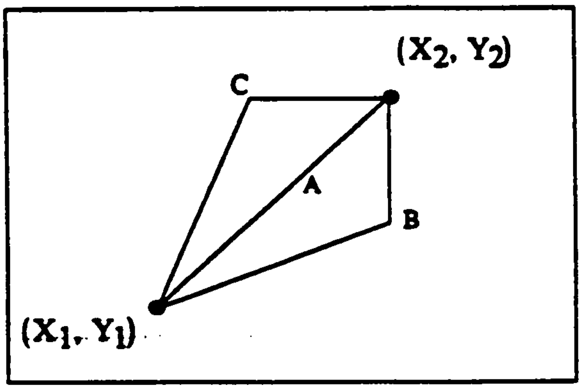


Figure 2-6: Possible XY trajectory tracking scenarios

3. Sensing Hardware

3.1 Vision

Acquisition and processing of visual information is a fundamental step for automated crack filling. Data which accurately represents pavement features is necessary for crack detection, path planning, blowing, and filling. A commercial Panasonic video camera already available in the Civil Engineering Laboratory was used to provide the raw image. A frame grabber accepting live video information was used to digitize the image obtained from the video camera for image processing. Since there were numerous companies which manufacture digitizing boards, the problem of selecting video imaging hardware was reduced to finding one which conformed to the following criteria.

1. Many complex frame grabbers can cost thousands of dollars. A fairly strict budget limited the search to a low-end frame grabber which still had the flexibility to meet specific processing demands.

2. The frame grabber needed to digitize an image and produce a 512x512 square array of digital information in a string file format. The square array was necessary to implement the quadtree structure which the data would ultimately possess. The (512x512) size was needed to insure that resolution of the image was fine enough to capture all details of an image.

3. The frame grabber had to produce pixel (picture element) areas with a 1:1 aspect ratio (the width to height ratio of the pixel). Most frame grabbers digitize visual elements with the same aspect ratio of a television screen (4:3). Correlation of vision data with range data was best accomplished when both data arrays had corresponding aspect ratios.

4. Each pixel had to represent a grey level corresponding to a brightness level. These pixels range from 0 to 255 representing absolute grey levels.

5. The frame grabber had to produce information in a convenient form not requiring time consuming data translating operations.

6. Software support had to accompany the board and have interfaces to Microsoft C 5.1.

A product search revealed that the number of boards meeting the requirements were few. The HRT512-8 manufactured by High Res Technologies was selected. It provided the functionality necessary for the project and offered features which would reduce the difficulty of data acquisition. One drawback concerning the frame grabber's ability to process data centered on the fact that processing of data was not done on board, which meant processing was handled by relatively slow programs running on a personal computer. A change made in the boards usual configuration, in order to meet project requirements, was the installation of new oscillator. Normally, the frame grabber board operated with a 40MHz clock frequency producing a pixel aspect ratio of 4:3.. This would have been very inconvenient for reasons cited earlier. However, the flexibility of the board allow installation of an oscillator with a higher frequency of 48MHz which generated images with a 1:1 pixel aspect ratio.

The frame grabber also offered a continuous video monitoring mode along with its capturing mode. Live video information could therefore be monitored before digitizing occurs. Upon digitizing, all analog signals within the specified area were digitized by a flash A/D converter and stored on the boards memory buffer. This buffer of information could then be transferred to a computer buffer for analysis and processing, which was done by means of software on the PC. Because of the oscillator change, a new initialization routine was developed to account for timing changes for synchronization. Another problem encountered was introduced by limitations of video information. Since a video camera and television signals only contain 480 scan lines of actual video information, the digitizing board filled the remaining lines of the buffer were filled with meaningless information. Modifications had to be made to the buffer array to adjust the image and fill empty lines with token data. In the end, each video pixel corresponds to an area on the scene surface of 3.9 x 3.9 mm. (0.2 x 0.2 in.).

3.2 Range Sensing

The range data was acquired using an infrared laser range sensor (Hamamatsu H3065-10 Optical Displacement Sensor and C3359-10 Controller). An A/D board was used to digitize the range data (Data Translation Single Board Analog and Digital I/O System DT2805/5716). The range sensor was mounted on the xy-table cart which was used to draw the sensor in a raster scan pattern over the scene. The "footprint" of the laser range finder was 15 mm (0.6 in.) or less, and its range was 350 to 650 mm. (14 in.

to 26 in.). Its range resolution was ± 0.5 to 4.0 mm (0.02 in. to 0.2 in.) depending on surface conditions including reflectance and texture. For the 2 m x 2 m (6 ft. x 6 ft.) work area of the range sensor, an array of 128 x 128 range values were acquired for each scene. More efficiency would have been possible by limiting the range scanning to areas of interest. This would require further software development to control subsequent perception operations, but it should be implemented in future systems. Each range value corresponded to an area on the surface of 15.6 x 15.6 mm (0.6 in x 0.6 in). For range scanning, the scene surfaces were covered with debris in a realistic pattern about the cracks. The range noise produced by this debris was apparent in the range data images, however the various algorithms effectively removed the noise from the final binary image.

4. Representation

4.1 Introduction

The pavement surfaces model applied in this project (Figure 4-1) is a combination of a surface representation model and an associated characterization process [Haas 90a]. The surface representation was comprised of characteristics at several levels of abstraction and aggregation. Characterization was comprised of four basic operations: filtering, aggregation, fusion and structuring. Characterization transformed the state of the surface representation. Common data structures supported characterization and facilitated the surface representation; in this way the elements of the model were unified.

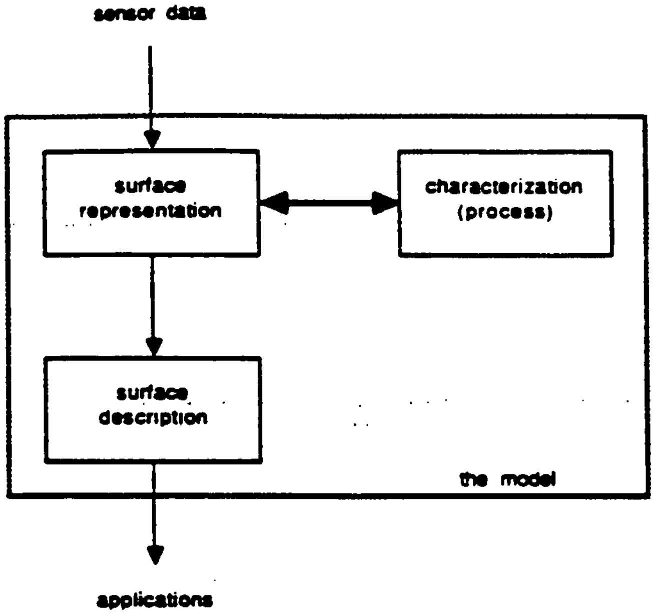


Figure 4-1: Overview of Pavement Surfaces Model

The model assumes pavement surfaces could be effectively characterized in two and one half dimensions, providing for the inclusion of pavement contour information. This assumption implied that all significant parts of the surface would be visible from directly above. Therefore pavement surfaces could be described in two dimensions with elevation as a scalar component. Surfaces can be represented by characteristics at different levels of aggregation and abstraction. The level of aggregation of a characteristic refers to its spatial extent and its distance hierarchically from original sensed data. Three levels of characteristics are defined, in hierarchical order:

- properties - derived from other properties or measured directly
- features - derived from properties and other features, and
- regions - spatial aggregations of sets of features and properties

The surface representation is composed primarily of two data structures. The first is a grid for mapping sensor data measurements. Measurements from different sensors are referenced to common points on the grid and thereby related to each other spatially. The grid supports sensor data filtering and reduction, and it forms the foundation of a generalized quadtree which is used to relate characteristics in a framework useful for data fusion and structuring. The generalized quadtree has advantages over other surface descriptions. It is compact because of its hierarchical structure and is unified because its nodes create a useful parallelism among surface characteristics. Each node is a data structure, with slots for each surface characteristic in a quadrant and with values for each slot. Descriptions of uniform characteristics spanning a wide area of the surface may be contained in higher nodes and propagated down the tree to access information at any level, including points. For example, Figure 4-2 illustrates the quadtree's hierarchical representation of pavement depression and cracking information. In their final state, each quadrant encompassed an area in which the property or feature value was relatively uniform.

The model was implemented as a software kernel or library using C++, an object oriented language [Haas 90b]. As a result, the grid, the tree types, and even nodes were implemented as data objects that have functions associated with them. For example the node object could be split, meaning it could grow four descendants, or pruned, meaning it could remove its four descendants. This form of implementation promotes modularity, independence and data hiding, and simplifies some algorithms.

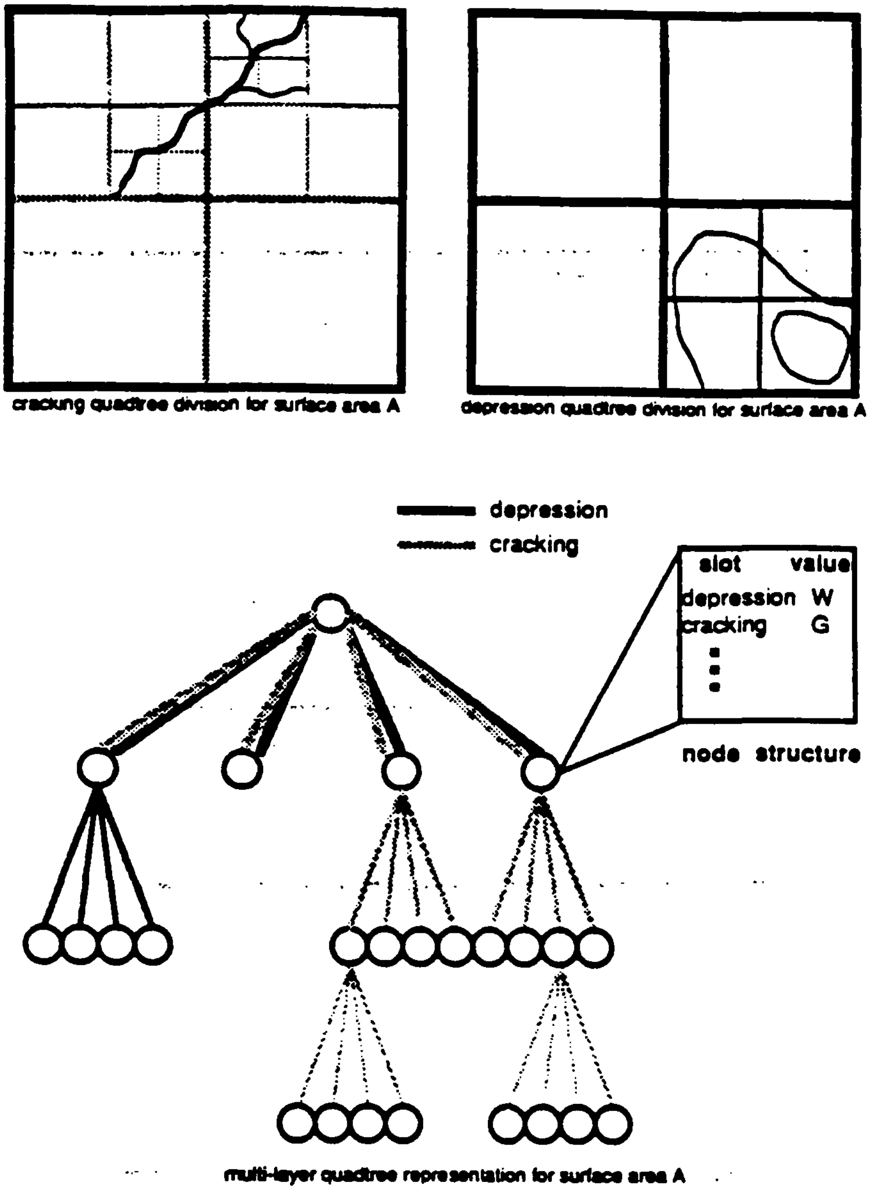

cracking quadtree division for surface area A
depression quadtree division for surface area A
depression
cracking
slot value
depression W
cracking G
node structure
multi-layer quadtree representation for surface area A

Figure 4-2: Multi-layer Surface Quadtree

4.2 Surface Characteristics

Data was stored at different hierarchical levels of surface aggregation and abstraction in the model. The levels defined included properties, features, and regions.

Properties are values associated directly with a grid point. A property is defined as *the value of a phenomenon measured or calculated at or about a point in the xy grid.* Properties may include:

- **elevation** - the distance in the negative z direction of a point with respect to an xy reference plane above the surface
- **color** - the intensity of the red, green, and blue spectrums of light sampled at a point on the surface
- **grey level** - the intensity of a pixel in a monochrome video image
- **infrared level** - the degree of infrared radiation detected at a position on the surface
- **electro-magnetic potential** - the magnitude and direction measured at a point on or just above the surface
- **gradient** - calculated for elevation, grey level, color, or other local properties
- **texture** - the phenomena of globally repetitive surface elements
- **edges** - the boundary between two different areas. Points on the grid could be identified as belonging to an edge through simple gradient thresholding operations.
- **deflection** - the center point

Only some of these properties were used in the laboratory prototype crack filling robot.

A feature was defined as *a spatial attribute of the surface which helps to characterize it.* Features were associated with areas of the surface. For example, the grey level of a pixel wass a property, but a number of pixels with low grey levels in an area of an image could form a feature called cracking which was associated with the corresponding surface area. In the generalized quadtree there was a state associated with each feature for each quadrant.

A region was defined as *a contiguous area of a set of features and/or properties.* Regions could be areas of hypothetical condition/cause pairs, areas of a particular condition, or areas where a set of features was relatively constant such as areas of a type of material or substance. In the generalized quadtree, regions were described by a set of adjacent quadrants, and they could be derived using manual, algorithmic, or knowledge based processes.

4.3 Surface Representation

The grid was the basic common structure by which sensor data was unified. It was an array of points laid out in a rectangular pattern in an xy reference plane. The plane was normally located above the surface with and arbitrary orientation The spacing between the points along an axis was constant, but the number of points along either axis was variable. Two conditions were imposed, however. First, the number of points in each dimension had to be a power of two. Second, the number of points in each dimension had to be divisible by a common denominator equal to the length of the smallest dimension or one half the length. Those conditions facilitated subsequent conversion to the generalized quadtree representation.

The grid dimensions were chosen so that all sensor data could be associated in a one to one mapping with points on the grid, which meant that no two datums from one sensor could be mapped to the same grid point. This condition was satisfied if the data type with the highest spatial resolution was used as a basis for the grid dimensions. The highest resolution data was the digitized image data in this application.

The definition of the grid ignored sensor performance in terms of spatial resolution, range resolution, scanner position accuracy, and dynamic range. Instead, those factors could be considered in a measure of variance associated with each sensor. This variance could then be used for a number of purposes including sensor fusion. In practice, sensor spatial and range resolutions should be evaluated using good engineering judgment when configuring a sensing system. In this research the grid was concerned only that sensor measurements were centered on points and that those points could be related to a common rectangular grid.

To permit efficient characterization of pavement surfaces and to provide a useful representation for applications, it was necessary to closely relate features and properties spatially and to access related characteristics quickly. This was a requirement for both the data fusion and data structuring operations which formed feature extraction. It is also required for applications such as pavement condition diagnosis where features have to be compared and related spatially in order to draw conclusions. The approach used in this research was to generalize the quadtree by merging the single quadtrees corresponding to individual properties and features of an area into one multi-layer quadtree (Figure 4-2). Thus each node contained information concerning the state of the properties and features in the quadrant that the node represented, a key property of the multi-layer quadtree when implementing spatial set operations. A feature is spatially a subset of another feature if the node at which its state is "black" is a descendent of the node where the first feature's state is black or is the same node. In order for this to be true physically, the sources of surface data must be aligned spatially.

A multi-layer quadtree implemented as a single data structure is a unique approach, and it provided several advantages over implementing a group of single feature parallel quadtrees. For n-feature comparison the multi-layer quadtree requires only one traversal versus order *n* for the single feature quadtrees, therefore it is much faster for set operations. The multi-layer quadtree was more space efficient for *n* types of data, because it required only one set of pointers versus order *n* required for implementing the separate quadtrees. Grafting new data sources onto the tree was faster than creating new separate trees, because the memory allocation process was minimized. Those advantages vary in magnitude with the level of aggregation and dispersment of the source data, and were affected by the method of data storage at the nodes. Another advantage of the multi-layer tree was the support for queries at nodes for co-occurance and adjacency information, because the information resides in one unified structure.

The multi-layer quadtree results in some descendent nodes having slots which had been as black or white in predecessor nodes. This raises the issue of value inheritance. While such node slots do inherit their predecessor's value in operations, the multi-layer quadtree could leave them undefined in the structure. Because of this and because slots could exist unfilled before and after some operations, the multi-layer quadtree defined four slot states:

- **black** - the characteristic filled the area encompassed by the slot's node

- **white** - the characteristic did not exist in the area encompassed by the slot's node
- **grey** - the characteristic existed in part of the area encompassed by the slot's node
- **undefined** - no information was retained concerning the characteristic at this node

The state of black could have several values. For example rutting could exist as low, medium, and high. In most cases it makes sense to discretize continuous valued properties into a few representative ranges such as those for rutting, however wide ranges of black values are acceptable. The multilayer quadtree required the definition of two types of leaves, "virtual" and "real". Real leaves had no descendants and were at the bottom of the tree. Virtual leaves were defined with respect to a particular slot. If the slot had a state of black or white, then its node were a virtual leaf with respect to the characteristic that uses the slot. Virtual leaves could be intermediate nodes or real leaf nodes.

Further generalization of the multi-layer quadtree could be achieved by extending it to represent non-square pavement surface areas in a functionally continuous manner. In practice, functional continuity means that for a section of road perhaps as much as 1 km long, its representation should consist of a single unified data structure. Quadtree algorithms should work without modification over the entire structure. For a square area such as that selected for automated surface work, the structure would be a standard multi-layer quadtree with a single root.

4.4 Surface Characterization

Characterization changes the state of the surface representation in order to produce a useful description of the surface. The four basic operations of the characterization process are described below.

1. **data filtering** - linear and non-linear transformations,
2. **data reduction** - deriving a representative value from a set of data,
3. **data fusion** - combining two or more spatially concurrent datums into a new datum, and
4. **data structuring** - linking and integrating data.

Computer processing and space constraints along with the nature of the application affect how the balance of these operations were divided between the grid and the generalized quadtree. Generally, data filtering and reduction were performed most effectively on grid data, and data fusion and structuring were performed most effectively on the generalized quadtree.

The characterization operations could be grouped into practical classes. The first included operations on the grid to prepare raw sensor data for conversion to the generalized quadtree. The next included operations to convert grid data to the quadtree representation. Quadtree set operations formed a class, and adjacency and region labeling another. These classes of operations are described in the following sections.

First, in order to understand the general relationship between the grid, the multi-layer quadtree and characterization, a hypothetical example may be useful. To extract the feature "fatigue cracking" the characteristics rutting, strength, and cracking might be used. This information could be acquired in raw form as property data using range, deflection, and vision sensors respectively and reported as arrays of data mapped onto points of the grid. The grid data is processed and converted at appropriate levels of aggregation to the generalized quadtree representation. Conversion is a structuring operation that places the properties in the generalized quadtree as slot values in node data structures. The slot values are datums which are combined by a feature extraction algorithm composed of set operations into datums in a new slot which represents fatigue cracking. This process is data fusion. Because it is achieved at the highest level of aggregation possible it is extremely efficient. The structure of the generalized quadtree relates the fatigue cracking feature spatially to the other characteristics and to the pavement surface.

The objective of grid data processing is to prepare raw sensor data for conversion to the generalized quadtree representation. It should result in each datum indicating with as high a degree of confidence as possible the existence or non-existence of the characteristic it represents in the area it corresponds to. Aggregation to as high a level as is practical and useful is also part of the objective since the subsequent conversion operation is costly. Reduced data must be arranged in an array format with the same row:column ratio as the underlying grid and with dimensions equal to the grid dimensions divided by two to a power. Also, each datum must have a white or black state value.

Filtering is used to segment grid data and to extract features. Linear filters are implemented using convolution operators [Ballard 82, Castleman 79]. Thresholding and edge detection are examples. Edge detection has been used on video image data of pavement surfaces to segment datums possibly located along cracks since cracks form an edge between regions in an image [Haas 84]. Non-linear filters include, for example, median filters which are used to remove spurious data.

Skeletonizing is an operation that derives the skeletons of blobs in a black and white image. It is one method of reducing data as illustrated in the next chapter. Another is to divide the grid into areas corresponding to quadrants at some arbitrary level of aggregation and derive a summary statistic for each area. For example, transverse profile data may yield approximate range values over a 2m x 2m area from which rut depth values can be derived along the wheel path and averaged for a summary value. Alternatively, thresholded (black and white) video data can be summarized in small 1.6" square areas that have or do not have cracking based on the number of black pixels in each area.

Continuous valued properties such as rut depth must be discretized to yield conversion ready values. Rutting below a threshold value is not considered significant so areas where this is the case are labeled white. Otherwise rutting can be discretized into three levels of severity, each one a "black" state value.

Several distinct conversion operations are necessary to convert grid data to the generalized quadtree representation. Conversion from grid data to a quadtree can be performed using existing bottom-up and top-down methods. The multi-layer quadtree structure makes an additional operation necessary, that of "grafting" layers of grid data onto an existing quadtree structure. The strip quadtree requires its own unique root structure and related algorithms to direct the standard conversion operations. Conversion from generalized quadtree data to raster data is also required to display and print graphic results.

Algorithms for converting grid data to a quadtree were cited earlier. A very simple algorithm was been implemented in the model's kernel making use of the node object's split and prune functions. The algorithm was slow, however, and created a bottleneck in overall processing. Replacement in a future generation of the model would thus speed processing. The algorithm functioned recursively, splitting and pruning the tree as it visited each point in the grid array. The grid data level of the tree was defined as 0 for this algorithm. Otherwise, the generalized quadtree treated the root node as level 0. The grid data to quadtree conversion function started at the root node. The algorithm is summarized in Figure 4-3.

```
maintain lower right array coordinates of the current node's quadrant, and
maintain current level of the tree for each recursion

IF  the level is 0
        insert the array cell value in the current node

ELSE
        split the node and call this function for each sibling

        IF  the siblings are all black or all white
                prune them, and
                put their value in the parent (current) node

        ELSE
                put the grey state value in the parent node

return
```

Figure 4-3: Grid Data to Quadtree Conversion Algorithm

This algorithm works equally well for multiple values of black. However black nodes with different values were not merged. A flexible slot-state predicate object was implemented that allowed a current black state value to be specified for algorithms acting on the tree. With the predicate object, a node could be identified as black, white, or grey with respect to a particular slot. The predicate object was also used to identify virtual leaves.

The generalized quadtree was converted to raster data for display and printing. The model's software kernel facilitated conversion of quadtree data from the root or any branch root level down any number of levels, or for any section along a strip quadtree up to a maximum length determined by the implementation. It also allowed graphic overlay of slot layers, and black states for a single slot using grey levels, and it could write quadrant division lines corresponding to any depth. The arguments for specifying output were described in the application developers guide to the kernel [Haas 90b].

The model implements two binary set operations, namely union and intersection, from which more complex set operations could be constructed (Figure 4-4). Each operation acted on two slot-value pairs building the result of the operation in a "target" slot and returning a value for the area of the result. This requires a traversal of the multi-layer quadtree comparing slot values at nodes. Each operator makes a full pass over the tree.

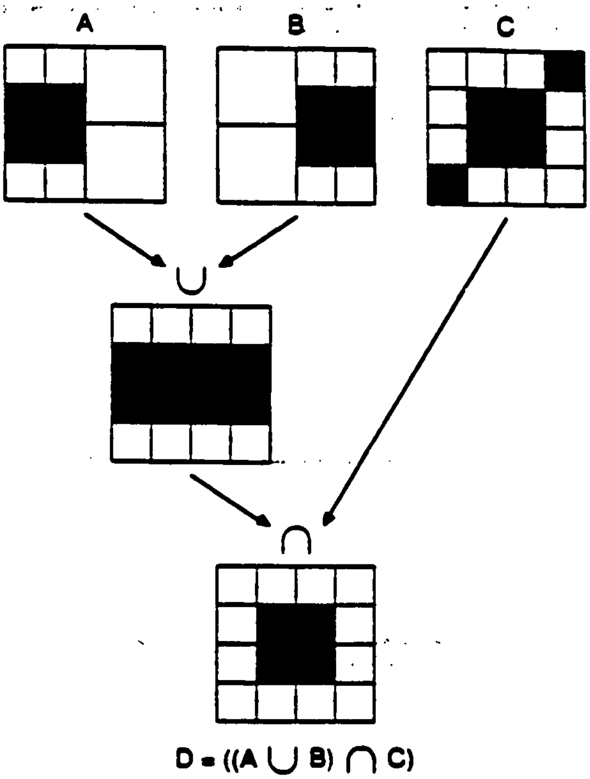


Figure 4-4: Set Operations

Storage and Retrieval of The Model

There are many well developed methods of storing quadtree information which use an encoding scheme to order and store only the leaves of the tree [Shaffer 86]. In a multi-layer quadtree however, because virtual leaves exist, the whole tree must be stored. A simple encoding scheme was implemented that dismantled and reconstructed the generalized quadtree for any subtree or section along the strip quadtree.

A function recursively traversed the quadtree writing the slot values for each node into a file. Each node's slot values were followed by a control character indicating whether the node was a physical leaf of the tree or an internal node. After this, the tree was pruned from the root node. The process was reversed using a simple recursive function for reconstruction.

4.5 Handling Uncertainty

Sources of uncertainty arise from instrument sensing errors, imprecision and slippage in mechanical components, missalignment, data registration, data fusion operations, and propagation of error through the aggregation and abstraction involved in the surface model's characterization processes. Sensing errors can sometimes be reduced by using more sophisticated technology such as an infrared range finder with built in automatic gain control. For imaging, increased precision and reduced exposure time can be achieved with special video cameras and lighting. In both these cases, there is a clear trade off between cost and quality of data. In most cases, some error is acceptable and unavoidable.

Incorrect registration of data can occur because the scanning mechanism exhibits mechanical imprecision. In a work system such as an automated pavement maintenance machine, there is potential for error due to vibrations and relative shift between different sensor components. Also, environmental factors such as dust, wind, heat, shadows, and even noise can affect sensor performance. Ultrasonic range sensors are especially sensitive to many of these factors.

Misalignment of scans results when the different sensing systems are not rigidly connected to each other, when the areas scanned are not precisely the same dimensions in terms of their outer boundaries, or when there are timing irregularities. In laboratory experiments, the first two problems were reduced by clearly marking the boundaries of the test scene, and adjusting scanning mechanisms to those boundaries. In practice, the scanning systems can be engineered for alignment, but there will always be some error. In a field system, it is therefore useful to have a calibration procedure for alignment. Alignment error can also be the result of cumulative round off errors in sensor scanning mechanism control algorithms.

The sources of error discussed so far are impossible to eliminate, and difficult to compensate for or to model [McNeil 89]. The pavement surfaces model provides mechanisms for handling some types of uncertainty in the form of its ability to work at different levels of aggregation and to associate confidence measures with characteristics using additional slots. There has been significant research in the areas of modeling and reducing sensor error in robotic and automated sensing systems. For instance, methods for reducing error have been developed that "fuse" data from multiple time and position displaced scans and from multiple sensors. There is a large body of literature in this area [Durrant-Whyte 87, Durrant-Whyte 86, Elfes 88, Duncan 87a, Duncan 87b, Faugeras 86, Luo 88, Allen 84, Chiu 86, Crowley 87, Huntsberger 87, Henderson 87, Richardson 84]. However it is probably not practical for pavement surveying and automated pavement work tasks to make multiple scans with the same sensor of the same general surface area, since such redundancy is not feasible in a field operational system because of real time constraints. Changing position would likely not create any useful new perspectives either given the generally two dimensional nature of the surface unless very precise stereo vision processing was applied. Modeling uncertainty within the context of the pavement surfaces model is worthy of further investigation however.

5. Software Implementation for Perception and Control

5.1 Introduction

Implementation of the pavement surfaces model should meet two key goals. First, the software that constitutes its core functions and data structures should be as generally applicable in practice as the model is in theory. Second, the software should be efficient and well designed. Good software design stresses modularity and independence, sometimes at the expense of efficiency.

Object oriented program design encourages data hiding and polymorphism for independence. Polymorphic functions perform the same task on objects of many different types. Data hiding is used to protect data from unintended damage by unrelated or unauthorized functions. Object oriented program design also encourages modularity. Because of this, an object oriented design methodology was followed. Top down design was also used to a limited extent. A design document was produced as a guide for coding and testing the kernel software [Haas 90b]. An alternative implementation approach that has some merit is to use a functional programing language. An interesting functional programing approach for quadtrees is described in [Burton 89]. It differs from C++ in its language's additional capabilities of higher order functions and lazy evaluation. However the language used is not well distributed or easily linked to other software packages, and the code produced is generally inefficient.

A C++ program is composed of objects which are instances of classes. A class includes data and the functions which may operate on and modify this data. A class is also considered a data type. The data in a class is normally hidden from other classes. The result is that the internal data structures may be modified without affecting other classes or parts of a program. Classes can be derived from other classes, inheriting their functions and data elements while at the same time adding new functions and data elements. The ability to declare class member functions to be virtual allows external functions to treat derived versions of a class as a single type. Virtual member functions are different versions of the same function specific to each derived class of a base class. This is an implementation of the principle of polymorphism. In practice it means that general quadtree operations can be implemented that operate on different versions of quadtree node objects as if they were the same type. So an implementation quadtree constructed with a derived quadtree node type is treated by the kernel functions as a generic quadtree type.

A key to good object oriented design is defining classes that correspond to objects in the real world such as an employee record which has data that can be updated, operated on, and accessed. Specific types of employees such as a "manager" can be derived from the base class type "employee" inheriting its elements and adding new ones specific to type manager including possibly a list of "employee" elements. The pavement surfaces model kernel is composed of a set of base classes which can be used directly for applications or from which application specific classes can be derived. The base classes correspond to objects in the model that have been described. They are described in the following paragraphs.

(a) the quadtree node class: **qnode**

The qnode includes the pointers that implement the quadtree structure. It also includes virtual declarations of modification and access functions that are redefined in derived qnode classes. The size of the qnode in memory is critical, so member functions must be limited. Slots are declared in the derived nodes and their implementation has a critical impact as well. They can be implemented as a fixed length vector or a variable length linked list. A variable length linked list minimizes the number of slots by making a slot at a node only if it is required. However because of the cost of pointers, depending on the number of layers and the scatter of the source data, vector implementation can be more memory efficient. It is also simpler and it permits faster data access.

(b) the grid view class: **view**

The view class has a 512 x 512 array that acts as a buffer for the grid data. It is a "view" of the grid. It includes all the conversion functions between the grid and quadtree representation, and it includes functions that read from and write to grid and raster data files. A derived class is implemented for the generalized quadtree called **strip_view**.

(c) the quadtree base tree class: **tree**

The tree class includes a pointer to the root qnode object of the quadtree. It includes several basic member functions for acting on the quadtree that are useful for applications. The members include functions that dismantle to and rebuild the quadtree from ascii files. The connected component labeling algorithm is also implemented as several private members and one public member function. A few other utility functions are also implemented. A derived class is implemented for the generalized quadtree called **strip_tree.**

(d) the traverser class: **traverser**

The traverser acts as an independent device somewhat like a cursor that can be moved over the tree structure in an arbitrary pattern. Among its members, it includes neighbour finding functions that use the qnode's parent pointer. No other kernel object uses the parent pointer, so if the traverser is not being used it may be worth removing the parent pointer from the qnode for some applications.

(e) special classes: **q_lookup_tables, equivalence_pairs, slot_predicates**

These classes provide necessary support. The q_lookup_tables class implements special predicate and lookup functions required for some of Samet's algorithms. The equivalence pairs class implements a set of functions and tables required for connected component labeling. The slot_predicates class provides predicate functions for slot states that take slot indices and values as arguments and return true or false. The values for the states can be assigned. This class is used by both the view and tree class members.

(f) the generalized quadtree's root structure: **strip_roots_index**

The generalized quadtree requires a root structure for a strip shaped area as opposed to a single root node. The strip_roots_index class implements the base elements of the root structure. A derived root structure is required for each application much as a derived qnode is required, but the kernel is polymorphic with respect to this class, because derived qnodes are used to flesh out the strip quadtree in the derived versions. These derived classes are the ones actually passed to the other kernel objects using base class pointers. The "strip" quadtree can be declared to have any length. For a particular application, standard lengths are used and the data for a whole pavement section length is encoded into or decoded from a single file.

The base class relationships to each other are illustrated in Figure 5-1. The connections indicate which classes either declare objects of other classes within their own objects or call members of other classes.

For a particular application some of the connections may not exist or are replaced by connections between derived versions of the classes. Also, applications programs may include new classes. An example of the relationship between base and derived classes is illustrated for the quadtree node classes in Figure 5-2.

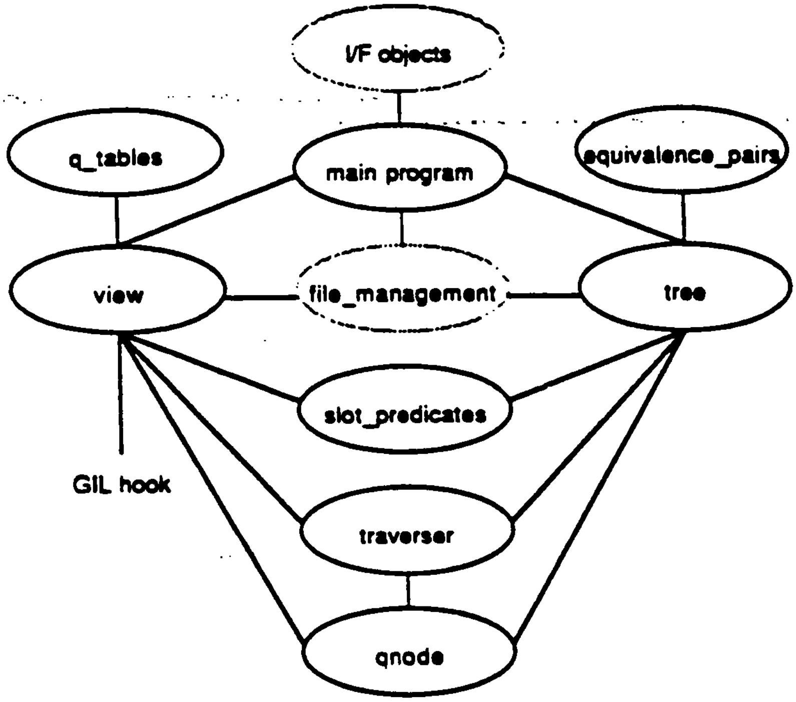


Figure 5-1: Kernel Class Connections

Figures 5-3 and 5-5 illustrate the progression of software development for the applications described in this report. The kernel classes are illustrated in Figure 5-3, and new and derived classes for each application are shown in Figure 5-5.

Several implementation improvements should be considered. Implementation of the model on a PC would increase its availability and its utility, but to do this, memory usage would have to be decreased. Currently, the model is implemented on a SUN 3 workstation running Unix. The dynamic memory

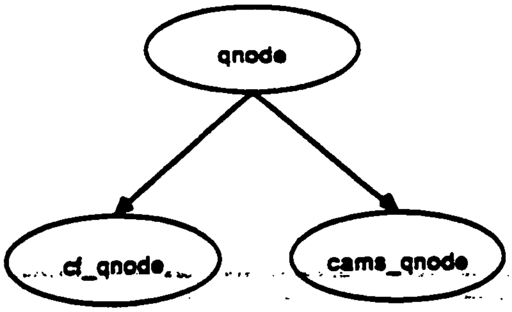


Figure 5-2: The Node Class and Derivations

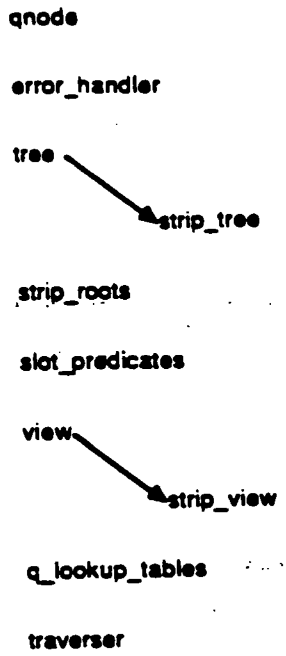


Figure 5-3: Kernel Classes

required by the generalized quadtree could be reduced by replacing its *qnode's* member functions with a separate class. However, this would increase the complexity of the kernel software. Memory use can also be reduced by converting at higher levels of aggregation. Dynamic memory allocation could be implemented more efficiently to improve the model's performance. Performance could also be improved by replacing a few of the algorithms, particularly the raster data to quadtree conversion algorithm. Faster alternatives exist. In addition, some filtering operations could be hardwired or run on special convolution processing hardware, thus increasing processing speed by as much as several magnitudes. Quadtrees are also a good candidate for exploiting multiple processor architectures since the kernel code consists largely of recursive independent function calls that could be allocated to separate processors.

In addition to the implementation issues discussed above, the efficiency of the generalized quadtree representation in practice is worth considering. The generalized quadtree nodes that are implemented in the GQL require 4 bytes per node pointer, 2 bytes per member function pointer, and 2 bytes per slot element. For many cases, one byte per slot element is sufficient. A generalized quadtree node with a single slot requires 30 bytes of memory. For each extra layer of data a slot must be added. At one extreme, adding a slot rather than generating a new quadtree for each layer can result in tremendous memory savings. This is the case where the data for each layer is in a checkerboard pattern. Such a pattern results in the largest possible quadtree for the layer of data. In this case, the separate quadtree for each additional layer would require 15 times more memory than that used by adding a 2 byte slot to the generalized quadtree and 30 times more memory than that used by a 1 byte slot. If the objects in each layer of data, however, are well separated spatially from other objects in other layers, then for two such layers approximately 1/15th more memory is used by the generalized quadtree than if the quadtrees for each layer were separate structures. For 15 such layers as much as twice the memory would be used for the generalized quadtree than what would be used for 15 separate data structures.. Again, this is an extreme case.

Whether the generalized quadtree results in memory savings in practice depends mostly on the distribution of the data in each layer and partly on each layer's resolution before conversion. Empirical analysis is required to estimate the actual magnitude of any savings using the generalized quadtree. Most of the applications considered in this report however should experience a decrease in memory usage. In any case, the cost of a small increase in memory usage is far outweighed by the benefits of faster and simplified set operator functions and the fact that grafting onto a largely pre-existing structure is faster than creating a new quadtree structure, because memory allocation procedures are not required where the grafted data has quadrant divisions corresponding to the current structure.

5.2 Raster Processing

The perception software is derived from the model's kernel. While some kernel functions such as conversion are used directly, others become the basis of application specific software. For instance, a new crack filler quadtree node is derived from the kernel's base class quadtree node (Figure 5-4). Figure 5-5 illustrates the progression of software development for the crack filler application using the kernel.

```
class cf_qnode : public qnode{
        short int         slot[4];
        // 0  -  VIDEO - video value for quadrant
        // 1  -  RANGE - range     "     "      "
        // 2  -  CC    - connected component label
        // 3  -  FLAG  - general flag for some operation
public:
                   cf_qnode();       // set all to zero
                   // destructor not necessary
        void              split();
        void              prune();
        short int         get(int);
        void              put(short int, int);
};
```

Figure 5-4: The Crack Filler Quadtree Node Declaration

The steps involved in crack perception are illustrated in Figure 5-6. The first step is to acquire a digitized video image of the pavement work area (Figure 5-7 (a)). The image is segmented by binarizing it (Figure 5-7 (b)). This means that pixels with a grey level value below a threshold value are labeled black and those with greater or equal values are labeled white. The threshold is set automatically but would be manually tuned by the truck driver for the local pavement section conditions.

The next step is to remove noise from the resulting binary image. This is done using a nonlinear filter implemented using a convolution operator. If the center pixel is in a local group of pixels where there are fewer than a threshold number of black pixels, it is made white since it is considered spurious, but it is

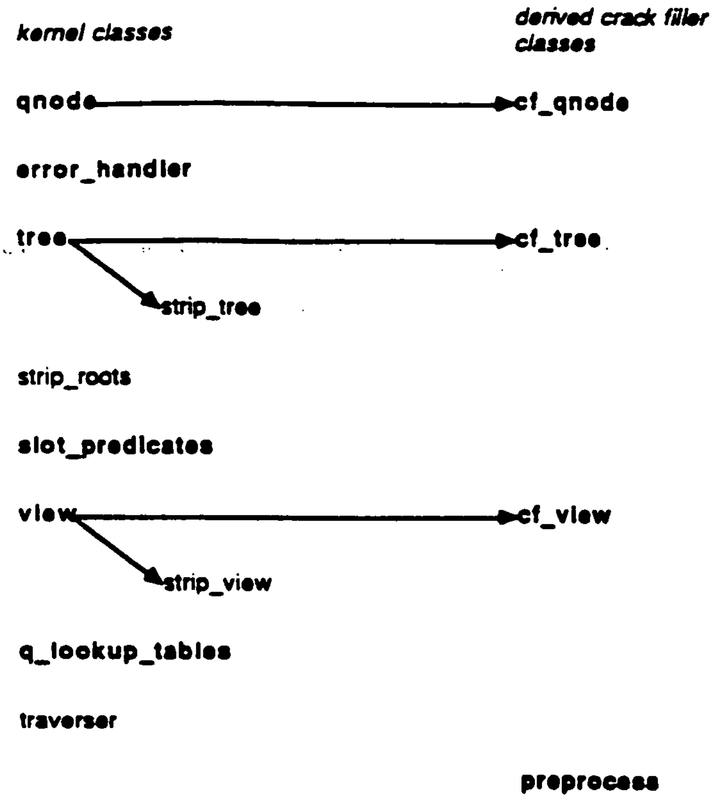


Figure 5-5: Crack Filler Perception Software Development

made black if the threshold number of black pixels exist. Thus with the proper threshold value such as 5 or 6, most black noise is removed, edges of black boundaries are smoothed, and white pixels in the midst of a black region are made black. The resulting image still includes many objects that are not cracks (Figure 5-8 (a)).

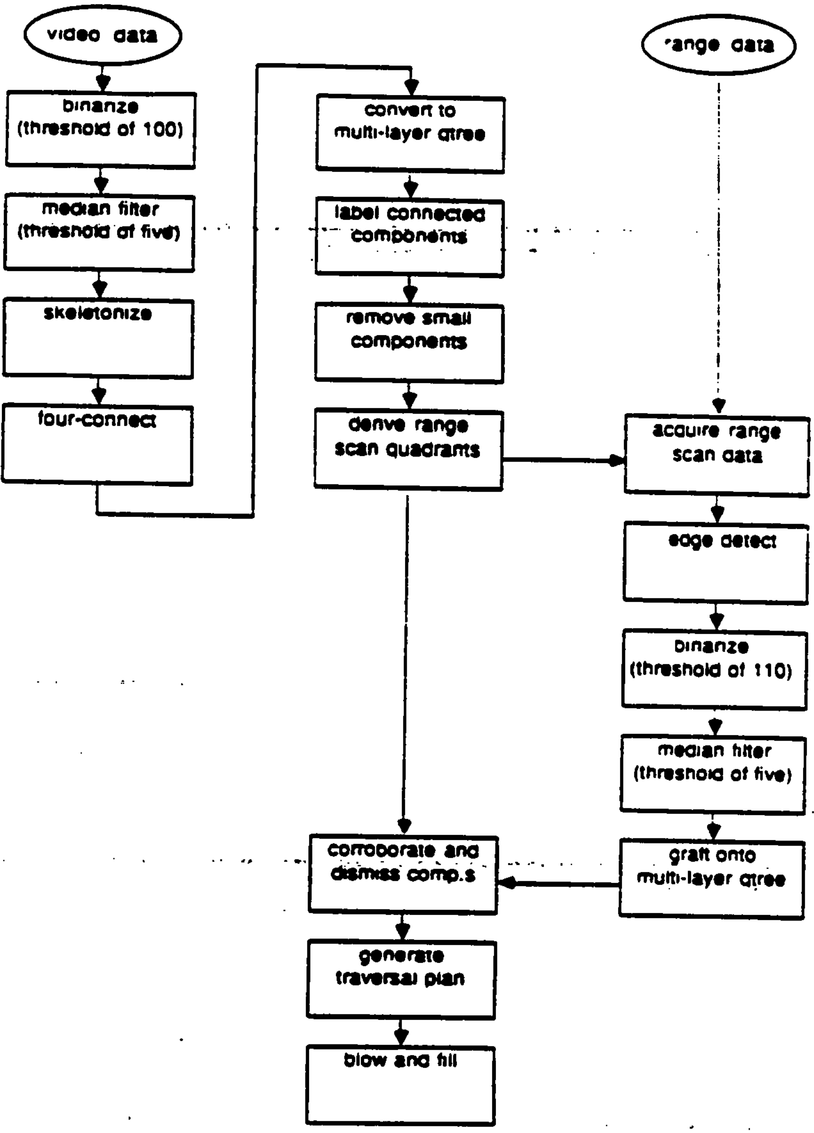
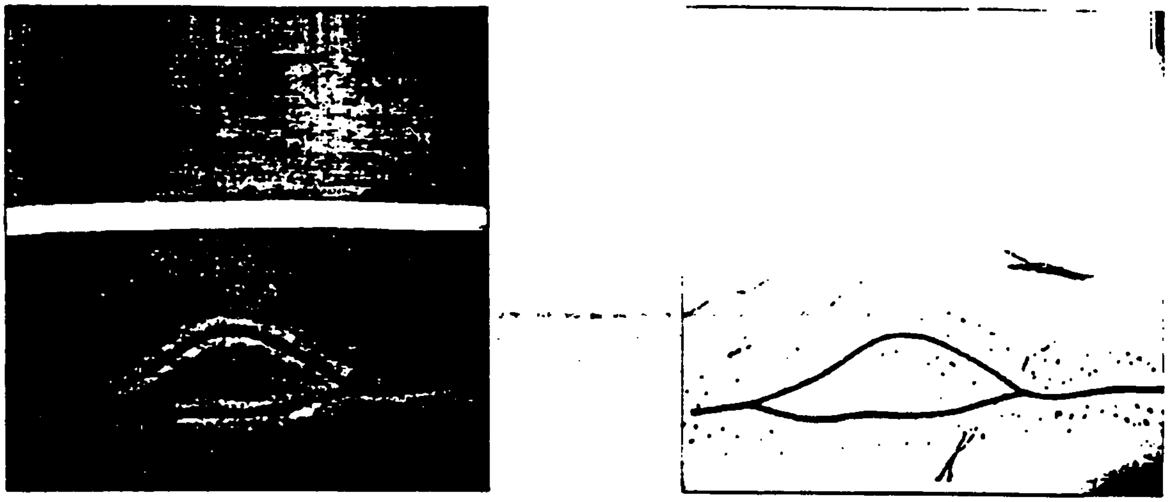
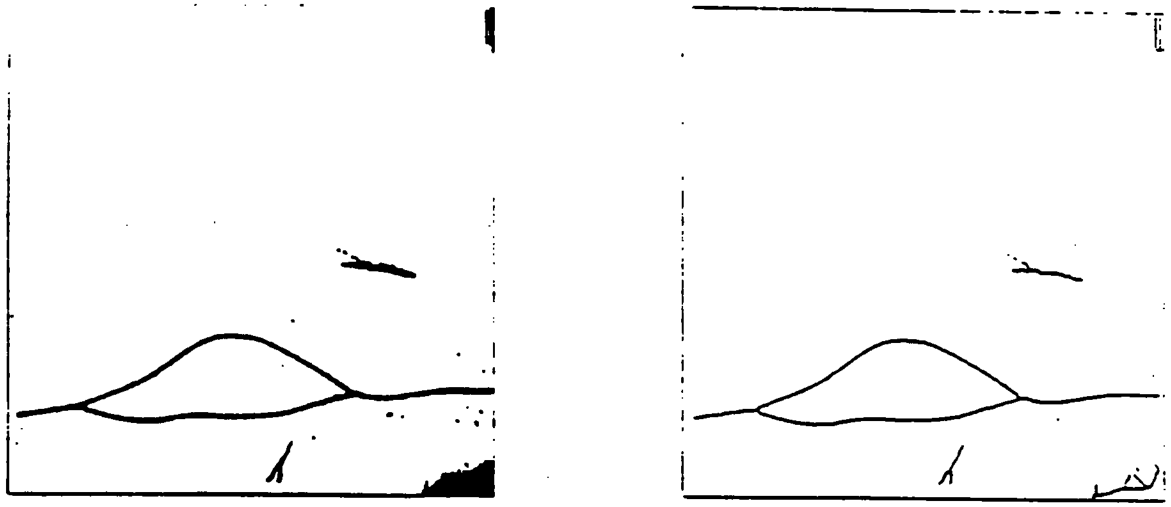

video data
binarize (threshold of 100)
median filter (threshold of five)
skeletonize
four-connect
convert to multi-layer qtree
label connected components
remove small components
derive range scan quadrants
range data
acquire range scan data
edge detect
binarize (threshold of 110)
median filter (threshold of five)
graft onto multi-layer qtree
corroborate and dismiss comp.s
generate traversal plan
blow and fill

Figure 5-6: Perception Steps



(a) (b)

Figure 5-7: Original and Binarized Video Images of Test Scene 1



(a) (b)

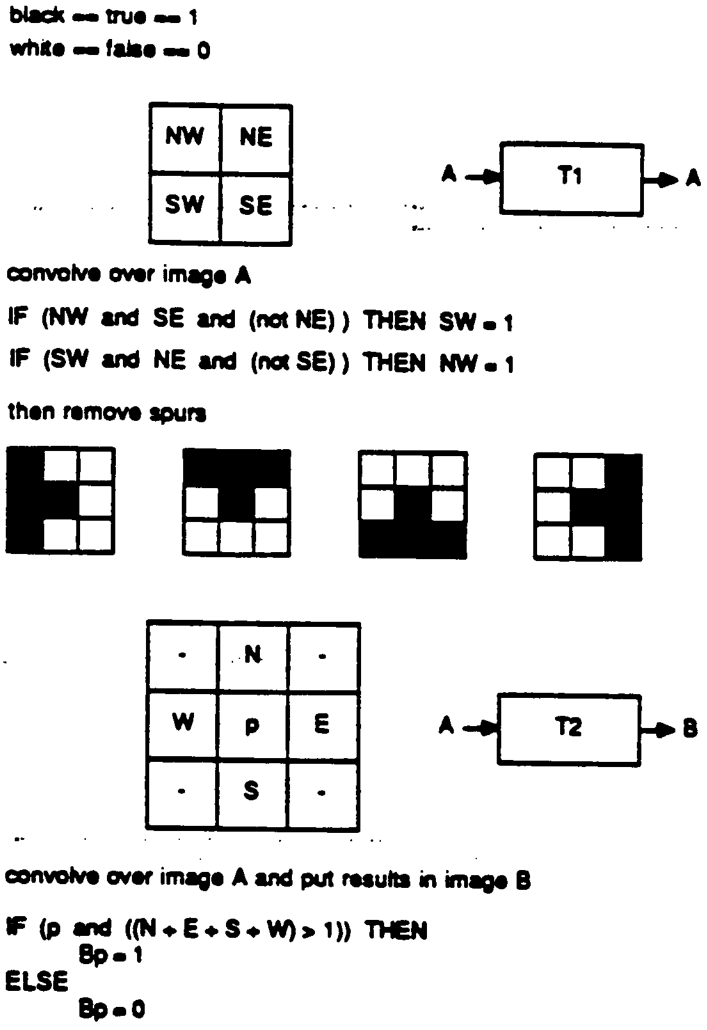
Figure 5-8: Filtered and Skeletonized Video Images of Test Scene 1

5.3 Quadtree Processing

To build a traversal plan of those objects that are eventually determined to be cracks, ordered lists of coordinate pairs along the objects' central or median axes must be derived. Strings of pixels can serve as ordered lists of coordinates. In order to derive these strings, an algorithm that thins image objects to something that resembles their skeleton can be used. For the crack objects, the skeleton normally corresponds to their central axes, so it serves as a useful "road map" for the blowing and filling operations. The crack skeleton (Figure 5-8 (b)) may however include side spurs resulting from rough edges on the original object boundaries, something which the prior noise filtering is useful for reducing. The noise filter reduces loops by filling in spurious white pixels in black regions. While the crack spurs complicate matters, the spurs and branches formed on spurious image object skeletons become inconsequential as a result of subsequent processing. A rigourous medial axis transform exists for the quadtree [Samet 83], but it can result in double quadrant thick axes that create very difficult interpretation problems for the traversal plan generation algorithms. The algorithm implemented for this application is taken directly from [Zhang 84] with corrections from [Lu 86].

An important operation is required next. The skeletons produced may be connected in some places only by their 8-neighbours (only diagonally). The kernel's connected component labeling algorithm that works on the quadtree representation only connects quadrants that are 4-neighbours. Extension to 8-neighbour connected component labeling is more costly, and potentially more complex crack network representations would ensue later in the perception process. An algorithm was therefore developed which is applied to the skeletonized image in the form of two logical operators convolved over the image that convert 8-neighbour connections to 4-neighbour connections while taking care not to create two pixel thick segments. The second operator removes spurs that would otherwise be interpreted as graph edges later. Its implementation ignores the corner pixels in the 3 x 3 convolution matrix thus increasing the efficiency of the operation at the expense of cutting off the ends of pixel strings. The operators are illustrated in Figure 5-9. The results are illustrated for test data in Figure 5-10. Figure 5-11 illustrates the results on the test scene objects' skeletons. The 4-connecting algorithm may be of general use to quadtree applications involving images composed primarily of lines.

The skeletonized and 4-connected image is still in grid data format. At this point, it is converted to the



```
black == true == 1
white == false == 0

convolve over image A
IF (NW and SE and (not NE)) THEN SW = 1
IF (SW and NE and (not SE)) THEN NW = 1

then remove spurs

convolve over image A and put results in image B
IF (p and ((N + E + S + W) > 1)) THEN
        Bp = 1
ELSE
        Bp = 0
```

Figure 5-9: Algorithm for 4-connecting

multi-layer quadtree representation filling the raw image data slot. Since the grid data is an array of 512 x

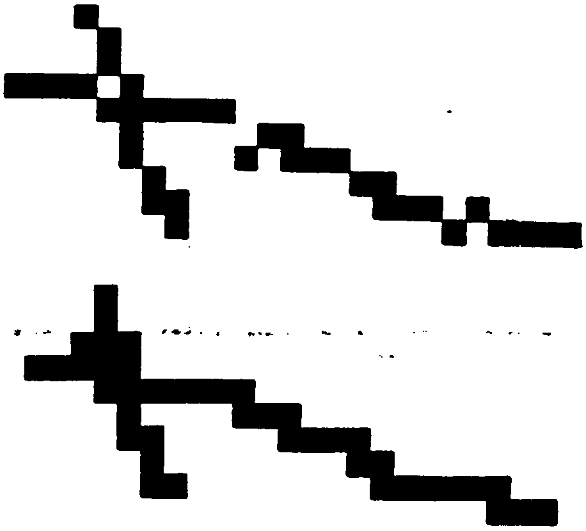


Figure 5-10: 4-connecting on Test Data

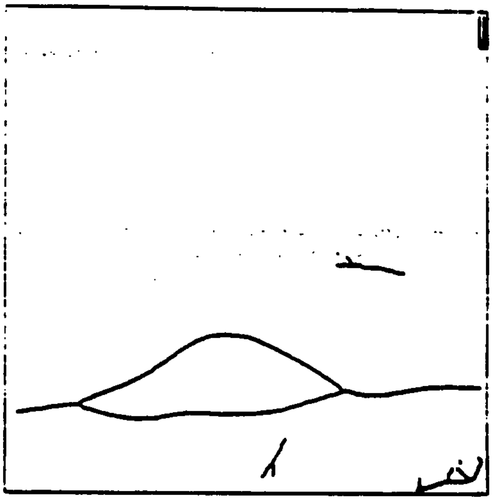


Figure 5-11: 4-connected Video Data of Test Scene 1

512 pixels, the resulting tree is up to nine levels deep. Figure 5-12 (a) illustrates the quadrant divisions down six levels overlayed on the binary video data. Each image object is then identified and labeled

using the connected component labeling algorithm described earlier. Both these procedures are implemented using the model's kernel. When the connected component labeling is done, each black image node has an object label number in the label slot. Also, the size of each object is stored in a look-up table accessed by label number.

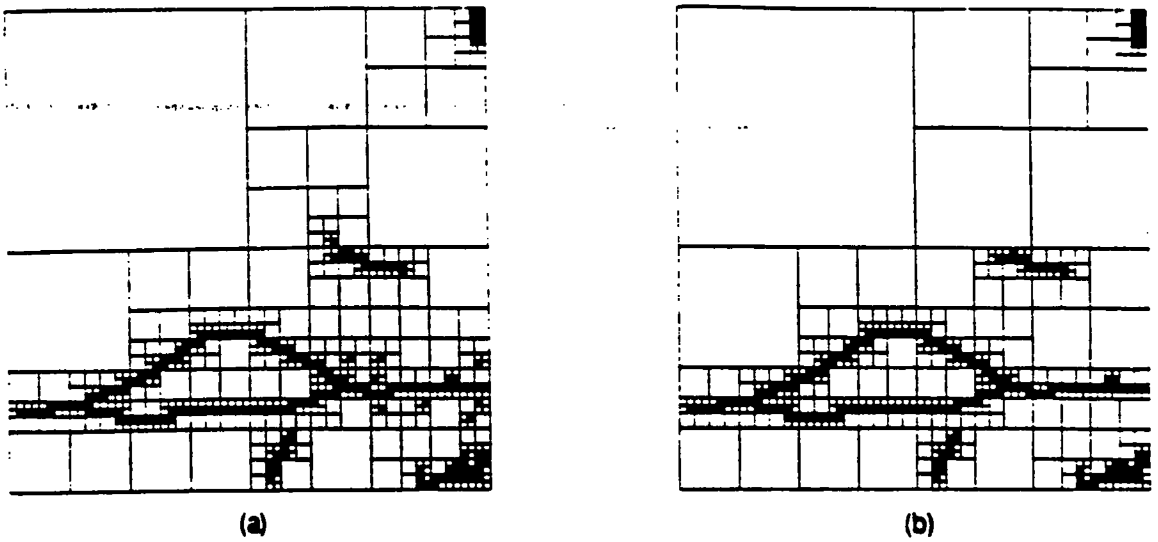


Figure 5-12: Quadtree Images for Processed and Pruned Video Data of Test Scene 1

Figure 5-12 (a) indicates that many spurious objects will still exist in the surface representation at this point. Many of the objects can simply be removed based on a size threshold. A kernel function is used to traverse the tree and prune those branches corresponding to objects below the threshold size (Figure 5-12 (b)). The result is that both the tree size (computer memory used), and the work required for subsequent algorithms are reduced. A concern is that if the crack skeletons are fragmented, significant portions could be removed by this procedure. Small gaps can be bridged at the traversal planning stage, but if small component removal creates large gaps with too high a percentage of images, then it need not be executed.

One of the purposes of integrating the video data into the multi-layer quadtree first is to use it to determine areas of interest in the scene. Areas are interesting if there are vision objects in them. Time is saved in the overall crack filling cycle by range scanning only the areas of interest, because range scanning is a time consuming operation. It requires physical traversal of the scene using the xy-table. The basic range scanning pattern implemented is a raster scan pattern, because it has the advantage of simple table control requirements and it facilitates precise grid registration. The quadtree is a useful tool for breaking down the scene into areas of interest that should be range scanned. A traversal of the tree down to a specific level will yield white quadrants that should not be scanned and grey and black quadrants that should be scanned (Figure 5-13). In a field system, the processed video data would be represented at the third level of aggregation using the quadtree representation, and the resulting black quadrants would be scanned for range data while the white quadrants would be ignored. For the laboratory experiments, full range raster scans were acquired as an interim measure to simplify the development process.

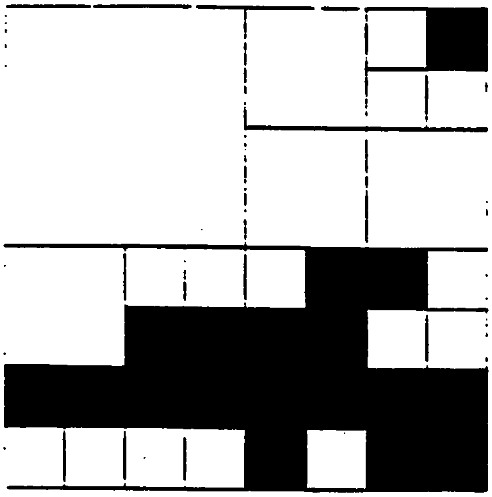


Figure 5-13: Quadtree Image of Area-of-Interest Quadrants for Test Scene 1

The range data must be processed at the grid data level into an array of values that represent the existence or non-existence of cracking with the highest degree of confidence possible. The original range data for the first test scene is presented in (Figure 5-14 (a)). A Laplacian filter is applied to enhance the routed cracks and eliminate the low frequency noise associated with gradual changes in elevation in the scene with respect to the table frame (Figure 5-14 (b)). The data is then thresholded (Figure 5-14 (c)).

Finally, the noise is filtered out prior to sending the range data to be grafted onto the generalized quadtree (Figure 5-14 (d)).

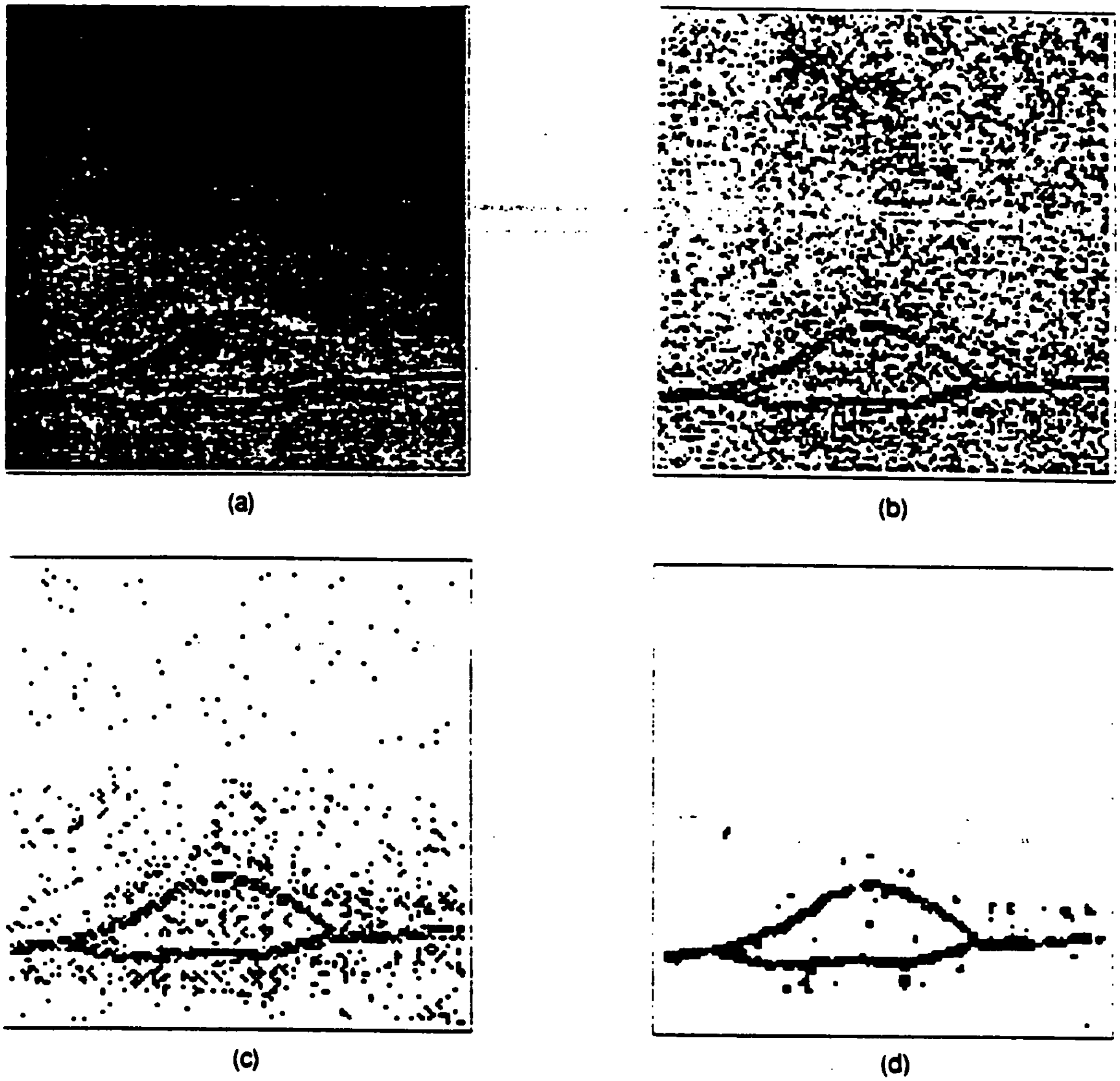


Figure 5-14: Images of Test Scene 1 Range Data Preprocessing Steps

Once the range data has been prepared, it is grafted onto the multi-layer quadtree using the kernel function for that operation (Figure 5-15 (a)). The range information is then used to corroborate or dismiss the vision objects. The percentage intersection of range black quadrants with each vision object is calculated. Vision objects with corroborating range data above a threshold percentage are retained while those below are removed from the quadtree. The remaining objects are considered the best estimate of the existence of cracking (Figure 5-15 (b)). An alternate approach to identifying routed cracks would have executed a pure intersection operation on raw vision and range information after both were incorporated in the multi-layer quadtree. Then connected component labeling would be applied to the result of the intersection operation. While this could have been done with kernel functions alone, the advantage of the connectivity of the vision data might be lost. That is, the vision objects that do correspond to cracks represent them well, and there is no point breaking them up with noisy range data.

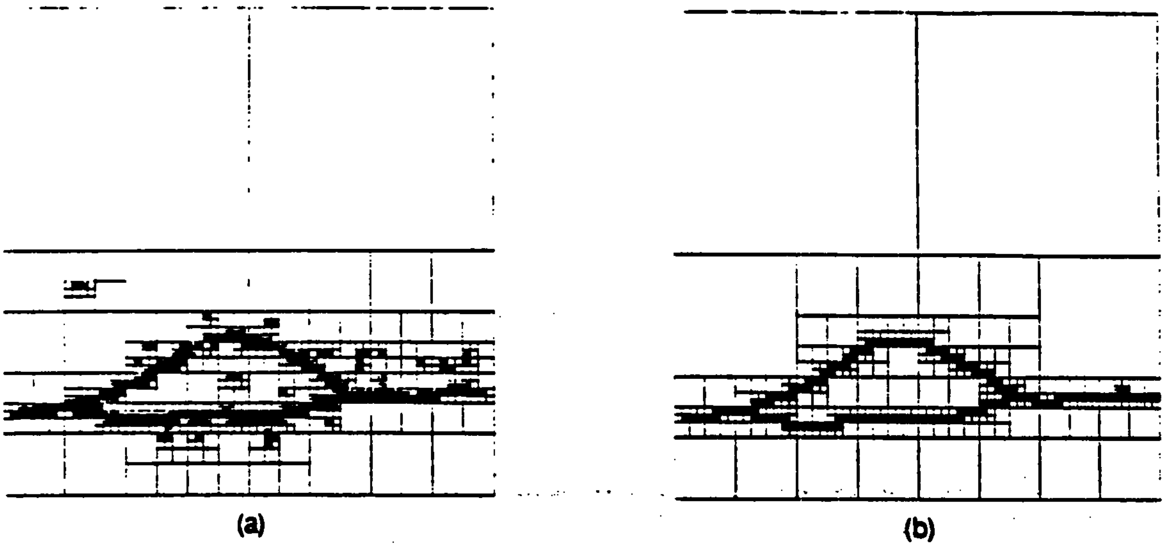


Figure 5-15: Quadtree Images for Range Data and Confirmed Crack Objects of Test Scene 1

5.4 Trajectory Processing

To derive a traversal plan of the corroborated routed crack network, it is useful to build a graph representation of the network. The graph representation provides an abstraction on which shortest route algorithms can operate (Figure 5-16). The graph is constructed of generic graph elements. An ordered traversal of the tree yields one or more linked networks of graph elements that correspond to the routed cracks. The networks are then traversed and the elements classified as edge or node elements. Associated with each node pair is a linked list of edge elements. The nodes and edges are cross indexed in respective table structures. Nodes can have one to four edges incident on them. The limit of four is set by the 4-connecting algorithm applied earlier. This restriction simplifies matters considerably. The spurs described earlier can be dealt with at this point in a number of ways including a short branch removal algorithm or by ignoring very short branches when the traversal plan is generated. A traversal algorithm is used to plan a path through the network of cracks. It includes a variation on a shortest route algorithm [Peters 90a]. From the path plan, movement and actuator commands are compiled and then sent as a list of instructions to the xy-table mechanism control subsystem [Peters 90b, Peters 90c].

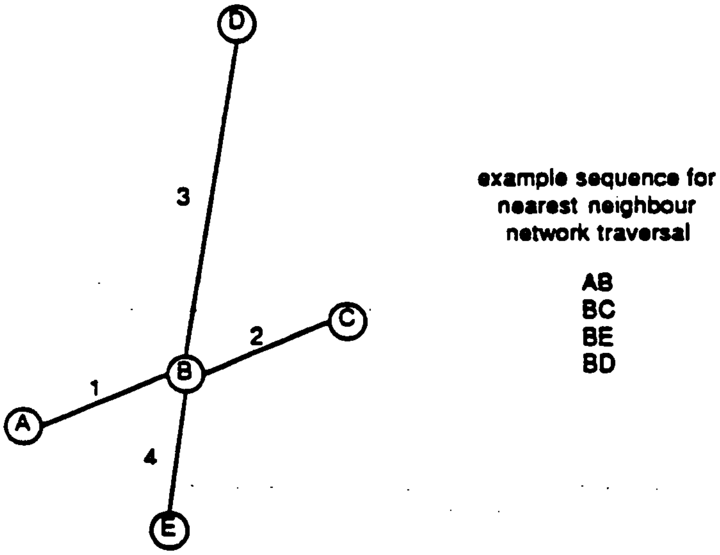


Figure 5-16: Graph Representation for Test Scene 2

6. Test Scene Perception and Processing

Three test scenes were constructed in order to develop and then verify the crack filler perception subsystem. The scenes were constructed of 2 inch thick polyethylene sheets. The routing groove dimensions were 2 x 2 cm (0.8 x 0.8 in), and cracks averaging 0.5 cm (0.2 in) in width run along the center of the grooves. The scenes were painted as realistically as possible with noise and spurious objects included. The vision scans were taken with the scene oriented perpendicular to the ground which allowed precise orientation of the video camera with respect to the scenes. A 2 x 2 m (6x6 ft) square work area was defined on each scene. Its bottom left corner is the origin for the grid coordinate system. The video frame grabber (High Res Technologies HRT512-8) acquires square pixels in a 512 x 478 array. The remaining 34 rows of pixels were filled in at the top of each scene. Each video pixel corresponds to an area on the scene surface of 3.9 x 3.9 mm. (0.2x0.2 in).

The processing results from the first scene were used to explain the perception procedures in the preceding chapter. The results from the remaining two scenes are illustrated here. The same filter parameter values were used for each scene, since it is an assumption of the automated crack filling system that only the threshold value will have to be manually tuned, and tuning should only be necessary as new pavement subsections are encountered. Figure 6-1 illustrates six of the steps of video data processing for scene two. The image data is binarized (b). The resulting binary image is filtered twice (c). It is then skeletonized and four-connected (d). Then it is converted to the quadtree representation (e) and components below a size threshold of 50 pixels are removed (f).

Figure 6-2 illustrates the range processing steps for the second scene. The second image displays the results of the Laplacian. The third displays the thresholding results. The fourth displays the results of filtering, and the fifth displays the quadtree divisions for the range data after it has been grafted onto the generalized quadtree.

Figure 6-3 displays the results of combining the range and vision data in the corroboration procedure. The remaining object corresponds to the routed crack in the test scene. The above results were repeated for an additional test scene in Figures 6-4, 6-5, and 6-6. The percentage of the verified routed crack object that exactly intersected the corroborating range data in each of scenes 1 (Figure 5-15), 2 (Figure

6-3), and 3 (Figure 6-6), was 90 %, 60 %, and 60 % respectively. In most cases, the error in these percentages was due to slight variations in the alignment of the video and range frames. Intersecting over a wider area would increase these percentages. Alteratively, cracks could be assumed to exist if some target percentage was met. These experiments used the latter approach.

In summary, the results from each of the three test scenes clearly illustrate the value of combining range and vision data for deriving the precise layout of routed cracks. The pavement surfaces model was useful for guiding data acquisition decisions, it provided a standard surface representation in which the data could be incorporated, and it provided kernel functions with which the perception steps were implemented. The experiments indicate that detection and selection of unrouted cracks for filling is feasible with a more sensitive range sensor and some modification to the perception algorithms. The experiments also suggest the potential value of combining range and vision data for automated pavement condition assessment. This potential is explored in the following chapter.

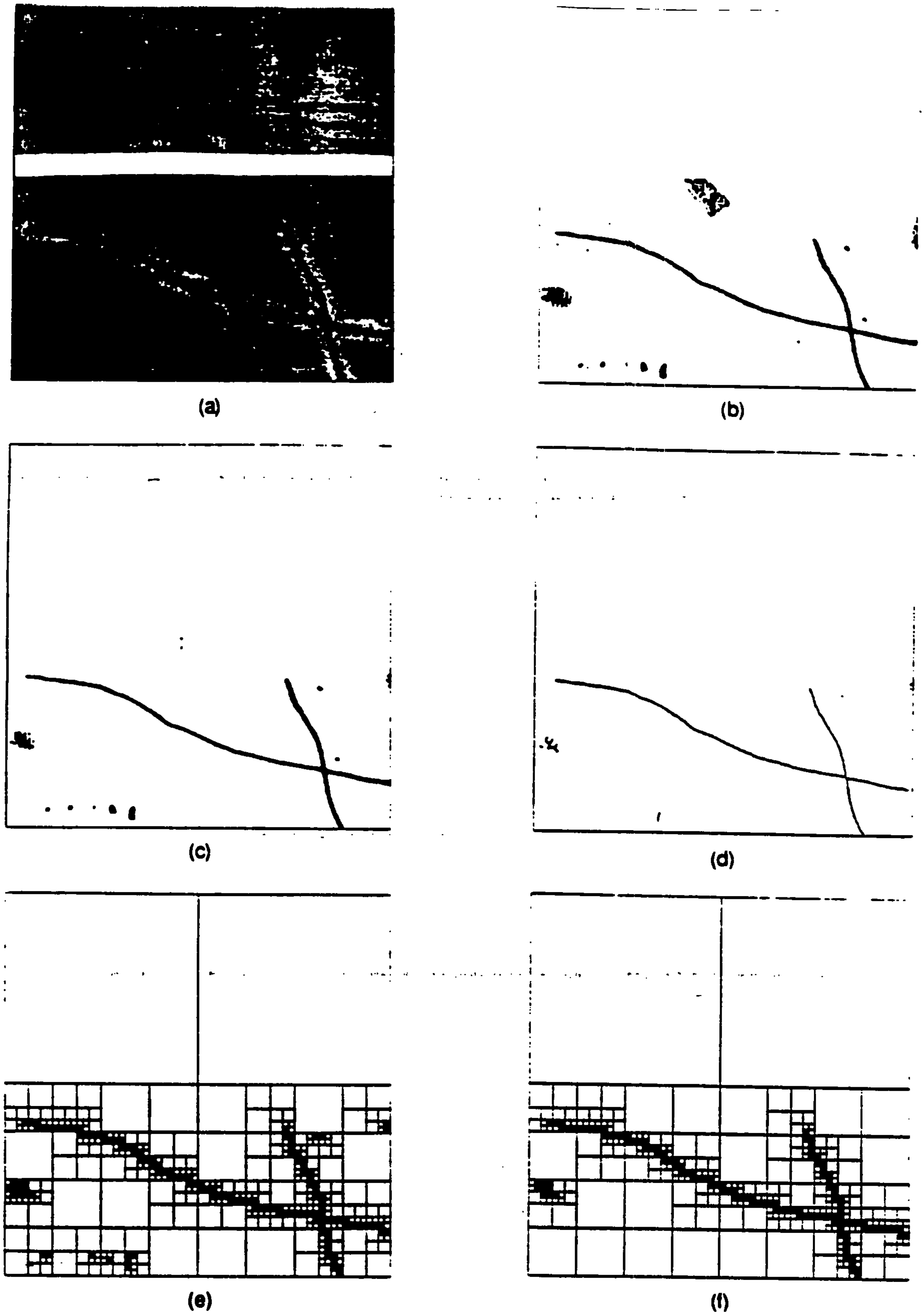

(a)
(b)
(c)
(d)
(e)
(f)

Figure 6-1: Video Data and Processing Results for Scene 2

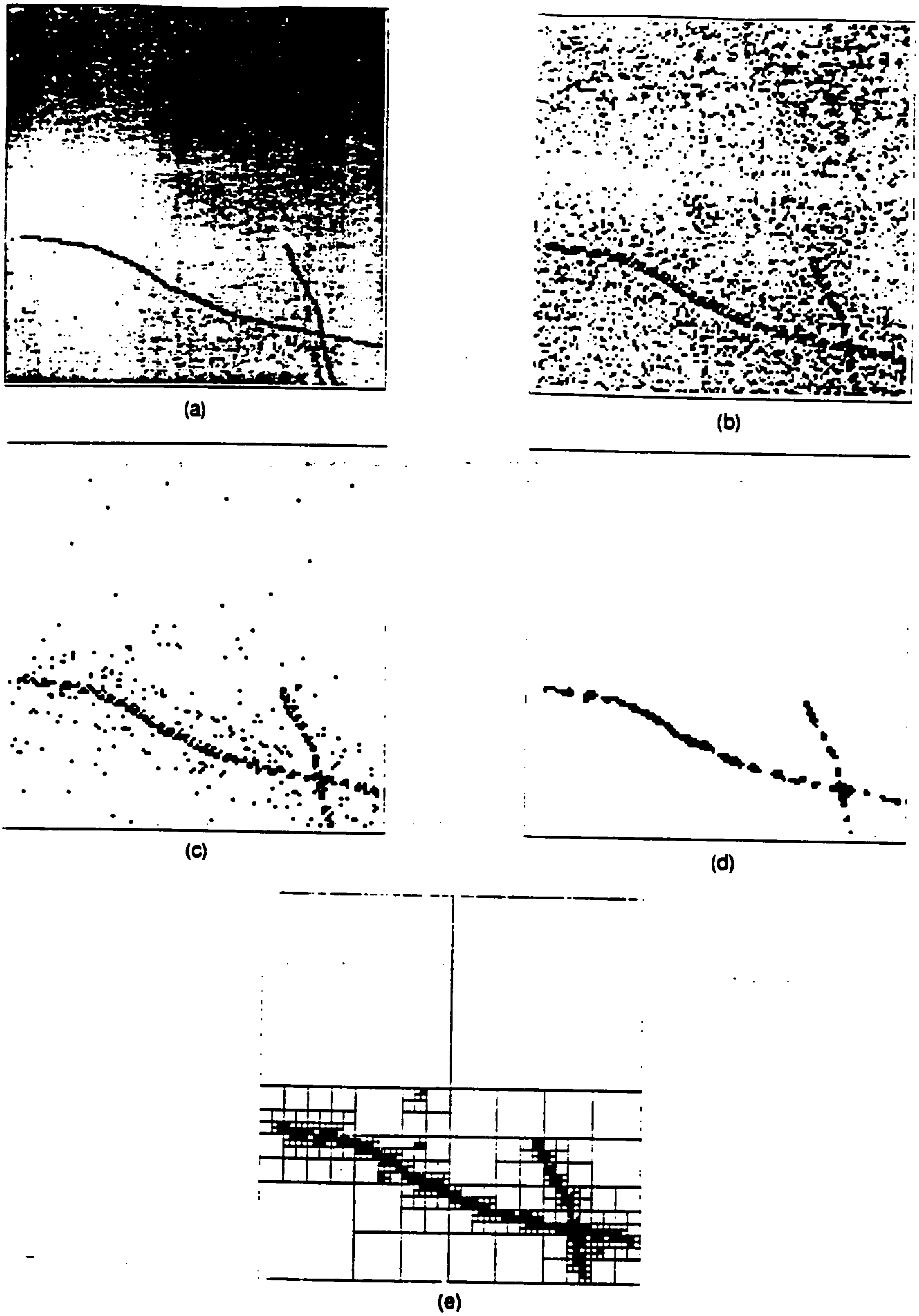


Figure 6-2: Range Data and Processing Results for Scene 2

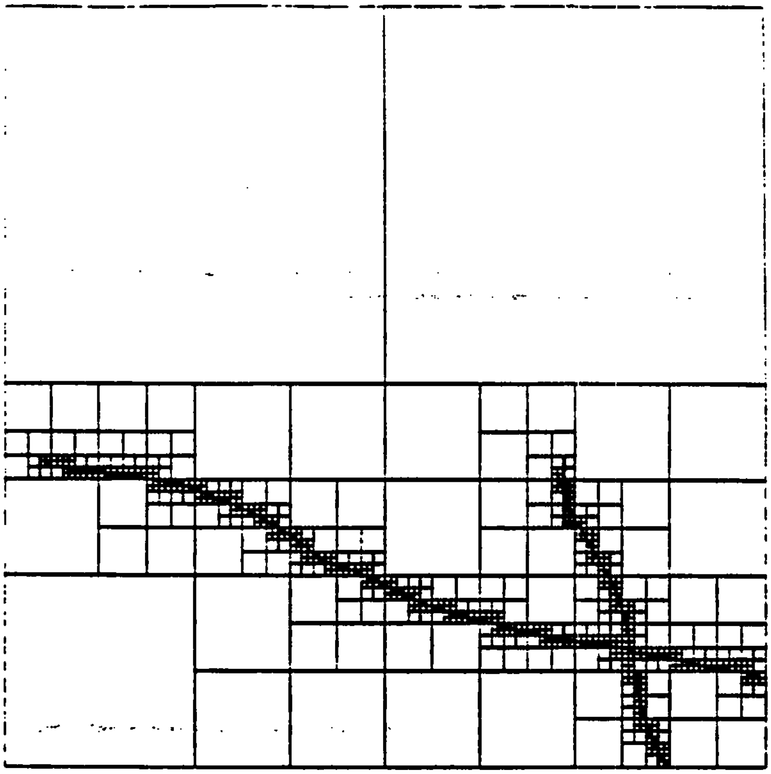
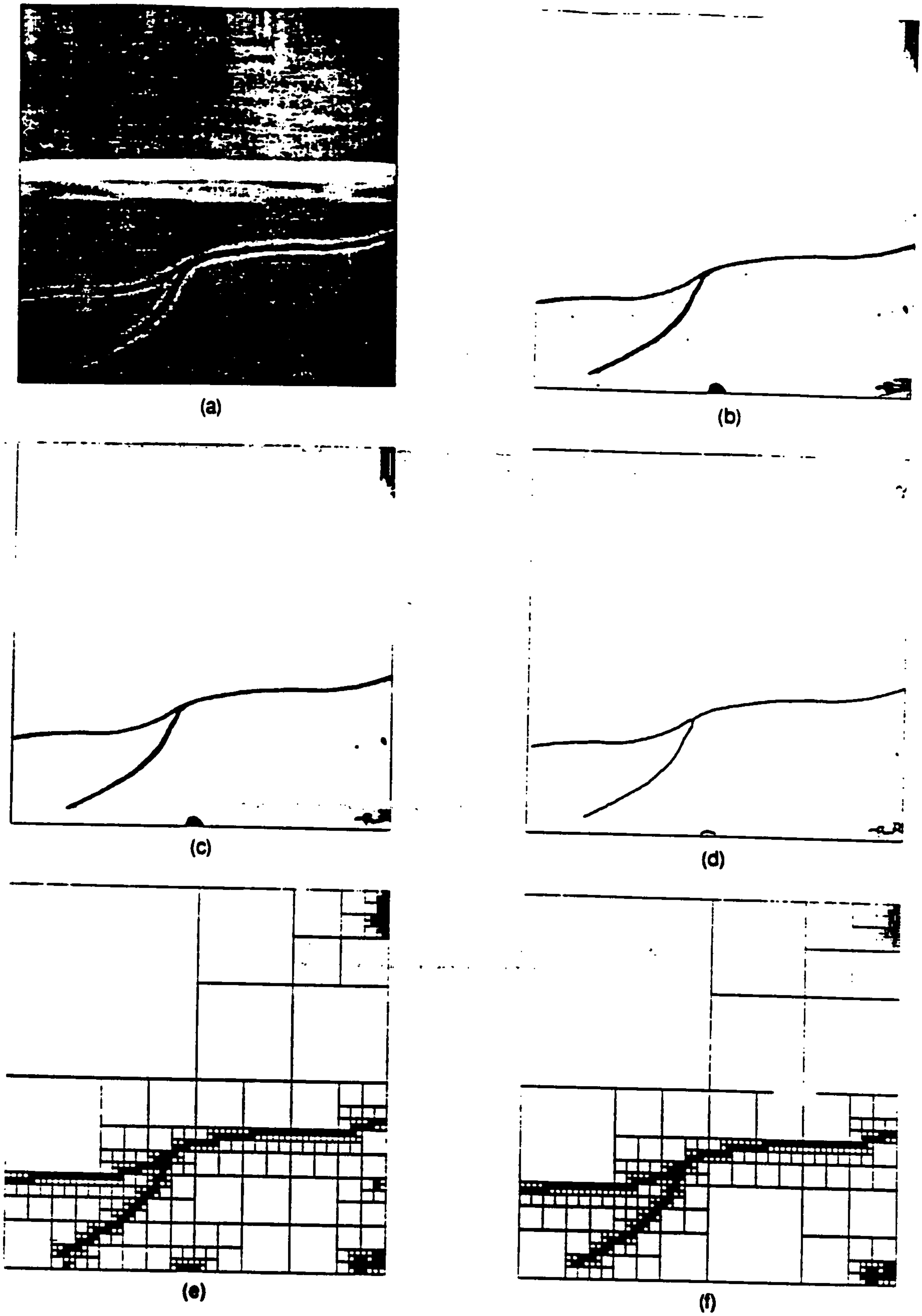


Figure 6-3: Corroboration Results for Scene 2



(a)

(b)

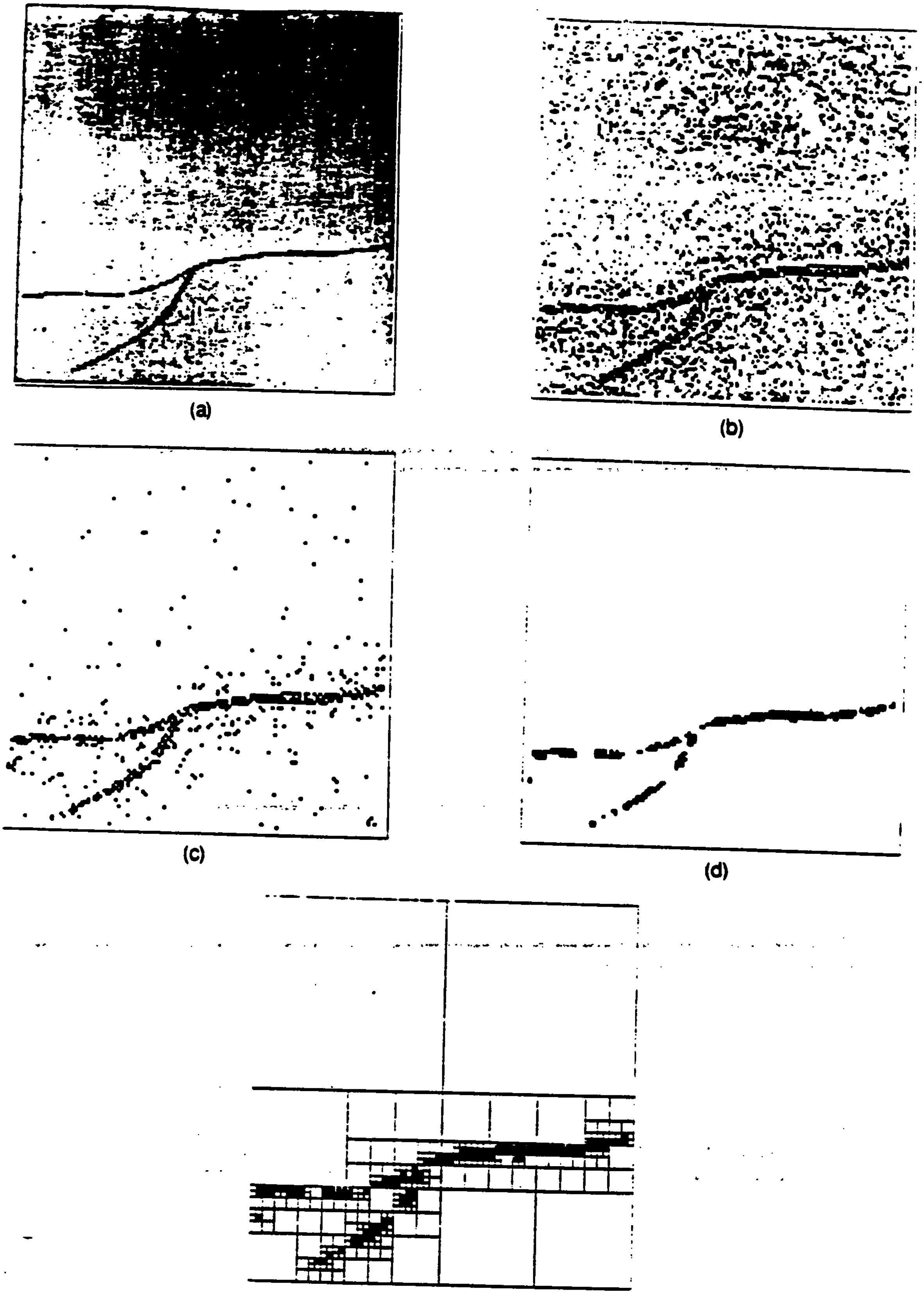
(c)

(d)

(e)

(f)

Figure 6-4: Video Data and Processing Results for Scene 3


(a)

(b)

(c)

(d)

(e)

Figure 6-5: Range Data and Processing Results for Scene 3

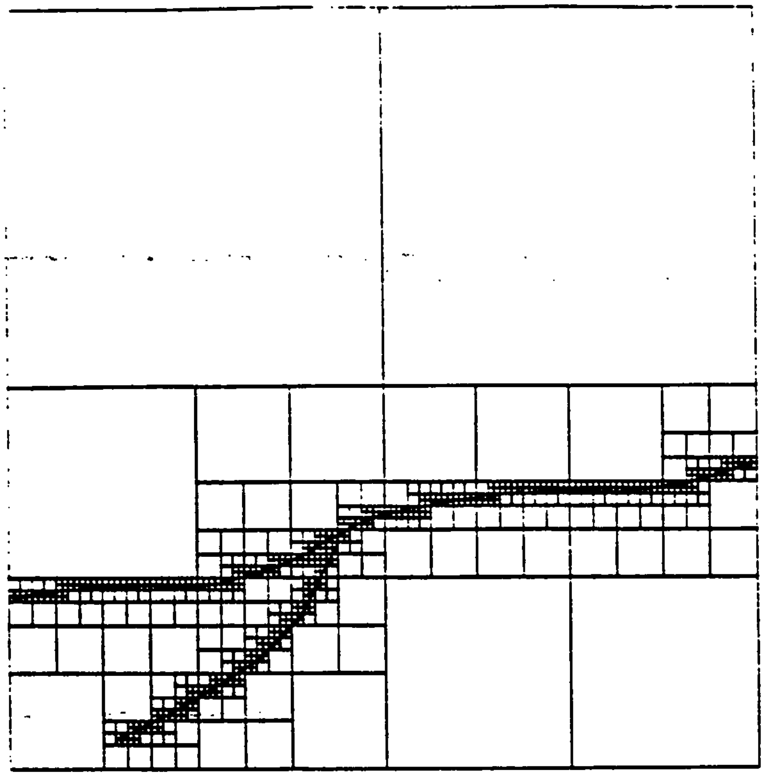


Figure 6-6: Corroboration Results for Scene 3

7. Second Generation Design and Field Demonstration

The previous chapters reported on laboratory and simulation efforts during Phase I of the project. During Phase II (1991/92), the following major activities were undertaken:

- the perception algorithms and software were extended to identify un-routed cracks
- the perception and mapping software were ported to operate on an MS-DOS based PC computer from the original UNIX implementation
- a second generation, field hardened crack sealer prototype machine was designed and built
- field demonstrations were performed in Sacramento, CA at the Caltrans Research Laboratory, and
- refined economic estimates were developed

This chapter describes this Phase II research, with the exception of the economic evaluation which appears in the following chapter.

7.1 Design Issues

The primary objective of a design for automation is to reduce the overall cost of crack filling. Improving the quality of the work and worker safety are also objectives. The final design must be evaluated based on these objectives.. Optimizing the design is a process of balancing several criteria within relevant constraints as summarized in Table 7-1.

There are a variety of functional approaches that can be considered for crack filling. Alternatives may have varying degrees of manual supervision. For example, multiple nozzles can be used for blowing and filling rather than individual effectors. Arrays of nozzle effectors would be costly, and the necessary switching patterns would be extremely difficult to implement. In particular, the viscosity of conventional sealant material makes short frequent bursts difficult. In contrast, use of individual effectors requires that each effector will somehow be drawn precisely along the length of the cracks to be filled. A multiple-degree of freedom manipulator is necessary to control the path of the individual effectors. Options include: (a) having the truck driver tele-operate the manipulator while the truck is stopped, (b) having the

Table 7-1: Criteria to Evaluate Automated Crack Filling Systems

System Constraints

- the ability to endure extremes of environmental conditions
- work quality that is equal to or greater than average manual quality
- nearly continuous up-time
- ability to endure a typical impact from a moving vehicle
- ability to maintain productivity equal to or greater than manual road crews (in lane miles per hour)
- automatic shut-offs to avoid injuries

Design criteria to be maximized:

- reliability
- transportability
- power to weight ratio
- speed of operation
- autonomy (ability to work without manual supervision)
- maintainability (simplicity of design and modular components contribute)
- consistency and quality of work

Design criteria to be minimized include:

- operating costs
- expected down time
- noise
- production of debris
- use of sealant material
- weight of equipment

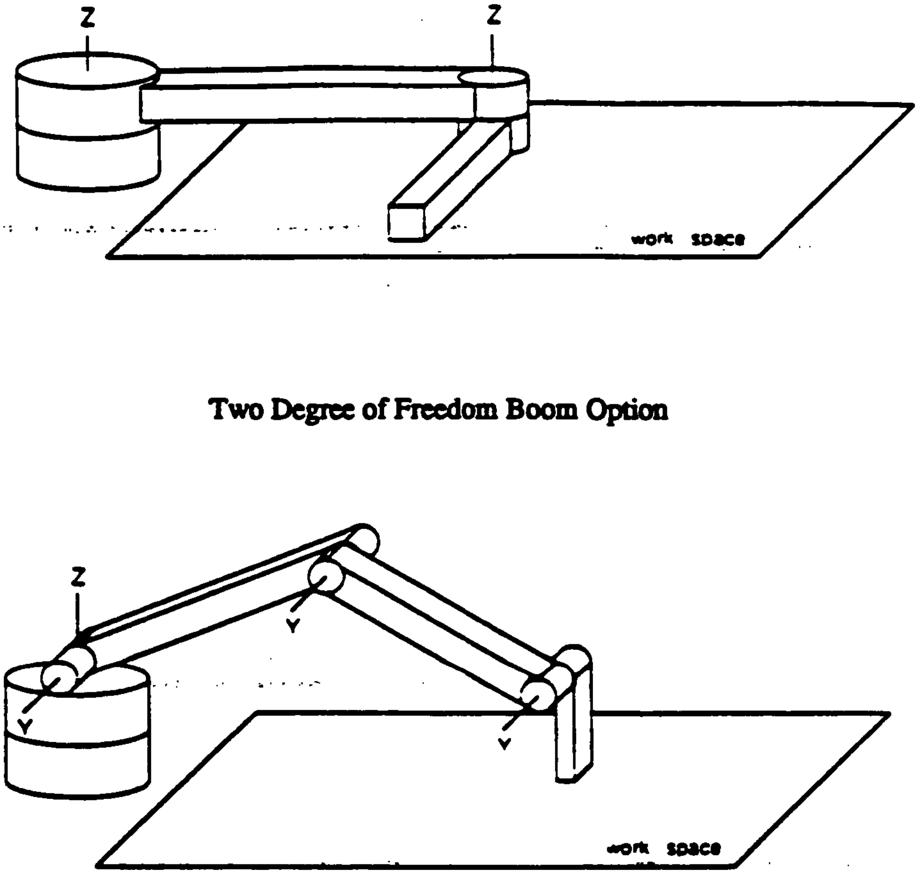
truck driver check or add crack location information to a partially automated system in which the manipulator would be controlled automatically, and (c) having the operator simply monitor with interrupt control a largely autonomous crack filling operation. In what follows, we shall consider the design of manipulators for option (c) to meet the criteria in Table 7-1.

A device that first suggests itself as a manipulator is a robot arm cantilevered out from behind the lead truck. Two constraints are imposed in this case. All parts of the arm must remain within the work space defined by a section of the pavement, and the arm must have at least two degrees of freedom. Within these constraints, numerous configurations are possible; such as the two degree of freedom boom or a backhoe-like configuration shown in Figure 7-1.

Most cracks describe irregular paths. Following such paths requires that the arm's controller calculate control sequence solutions for many small incremental movements. Heuristics might also be required to keep the arm within its workspace. As a result, computational requirements for arm control are severe. A robot arm may be made to satisfy the design constraints, but its resistance to an impact would be questionable given the potential moments about its joints. Concerning design criteria, an arm may be portable, but the combined weight of two effectors may exceed the load specifications for commercial robot arms. A commercial arm measures well against the remaining design criteria, however purchase and operating costs can be very high.

A simpler solution for manipulation is a an xy-table. Such a device works much like a large scale plotter with a gantry and mounted cart implementing x and y motions respectively (Figure 7-2). Control is much simpler than a robot arm. With a frame-work constructed of I-beams, a table is more impact resistant and stable than an arm in that reactions are always within the framework and distances to points of support are minimized. The effectors are easily kept perpendicular to the pavement surface. All the design constraints can conceivably be met. As for the design criteria, the device may be simply constructed and therefore maintainable, but its transportability is an issue of concern. It measures well with respect to the remaining design criteria.

In the following sections, requirements and options for automating the actions of the crew members who blow and fill the cracks are described. The crews' actions include identifying the cracks to be filled and then, in effect, tracing the cracks with the torch and the wand.



Two Degree of Freedom Boom Option

Four Degree of Freedom Backhoe Option

Figure 7-1: Two Manipulator Options

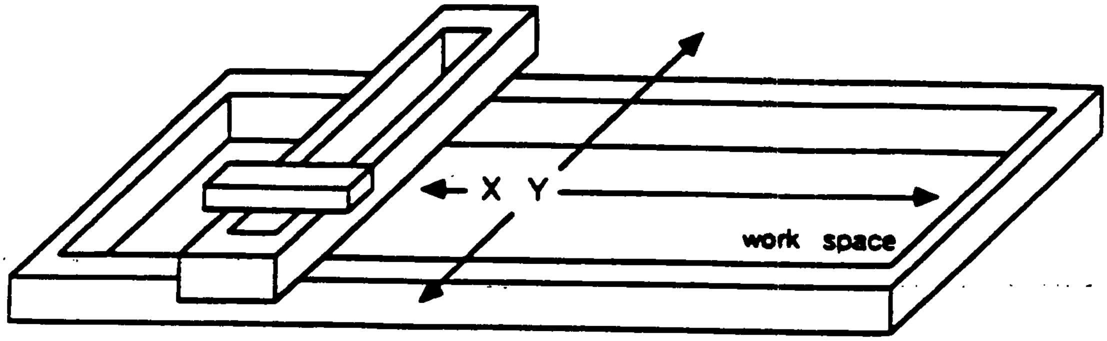


Figure 7-2: XY-Table Conceptual Design

7.2 Surface Perception and Modeling

Identifying cracks in the road surface automatically is not an easy problem [Haas 84, Fukuhara 90, Ballard 82]. Mapping the layout of the cracks in detail and selecting those to be filled increases the difficulty. Only in the case of routed cracks is the problem simplified by distinct visual patterns of debris and by consistent groove dimensions. To identify the cracks, characteristic surface data is required.

Surface data can be acquired at a suspended distance above the surface or near the surface by locating a sensor with the effectors. It can be acquired in raster scan or arbitrary patterns. It can also be acquired with noncontact or with contact sensors. Contact sensors such as a pin cushion type roller or a linear array of brush sensors are not feasible because of their cost and their probable insensitivity to narrow cracks. Noncontact sensors include vision, range, and forward looking infrared devices. Video cameras can acquire a raster pattern of digitized surface grey level values very efficiently. Range sensors such as ultra-sonic and infrared laser devices can be drawn over the surface in any acquisition pattern by the effector equipment. In practice, all these sensors experience noise because of the varied topological and color conditions of the pavement surface, and because of environmental factors such as wind and sunlight.

The range of potential pavement surface characteristics that may be encountered in any particular crack filling operation requires a powerful general surface modeling tool. The model should include a perception element which acquires characteristic surface data with sensors and then processes the data to extract useful features. The model should also include a surface representation from which useful descriptions such as routed crack layouts can be derived. Even with good data, a single sensor perception system can be fooled. Analysis of a video image alone shows that it is almost impossible to automatically detect the difference between a routed crack, a filled crack, and a strip of dark oil. With the corroboration of range information however, the routed crack can be distinguished from the imposters. Combining information from multiple sources in a common surface representation can increase the accuracy of crack perception.

7.3 Crack Filling Control

Control of the crack filling process is exercised at several levels. Control of a system that moves continuously down the road at a constant speed introduces complexities in terms of perception, planning, and especially manipulator control that are simplified by operating the system in a stop-start manner. Assuming a stop-start strategy, the highest level of system control implements the following steps repetitively as the system moves down the road:

1. acquire sensor data and develop a representation of the pavement surface

2. map the cracks to be filled

3. develop a work plan

4. execute the blowing and filling operations

5. repeat the above four steps

In the process of developing the surface representation for the current area to be worked, the system may compile a list of commands to the equipment to enact scanning patterns and acquire data. Once the surface representation is complete, the system must also choose the order and direction of the cracks to be traversed. This plan must then be compiled into a list of commands to the manipulator and to actuators such as the open and close valve on the sealant wand.

The movement commands to the equipment and the actuator commands result in machine level control problems that are more or less complex depending on the equipment configuration. In the case of an xy-table a command to move from point (x1,y1) to (x2,y2) can be translated directly to a number of revolutions for each one of an X and Y drive motor. Servo-control motors allow additional control of acceleration and velocity. Higher level feedback may be required for at least two purposes:

- confirming progress and accuracy of the effector mechanisms
- registering changes in position of the xy table with respect to the pavement surface

It might also be possible to alter this process by changing the type of material being used. Low viscosity sealant could be used to flow into cracks while excess sealant material could be scraped from the surface. However, identifying new materials with the proper amount of strength and low cost is difficult.

7.4 A Second Generation Design

A schematic illustration of the prototype final system design appears in Figure 7-3. A lead vehicle would tow an xy-table assembly and the sealant and propane tanks. The lead vehicle would carry the power source for the manipulator and the necessary computing equipment. A camera is mounted on an extendable boom so that it can be suspended above the work surface during operation. A video monitor is located next to the driver for monitoring and manual over-ride. The monitor could also be used in an interactive configuration in which the driver would enter some sort of rough directions to the system via a joy stick in the form of graphics overlaid on the scene on the monitor. Three tools are mounted on the xy-manipulator. They include the heated air torch, the sealant wand, and an infrared laser range sensor. The table structure may be steel I-beams vertically oriented with the top flange being used as a guide rail for the gantry. In turn the gantry may be constructed in a similar fashion with its two main members serving as rails for a cart on which the tools are mounted.

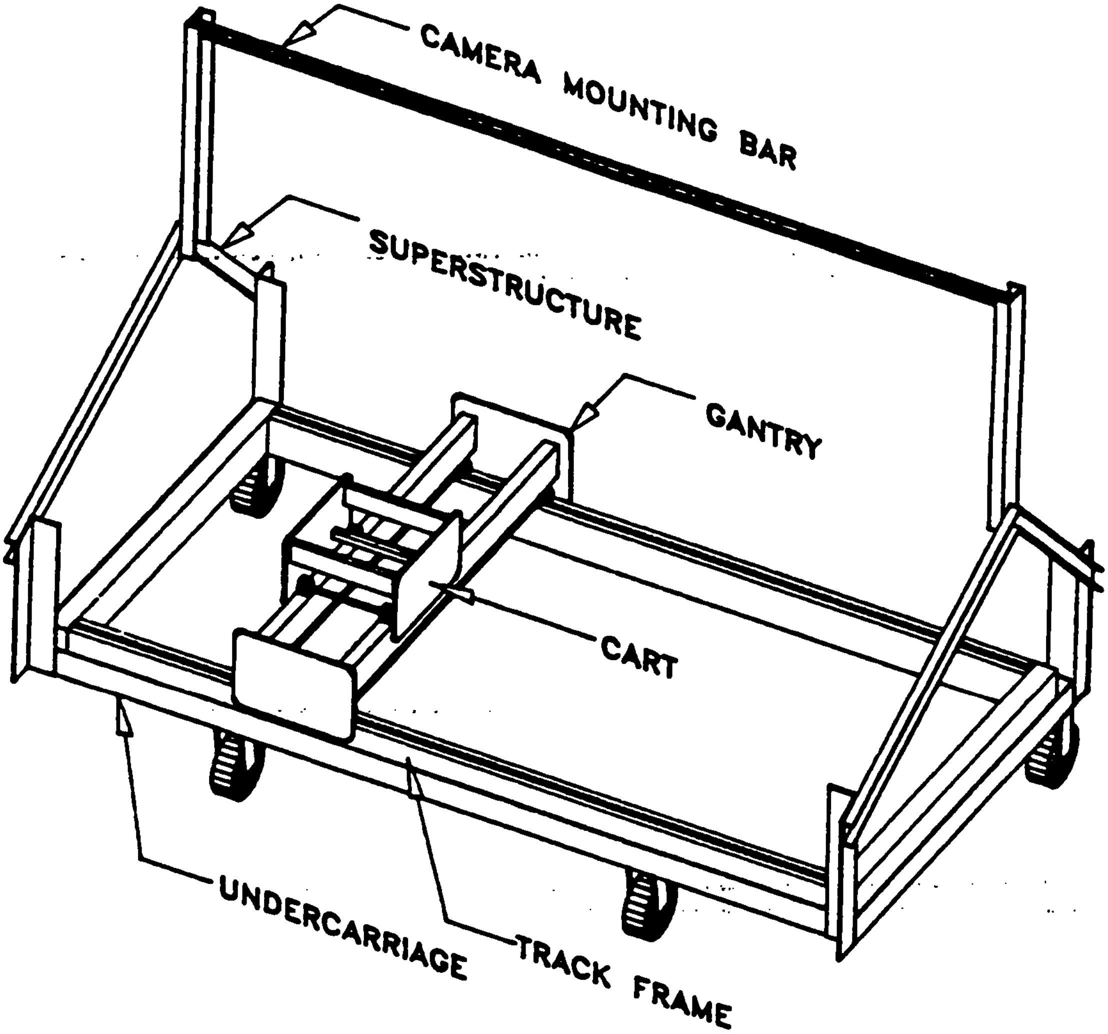


Figure 7-3: A Schematic Illustration of the Crack Filling Robot System

7.5 Field Demonstration

Field demonstrations of the prototype system were made in August, 1991, in the parking lot of a Caltrans research facility in Sacramento, CA. Figure 7-3 illustrates the field apparatus. The perception and control example of the field system is summarized here.

In the automatic crack detection and mapping process (Figure 7-4) a video image of the work area is acquired first (in this description, data from a field trial is used). It is digitized and stored in computer memory as a 512 x 512 array of pixels. The image is then segmented using a grey level thresholding operation which is manually tuned. More sophisticated and autonomous segmentation procedures are unnecessary because of the slow moving equipment train and the availability of the driver for tuning. The image is divided into cells of 8 x 8 pixels and each cell is classified as occupied or unoccupied by what looks like a crack. The result is a cell occupancy array of 64 x 64 cells. The cell occupancy array is then filtered to remove stray cells. Clumped strings of cells correspond to cracks on the road. These clumps are thinned into singly connected strings of cells using a skeletonizing algorithm, so that the strings can be converted into an ordered set of work area coordinates that can be used to generate instructions for the manipulator (Figure 7-5). The occupancy array is then converted to the multi-layer quadtree data structure. The strings are identified as connected groups of cells using a connected component labeling algorithm, and very small strings are removed automatically (Figure 7-6). The remaining strings may still represent marks of filled cracks, so range information is required to corroborate or dismiss each string, as in Figure 7-7.

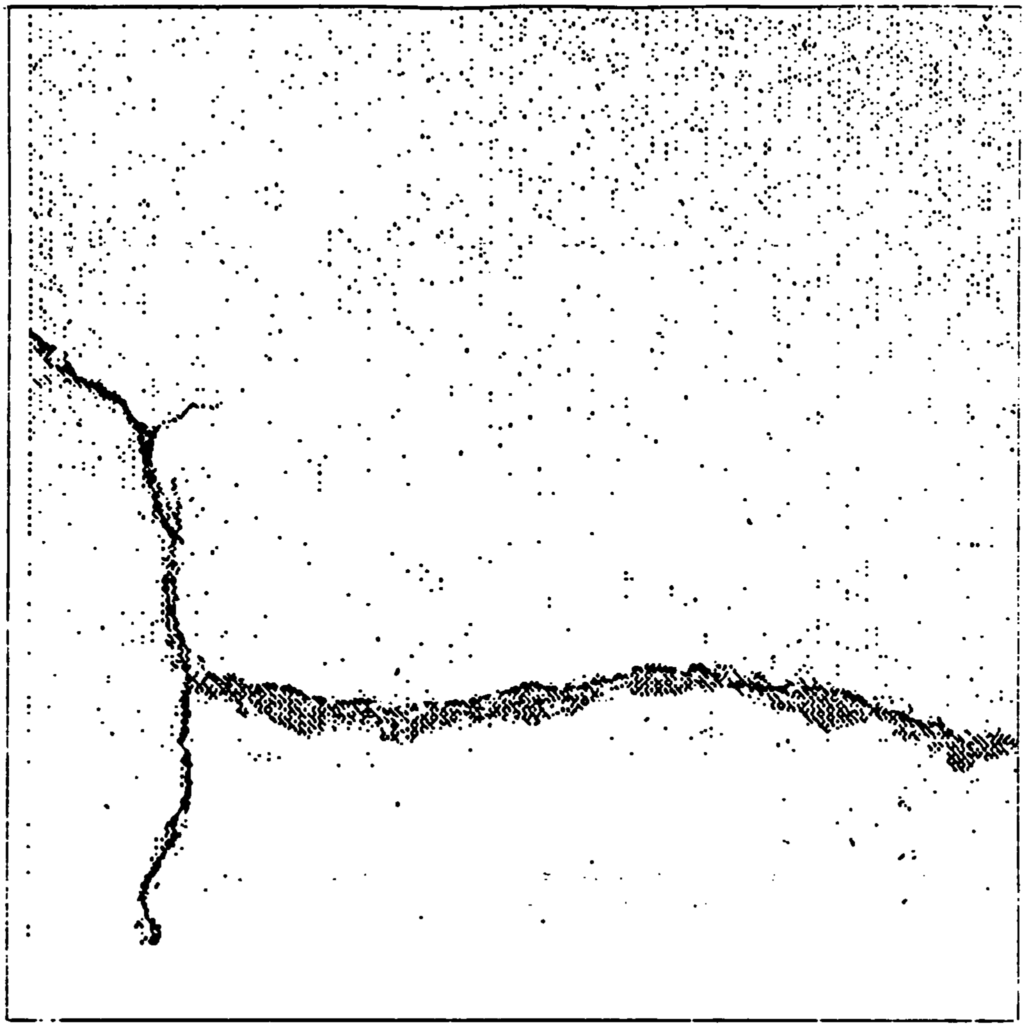


Figure 7-4: Raw Video Data

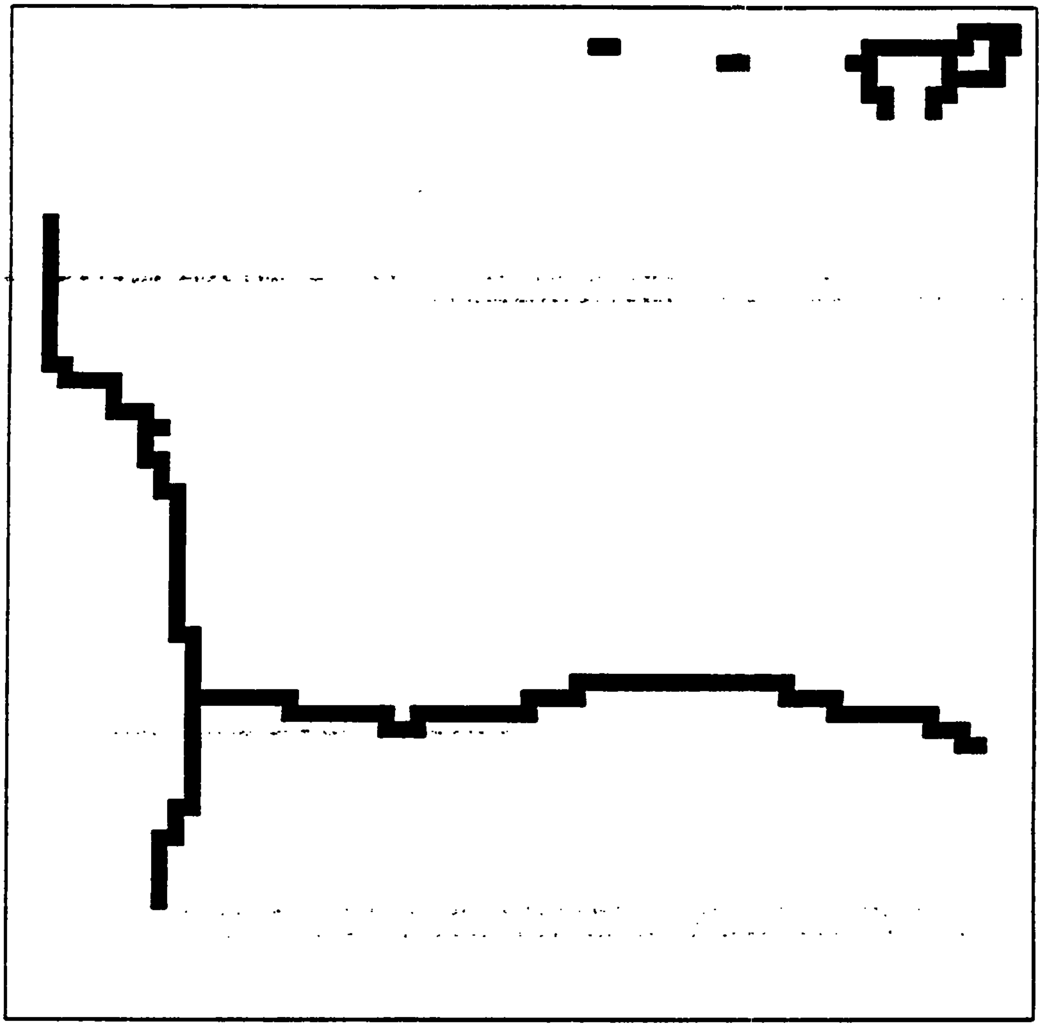


Figure 7-5: Occupancy Array of Video Data

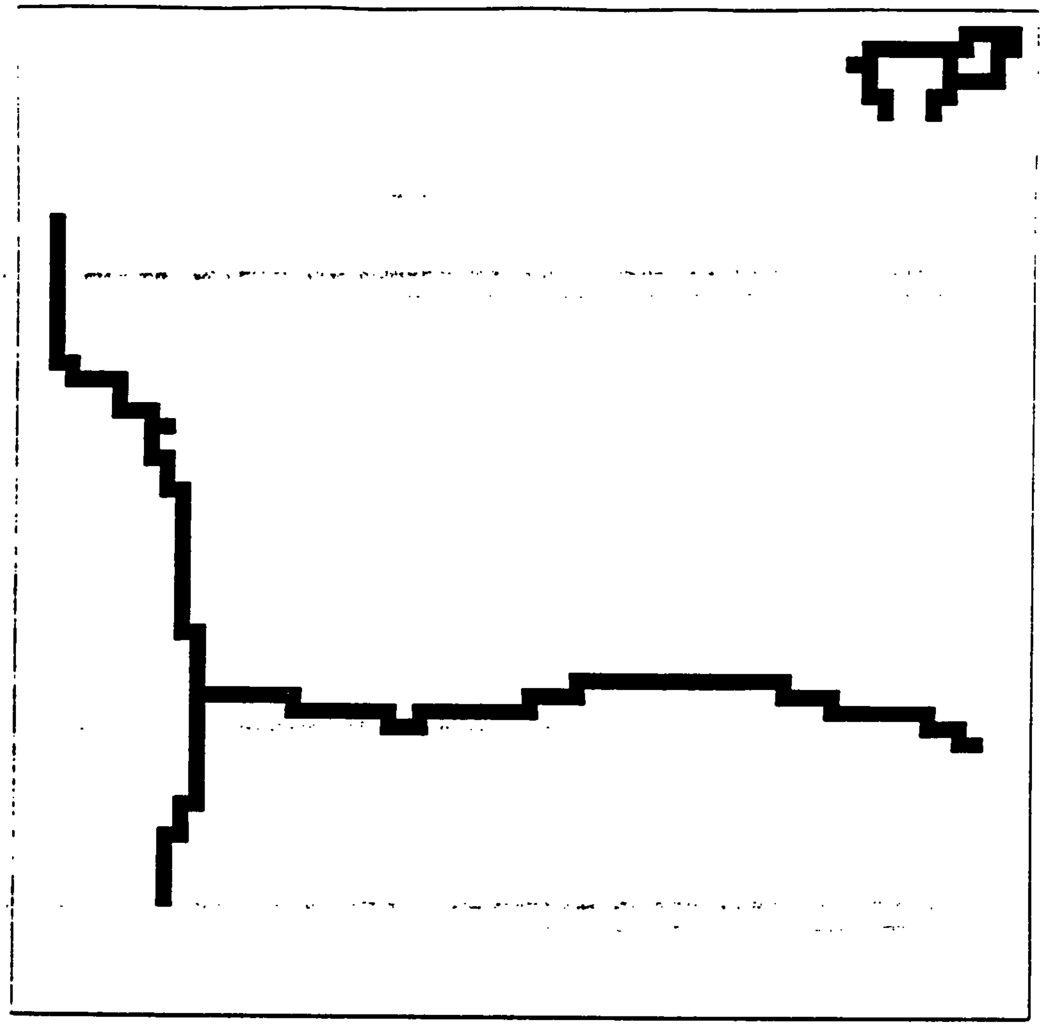


Figure 7-6: Median Filter of Video Data

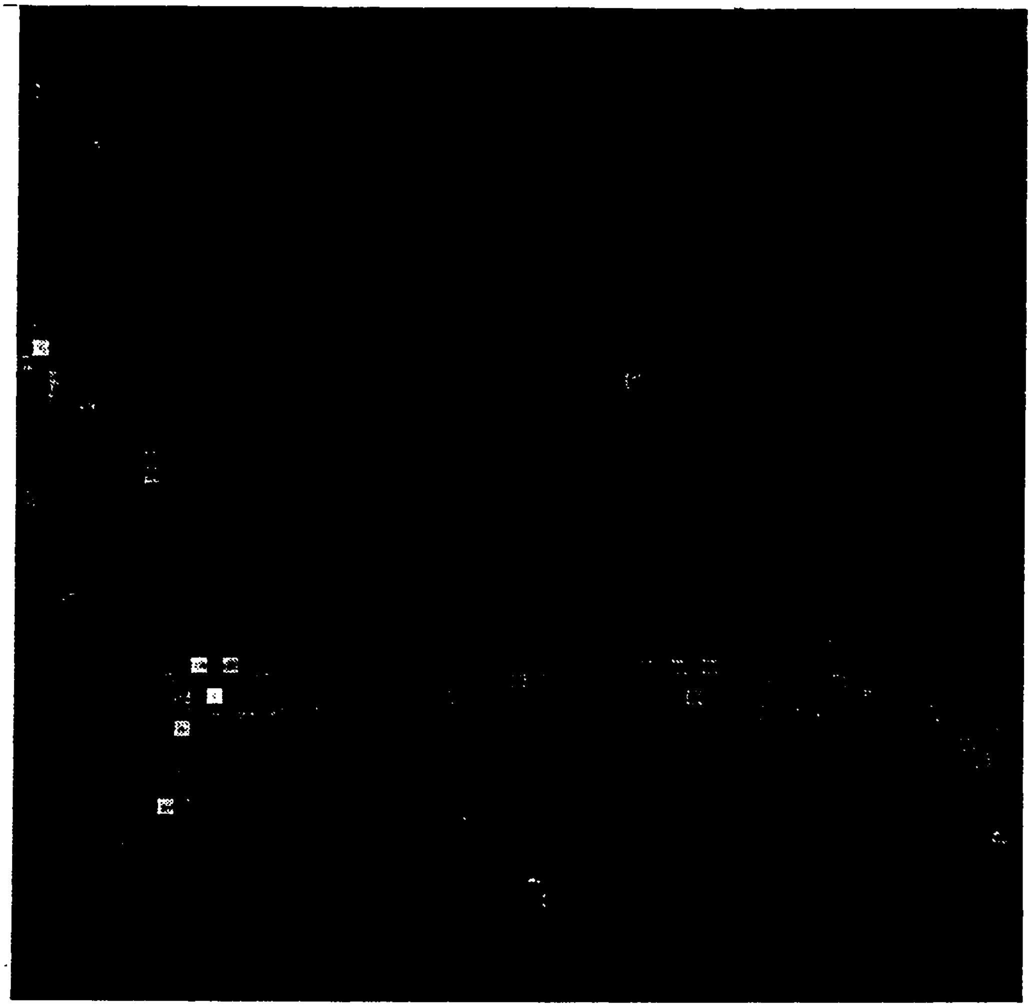


Figure 7-7: Raw Range Data

7.6 Areas for Improvement in Further Research

Improvements in the hardware and software remain to be made before the crack filler is ready for production. Hardware improvements include enlarging the frame of the crack filler to achieve a larger scan area and hence reduce the number of times the frame must be moved to cover a given area, as well as refining camera mounts, upgrading several components to achieve higher speeds, and adding brakes. Software improvements center around integration and generalization of existing software components.

In the next generation of hardware, improvements would be made which would lead to a larger scan area with a shorter scan time. The following would increase the size of the scan area:

- The frame could be widened such that the available scan width is at least one lane width. Since the frame is not permitted to extend into the adjoining lane, provisions would have to be made to permit the sensors and effectors to extend to the very edge of or beyond the frame so that the complete land width could be repaired in one pass.
- The length of the frame could be increased to the limits imposed by deflection and stability to permit more area to be scanned and repaired before movement of the frame is required.
- The size of motors and placement of sideway bracing interfere with the movement of the cart and gantry. Smaller motors and improved bracing design would permit the cart and gantry to move over a wider area.

Second, the hardware could be improved to permit faster movement of the effectors for crack verification and filling. Currently, the cart and gantry require about 2 minutes to complete the range scan of the framed area. This speed limitation is primarily due to lack of stability, requiring that acceleration of the cart and gantry be minimized to prevent movement of the frame out of position during sensing. While software improvements will vastly improve the robustness of the system in accommodating variations in precise orientations, improvements in hardware can also solve the problem of vibration. Proposed improvements include:

- Adding brakes to the frame will stabilize the system against movement. Improved bracing of the frame, especially if the frame is enlarged, is essential.

- A complete structural analysis of the frame should be undertaken to determine the optimal bracing configuration and size of structural members.
- An integrated z-y controller would permit more exact curve-following, eliminating the separate x and y movements currently required to move the cart and gantry along a nonlinear path.

Finally, the fieldworthiness of the crack filler must be addressed in future generations of hardware:

- Sturdier wheels would permit the crack filler to be towed at highway speeds behind a vehicle instead of limiting speeds to five miles per hour and hence requiring truck transport and additional assembly and disassembly.
- Improved cable management would permit more freedom of movement of the cart and gantry and would reduce the possibility of damage to the system in the field.
- A new camera mount should be designed which will permit rapid assembly and alignment in the field and will diminish camera movement due to vibration.
- At least two cameras should be utilized in future generations of the crack filler, to improve scan time and reduce spatial distortion.
- A study of effectors for sealant should be undertaken to determine what limitations are imposed by them, as well as optimal configurations of the hardware and software with sealant effectors included.

Improvements in the software which operate the crack filler are essential in future generations of the project. In fact, significant improvement could be achieved in the existing system by improving the software and utilizing existing hardware. Hence, further research should focus on software improvement before hardware modification is undertaken. Software improvements should include the ability to calibrate video and infrared images after perception, path planning for crack transversal after video imaging, and development of a robust, integrated software package with a user interface for non programmers.

First, sensor calibration is most efficiently achieved by software calibration of video and infrared images. Currently, adjustments must be made iteratively to the video camera in order to align the images prior to sensing the image processing. This is a lengthy and arduous process which requires several persons and several days to complete. Moreover, if movement of the camera occurs from vibration during sensing, the system must be recalibrated. An algorithm should be created to allow the range sensor to be calibrated to video images. Related software must be generalized to accommodate variations in image size caused by this calibration.

Second, path planning should occur after video imaging. Currently, path planning occurs after infrared imaging. An infrared scan is undertaken for the complete scan area, in addition to a complete video scan. The video scan is currently completely redundant. The software should be modified to allow path planning after video scanning. This would reduce the amount of infrared scanning to only verification of video images.

Finally, all software should be evaluated and integrated into a turnkey package. In order to operate the crack filler at present, several programs must be run independently, and files transferred between programs. Software must be used interfaced, generalized, and thoroughly documented in order to be widely useful. With these software improvements, the robot pavement crack filler will evolve into a usable product with only minor hardware modifications.

A field prototype system for testing is currently under development by a team involving Caltrans, University of California at Davis and Bechtel Corporation.

8. Evaluation of Automated Crack Filling

8.1 Introduction

A field prototype of a robotic pavement crack sealer has been developed at Carnegie Mellon University. The system identifies pavement cracks using video imaging and verifies that the cracks actually have depth using a laser range sensor. The system then develops a map of the pavement cracks and can be extended to proceed automatically to clean and fill the cracks. The system is intended to reduce labor costs, improve worker safety due to reduced exposure to traffic and improve the quality of the crack sealing operation. A preliminary analysis of the costs and benefits of automated crack sealing indicated that the system is economically feasible [McNeil 90]. This analysis was based on a limited survey of current practice. To obtain better estimates of the costs and benefits of automation, a more comprehensive survey was administered to determine

- current crack sealing practice,
- the expected extent to which an automated system would be adopted, and
- the expected labor savings due to automation.

This report summarizes the survey responses and analysis of the economics of automation based on the survey responses.

8.2 Data Sources

A two page survey was developed to obtain information on current crack sealing practice including materials, crew organization, costs and safety record. The limited survey used in 1990 served as a test for the range of responses and wording of the questions. The survey in 1991 was more comprehensive in terms of the questions asked and its distribution. The survey was mailed to the department of transportation or public works in:

- 50 states
- 19 turnpike and toll authorities
- 14 cities and townships

- 12 countries
- 12 provinces

The survey form is included in Appendix I.

Responses were received from:

- 42 states representing an 84% response rate. (Alabama, Alaska, Arizona, Arkansas, California, Colorado, Connecticut, Florida, Georgia, Idaho, Illinois, Indiana, Iowa, Kansas, Kentucky, Louisiana, Maine, Maryland, Michigan, Minnesota, Mississippi, Missouri, Montana, Nebraska, Nevada, New Hampshire, New Jersey, New Mexico, New York, North Dakota, Oklahoma, Pennsylvania, South Carolina, South Dakota, Tennessee, Texas, Utah, Vermont, Washington, West Virginia, Wisconsin, Wyoming) For some states multiple responses were received as different regions have different procedures.
- 4 turnpikes and toll authorities representing a 21% response rate. (Indiana, Maine, Massachusetts, Texas)
- 7 cities and towns representing a 50% response rate. (Charlotte, NC, Dallas, TX, Kansas City, MO, Napierville, IL, Omaha, NE, Oklahoma City, OK, Pittsburgh, PA)
- 5 counties (42% response rate - Jefferson County, CO, Lake County, IL, Montgomery County, MD, Oakland County, MI, Prince Georges County, MD)
- 6 provinces (50% response rate - Alberta, British Columbia, Manitoba, New Brunswick, Nova Scotia, Saskatchewan)

Complete mailing lists and responses are included in [McNeil 91]. The data were entered into a spreadsheet to facilitate summarizing and analysis.

8.3 Current Practice for Crack Sealing

The survey responses indicate tremendous variability in crack sealing practice from organization to organization in terms of both the extent of crack sealing and the methods used. Tables, figures and descriptive statistics (maximum, minimum value, mean and standard deviation) are included to summarize the survey responses. Where possible the responses have been extrapolated to provide national estimates. Table 8-1 summarizes the responses for several questions. The minimum and maximum value, the mean and standard deviation for the responses by individual states are reported. Where appropriate totals are also included. The following subsections provide more detail on expenditures, method of accomplishment, crack preparation, crew size and organization, labor costs, materials, crack sealing periods, safety and expected usage of an automated system. This summary focuses on the responses by the state transportation departments. The small samples for the other government units make it more difficult to draw conclusions or characterize the response in terms of average values.

8.3.1 Expenditures

Survey respondents were asked for crack sealing expenditures and their total maintenance budget. The proportion of the maintenance budget used for crack sealing is used to indicate the importance of crack sealing for a state. Three states - Alaska, Louisiana, and Wisconsin reported that in general they did not do crackfilling. Others, such as Illinois, indicated that it varied from district to district. Figure 8-1 shows the variability in the importance of crack filling for the states responding to the survey. Of the 42 states responding, 26 states spend less that 1% of their maintenance budget on crack sealing, compared with 8 states that spend more than 6% of their maintenance budget.

The surveys indicated that an average percentage of maintenance budgets spent on crack filling is 2.8% with a high of 13.3% and a low of 0% for the agencies surveyed. Table 8-2 provides similar descriptive statistics for each agency type. An estimate of the percentage of budget spent on crack filling for each type of agency is also given in Table 8-2[1]. The former quantity provides an indication of importance of

[1] The average of the percentage expenditure is the mean of the ratios for each agency where the estimated percentage expenditure is the ratio of the mean expenditure on crack sealing divided by the mean maintenance budget [Cochran 77].

crack sealing in states where the latter can be used to estimate national expenditures as follows. Total expenditures for crack filling are computed by agency type by determining

1. the percentage of maintenance expenditures used for crack filling based on survey responses (Table 8-2), and
2. total maintenance expenditures by agency (Table 8-3).

Table 8-1: Survey Responses

		Min	Max	Mean	Std Dev	Total
Number Months Crack Sealing Conducted		0	12	5.98	3.35	
% Time Crews Involved in Crack Sealing	(reported)	1	100	14.7	25.3	
	(calculated)	1	97	28	28	
No of Crews Involved in Crack Sealing		1	345	67	72	1802
Crew Size		3	14	7.16	1.95	
Annual Accidents (for all maintenance activities):						
Fatalities		0	2	0.1	0.4	4
Injuries		0	975	131	250	368
Property Damage[2]		0	550	85	145	2294

Using this method, roughly $53 million per year is spent by states on crack filling. The survey responses indicated that in 1990 $48 million was spent on crack filling in 38 states which is comparable. The total value of expenditures in Table 8-3 is approximately $190 million representing national expenditures on crack sealing, but excluding expenditures by private organizations, the military and airports.

[2] Fewer property damage accidents are reported than injury accidents due to unreported accidents and inconsistencies in reporting requirements.

Table 8-2: Percentage of Budget Spent on Crackfilling Based on Survey Responses

Agency	Min	Max	Std Dev	Mean of % Expenditure	Estimated % Expenditure
States	0	6.00	1.45	1.22	0.69
Provinces	0	5.71	2.55	1.93	1.23
Cities	0.62	13.33	5.96	4.44	1.50
Counties	0	8.33	4.40	3.35	0.83
Turnpikes (1 observation)	0.18	0.18	0	0.18	0.18

Table 8-3: Maintenance Expenditures by Agency

Source: *Highway Statistics*, 1989

Agency	Total O&M million $	Amt for Crack Filling million $
States	7,761	53.3
Municipalities	5,707	85.9
Counties & Townships	5,529	46.1
Toll Facilities	1,212	2.2
Total		187.5

8.3.2 Method Of Accomplishment

Crack sealing may be undertaken by agency forces or by contractors or both. Figure 8-2 summarizes the method of accomplishment (contract, agency forces or both) for each of the states. The majority of states use their own labor forces to seal cracks.

Figure 8-1: % of Maintenance Budget Used for Crack Sealing

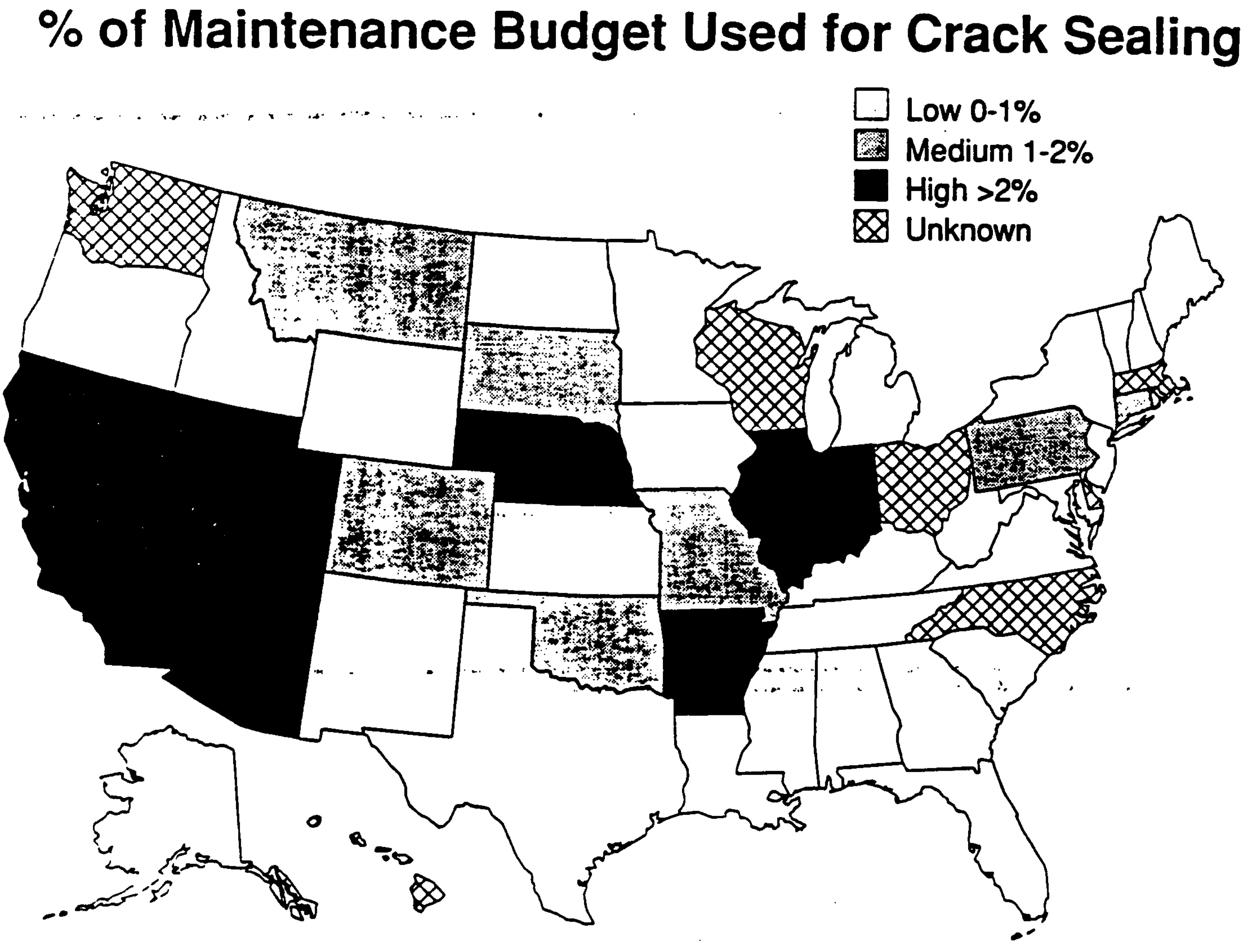
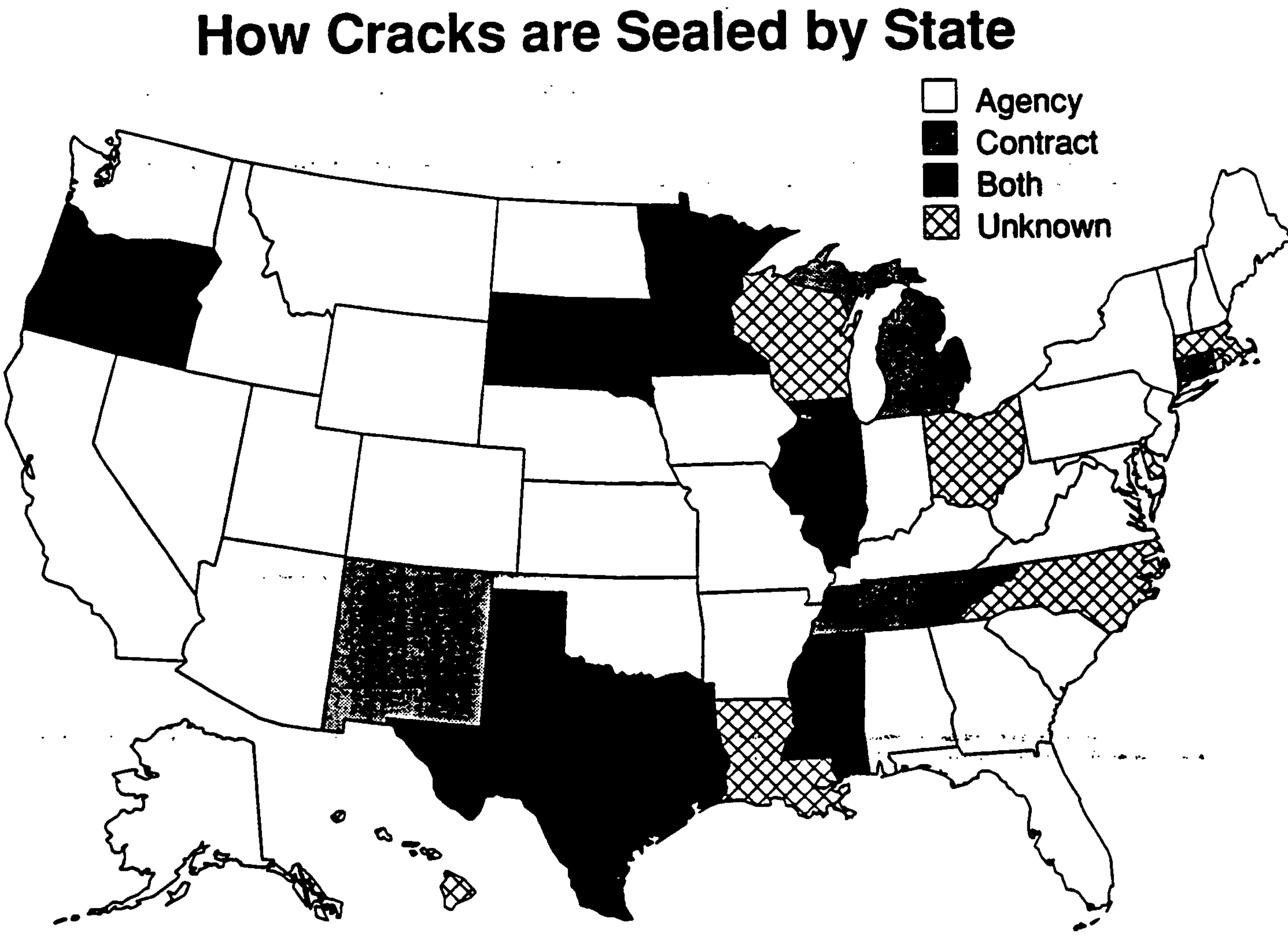


Figure 8-2: How Cracks are Sealed by State



8.3.3 Crack Preparation

States prepare cracks for sealing using a variety of procedures, either individually or in combination. The procedures include hot air lance, routing, sweeping, compressed air and sandblasting. The number of states using each procedure is shown in Figure 8-3. All states responding used compressed air to clean cracks but only 16 routed cracks prior to sealing.

8.3.4 Crew Size and Organization

The procedures used for crack sealing differ by state in terms of the activities involved in crack sealing. For example, some states rout cracks and some states use contract rather than direct labor forces. As a result crew size and organization varies. For the states using agency forces, crack filling appears to be an activity undertaken by multi-functional maintenance crews on an as required basis. The survey responses (Table 8-1) indicate that on average crews are involved in crack sealing almost 10% of the time. However, the reported crew utilization showed some inconsistencies. For example, it was not clear if reported utilization rates were over a whole year or just the season during which crack sealing was undertaken. Therefore, crew utilization was also calculated as follows:

$$\% \text{ Utilization} = \frac{\text{\$ spent on crack sealing}}{\text{Sum(daily costs)} * \text{\#months} * \text{20 day/month} * \text{\# crews}}$$

The summary statistics for the calculated utilization are included in Table 8-1. The surveys also provided details of crew compositions. Crew compositions for representative states are shown in Table 8-4. The average crew size is seven with a maximum of 14 and a minimum of 3.

8.3.5 Labor Costs

Labor costs vary from $5.11/hr to $23.04/hr with an average of $13.26/hr for a laborer, not necessarily including overhead and profit. These values are significantly lower than Means [Means 89] which gives $26.05/hr for a highway laborer including overhead and profit. Crack sealing is relatively labor intensive with labor costs representing over 61% of costs on a per lane mile basis.

Figure 8-3: Number of States Using Crack Preparation Procedure

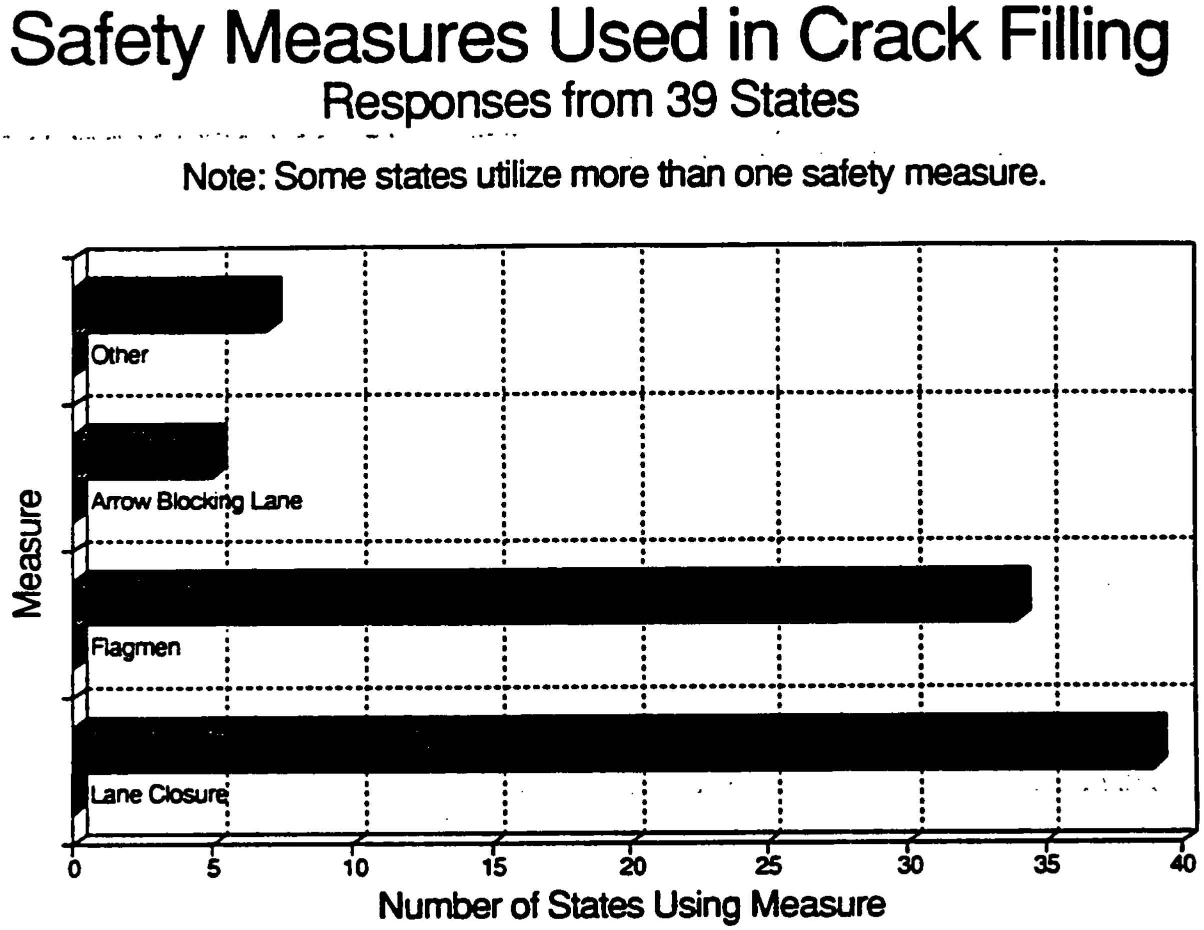


Table 8-4: Crew Composition for Crack filling (1990 $)

California Composition	Cost	Connecticut Composition	Cost	Pennsylvania Composition	Cost
Traffic Control		Kettle Operator	$11.33/hr	Foreman	$12.27/hr
Clean, Fill, Squeegee		Compressor Operator	$11.33/hr	Driver	$10.23/hr
Cover with Sand,	$15.00/hr	2 Truck Drivers	$10.97/hr	3-5 Non-Operators	$8.18/hr
Sweep	each	4 Laborers	$10.63/hr	2 Flaggers	$8.18/hr
(6-10 members)		2 Flaggers	$10.63/hr		

Illinois Composition	Cost	Missouri Composition	Cost
2 Flaggers	8 @	1 Supervisor	$9.82/hr
2 Laborers	$15.35/hr	2 Truck Operators	$8.74/hr
(compressed air)		3 Maintenance	each
2 Laborers		Workers	
(routers)			
1 Wand Operator			
1 Squeegee Operator			
1 Driver	$16.13/hr		
1 Supervisor	$16.42/hr		

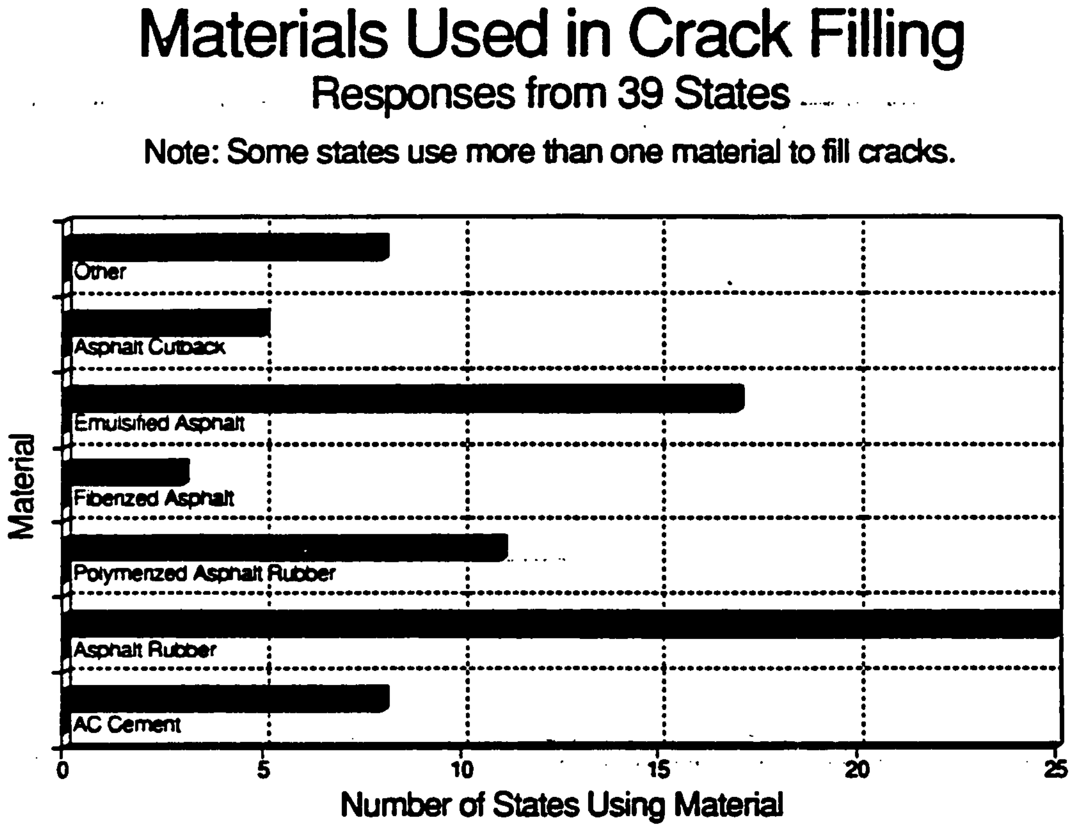
8.3.6 Materials

A variety of materials are used including AC Cement; Asphalt Rubber, Polymerized Asphalt Rubber, Fiberized Asphalt, Emulsified Asphalt, and Asphalt Cutback. The number of states using each material is shown in Figure 8-4. Some states use different materials for different applications or in different areas so the total number of users exceeds the number of survey respondents.

8.3.7 Crack Sealing Periods

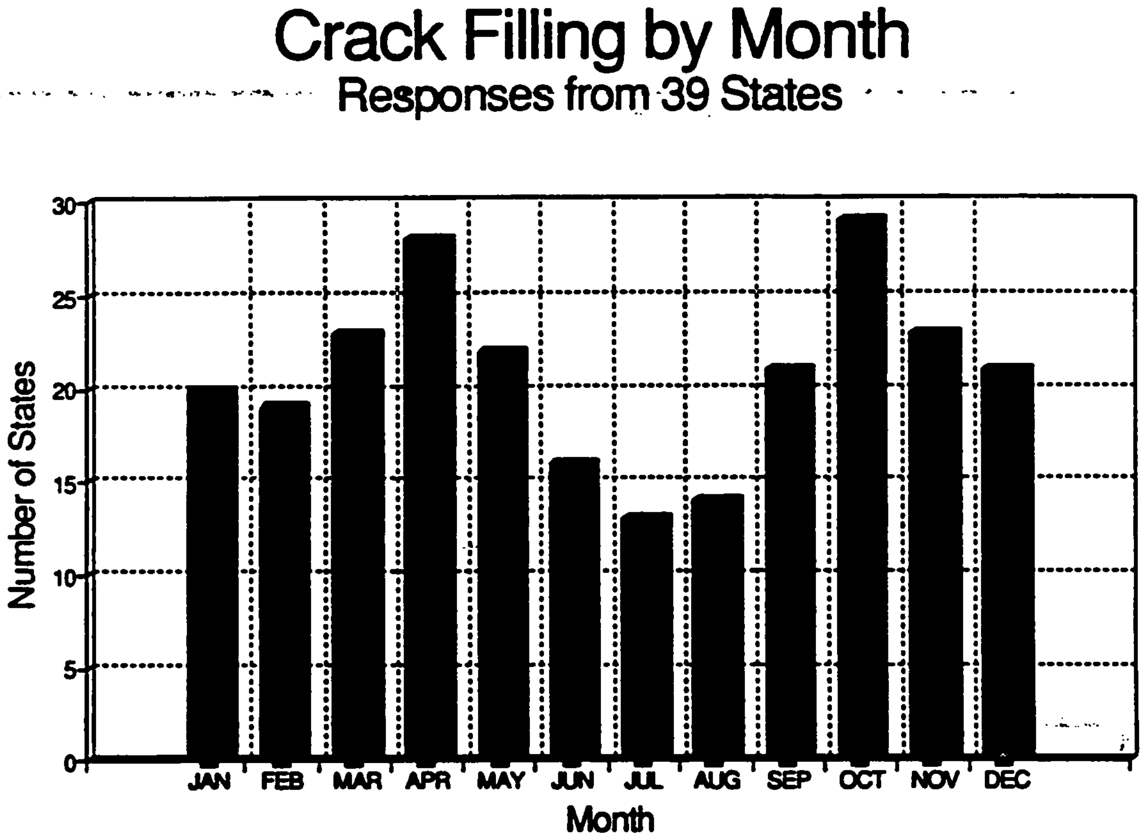
The months of the year in which states undertake crack sealing vary significantly depending on the location, the temperature and precipitation of the area and the deterioration of the pavement to be sealed. Figure 8-5 shows the number of states undertaking crack sealing in each month of the year. The most common times for crack sealing are Spring and Fall, but some agencies seal all year (or in all but the very

Figure 8-4: Number of States Using Material



cold months). As shown in Table 8-1, states undertake crack sealing six months of the year, on average. The surveys also indicated that approximatley 70% of states use pavment condition to detemine when crack sealing is required.

Figure 8-5: Number of States Crack Sealing in Each Month



8.3.8 Safety

All states reported that they close a lane to do crack sealing and most (33) states use flagmen. The number of states using various safety measures is shown in Figure 8-6. However, despite routine safety practices, a significant number of accidents involving maintenance workers were reported as shown in Table 8-1. With a total of 3681 reported injury accidents involving maintenance workers a slight reduction can have a significant impact.

8.3.9 Expected Usage of Automated System

The survey responses indicated that 35 of the 42 states would adopt the automated method if it was cost effective. That is, of the states sealing cracks, 90% would adopt the automated system.

8.4 Analysis of the Costs and Benefits of Automation

In order to analyze the costs and benefits of automation, estimates for the number of crack sealing units to be used, the expected costs and the expected savings need to be developed.

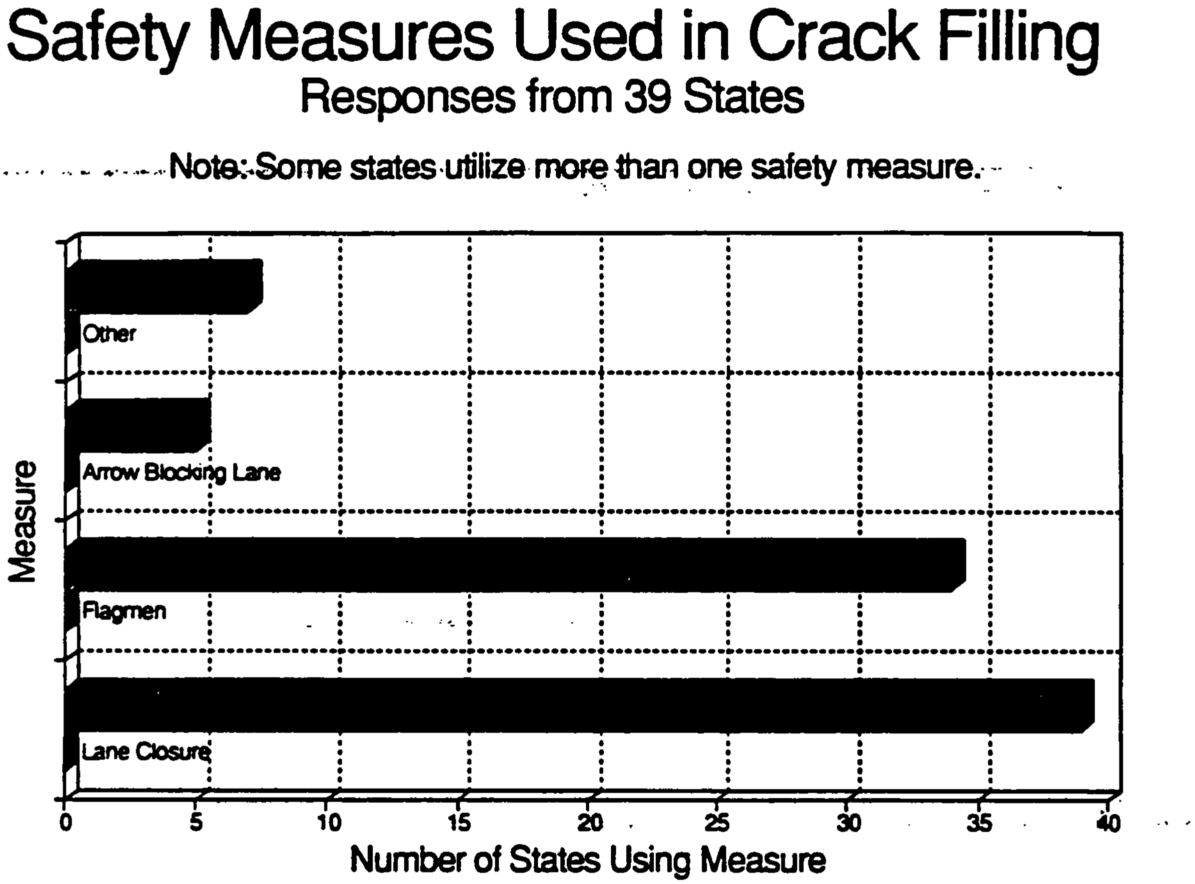
8.4.1 Estimate of the Market for Automated Crack Sealing Units

To develop an estimate of the number of crack sealing units likely to be used by states, the analysis focussed on crack sealing by agency forces. The 35 states indicating that they would use an automated crack sealing system represent 1800 crews. Based on the survey responses, a crew is involved in crack sealing 25% of the time on average. Therefore, it is assumed that a crack sealing system could be shared between 4 crews. Therefore, it is expected that 450 units (1800 crews/ 4 crews per unit) are required nationally by states intending to used automated crack sealing for work performed by agency forces.

8.5 Reduced Labor Costs due to Automation

An important benefit of automation is the reduced labor costs. The data presented in Table 8-4 indicate that three (3) laborers could be eliminated from the process while still maintaining adequate supervision of the-equipment. These three laborers would normally be involved in cleaning the crack, and using the filling wands. Using the automated system, a crew would then consist of a supervisor, a driver and two flagmen. The expected labor savings are estimated to be:

Figure 8-6: Number of States Using Various Safety Measures



$\frac{3\ \text{Laborers}}{\text{Crew}}$ x \$12/hr * 0.25 * 2000 hrs/yr * $\frac{6\ \text{Months}}{12\ \text{Months}}$ = \$9,000 per year per crew.

This is based on the following assumptions:

- The crew uses the equipment 25% of the time for 6 months of the year.

- Average labor rate is $12/hr.

- Available work time is 2000 hours per year.

8.6 Life Cycle Costs for Automated System

Life cycle costs for the system include acquisition costs, and annual operating and maintenance costs [McNeil 90]. The system acquisition costs are estimated to be $100,000 per unit based on the breakdown of costs given in Table 8-5.

Item	Cost ($ 000s)
Computing	10
Generator and UPS	8
Controllers and Motors	5
Camera and Boom	10
x-y table and trailer	10
Other	17
Engineering, Assembly and Manufacturing	40
Total Capital Cost	100

Table 8-5: Capital Cost Breakdown

Annual maintenance and operating costs of $10,000 per unit per year are based on the following assumptions:

- Software maintenance - $2,500/year

- Energy expenditures - $2,500/year

- Set-up and dismantling costs - $500/year

- Transportation costs between job sites - $1,000/year

- Maintenance and repair - $3,500/year

The system life is assumed to be 6 years based on 6 months operation per year and regular maintenance.

8.7 Net Benefits due to Reduced Labor

Using these cost estimates for the automated system and estimates for the labor savings, the difference in expenditures using the automated rather than the manual method can be developed. Using a discount rate of 5%, a system life of 6 years (with the equipment only being utilized for 6 months in any one year), and productivity rates comparable to existing procedures, the net present value of the additional cost (acquisition, maintenance and operating) of the automated system is \$150,760[3]. When subtracted from the labor savings over the life of the system of \$182,736[4] the net labor savings are \$31,976 per unit over the life of the system or \$6,300 per unit per year. Annual crackfilling expenditures by states were estimated to be \$53.3 million per year and it is expected that 450 units will operate nationally. This gives a national saving of approximately \$14.4 million over the 6 year life of the equipment or \$2.84 million per year, or 5.3% of estimated expenditures by states for crack sealing.

8.8 Other Benefits of Automation

8.8.1 Improved Safety

By substituting robotic systems for manual work in the field, the exposure of workers in unsafe roadway conditions is greatly reduced. With typical injury accident costs of \$1,100 for medical cases and \$21,100 for for restricted activity/lost work day cases [Hinze 91][5] assuming a 1% reduction in reported injury accidents in each year represents a savings of \$180,700 based on thirty (30) medical cost injury accident and seven (7) restricted activity/lost work day accident.

[3] (\$100,000 + (PIA 5%, 6) * 10,000) where (PIA, 5%, 6) is the present value of an annual amount over a 6 year period at a 5% discount rate and is equal to 5.076.

[4] (9,000 * 4 * (PIA, 5%, 6))

[5] Based on reported costs for 249 medical cases and 65 restricted activity/lost workday cases, including indirect costs as 118% and and 206% of direct costs respectively.

In addition to exposure to uncontrolled vehicular traffic, roadway workers applying crack filling material are exposed to volatile organics that can cause dermatoses and respiratory problems, and the equipment and traffic noise may lead to impaired hearing. Furthermore it appears that this procedure can be extended to joint sealing where workers routinely used sand blasting equipment for cleaning.

8.8.2 Improved Quality

While the automated system is not expected to increase crew productivity, the consistency of the crack filling operation can be improved as cracks are accurately identified, uniformly cleaned and potentially, material delivered at a rate appropriate to the depth and width of the crack. Benefits will be derived from improved durability of the pavement due to proper sealing of the crack, longer pavement life and reduced user costs due to delays for resealing pavement cracks or undertaking other maintenance activities.

To illustrate the potential benefits of improved consistency the following example is based on data from [Chong 88]. Consider a crack sealing operation that is to be undertaken 2 years after the pavement is rehabilitated. This extends the pavement life from 12 years to 16 years at which point rehabilitation is required. Assume that more consistent crack filling using the automated method extends the pavement life an additional year. If rehabilitation costs including user delay during rehabilitation are $40,000 per lane km, savings are achieved from the time value of money when the rehabilitation is deferred one year and from the additional year of life the pavement has gained. Over the pavement life this is equivalent to a savings of $145 per lane km[6]. Based on the survey response approximately 77,000 lane km of cracked road are sealed annually. Conservatively, assuming that only 50% of sealed cracks extend the life of the pavement by an additional year (or that the automated system is only used to seal 50% of the cracks) gives a saving of $5.6 m per year. This represents an additional savings of approximately 10% of the cost of crack sealing.

This analysis indicates that significant savings may be realized from improved consistency.

[6] Based on a savings of $(AIP,5\%,17)*40{,}000/(1.05)^{17}-(AIP,5\%,16)*40{,}000/(1.05)^{16}$ where (AIP, i, n) is the annual equivalent over n years at interest rate i.

8.9 Limitations of this Analysis

As this analysis is based on survey responses, the limitations should be noted:

- The survey responses may not represent a random sample due to biases introduced by non-responses.
- Actual labor costs may be higher, as reported labor cost are significantly less than Means figures including overhead. Therefore, larger savings may be realized.
- Analysis focuses on crack sealing by state public works or department of transportation crews that work on crack sealing as just one of many maintenance operations. Additional crack sealing units and ultimately labor savings will be realized as contractors adopt automated crack sealing. Additional savings may also be realized if organizational changes occur and specialized crack sealing crews are used to ensure more effective utilization of the equipment.

To determine the sensitivity of the results to changes in parameters and costs, a sensitivity analysis was performed. The base case with 5% interest rate, 6 year life of system, $100,000 system acquisition costs, $10,000 annual operating and maintenance costs, $12 per hour wage rates was used. The net present value of costs, labor savings and benefits were determined with combinations of the following parameters and the base case:

- Interest rate 3% and 8%
- Expected system life 4 years and 8 years
- Acquisition cost $250,000
- Annual operating and maintenance costs of $8,000 and $12,000
- Hourly wage rate $20 per hour.

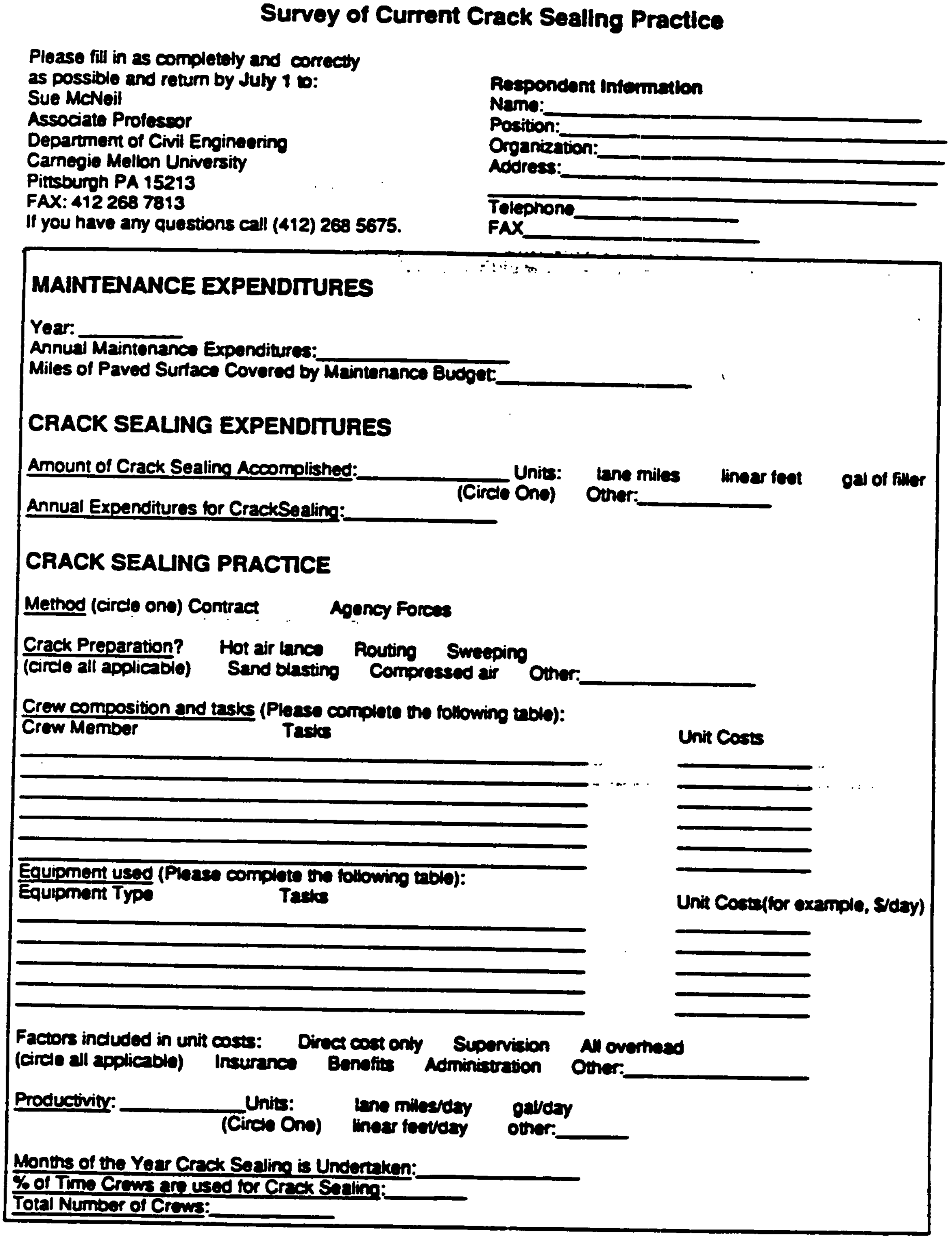
The results of the analysis are included in Appendix II. Review of the results indicates that it is important to minimize costs and maximize the life of the equipment.

8.10 Conclusions

The analysis shows that automated crack sealing is economically feasible and desirable. Assuming the elimination of three laborers, the labor savings of $9,000 per year per crew represents 5.3% of annual crack sealing costs. The analysis is based on crack sealing by agency forces and assumes crack sealing is undertaken 6 months of the years and a crack sealing system (one piece of equipment) is shared between 4 crews. Automation of this process is expected to require about 450 crack filling units nationwide. Furthermore, the economic impacts of improved consistency may be significant.

Appendixes

I. Survey Form - 1991



Survey of Current Crack Sealing Practice

Please fill in as completely and correctly
as possible and return by July 1 to:
Sue McNeil
Associate Professor
Department of Civil Engineering
Carnegie Mellon University
Pittsburgh PA 15213
FAX: 412 268 7813
If you have any questions call (412) 268 5675.

Respondent Information
Name:____________
Position:____________
Organization:____________
Address:____________
____
Telephone____________
FAX____________

MAINTENANCE EXPENDITURES

Year:__________
Annual Maintenance Expenditures:____________
Miles of Paved Surface Covered by Maintenance Budget:____________

CRACK SEALING EXPENDITURES

Amount of Crack Sealing Accomplished:____________ Units: (Circle One) lane miles linear feet gal of filler Other:__________
Annual Expenditures for CrackSealing:____________

CRACK SEALING PRACTICE

Method (circle one) Contract Agency Forces

Crack Preparation? (circle all applicable) Hot air lance Routing Sweeping Sand blasting Compressed air Other:____________

Crew composition and tasks (Please complete the following table):

Crew Member	Tasks	Unit Costs

Equipment used (Please complete the following table):

Equipment Type	Tasks	Unit Costs(for example, $/day)

Factors included in unit costs: (circle all applicable) Direct cost only Supervision All overhead Insurance Benefits Administration Other:____________

Productivity:__________ Units: (Circle One) lane miles/day gal/day linear feet/day other:______

Months of the Year Crack Sealing is Undertaken:____________
% of Time Crews are used for Crack Sealing:________
Total Number of Crews:__________

MATERIALS USED FOR CRACK Sealing

CrackSealing Materials Used: AC cement Asphalt rubber Polymerized Asphalt rubber
(Circle All Applicable) Fiberized Asphalt Emulsified Asphalt
Asphalt cutback Other:____________

Cost/unit:____________ Unit: gal liter lb
(Circle one)
Utilization Rates:____________ Units: gal/lane mile gal/linear foot
(Circle one) l/lane mile l/linear foot

USAGE AND EFFECTIVENESS

1. Are crack Sealing decisions based on condition surveys? Yes No
(Circle One)
2. If yes, what are the indicators for crack Sealing (e.g. range of crack widths such as 5-20 mm)? ____________
3. Is the effectiveness of crack Sealing measured? ____________
4. If yes, how? ____________
5. What is the expected impact of crack sealing on pavement life?____________
6. Over the life of a typical pavement, how may times ____________ and at what intervals ____________is crack sealing applied?

SAFETY

Procedures
What safety procedures are used for crack sealing crews? Lane closure Flagmen
(Circle all applicable) Other:____________

Are these procedures significantly different from other maintenance operations? Yes No
(Circle one)
If, yes describe how they differ?____________

Safety Record

Total Maintenance Forces: __________employees on roads __________crews on roads

Annual number of accidents __________ Property damage only Year________
__________ Injury
__________ Fatality

EXPECTED USAGE

Would your organization consider using an automated crack sealing system assuming it followed state practice and savings exceeded cost of acquisition and maintenance? __________

COMMENTS

Please use this space for any comments or additional relevant information.

II. Sensitivity Analysis

)

ECONOMIC ANALYSIS
BREAK EVEN

MARR		0.1	Maint & Op	25000
	Life			
Savings		3	5	
$104,000		$196,000	$299,000	
$60,000		$87,000	$133,000	

MARR		0.05	Maint & Op	25000
	Life			
Savings		3	5	
$104,000		$215,000	$342,000	
$60,000		$95,000	$152,000	

MARR		0.1	Maint & Op	15000
	Life			
Savings		3	5	
$104,000		$221,000	$337,000	
$60,000		$112,000	$171,000	

MARR		0.05	Maint & Op	15000
	Life			
Savings		3	5	
$104,000		$242,000	$385,000	
$60,000		$123,000	$195,000	

MARR		0.1	Maint & Op	35000
	Life			
Savings		3	5	
$104,000		$172,000	$262,000	
$60,000		$62,000	$95,000	

MARR		0.05	Maint & Op	35000
	Life			
Savings		3	5	
$104,000		$188,000	$299,000	
$60,000		$68,000	$108,000	

COST BENEFIT

MARR	0.1
Maint & O	$25,000
Acq cost	$100,000
Labor sav	$104,000

	3	4	5
Savings	$96,000	$150,000	$199,000
Annual Savings	$39,000	$60,000	$80,000
National	$15,600,000	$24,000,000	$32,000,000
% Nat Sav	12	19	26

MARR	0.05		
Maint & O	$25,000		
Acq cost	$100,000		
Labor sav	$104,000		
	3	4	5
Savings	$115,000	$180,000	$242,000
Ann Sav	$42,000	$66,000	$89,000
National	$16,912,000	$26,458,000	$35,550,000
% Nat Sav	14	21	28

MARR	0.1		
Maint & O	$15,000		
Acq cost	$100,000		
Labor sav	$104,000		
	3	4	5
Savings	$121,000	$182,000	$237,000
Ann Sav	$49,000	$73,000	$95,000
National	$19,515,000	$29,293,000	$38,182,000
% Nat Sav	16	23	31

MARR	0.05		
Maint & O	$15,000		
Acq cost	$100,000		
Labor sav	$104,000		
	3	4	5
Savings	$142,000	$216,000	$285,000
Ann Sav	$52,000	$79,000	$105,000
National	$20,912,000	$31,667,000	$41,909,000
% Nat Sav	17	25	34

MARR	0.1		
Maint & O	$35,000		
Acq cost	$100,000		
Labor sav	$104,000		
	3	4	5
Savings	$72,000	$119,000	$162,000
Ann Sav	$29,000	$48,000	$65,000
National	$11,515,000	$19,096,000	$25,987,000
% Nat Sav	9	15	21

MARR	0.05

Maint & O	$35,000		
Acq cost	$100,000		
Labor sav	$104,000		
	3	4	5
Savings	$88,000	$145,000	$199,000
Ann Sav	$32,000	$53,000	$73,000
National	$12,912,000	$21,250,000	$29,191,000
% Nat Sav	10	17	23

MARR	0.1		
Maint & O	$25,000		
Acq cost	$100,000		
Labor sav	$60,000		
	3	4	5
Savings	($13,000)	$11,000	$33,000
Ann Sav	($5,000)	$4,000	$13,000
National	($2,085,000)	$1,761,000	$5,256,000
% Nat Sav	-2	1	4

MARR	0.05		
Maint & O	$25,000		
Acq cost	$100,000		
Labor sav	$60,000		
	3	4	5
Savings	($5,000)	$24,000	$52,000
Ann Sav	($2,000)	$9,000	$19,000
National	($688,000)	$3,541,000	$7,569,000
% Nat Sav	-1	3	6

MARR	0.1		
Maint & O	$15,000		
Acq cost	$100,000		
Labor sav	$60,000		
	3	4	5
Savings	$12,000	$43,000	$71,000
Ann Sav	$5,000	$17,000	$28,000
National	$1,915,000	$6,859,000	$11,353,000
% Nat Sav	2	5	9

MARR	0.05		
Maint & O	$15,000		
Acq cost	$100,000		
Labor sav	$60,000		
	3	4	5
Savings	$23,000	$60,000	$95,000
Ann Sav	$8,000	$22,000	$35,000
National	$3,312,000	$8,750,000	$13,928,000

% Nat Sav	3	7	11

MARR	0.1		
Maint & O	$35,000		
Acq cost	$100,000		
Labor sav	$60,000		
	3	4	5
Savings	($38,000)	($21,000)	($5,000)
Ann Sav	($15,000)	($8,000)	($2,000)
National	($6,085,000)	($3,338,000)	($841,000)
% Nat Sav	-5	-3	-1

MARR	0.05		
Maint & O	$35,000		
Acq cost	$100,000		
Labor sav	$60,000		
	3	4	5
Savings	($32,000)	($11,000)	$8,000
Ann Sav	($12,000)	($4,000)	$3,000
National	($4,688,000)	($1,667,000)	$1,210,000
% Nat Sav	-4	-1	1

III. Project Bibliography

Haas, C. and C. Hendrickson, "Computer-Based Model of Pavement Surfaces," *Transportation Research Record*, No. 1260, pp. 91-98, 1990.

Haas, C., "A Model of Pavement Surfaces", PhD Thesis, Carnegie Mellon University, Department of Civil Engineering, 1990.

Haas, C., C. Hendrickson and S. McNeil, "A Design for Automated Pavement Crack Sealing," *Preparing for Construction in the 21st Century*, (L.-M. Chang, ed.), ASCE, 1991.

Hendrickson, C., S. McNeil, D. Bullock, C. Haas, D. Peters, D. Grove, K. Kenneally and S. Wichman, "Perception and Control for Automated Pavement Crack Sealing," *Applications of Advanced Technologies in Transportation Engineering*, ASCE, 1991.

Haas, C., S. McNeil, C. Hendrickson and R. Haas, "A Pavement Surface Model for Integrating Management Data," *Pavement Management Implementation*, ASTM STP 1121, (F.B. Holt and W.L. Gramling, eds), American Society for Testing and Materials, Philadelphia, 1991.

Hendrickson, C., S. McNeil, D. Bullock, and C. Haas, "Crack Filling Robot", Proceedings of ASCE Specialty Conference on Strategic Highway Research Program Products, Denver, 1991.

Hendrickson, C. and S. McNeil, "Robotic Pavement Crack Sealing", Final Report, SHRP IDEAS Project 87-ID017, Strategic Highway Research Program, Washington, DC, 1991.

McNeil, S., "An Analysis of the Costs and Impacts of the Automation of Pavement Crack Sealing," Proceedings Sixth World Conference on Transport Research, Lyon, 1992.

Haas, C., C. Hendrickson, S. McNeil and D. Bullock, "A Field Prototype of a Robotic Pavement Crack Sealing System," Proc. Ninth ISARC Conference, Tokyo, 1992.

McNeil, S., C. Haas and C. Hendrickson, "Automated Pavement Crack Sealing: Prototype Design and Evaluation," *Transportation Research Record*, to appear, 1992.

Haas, C. and C. Hendrickson, "Integration of Diverse Technologies for Pavement Sensing," *Transportation Research Record*, to appear, 1992.

McNeil, S., C. Hendrickson, D. Bullock, W. Shi, P. Simon, A. Pearce, C. Haas, J. Deng and A. Zacherl, "Investigation of a Pavement Crack-Filling Robot," Final Report, Strategic Highway Research Program, 1992.

K. Kenneally, "Process Overview", Department of Civil Engineering, Carnegie Mellon University.

D. Peters, "Path Planning for a Robotic Pavement Crack Filler", Department of Civil Engineering, Carnegie Mellon University.

D. Grove, "Crack Filler Command Language Interpreter (CFCLI) Users Guide", Department of Civil Engineering, Carnegie Mellon University.

D. Grove, "Crack Filler Command Language Interpreter Design Document", Department of Civil Engineering, Carnegie Mellon University.

C. Haas, "Generalized Quadtree Library Applications Development Guide", Department of Civil Engineering, Carnegie Mellon University.

S. Wichman, "Documentation of Perception Experiments", Department of Civil Engineering, Carnegie Mellon University.

S. Wichman, "Documentation of Imaging Hardware and Software", Department of Civil Engineering, Carnegie Mellon University.

K. Kenneally, "Technical Documentation for Table and Cart Assembly", Department of Civil Engineering, Carnegie Mellon University.

D. Peters, "The DT2805 Software Interface", Department of Civil Engineering, Carnegie Mellon University.

S. McNeil, "Economic Evaluation of Automated Pavement Crack Filling", Department of Civil Engineering, Carnegie Mellon University.

References

[AASHTO 87] AASHTO committee.
AASHTO Maintenance Manual
Published by the American Association of State Highway and Transportation Officials, Washington, D.C., 1987.

[Allen 84] Allen, P.
Surface Descriptions From Vision and Touch.
In *IEEE 1984 International Conference on Robotics*, pages 394. Computer Society Press, 1984.

[Ballard 82] Ballard, D.H., and Brown, C.M.
Computer Vision.
Prentice Hall Inc., Englewood Cliffs, NJ., 1982.

[Bomar 88] Bomar, L.C., W.F. Horne, D.R, Brown and J.L. Smart.
NCHRP PROJECT 10-28:A Method to Determine Deteriorated Areas in Portland Cement Concrete Pavements.
Technical Report, Gulf Applied Research, Electromagnetics Facility, Marietta, Georgia, January, 1988.

[Brown 88] Brown, E.R.
Preventive Maintenance of Asphalt Concrete Pavements.
Transportation Research Record (1205), 1988.

[Burton 89] Burton, F.W., and Kollias, J.G.
Functional Programming with Quadtrees.
IEEE Software 6(1):90-8, Jan, 1989.

[Butler 89] Butler, B.C. Jr.
Pavement Surface Distress Segmentation Using Real Time Imaging.
In C.T. Hendrickson and K. Sinha (editor), *Proceedings of the First International Conference on Applications of Advanced Technology to Transportation.* ASCE, February, 1989.

[Castleman 79] Castleman, K.R.
Signal Processing Series: Digital Image Processing.
Prentice-Hall, 1979.

[Chiu 86] Chiu, S.L., Morley, D.J., and Martin, J.F.
Sensor Data Fusion on a Parallel Processor.
In *Robotics and Automation - Proc. of 1986 IEEE Conf.*, pages 1629. Computer Society Press, Washington, D.C., 1986.

[Chong 88] Chong, J.G. and W.A. Phang.
Improved Preventive Maintenance: Sealing Cracks in Flexible Pavements.
Transportation Research Record (1205), 1988.

[Cochran 77] Cochran, W.G.
Sampling Techniques.
Wiley, 1977.

[Crowley 87] Crowley, J.L. and Ramparany, F.
Mathematical Tools for Representing Uncertainty in Perception.
In Kak, A., and Chen, S. (editor), *Spatial Reasoning and Multi-Sensor Fusion - Proceedings of the 1987 Workshop*, pages 293. Morgan Kaufmann Publishers, Inc., Los Altos, California, October, 1987.

[Duncan 87a] Duncan, J.S., and Staib, L.H.
Shape Determination from Incomplete and Noisy Multisensor Imagery.
In *Spatial Reasoning and Multi-Sensor Fusion - Proceedings of the 1987 Workshop*, pages 334. Morgan Kaufmann Publishers, Inc., Los Altos, California, October, 1987.

[Duncan 87b] Duncan, J.S., Gindi, J.R., and Narendra, K.S.
Low Level Information Fusion: Multisensor Scene Segmentation Using Learning Automata.
In *Spatial Reasoning and Multi-Sensor Fusion - Proceedings of the 1987 Workshop*, pages 323. Morgan Kaufmann Publishers, Inc., Los Altos, California, October, 1987.

[Durrant-Whyte 86]
Durrant-Whyte, H.F.
Consistent Integration and Propogation of Disparate Sensor Observations.
In *Robotics and Automation - Proc. of the 1986 IEEE Conf.*, pages 1464. Computer Society Press, Washington, D.C., 1986.

[Durrant-Whyte 87]
Durrant-Whyte, H. F.
Sensor Models and Multi-Sensor Integration.
In *Spatial Reasoning and Multi-Sensor Fusion: Proceedings of the 1987 Workshop*, pages 303-312. Morgan Kaufmann Publishers, Inc., Los Altos, California, October, 1987.

[Elfes 88] Elfes, A., and Matthies, L.
Integration of Sonar and Stereo Range Data Using a Grid Based Representation.
In *Robotics and Automation - Proc. of the 1988 IEEE Conf.*, pages 727. Computer Society Press, Washington, D.C., April, 1988.

[Faugeras 86] Faugeras, O.D., Ayache, N., Faverjon, B., and Lustman, F.
Building Visual Maps By Combining Noisy Stereo Measurements..
In *Robotics and Automation - Proc. of the 1986 IEEE Conf.*, pages 1433. Computer Society Press, Washington, D.C., 1986.

[FHWA 87] Federal highways Administration.
Value Engineering Study of Transverse Crack Repair in Asphalt Concrete Pavements.
Final Report DTFH61-87-6-0064, Federal Highway Administration, 1987.

[Fukuhara 90] Fukuhara, T., Terada, K., Nagao. M., Kasahara, A., and Ichihashi, S.
Automatic Pavement-Distress-Survey System.
Transportation Engineering 116(3):280-286, 1990.

[Grove 90] Grove, D.E.
Crack Filler Command Language Interpreter.
Technical Report 1.1, Civil Engineering, CMU, Pittsburgh, PA, 8, 90.

[Haas 84] Haas, C., Shen, H., and Haas, R.
Application of Image Analysis Technology to Automation of Pavement Condition Surveys.
In *Proc., Int. Transport Congress, Montreal.* September, 1984.

[Haas 90a] Haas, C. and C. Hendrickson.
A Model of Pavement Surfaces.
Technical Report R90-191, Dept. of Civil Engr., Carnegie Mellon Univ., Pittsburgh, PA 15213, September, 1990.

[Haas 90b] Haas, C.
Generalized Quadtree Library (GQL) Software Development Guide.
Technical Report R90-190, Department of Civil Engineering, CMU, Pittsburgh, PA, 1990.

[Henderson 87] Henderson, T.C., Allen, P., Cox, I., Mitiche, A., Durrant-Whyte, H., and Snyder, W. (editor).
A Report on the Workshop on Multisensor Integration in Manufacturing Automation.
Sponsored by U. of Utah and the NSF, Utah, 1987.
pub. unknown - work supported by NSF grant no. DMC-8612180.

[Hinze 91] Hinze, J.
Indirect Costs of Construction Accidents.
Source Document 67, Construction Industry Institute, 1991.

[Huntsberger 87] Huntsberger, T.L., and Jayaramamurthy, S.N.
A Framework for Multi-Sensor Fusion in the Presence of Uncertainty.
In Kak, A., Chen, S. (editor), *Spatial Reasoning and Multi-Sensor Fusion - Proceedings of the 1987 Workshop.* Morgan Kaufmann Publishers, Inc., Los Altos, California, October, 1987.

[Lu 86] Lu, H.E., and Wang, P.S.P.
A Comment on 'A Fast Parallel Algorithm for Thinning Digital Patterns'.
Communications of the ACM 29(3):239-42, March, 1986.

[Luo 88] Luo, R.C., and Lin, M.H.
Robot Multisensor Fusion and Integration: Optimum Estimation of Fused Sensor Data.
In *Robotics and Automation - Proc. 1988 IEEE Conf.*, pages 1076. Computer Society Press, Washington, D.C., April, 1988.

[Maser 87] Maser, K. R.
Computational Techniques for Automating Visual Inspection.
Internal CCRE Report, Dept. of Civil Eng., MIT.
November, 1987

[McNeil 89] McNeil, S. and Humplick, F.
Evaluation of Errors in Pavement Surface Distress Data Acquisition.
In Hendrickson and Sinha (editor), *Proceedings of the First International Conference on Applications of Advanced Technology to Transportation.* Purdue, February, 1989.

[McNeil 90] McNeil, S.
An Analysis of the Costs and Benefits of Automated Pavement Crack Filling.
Working Paper - SHRP IDEA 87 ID017, Carnegie Mellon University, 1990.

[McNeil 91] McNeil, S.
Survey of Current Crack Sealing Practice.
Working Paper, Strategic Highway Research Program, Carnegie Mellon University, 1991.

[Means 89] Cleveland, A.B. (editor).
Building Construction Cost Data 1990.
R.S. Means Company Inc., Kingston MA, 1989.

[Mendelsohn 87] Mendelsohn, D.H.
Automated Pavement Crack Detection: An Assessment of Leading Technologies.
In *Second North American Conference on Managing Pavements: Proceedings*, pages 3.297. MTC of Ontario and FHWA, November, 1987.

[Peters 90a] Peters, D.
Path Planning for a Robotic Pavement Crack Filler.
Technical Report 1.11, Civil Engineering, CMU, Pittsburgh, PA, August, 1990.

[Peters 90b] Peters, D.
The DT2805 Software Interface For 1st Generation Robotic Pavement Crack Filler.
Technical Report 1.3, Civil Engineering, CMU, Pittsburgh, PA, August, 1990.

[Peters 90c] Peters, D.
The Table Control Language For 1st Generation Robotic Pavement Crack Filler.
Technical Report 1.4, Civil Engineering, CMU, Pittsburgh, PA, August, 1990.

[Richardson 84] Richardson, J.M., Marsh, K.A., and Martin, J.F.
Techniques of Multisensor Signal Processing and Their Application to the Combining of Vision and Acoustical Data.
In *Fourth International Conf. on Robot Vision and Sensory Controls.* IEEE, October, 1984.

[Samet 83] Samet, H.
A Quadtree Medial Axis Transform.
Communications of the ACM 26(9):680-93, September, 1983.

[Shaffer 86] Shaffer, C.A.
Application of Alternative Quadtree Representations.
PhD thesis, University of Maryland, June, 1986.
also Computer Science Tech. Report CS-TR-1672.

[Skibniewski 90] Skibniewski, Miroslaw and Chris Hendrickson.
Automation and Robotics for Road Construction and Maintenance.
ASCE J. of Transportation Engineering , May, 1990.

[Stroustrup 87] Stroustrup, B.
The C++ Programming Language.
Addison-Wesley Publishing Company, Reading, MA, 1987.

[Wigan 87] Wigan, M. and Cullinan, M.C.
Image Processing Applied to Roads: Surface Texture, Mensuration, Vehicle Shape and Number Detection.
Special Report 145, Australian Road Research Board, January, 1987.

[Zhang 84] Zhang, T.Y., and Suen, C.Y.
A Fast Parallel Algorithm for Thinning Digital Patterns.
Communications of the ACM 27(3):236-9, March, 1984.