Automated Find Fix & Eliminate for Automotive Paint Systems

Valerie Bolhouse
Advanced Manufacturing Technology Development
Ford Motor Company

Find Fix & Eliminate (hereafter referred to as FFE) is a new process for automotive paint that automates the inspection and repair process seen after e-coat or prime. The system is the product of multiple discrete technologies integrated into an automated system that can improve the repair process in a paint shop by providing a computer controlled repair operation with real time automated data logging of defects. The automated process will reduce in-plant repair costs and customer warranty claims by:

- Fewer missed dirt locations
- Reduced sand-through and sand scratch caused by aggressive manual sanding
- Defects detected where repair is minimally invasive to final finish
- Less dirt and sludge generated by eliminating unnecessary sanding and restricting the sanding operation to the location of the defect
- Real time data feedback to eliminate the sources of dirt defects.

FFE was developed at Ford's Advanced Manufacturing Technology Development Center (AMTD) with three supplier partners and piloted at the Ford Wayne Assembly Plant. There are four primary subsystems developed and integrated to perform the dirt-in-prime "find and fix" function. The block diagram for the FFE system is shown in Figure 1. Each of the major subsystems will be discussed separately below.

1. A vision scanner (PSI®) which finds defects in the paint surface;

2. A vision cell controller (RepairMan®) which receives defect information from the optical scanners, provides coordinate transformation between the vision, robot, and vehicle, stores the defect data in a database, and provides an analysis of defect trends for process improvement;

3. A robot cell controller (RepairWorks®) which retrieves defect data from the database, plans the repair strategy, generates repair paths which get communicated to the specified robots, and updates the defect database with the repair history; and

4. An automated robotic repair system from Fanuc Robotics with force-feedback sensors that perform the repair process.

The "eliminate" function uses the real time data logging of dirt defects to immediately detect trends or alarm conditions for elimination of the source of defects. A defect database is maintained that can be sorted by any number of attributes, such as time, vehicle style, vehicle color, or dirt location. Data clustering is used to analyze historical data for trends not recognizable to an individual observer of the process. Current dirt data trends are used to immediately notify upstream production when an event causes dirt counts to spike.

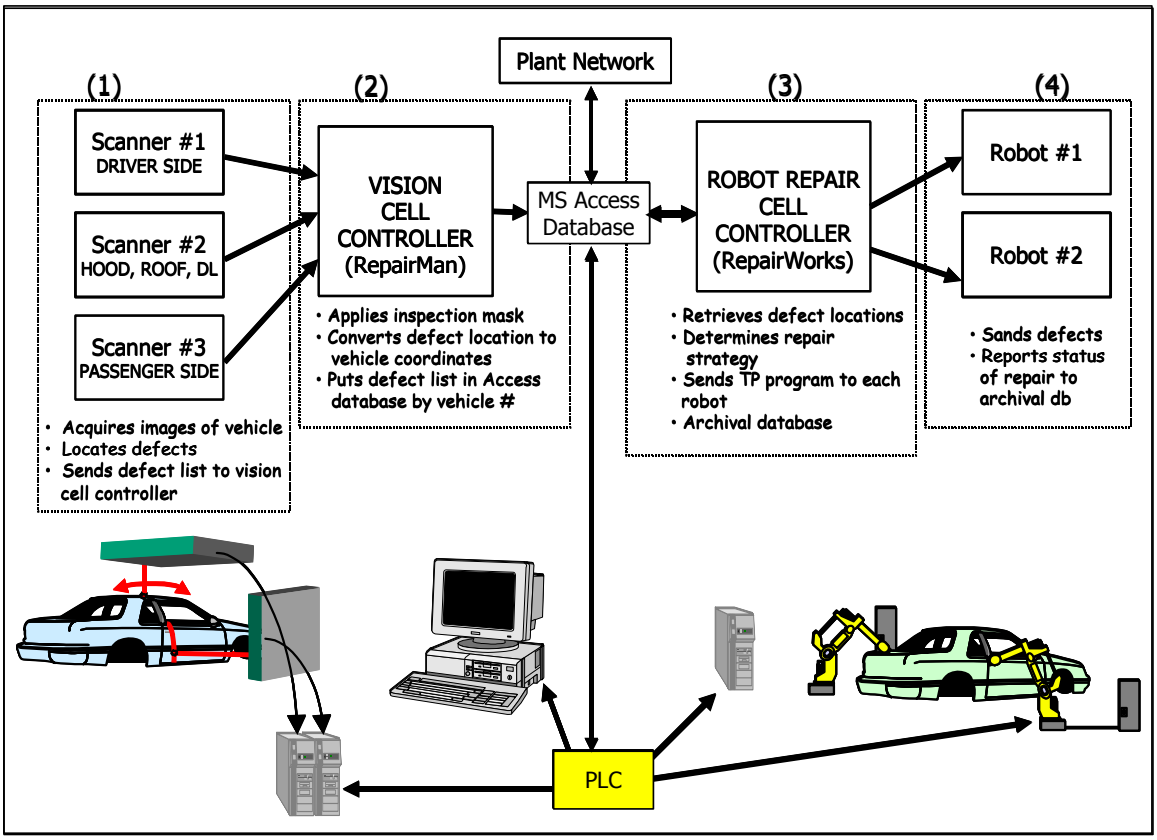

Figure 1. System Architecture of the Find Fix & Eliminate System

FUNCTIONALITY OF THE FIND & FIX SUB-SYSTEMS.

1. VISION SCANNER

The vehicle is inspected on a moving line. A fixed arch of optical detectors is set up around the line to acquire a full body image of the vehicle surface. The system developed at AMTD uses a custom laser scanner developed under contract for Ford by Adaptive Optics Associates (AOA), Cambridge, MA. The Paint Surface Inspection (PSI®) scanner obtains an image of the vehicle surface by raster scanning the laser light on the vehicle moving on a conveyor at line speeds up to one foot per second. Multiple scan lines are assembled to form a two-dimensional grayscale image of the entire vehicle. The resolution and image format are user-programmable by scan rate, line rate, digitization rate and the total number of scan lines. The image shown in Figure 2 is a 30" X 180" picture of the Ford Focus wagon scanned at about 100 dots per inch, resulting in a two dimensional (2D) bitmap of 3,000 X 18,000 pixels.

Topographic defects in paint are identified as regions with rapid changes in grayscale intensity. A vision computer processes the grayscale image data using conventional gradient-based image processing techniques. Potential defects are identified, located, and sized, then stored in a defect

table. The scanner can find topographic defects that cause a 200-micron or greater disturbance in the paint surface (Figure 3).

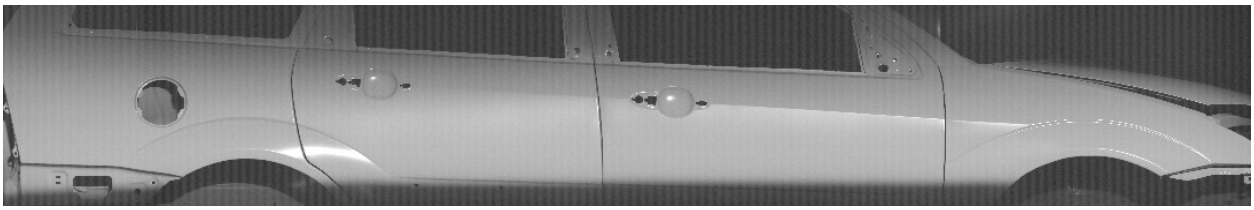

Figure 2. Grayscale Image of the Ford Focus Wagon. The image size is 3,000 by 18,000 pixels with 12 bits of grayscale resolution.

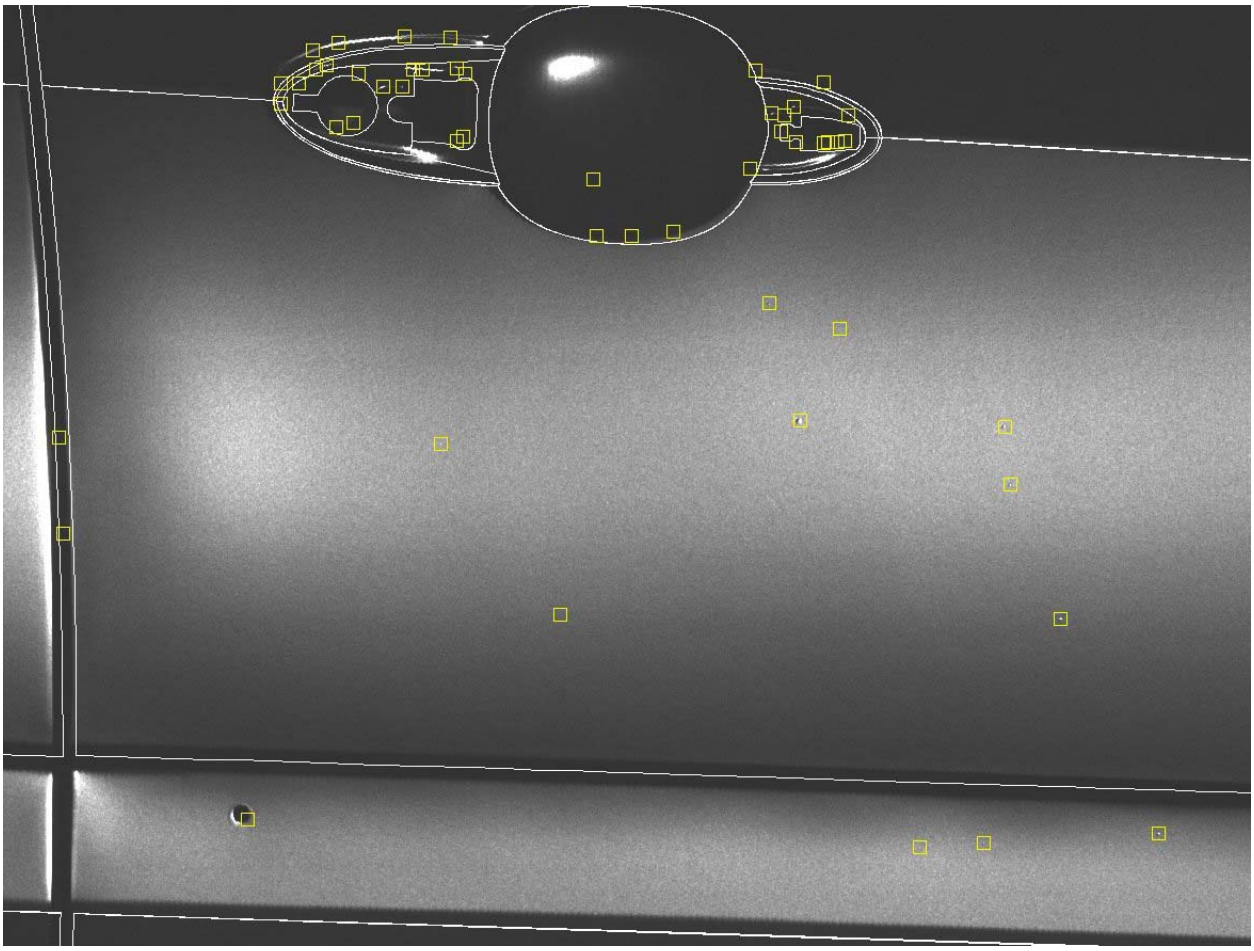

Figure 3. Image of passenger side front door with dirt defects and also glints from edges and character lines highlighted by the vision system.

2. VISION CELL CONTROLLER

The vision scanner locates defects in x, y image or pixel space. This coordinate system would be sufficient for a manual repair operation where an operator could view an image of the vehicle with the defects labeled to assist him in locating the defect on the identified panel. However, if you wish to automate the system using robotic repair, it is necessary to know the location of the defect in real world coordinates with six degrees of freedom.

Surface angle is important because the sandpaper must be placed flat against the vehicle surface at the location of the defect to ensure a repair process that cannot be seen after base and clear coat application. This repair position is fully described using the x, y, z point and the surface normal vector. All points on the vehicle surface are contained in the CAD geometry of the vehicle, which is used as the master for the Find Fix & Eliminate system.

Using the vehicle CAD as the master for the FFE coordinate system provides multiple advantages:

1. It dissociates the inspection from repair in time and space, a major benefit when implementing a cell in an existing facility. Inspection can occur right after the oven while the repair can be implemented at any convenient location down the line.

2. The calibration of the robot workspace is independent of the vision coordinate space. Each can be accomplished and verified off-line.

3. Only one system in the integrated cell requires the CAD data—all other equipment can work in their own native coordinate system.
    a. The vision scanner uses pixel coordinates. The vertical or horizontal resolution can be adjusted as needed without impacting the calibration or coordinate system of the rest of the cell.
    b. The robot cell uses world coordinates, and does not require any vehicle CAD data. Vehicle models can be modified or added without any modification, training, or recalibration in the robot cell.

4. Alignment of the vision system to the vehicle is not critical and can be done with a tape measure, level, and plumb bob. Calibration of the vision coordinates to CAD is accomplished through a simple process on the vision cell computer with an intuitive Graphical User Interface (GUI) to line up edges on a grayscale image.

Coordinate Transformation.

The process for the coordinate transformation between image space and real world vehicle coordinates, as well as a number of the setup and alignment features of the implemented system are unique to the Ford FFE development (patent pending). These functions occur in the vision cell controller. This controller, an AMTD and Matthews Software Inventions product, is a Windows 2000 computer networked to the PSI vision scanner and the plant local area network. The system software, RepairMan®, is developed to provide an intuitive graphical interface such that the system can be easily setup and calibrated with minimal training and no special tools.

The accuracy of the coordinate translation between the x, y image space and the 3D vehicle coordinate space is critical in defining system performance. It determines the minimum sanding disk size you can use with assurance that you will remove all identified defects. It also defines the minimum mask, or keep-out regions, on the vehicle surface for safe repair and valid inspection data.

Inspection and Repair Masks.

RepairMan® automatically generates both inspection and repair masks, based upon user-defined design rules and system tolerances. The masks need to accommodate variations in vehicle build, location of the vehicle on the skid, robot accuracy, and any other expected system variations, as well as the mechanical constraints of the tooling.

The inspection mask is required to filter out highlights and glints from edges and character lines that are flagged by the vision system but are not real defects. The mask is also set up to ignore flagged defects within a preset region from any edge or character line. The size of this region is set by the alignment tolerance of the acquired images to the calibration image. It is determined by the expected variability seen in vehicle build and location on the skid, as well as vision system variability such as line rate, trigger delay, and image distortion.

A repair mask identifies how close the sanding tool can get to edges, character lines and corners without risk of sanding through the prime. This tolerance includes all the error seen in the vision cell plus expected error in the robot system. If tolerances can be kept tight, then the automated find and fix operation will cover a greater portion of the vehicle surface for maximum benefit.

How Masks Are Auto-Generated.

The inspection masks are automatically generated by Repairman®. Boundary regions are grown inward from the edges defined in the 2D CAD line image the preset number of pixels to equal the inspection mask parameter dimensions. A global tolerance is set up for all lines in the CAD image, with the ability to select individual lines and associate a unique local parameter. This allows you to sand closer to some edges (outside door edge, for example) while keeping the tool clear of other obstructions (hinged door edge).

RepairMan® also has the ability to automatically generate repair masks—regions of the vehicle that can be automatically repaired by the robots. The repair mask generation is considerably more complex than the inspection mask. It requires additional processing to accommodate geometries identified by the repair tooling. Circular orbital sanders are used for repair in prime. The entire surface of the sanding tool must fit within the geometry of the surface being repaired. The sander cannot touch a character line in order to prevent "burn throughs" (areas where the sanding operation removes well below the primer and exposes bare metal). Tight corners and small regions constrained by character lines create accessibility issues for a circular sanding tool of finite dimension.

RepairMan® uses the calculated inspection mask and user-defined sanding parameters to determine the region within all boundaries where the sanding tool can fit. It does this by fitting a circle the diameter of the sander within the closed boundaries. This region is shown in white on Figures 4 and 5. Defects outside of the white region cannot be reached by the repair tool without

portions of the tool surface (sanding disk) touching a character line or extending over the edge of the panel.

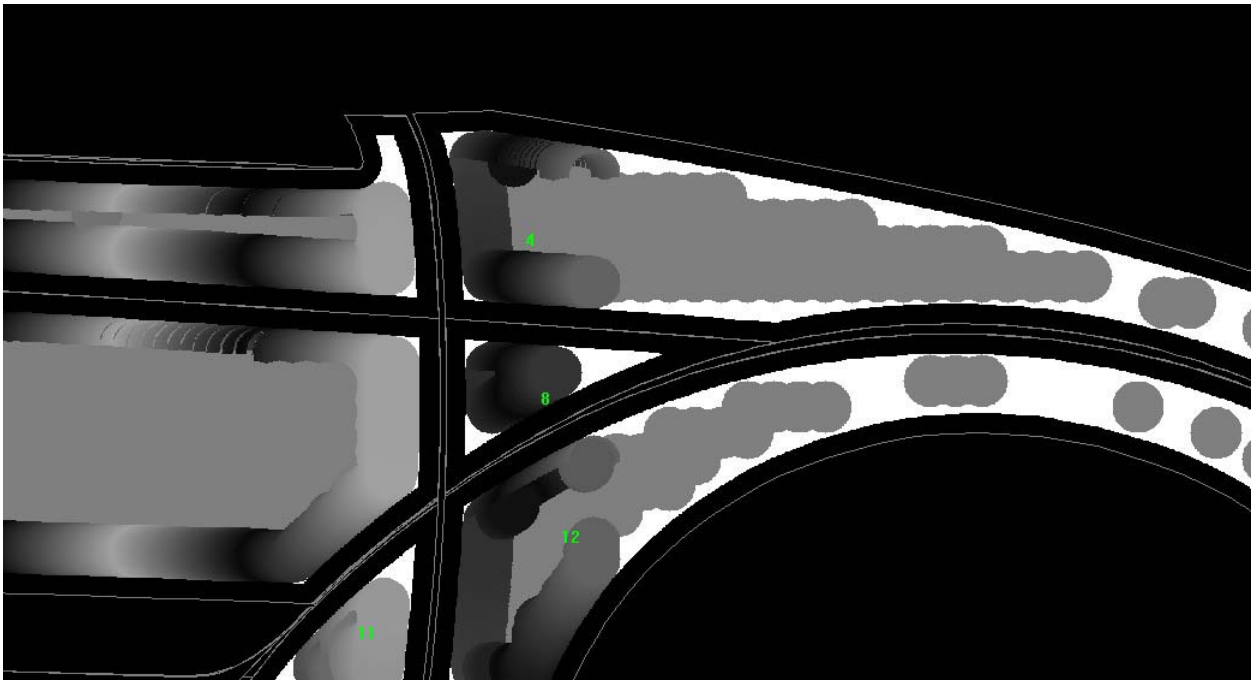

Figure 4. Computer Fitting Circular Sanding Disk in the Inspection Mask to Determine Valid Repair Regions

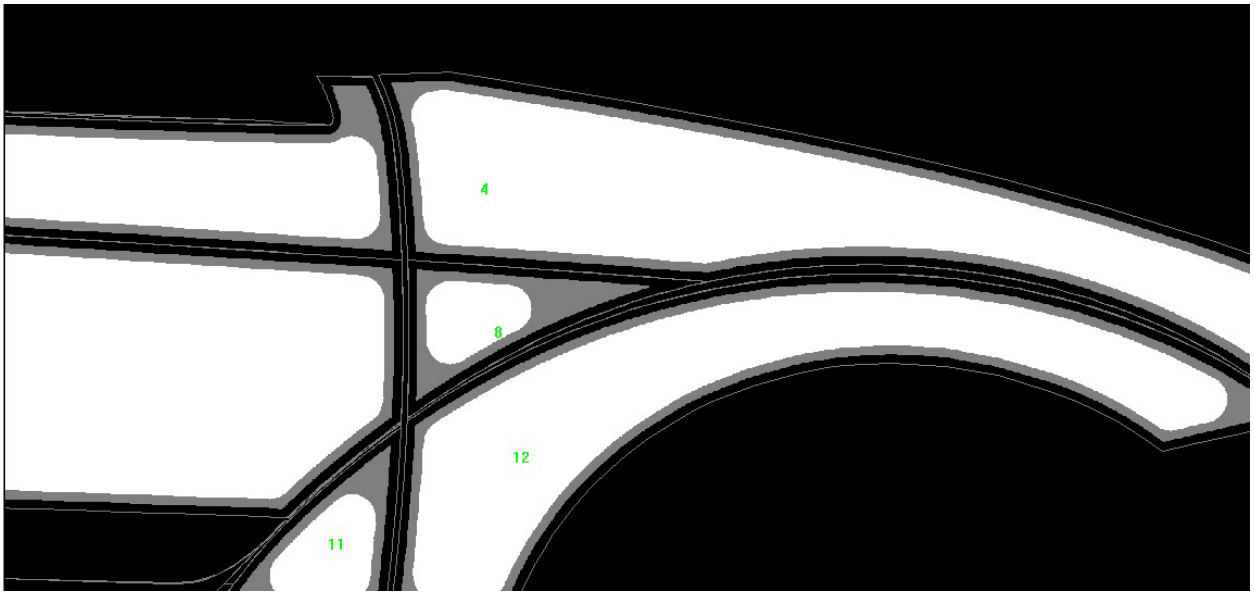

Figure 5. Resulting Inspection and Repair Masks. White defines regions where the sander fits without touching edges or character lines. Gray defines valid inspection region that cannot be automatically repaired. Black is non-valid region.

How Masks Are Used to Filter and Process Defect Data.

During system operation, defects are passed from the scanner to the RepairMan® vision cell controller in x, y pixel space. The defect locations are compared to the locations in the masks. If the location falls within a white region, it is classified as a repairable defect. If the location falls within a gray region, it is classified as a non-repairable defect. If it falls outside of the gray region in the black area, it is ignored. Defects identified by the scanner in the black region are typically caused by glints on the vehicle character lines, weld marks, edges, underbody, or vehicle skid. Only those points identified as real defects (both repairable and nonrepairable) are transformed into 3D vehicle coordinates using an inverse ray tracing process.

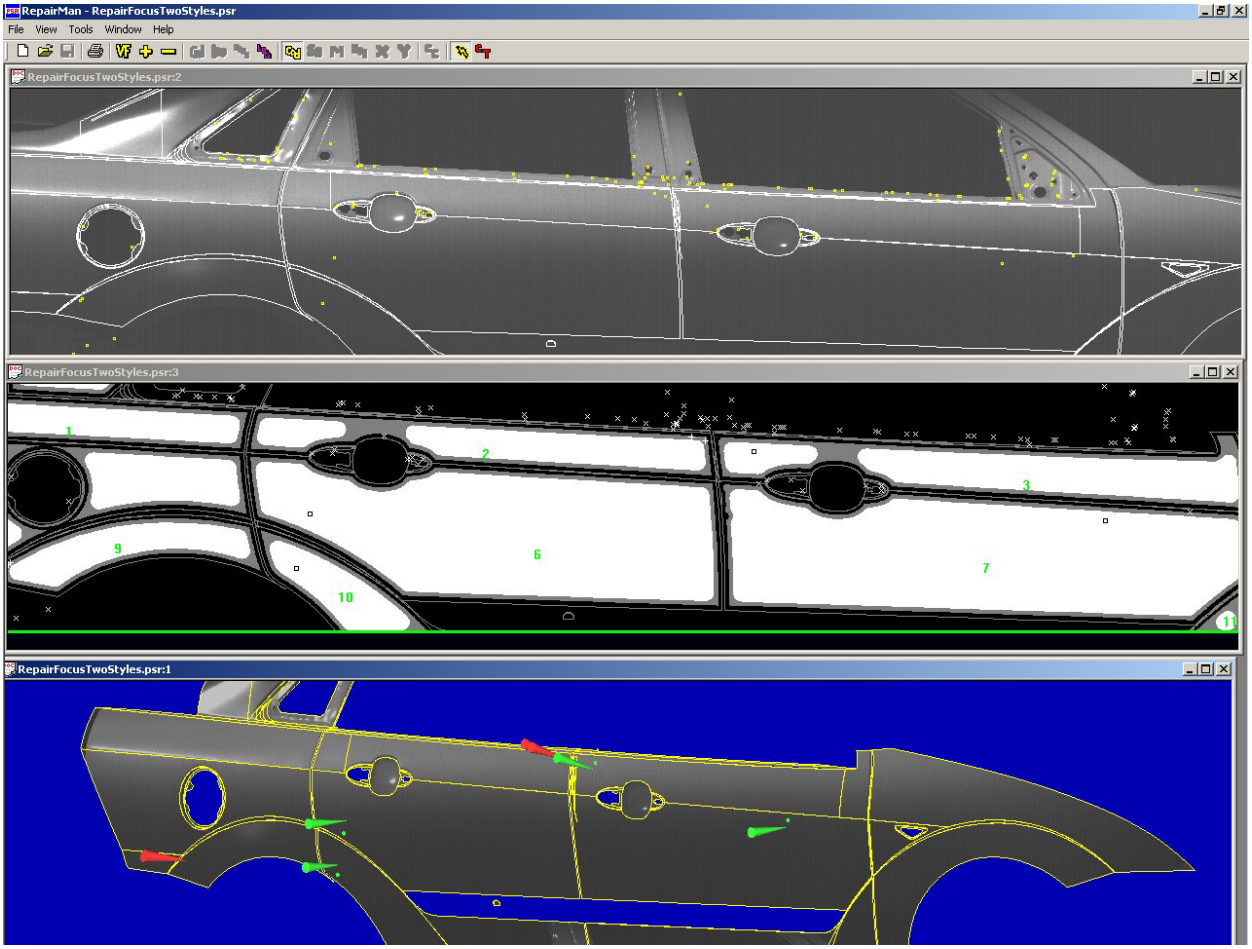

Figure 6. Vision cell controller operator screen. The top window shows defects as found by paint scanner, middle window shows the inspection and repair mask, and the bottom window shows defect locations as a 3D point plus surface normal vector, as required by the robot repair cell. The dot next to the defect vector is the closest mask edge, additional data required by the robot cell controller to keep the sanding tool away from the edges and character lines.

Figure 6 shows the three steps performed by the vision cell controller to transform vision defect data to repair coordinates:

1. Defects identified by the paint scanner are acquired by the vision cell controller.
2. Defect locations are checked against the inspection and repair masks.
3. Defect locations are transformed from 2D pixel coordinates into 3D vehicle coordinates. Transformed defects are stored in a Microsoft Access database and sorted by vehicle ID. See Figure 7 for the database structure.

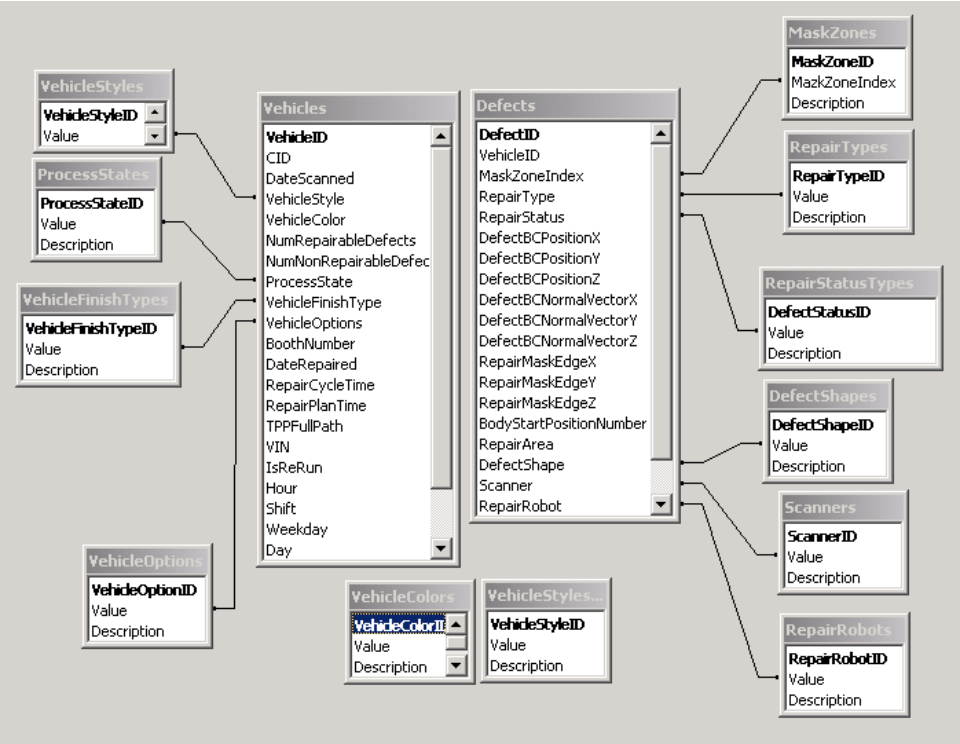

Figure 7. Defect database structure.

3. ROBOTIC REPAIR CELL CONTROLLER

At some point later in time, the vehicle enters the repair cell. The ID number is read from the vehicle tag, and the database is searched for the status of the inspection. If the vehicle was inspected by the scanner and repairable defects found, the vehicle is stopped in the station for automated robotic repair. Defects are retrieved from the database by the robotic repair cell controller, Repair Works®, and a repair program to be executed by the robot is generated.

This controller, a Fanuc-developed product, is a Windows 2000 computer networked to the vision cell controller and the robot controllers. Its function is to develop the repair strategy and generate the robot program for each vehicle. The repair strategy would include assigning the defect locations to a repair robot, determining repair procedure, attack angle, and contact pressures dependent upon size, type, and location of the defect. The robot program contains all moves and logic required to accomplish the repair. The RepairWorks® cell controller

downloads the auto-generated program to the Fanuc robot controller, initiates the repair, and receives confirmation that all repairs have been completed.

How the Repair Process Is Automated.

The vehicle is repaired in a stop station. Future developments in line tracking could enable repair on a moving line. A skeleton path is trained for each vehicle model. This defines a safe travel path for the robot and tool about the vehicle. Generally, each panel of the vehicle (rear fender, rear door, front door, front fender) will require two trained points to define a region. The path definition is not critical—close enough to the vehicle to minimize robot travel for best cycle time, yet far enough away to handle the unforeseen in production. Six to eight inches works well.

The RepairWorks® cell controller inserts the required points and motion parameters into the skeleton path to do the sanding repair. A teach pendant (TP) program is generated which includes the skeleton path plus all moves for the repair operations. During operation, the robot moves from the trained skeleton path to the defect location at a pre-determined stand off distance from the surface of the vehicle. It then moves along the normal to the surface to the contact point. A circle or figure eight is traversed on the surface. The radius of the circular motion, forward velocity, contact force and total contact time are user-settable parameters that are incorporated into the repair program based on defect size and location. Once the repair motion is complete, the robot moves off the vehicle surface to a new location above the next defect in that panel. Any number of repair motions can be inserted within the skeleton path to complete all repairs on the vehicle.

Development of an Intelligent Repair Strategy for the Vehicle.

Intelligence is incorporated in the automated path generation to handle special cases: clustered defects (defined as defects within a region smaller than the radius of the sander) and defects close to edges and character lines ("close" is defined to be a distance smaller than the radius of the sander). This is necessary to optimize cycle time and ensure uniform material removal. If the repair sequence was set up to send the center of the sanding disk (robot tool center point) to each and every defect location stored in the database and sand about that point, the following undesirable consequences could occur:

1. Defects close to an edge or character line would result in sanding over the edges or character lines, possibly burning through the prime to e-coat or bare metal.

2. Clustered defects would result in sanded regions that overlap, causing too much material removal, possibly sanding clear through the prime.

3. Additional cycle time for handling close defects separately, compared with a grouped defect strategy.

The user defines the sanding disk radius and the effective sanding diameter. This information is stored in a process database, and is used by the Repair Works® system to determine how close the robot tool center point can be to an edge or character line and the maximum distance between defect locations to be grouped into one repair operation. Clustered defects are grouped into one

repair step. The defect database also contains information on the closest mask edge. If a defect is closer to a mask edge than the radius of the sanding tool, the tool center point of the sander will be offset from the defect location to prevent hitting the edge or character line.

The user controls the repair process by defining parameters and sequences to be used by RepairWorks® in auto-generating the repair paths. Sanding parameters are entered in a table to define motion and forces. The user can select the repair sequence from front to back or top to bottom. The automated repair operation in FFE is an industry-first, true parametrically programmed application.

4. AUTOMATED ROBOTIC REPAIR

Repair Works® downloads the auto-generated program to the robot controller as described above. We are now ready for the automated repair. VisLoc®, Fanuc's 3D machine vision system locates the vehicle in the repair cell. It uses four cameras to view tooling holes on the underside of the body. A coordinate frame is developed to provide correction to the auto-generated path to accommodate the variability of vehicle location in the cell.

Upon receipt of a start signal, the robot executes the repair program. We are using a pneumatic orbital sander with a 3.5" sanding pad, the same tool used in manual repair. A Pushcorp® force feedback device sets and maintains the contact forces during the sanding process. The desired contact force is a user-settable parameter in the Repair Works® system. A vacuum shroud surrounding the sanding tool is used to collect airborne dust. The repaired area will still need to be wiped to remove the remaining fine sanding debris on the surface, but redeposit of sanding contaminants is reduced.

The sanding disk must be flat to the surface to within +/- 1.5 degrees to obtain good sanding performance without sand scratches. This is achievable because the paths were generated using the vehicle CAD geometry such that the robot approaches the vehicle surface along the surface normal. The automated repair process is also more repeatable because the Pushcorp® device uses feedback to maintain a contact pressure per the program to +/-0.1 pounds force, a tolerance not achievable with manual sanding.

When the repair operation is complete, the defect database is updated with the repair history. This information includes time of repair, cycle time, which robot performed the sanding operation, which defects were repaired, and which ones were grouped for a common repair. The cell also logs usage of the sandpaper to initiate an automated paper change when required.

Benefits of CAD-Based Parametric Programming.

An important side benefit derived from the architecture of the repair cell is the simplicity of setup and maintenance. It does not take a robot or paint process expert to set up the cell. The complex repair programs are auto-generated in the Repair Works® cell controller using true parametric programming. The only robot path that needs to be manually programmed is the skeleton path, where the position of the tool relative to the car is not critical as long it describes a safe path. Tolerance of setup is within inches, and angular orientation is inconsequential. If something changes in the paint or repair process, and a new sanding disk size, contact force, length of sanding time, etc., are required, these parameters are entered into the process database

table accessible from the Repair Works® GUI. The new parameters are automatically incorporated into the next vehicle repaired. A process engineer can focus on the repair process, and does not need to be knowledgeable in programming robots.

The system is also modular for debug and maintenance. The sanding forces can be measured and validated independent of the repair process by using a scale. The coordinate transformation and accuracy of the robot in the cell can be verified by moving a pointer to features on the vehicle called out in the CAD geometry defined in the vision cell controller (such as the intersection of a door edge and character line). Since the repair cell uses real world coordinates and does not require maintaining vehicle CAD, no action is required for product changes to the vehicle geometry.

USING THE SYSTEM TO ELIMINATE SOURCE OF DEFECTS

A major advantage of the automatic FFE operation over manual inspection and repair in a prime scuff booth is the real time data logging of paint defects. The defect database contains a complete history of each vehicle: model, color, booth and time painted, scanner and time inspected, defect location and size, repair status and history. Data clustering techniques can be used to highlight trends based on time, primer color, model, location of defects, etc. See Figure 8 as an example of an operator screen for defect reporting.

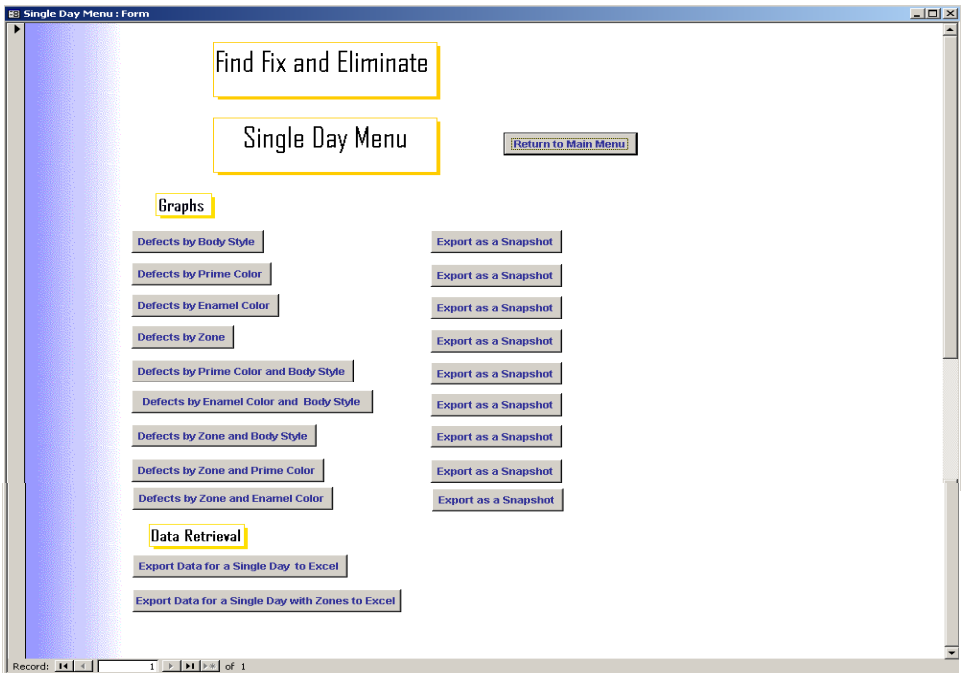

Figure 8. Access menu screen developed for defect report generation.

An automated dirt counter can provide an objective measure of paint shop performance, useful for decreasing dirt in the process. The criteria to determine what is a defect lose its dependence on the operator and outside influences (time of day, length of time into the shift, operator or supervisor attitude). Observed trends are identified and can be analyzed for root cause.

Of course, eliminating the source of defects to improve quality requires action on the part of someone in production to understand what causes dirt counts to spike at 12:00 PM, or why Tuesdays and Wednesdays are best, or why dirt counts are higher on days than afternoons, or why black is consistently dirtier than white or gray. This system provides a capable tool for logging dirt counts, it does not analyze root cause for the dirt.

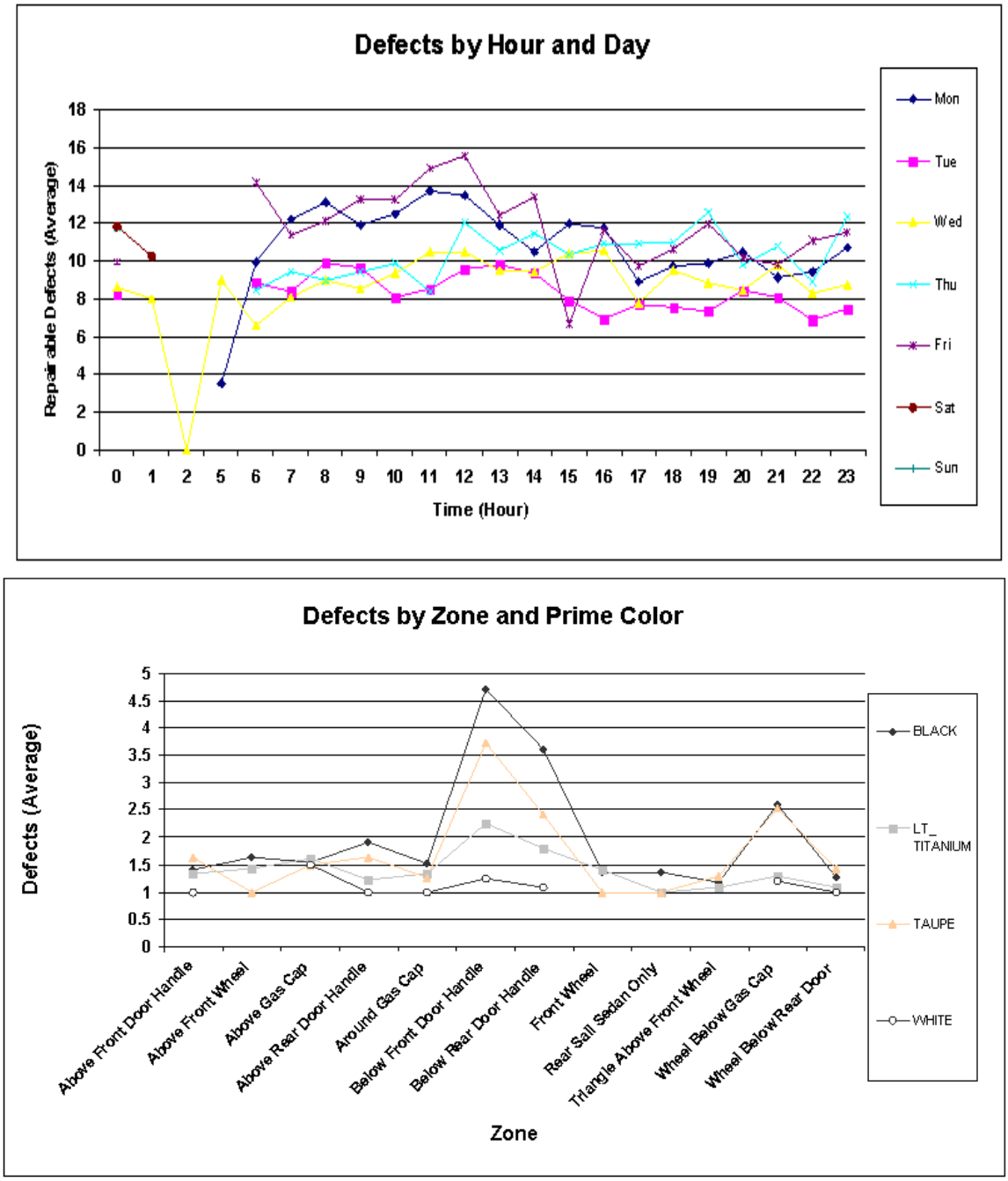

Figure 9. Examples of trend charts for dirt counter. Both reports are for data accumulated over one month period. Data is cut by day of the week and plotted by hour for the top report, and by color of prime and plotted by position on vehicle for the bottom report.

CONCLUSION

We have successfully demonstrated the feasibility of automating the find and fix of dirt defects in the prime finish. Throughout this development, there were many lessons learned as well as some delightful surprises. A few of the lessons learned:

1. The position of the vehicle on the skid can vary tremendously vehicle to vehicle. We used a photoeye on the line to start the capture of the image when the vehicle crossed a certain point. The object moved in the field of view by about 1" in the vertical and 0.75" in the horizontal. We had to add functionality in the software to locate the vehicle in the field of view, and provide an offset to the reported defect locations. If we did not provide the software fiducial function, the masks would have to be increased by 1" all the way around, and very little of the actual surface would have been inspected or repaired.
2. We learned that through-beam sensors are more robust than reflective photoeyes, especially when you are working with parts that can be white or black (this should have been obvious, but was something we overlooked).
3. Software developed for Windows is very portable. However, it is important to build the software product and test the portability by running an installation program on a target machine that was not used for development. Throughout the development cycle, we used Windows 98, NT, and 2000. Two of the three products could be moved to different machines using any of the Operating Systems. By loading the software on multiple machines during the development, we identified inconsistencies in the libraries that would crash the software, and fixed them on an ongoing basis. If this is not done, you have a complex product than can only run on one computer (the development computer), and an undefined amount of debug time will be required to complete the system to a shippable product long after you have demonstrated feasibility.
4. Using a force feedback device that was designed for edge grinding (25 pounds force) for fine finishing will not provide the force resolution required to truly optimize the sanding process (4 pounds).

Some of the pleasant surprises:

1. Using CAD as the master provides tremendous benefit for setup and calibration. We regularly modified the vision system resolution and position during development. It would take at most a couple of minutes to bring the system back into calibration. When Fanuc installed the repair cell, all of the robot calibration occurred in the robot cell without any influence from the vision scanner. The scanner and repair functions were truly separate and distinct.
2. Using auto-generated, parametric programming for the robotic repair operation meant that once we had the robot calibrated to the workspace, we did not have to spend time tweaking points. This was a real plus, because we did not need to have the robot expert there to do process development.
3. Twelve bits of grayscale resolution on the paint scanner provide adequate dynamic range for inspecting any color of paint (white, black, and gray) without making adjustments to the system imaging or processing parameters.

4. A lot of time was spent up front in development of the user interface. This was time well spent, since the graphical interface provided the information we needed to debug the system in an easy to understand format.
5. Windows was surprisingly robust, even for real time functions (hear that, Unix supporters). We were processing large images (400 Meg every 80 seconds) using the Pentium processor, and could run the system for weeks at a time without memory leaks or software crashes.
6. Microsoft Access works well for storage and analysis of process data. We had the system shared by multiple computers (vision cell controller and robot cell controller) using standard Microsoft folder sharing. We tested the system and simulated a year's production of vehicles without memory leaks, increase in cycle time, or other failure.
7. Using a standard Windows platform enabled us to take advantage of inexpensive, and sometimes free, software to perform networking and reporting functions. One example was the AT&T Laboratories Virtual Network Computing (VNC™) software. Since our vision cell controller RepairMan® resided on the plant network, we were able to monitor and debug the system at Wayne from our office 20 miles away in Redford as if we were at the plant. Using the same free software, we were able to daisy chain the VNC window to work on the other computers in the repair cell, which were connected to the cell LAN, but were not on the plant network. All at no cost, and without any debug.
8. Using Access for the defect database meant that we could utilize the query and reporting wizards familiar to most Microsoft users, enabling us to easily analyze the large amounts of data in multiple different ways to chart and identify trends. Minimal development time or expense was required to complete this portion of the project. Effort was expended in the analysis and understanding of the data, not in the generation of software tools and reports. There is much to be gained from using standard, familiar tools.
9. Finally, we demonstrated that it is possible to automate the inspection of painted vehicles at line rates in our fastest plant and consistently detect even the smallest dirt particles objectionable to customers. Armed with that data, a robotic repair cell can then auto-generate programs to remove the dirt using force-controlled sanding. It was quite remarkable to watch the robot move through its complex repair program, always normal to the surface, staying clear of edges and character lines, knowing that we had manually programmed only 8 crudely positioned points. And they said it could not be done!

The author would like to acknowledge the contributions of the three supplier partners, who believed in the vision that an automated seek and repair was possible and worked diligently to make it happen:

Adaptive Optics Associates, Cambridge, Massachusetts (paint scanner)
Fanuc Robotics, Auburn Hills, Michigan (automated robotic repair)
Matthews Software Inventions, Farmington Hills, Michigan (cell controller).

Patent pending on technologies developed under this project.