Statistical Tools for Forensic Analysis of Toolmarks

Final Report

IS-5160

David Baldwin[†], Max Morris[‡,§], Stan Bajic[†], Zhigang Zhou[‡], and M. James Kreiser[*]
†Ames Laboratory, Iowa State University (ISU),
‡Department of Statistics, ISU,
§Department of Industrial and Manufacturing Systems Engineering, ISU,
*retired Illinois State Police forensic scientist

ABSTRACT

Recovery and comparison of toolmarks, footprint impressions, and fractured surfaces connected to a crime scene are of great importance in forensic science. The purpose of this project is to provide statistical tools for the validation of the proposition that particular manufacturing processes produce marks on the work-product (or tool) that are substantially different from tool to tool. The approach to validation involves the collection of digital images of toolmarks produced by various tool manufacturing methods on produced work-products and the development of statistical methods for data reduction and analysis of the images.

The developed statistical methods provide a means to objectively calculate a "degree of association" between matches of similarly produced toolmarks. The basis for statistical method development relies on "discriminating criteria" that examiners use to identify features and spatial relationships in their analysis of forensic samples. The developed data reduction algorithms utilize the same rules used by examiners for classification and association of toolmarks.

I. INTRODUCTION

The proposition that particular manufacturing methods produce marks on tools that are substantially different from tool to tool is generally accepted by the toolmark examiner community. This premise is based on a handful of limited studies in the literature, training, and years of proficiency testing and experience of toolmark examiners in making relative correspondences and associations of known matches and non-matches to develop a knowledge of uniqueness. Recently this premise was challenged in a U.S. court (*Ramirez* vs. *State of Florida*, Supreme Court of Florida, *Docket #SC92975*, 2000) and a critical toolmark match was deemed inadmissible.[1]

The purpose of this project is to provide the statistical tools necessary for validation of the proposition that marks produced on tools during the manufacturing process are unique from tool to tool. There have been several studies that have shown the impact of various tool manufacturing methods on the individuality of toolmarks and striations produced on tools.

These studies are reviewed in an article by Nichols[2] and have routinely shown that similar and/or sequentially produced tools have toolmarks that are distinctly distinguishable. The major shortcomings of these studies is that they concentrated on only a single manufacturing method (e.g., broaching) and they typically used a small number (<10) of samples.[3-6] This project extends these previous studies by conducting a more comprehensive statistical study of toolmark variation produced by several manufacturing methods (filing, grinding, whetstoning, broaching, stamping, and milling). Digital image databases of toolmarks produced by the different manufacturing methods have been generated. Digital images of surfaces are compared using a statistical analysis that yields an index of the degree of similarity.

The major innovation of this project is that the data reduction software developed provides a means of determining the level of difference between known matches (replicate images of the same surface) and known non-matches (images of different surface but same manufacturing process) for each manufacturing method studied. The developed statistical software is capable of determining the significance of a toolmark match, since the basis for development relies on "discriminating criteria" that examiners use to identify features and spatial relationships in their analysis of forensic samples.

II. EXPERIMENTAL

Images of samples were acquired with a Leica UFM4 Universal Forensic Microscope equipped with a comparison bridge, a Leica DC300 digital camera and a 0.63x c-mount adaptor. The DC300 has a 3132 x 2325 pixel CCD. Sample images were acquired at two magnifications, 15x and 25x, and saved as 8-bit grayscale jpeg files. The same area of each sample was imaged at the two magnifications. The Leica UFM4 is equipped with fluorescent and incandescent lighting for sample illumination. Images of at least 100 of both consecutively in-house made and commercially made samples were acquired for each studied manufacturing process. All 100 samples of each process, or set, were imaged, and the first 20 in each set were acquired in triplicate. If a sample or tool had more than one side, image sets were collected for each side, with three replicate images of the first 20.

Commercial tools were acquired from local tool and equipment distributors. The tools were acquired in lots of 100 and represented a random generation of toolmarks, which would normally be encountered in a real-world situation. The following commercial tools (and the manufacturing process they represent) were acquired: screwdrivers (coarse grinding); cold chisels (coarse grinding); pliers (broaching); metal snips (fine grinding); bolt cutters (milling and grinding); wire cutters (filing); and wood chisels (whetstoning). Appropriate jigs were made to reproducibly present the same area of each tool to the microscope.

Commercial tools typically are produced via several manufacturing processes before being distributed for sale. As such, class characteristics for certain manufacturing processes on commercial items may have different distinctive characteristics than on samples produced in-house by only one process for this project. For example, commercially produced pliers were initially shaped by broaching the appropriate tool steel blank. Following the broaching process,

the teeth on the pliers were laser-hardened followed by a thermal hardening of the entire tool. The hardening processes subsequently obliterated the marks produced by the broaching process.

Consecutively made toolmark samples were fabricated in-house at the Ames Laboratory machine shop. The following manufacturing processes were emulated: grinding, filing, milling, and broaching. All of these samples were generated from tool-steel blanks. For grinding, filing, and milling, the marks were made on three-foot lengths of tool steel and then cut into one-inch pieces. Care was taken to ensure that consecutively made samples were marked accordingly with a number-stamp punch. For the grinding process, the grinding stone needed to be resurfaced after each three-foot section. The filing was done by hand using a metal file. Care was taken to ensure that approximately the same pressure, angle, and direction were maintained during the filing process. A fly-cutter was used to generate the milling marks on the steel sample. The fly-cutter was angled one-half degree from normal to eliminate toe marks and set to a depth of 0.005 inches. Only the heel of the fly-cutter generated the milling marks. The velocity of the fly-cutter was 260 rpm and the steel was fed at a rate of 0.75 fpm. Broach marks were made in a two-inch by two-inch by half-inch piece of steel with a one and one-half inch hole drilled through it. A keyway broach was used to generate a 7/16-inch wide broach mark. Only one pass of the broach was made for each mark. The broach bushing guide was then rotated 180 degrees and another broach mark was made in the steel. These two marks were considered consecutive marks and were numbered as such. The rectangular steel piece was subsequently cut in half to allow access for visual inspection and imaging of the marks.

A fabricated indexed plate was attached to the mechanical translation microscope stages to allow the in-house samples to be positioned on the stages. One of the in-house samples for each of the processes was placed on the indexed plate and marked to identify the portion of the sample that was in the field-of-view of the microscope. This was done to ensure that the same portion of each sample was presented to the microscope. At 15x magnification the field-of-view for the collected image is nominally three-quarters of an inch; at 25x magnification, approximately 0.45" of the sample is imaged.

III. Algorithm and Data Analysis

A. Forensic Toolmark Matching

The surfaces of tools display marks that are characteristic of the methods and equipment used in their manufacture. While these marks may be generally similar in appearance for similar tools, the microscopic details of a tool's surface are assumed to be unique to that specific tool. Therefore, microscopic comparison of test scrape marks made by a suspect tool to scrape marks on a surface such as a door or window frame is an important procedure in forensic investigations. A tool recovered from a suspect, which can be identified as having made the toolmark found at a crime scene, is of great evidentiary value.

Forensic scientists trained in identification procedures perform forensic examination of tool surfaces and toolmarks. Test marks made with a suspect's tool along with toolmarks from a crime scene are examined together under a two-stage microscope that allows independent

rotation and translation of each specimen. The microscope produces adjacent images of the two specimens separated by a vertical "hairline" across the field of view. The forensic scientist attempts to find a physical orientation of the test mark and the evidence mark such that the complex pattern of the surfaces is aligned along the hairline and the visible difference between the two surfaces at the hairline is minimized. In a positive identification, the hairline may almost disappear as details on each side are optimally aligned.

Figures 1A and 1B depict characterizations of toolmark patterns that might be seen through the microscope before, and after, alignment by the forensic scientist. The two dimensional parallel lines shown in each figure approximate the three dimensional patterns associated with, for example, grinding processes. Figure 2 is an image of a laboratory specimen produced by a grinding process.

For simplicity, Figure 1 depicts the marks in two colors, as dark lines of varying widths against a light background. In reality, specimen surfaces are obviously much more complex and appear in more gradually changing grayscale. A critical feature of toolmark comparisons as practiced by forensic scientists is that specimens must significantly correspond in pattern detail in order to be considered a positive identification or match. In reality, the technical complexity of a skilled forensic scientist's work goes far beyond the mechanical trial-and-error process described here, but this serves as an introduction to the digital image processing problem to be described next.

From an algorithmic perspective, the forensic scientist's job (as described here) can be loosely thought of as an optimization problem, requiring a search over:

- angles of rotation for each specimen,
- left-to-right translation of each specimen, and
- vertical translation of one specimen relative to the other.

This requires as much as a 5 degrees-of-freedom comparison or optimization of the toolmark images. Note that if it were not important to find rotations that maximize the complexity of the striations running parallel to the hairline for each specimen, the dimension of the problem could be reduced by one by requiring only a search of rotation angle for one specimen relative to the other. But this could lead to identifications or "matches" of the form shown in Figure 1C, in which the patterns correspond along the hairline, but do not offer meaningful evidence that the images are of the same surface.

The algorithm described in this report is an attempt to mimic the comparison process used by a forensic scientist, when the surfaces to be compared are represented by grayscale digital images. Specific computer file formats vary, but for practical purposes the data recorded for a specimen can be regarded as a rectangular array of positive numerical values representing the intensity of light recorded at the corresponding image pixel locations.

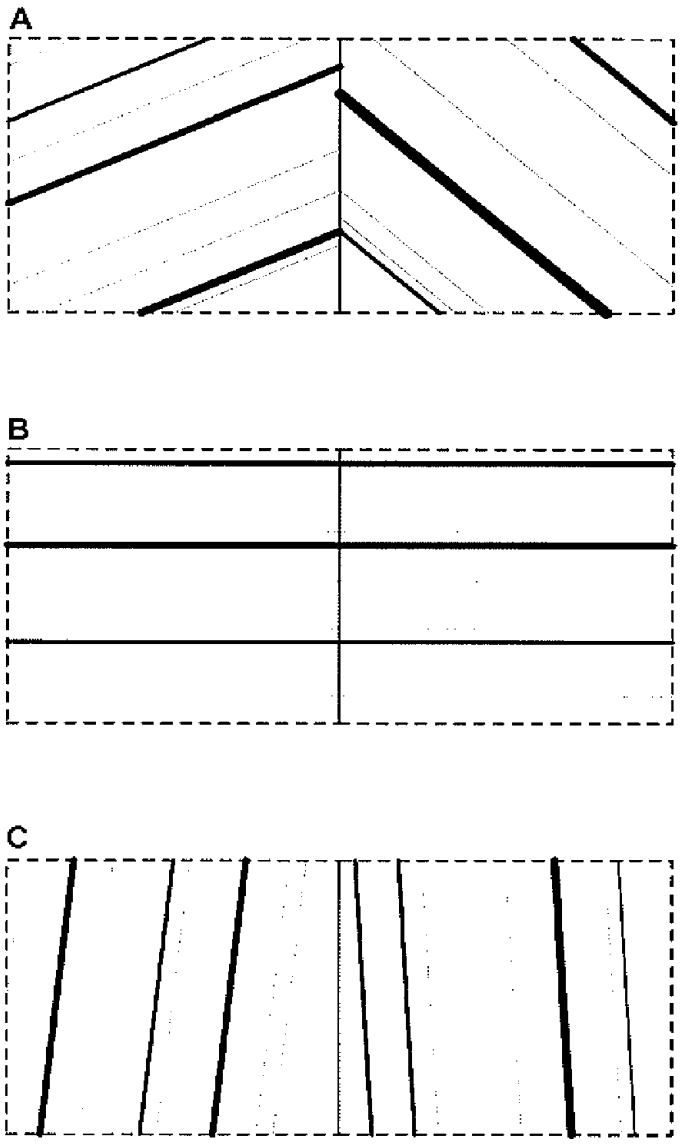

Figure 1. Characterization of split-view comparisons of specimens. A: Split-view before alignment. B: Split-view after alignment. C: Disallowed "match" in which specimen detail is not aligned to either side of the hairline.

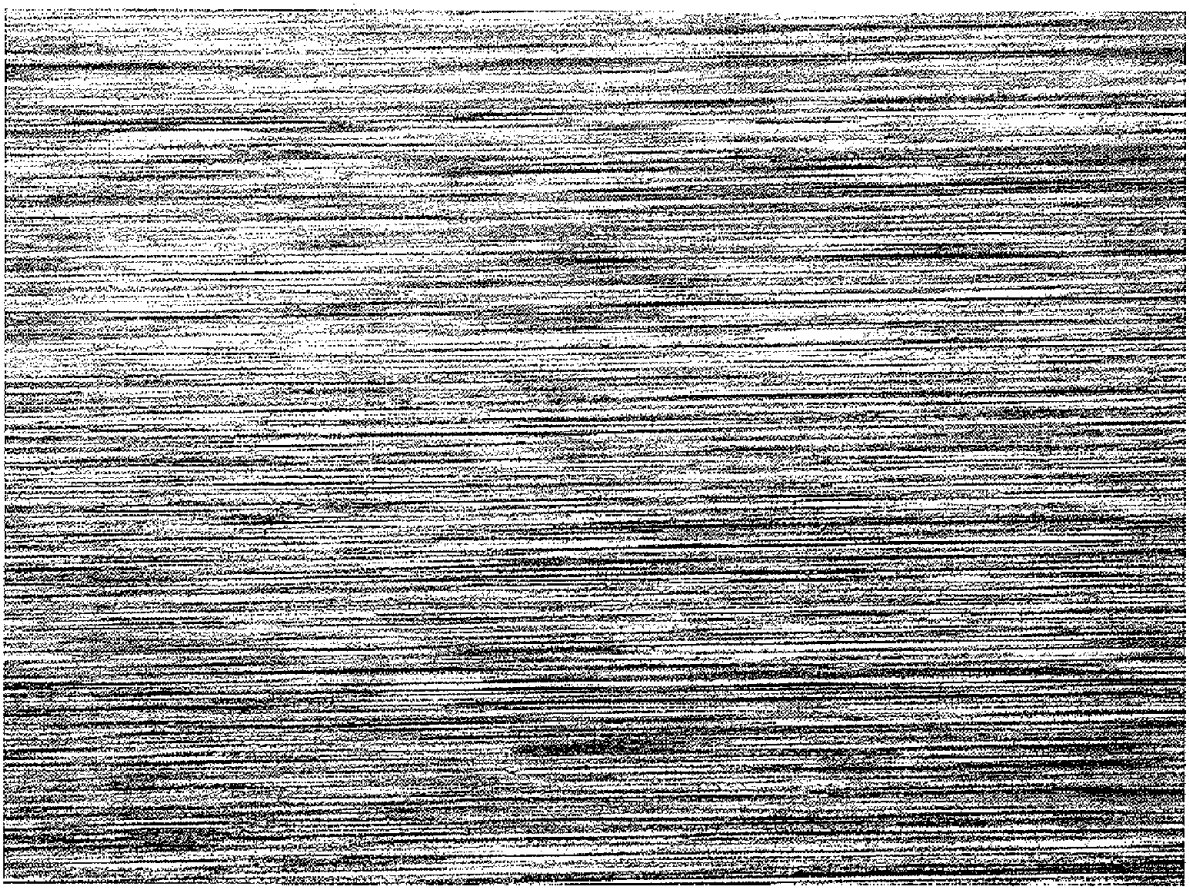

Figure 2. Digital Image of a laboratory specimen prepared using a grinding process, at 25x magnification under fluorescent lighting.

B. AN ALGORITHM FOR MATCHING VIA NUMERICAL OPTIMIZATION

Recall that the process of comparing two specimens was loosely described above as an optimization of quality of fit at the hairline separating the microscopic images of the two objects, with respect to the 5 quantities:

- angles of rotation for each specimen,
- left-to-right translation of each specimen,
- vertical translation of one specimen relative to the other.

However, framing the problem simply as an unstructured numerical optimization is not likely to be fruitful for two reasons. First, any useful measure of match quality is likely to have many local maxima with respect to these five quantities, owing to the complexity of the typical pattern in each image. Second, the high resolution of these images suggests that a truly global search over all possibilities would require a number of rotation/translation evaluations that for practical

purposes is impossibly large. However, note that a laboratory forensic scientist does not explicitly examine all possible trial matches, but relies on "experienced-based" rules based on the important features of this problem to form an efficient search for an optimal match. We shall also follow this approach in developing a numerical analogue to the comparison process used by the forensic scientist. The iterative numerical strategy to be developed is based on three steps, which we call rotation and interpolation, lag optimization, and grid augmentation, each of which is repeatedly used in the overall algorithm.

1. Rotation and Interpolation

Image rotation is motivated primarily by the requirement that the variation in the character of the striations on the specimen surfaces is maximized - or at least substantial - along the hairline. Because the direction of greatest variation may not be consistent across the entire range of the image, we define it locally. From a given point (pixel) in an image, rotation is the process of determining the direction (across the image) corresponding to the greatest variation in the striation pattern, which will be referred to as the response or recorded intensity. (The process of selecting the "given point" where this is done is described in the grid augmentation step below.) The analysis tool we use to accomplish this is based on the geostatistical method called kriging (e.g. Cressie, 1993)[7], originally introduced as a procedure for estimating the contours of the response as a function of longitude and latitude, given only response values at a distinct set of locations. Briefly, our use of kriging is primarily through the calculation of a set of direction-specific variogram functions using the data at pixels within the neighborhood of the selected point. The variogram function has distance as its argument (here measured in units of adjacent pixel separation), and its value is closely related to the covariance of response values observed at locations separated by the corresponding distance. Variograms for which the value changes rapidly across small increments of distance indicate a surface for which spatial correlation "dies quickly" with increasing separation distance. We compute variogram estimates corresponding to 180 equally spaced directions of separation in the neighborhood of the selected point, and note the direction for which this function changes most rapidly over short distances, as depicted in Figure 3.

Once directions of greatest variation are determined through given points in two images, the next step is to compare "strips" of each image taken along these directions. The forensic scientist accomplishes this mechanically through physical rotation of specimens and optical comparison across the microscope hairline. Our analogue of this will not actually involve rotation, but rather construction of a linear sequence of intensity values along the determined direction in each image. Note that because each digital image actually consists of values at only a rectangular grid of pixel values in a fixed orientation with respect to the image, this requires that we interpolate values along the direction of interest - at what may be though of as "pseudo-pixel" values - for direct comparison. This interpolation of the image to points "between pixels" is based on the use of a kernel smoothing technique to estimate the set of values needed. The constructed linear data series is of standard length, half on each side of the point about which the direction of maximum variation is determined. Interpolation is done so as to construct a sequence of intensity values corresponding to a line segment in the direction of greatest variation, so that these values change "most quickly" across the striation pattern in the image.

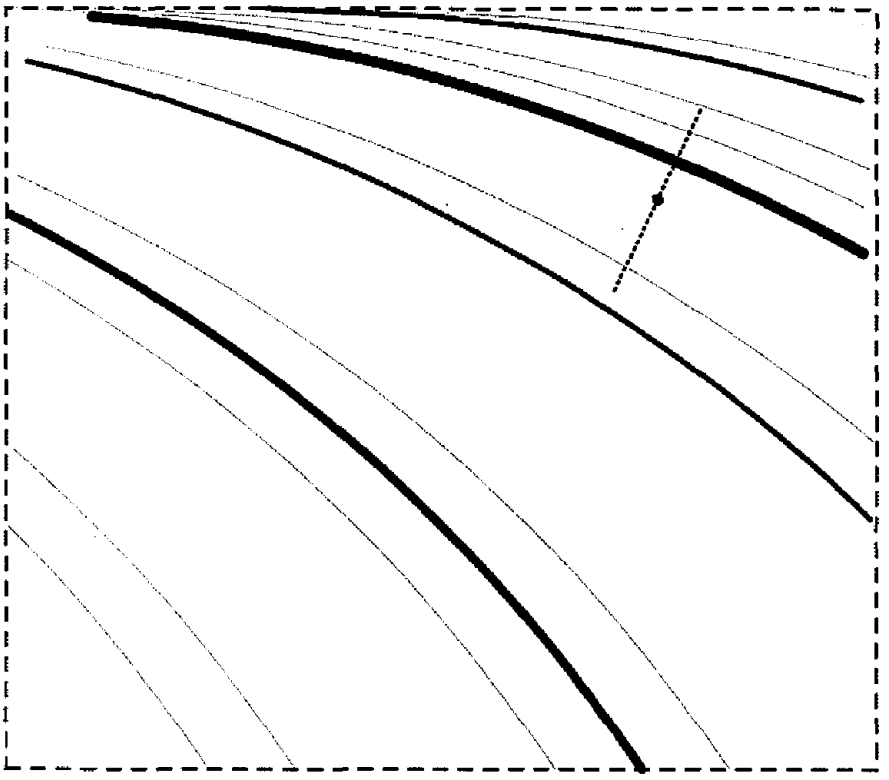

Figure 3. Characterization of local direction of greatest variation through a given point in the image.

The individual values in each sequence cannot simply be taken as intensity values from the image, because the "rows" and "columns" of pixels do not generally correspond to the direction of greatest variation. Instead, intensity values are estimated along the segment at intervals equal to the distance between pixels in the image; these mimic the intensity values we *would* have had in a segment of one "column" of pixels had the vertical image dimension been perfectly aligned with the direction of variation. So, for example, if the direction of greatest variation is determined to be 15 degrees relative to the "natural" x-axis of the digital image, a relatively narrow "strip" is identified in the image, centered on the 15-degree segment of interest. "Slices" cut across this strip (at 105 degrees) contain pixels for which the intensity values are relatively consistent (since they follow "ridges" in the image). Estimation of each intensity value along the segment of interest is accomplished using a nonparametric smoothing technique (specifically, kernel smoothing with a Gaussian kernel with a relatively narrow window). At each location, the smoothing function is applied only to actual intensity values associated with pixels found in a slice of the strip, centered on the point of interest. This approach minimizes the variability in the data used for interpolation.

2. *Lag Optimization*

For given points in each image, once directions of maximum variation have been established, and linear data series of one-pixel separation constructed along these directions, the next step is to determine how well these two series can be aligned. This operation is analogous to the forensic scientist "sliding" one specimen along the hairline relative to the other, in an attempt to find the best alignment of striation patterns. For our purposes, we use lag correlations, often used in statistical time-series analysis. Recall that at this point, each image is represented by a linear series of interpolated values. Ordinary correlations are calculated between the two series, using different "lags" or offsets of one series relative to the other, to find the alignment for which agreement is best, i.e. the lagged correlation is closest to +1 or -1. Because negative correlations may also indicate good agreement, e.g. if one image appears to be the "negative" of the other due to differences in illumination of the two specimens during image acquisition, the largest absolute correlation (sign removed) is selected. This maximized absolute correlation is our measure of how well the two images agree (locally) along linear sections selected as described above.

3. *Notation*

Some notation is now necessary in order to describe these ideas precisely. In discussing the rotation and interpolation and lag optimization steps, we begin by assuming that "given points" have been selected in each image denoted as $\underline{p}_1=(x_1,y_1)$ and $\underline{p}_2=(x_2,y_2)$, respectively, for images 1 and 2. Rotation determines a direction of maximum variation within a neighborhood of the selected point in each image; denote the direction vector as $\underline{v}_1$ in image 1 and $\underline{v}_2$ in image 2, where $\underline{v}_1$ and $\underline{v}_2$ are each of length L. Interpolation produces a sequence of image intensity values corresponding to "pseudo-pixels" along a line determined by the selected point and direction vector. Denote these two sequences of values as s_1 and s_2, respectively, for the two images. Within a sequence, index the initial pixel, e.g. $\underline{p}_1$ in s_1, with the index value 0, and interpolated values within the sequence with consecutive positive or negative integers (depending on direction). Lag optimization determines the offset or lag that maximizes the linear correlation coefficient for these two sequences of values. If this optimal lag associates $\{s_2\}_L$ with $\{s_1\}_0$, identify this as a lag-L correspondence. Mechanically, this means that the best translation along the hairline shifts the second specimen a distance L relative to the first, or that each specimen can be shifted a distance L/2 in opposite directions, to achieve the same effect. Hence a symmetric "correction" for points $\underline{p}_1$ and $\underline{p}_2$ can be defined as:

$$\underline{p}_1' = \underline{p}_1 + (L/2)\,\underline{v}_1$$
$$\underline{p}_2' = \underline{p}_2 - (L/2)\,\underline{v}_2$$

The rotation and interpolation and lag optimization are iteratively applied, using $\underline{p}_1'$ and $\underline{p}_2'$ as new "given points" in an attempt to find even better local agreement between the images. Iteration stops when the lag correlation cannot be further increased, and the largest absolute correlation obtained is denoted as r'. Note that since the (real) pixel locations are a rectangular grid in the original (x,y) coordinate system, iterative rules such as these for selecting new locations will typically require numerical rounding to actual pixel locations.

4. *Grid Augmentation*

The "outer loop" of the optimization consists of the rule for selecting what are referred to as "given points" in the text above. Initially, a fixed, relatively sparse grid of pixels is specified in each image. We use a regular, uniform point grid for this purpose, but other fixed or random point patterns that are relatively uniformly scattered across the image might also be effective. The rotation and interpolation step is applied to each of these points, and lag optimization is applied to each pair of points, one taken from each image. So for example, if the initial grid consists of 25 locations in each image, $25^2 = 625$ such comparisons will be made, each resulting in an adjusted pair of points and a locally optimized absolute correlation:

$$\underline{p}_1', \underline{p}_2', r'$$

We then identify a relatively small number of such pairs that lead to the greatest values of r'. Recall that the use of lag optimization is the analogue of translation of specimens along the directions $\underline{v}_1$ and $\underline{v}_2$, respectively, so as to maximize agreement of the images at the hairline. Further improvement in the match would then need to come from what would be analogous to left-to-right translation of the specimens, e.g. in directions perpendicular to $\underline{v}_1$ and $\underline{v}_2$ in the digital images. Denote these perpendicular directions as $\underline{v}_1^P$ and $\underline{v}_2^P$, respectively. For the relatively small number of pairs for which r' is largest, determine new grid points as:

$$\begin{array}{ll} \underline{p}_1' + \Delta\underline{v}_1^P & \underline{p}_1' - \Delta\underline{v}_1^P \\ \underline{p}_2' + \Delta\underline{v}_2^P & \underline{p}_2' - \Delta\underline{v}_2^P \end{array}$$

where Δ is a standard increment specified in the algorithm. The rotation and interpolation step is executed for each of these new points, and lag optimization is performed for all point pairs that include at least one new point. Grid augmentation continues iteratively so long as the best (largest) value of r' increases from step to step; the algorithm is finished when no further improvement is realized.

5. *Internal Calibration*

In their usual use in statistical analysis, correlation coefficients are generally computed from paired sequences of data for the purpose of establishing whether a direct or inverse association exists between the associated pair of variables. In these traditional studies, only one correlation coefficient is calculated. In contrast, we use a relatively complex optimization process to find linear segments in each image (each represented by a data vector of intensity values) for which the correlation is most extreme. In this process, many thousands of correlations are computed, but only the value closest to +1 or -1 is retained. As a result, the optimized correlation found by our algorithm is not comparable to the values that would ordinarily be observed in simpler sampling plans, but will be more extreme (e.g., closer to +1 or -1) even when the images do not match. The actual magnitude of the correlation that can be obtained, whether the images match or not, depends upon many factors including the general characteristics of the patterns present in each image and the specific form of the rules used in the numerical search for an optimal match. In order to place the obtained correlation value on a more meaningful scale, the last step of the

analysis is an internal calibration to provide an index of greater value in assessing the degree of agreement achieved by the matching process. This is done by generating and comparing two additional sets of correlation coefficients, one set representing the correspondence between analogous sections from the two images, relative to the initially identified areas of high correlation, and the other representing the correspondence that can be found between randomly selected sections from the two images (i.e. arbitrary comparisons of areas). These two sets of correlations correspond to two sets of points selected as described below, depicted in Figure 4.

The optimization algorithm stops, as described above, when additional iterations do not yield improvements, i.e. more extreme values, in the optimized correlation. The correlation corresponds to two "strips," one taken from each image, each of which can be identified by a central pixel and a direction of greatest change. If the algorithm has been successful in finding sections of the images corresponding to the same area of a physical specimen, it is easy to identify other pairs of nearby regions in the images that also match. To be specific, let $\underline{p}_1$ and $\underline{p}_2$ denote the center points of the identified strips in the two images, and $\underline{v}_1$ and $\underline{v}_2$ denote the directions of greatest change corresponding to each of them, for which the correlation has been optimized. If $\underline{p}_1$ and $\underline{p}_2$ actually correspond to the same (or nearly the same) point on a common physical surface, and $\underline{v}_1$ and $\underline{v}_2$ represent the same vector across the common physical surface, then new points identified as perturbations of $\underline{p}_1$ ($\underline{p}_2$) along a selected distance d in the direction of a selected angle θ relative to $\underline{v}_1$ ($\underline{v}_2$) should also correspond to matching (or nearly matching) points on the surface. For given d and θ, these new points are:

$$\underline{p}_1^* = \underline{p}_1 + d\, T_\theta\, \underline{v}_1$$
$$\underline{p}_2^* = \underline{p}_1 + d\, T_\theta\, \underline{v}_2$$

where T_θ is the 2-by-2 rotation matrix defined by the angle θ:

$$T_\theta = \begin{vmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{vmatrix}$$

Hence, correlations calculated in the first calibration set are computed by selecting random values of θ, computing the corresponding points p_1^* and p_2^* (closed dots in Figure 4), and performing the rotation and interpolation and the lag optimization steps described above to find a "locally" optimized correlation. If the two images actually represent the same pattern and the matching algorithm has been successful in identifying corresponding points in the images, these new correlations representing "rigid rotations and translations" from the identified matching points should also be relatively extreme (correlations very close to +1 or -1). However, if the optimized correlation is an anomaly, representing only a random similarity between the two identified strips, the new correlations should be considerably less extreme (closer to zero). For clarity, let n represent the number of such correlations calculated (each beginning with new randomly chosen values of θ in the fixed distance d) in this group.

The correlations in the second calibration set are computed in a similar fashion, but from randomly selected pairs of points rather than pairs selected to physically correspond in the case of a successful match. So here, p_1^* and p_2^* are selected randomly from images 1 and 2,

respectively (open dots in Figure 4), and the rotation and interpolation and the lag optimization steps are performed for this pair of points. Because no effort is made to identify physically corresponding locations in these pairs, the resulting correlations should represent the degree of "accidental" agreement that can be found by the lag optimization procedure. Because they will each be the most extreme value found in the process of computing correlations for many different lags, they will typically be more extreme than single correlations calculated from randomly generated data. Suppose as with the first group, n such correlations are calculated (each beginning with new randomly selected p_1^* and p_2^*) in this group.

If the correlations computed in the first group are to be taken as evidence for a match, they should overall be more extreme than those in the second group. Therefore, comparisons are made between all n^2 pairs of correlations, one taken from group 1 and the other from group 2. The final index of evidence for a match is the proportion of such pairs in which the absolute value of the correlation from the first group is larger than the absolute value of the correlation from the second group. Index values near 1 indicate situations in which most of the correlations might represent physical matches that are more extreme than locally optimized correlations corresponding to randomly chosen points. Index values near 0.5 indicate little difference between the two groups, suggesting that while the optimized correlations may be large, they cannot be interpreted as evidence for a physical match. Index values substantially below 0.5 are not likely because they would suggest that randomly selected points display more correlation than those selected by the iterative procedure.

There is some statistical theory suggesting an idealized distribution of values for this index in cases that are not true matches. The Mann-Whitney U-statistics (Mann and Whitney, 1947)[9] is used as the basis of a classical nonparametric two-sample comparison procedure. The U-statistic for comparing the two sets of calibration values is, in fact, the same as the index defined, apart from the divisor n^2 used to scale the index to the unit interval. Standard asymptotic arguments discussed, for example, by Gibbons[8] imply that for large samples (large n), when the two sets of values are independently drawn from the same population, this index should:

- be approximately normally distributed
- have mean 0.5, and
- have standard deviation approximately $(6n)^{-1/2}$.

This would imply, for example, that index values of more than $0.5 + 3(6n)^{-1/2}$, or approximately 0.622 for n=100, should be relatively rare unless the two sets are drawn from different

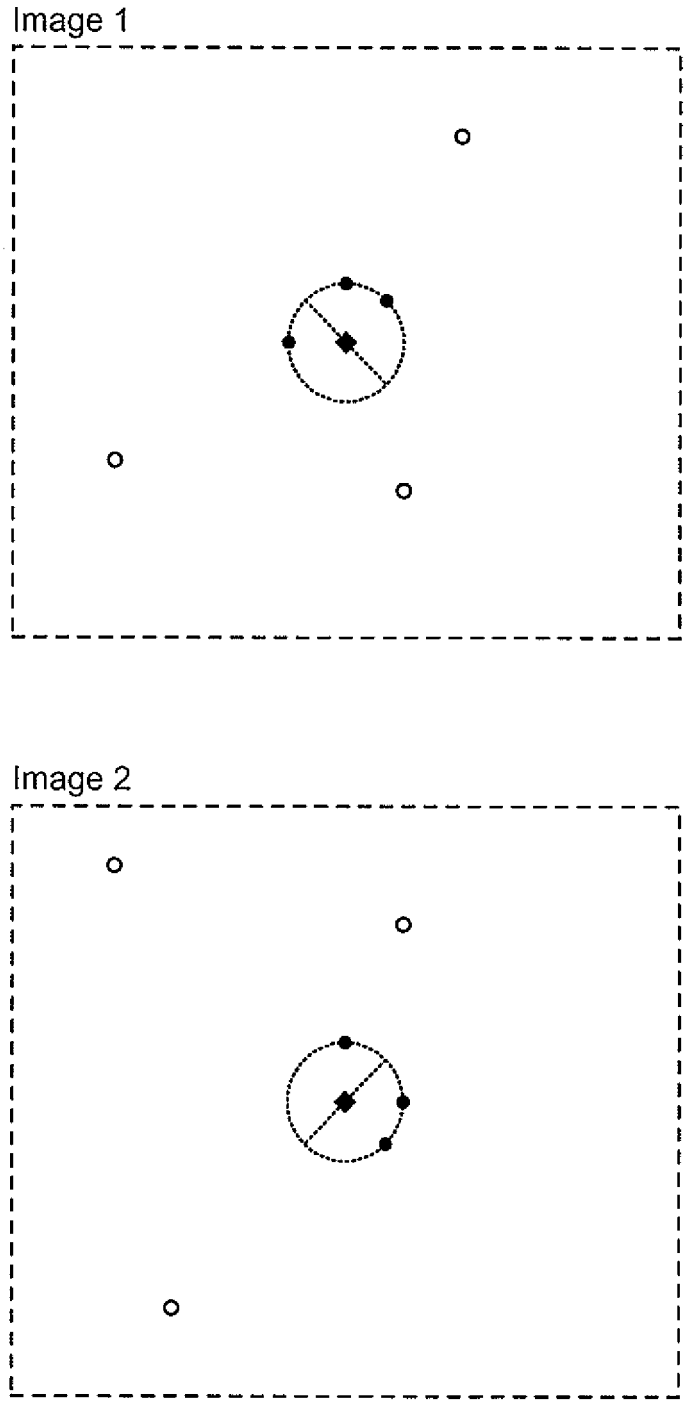

Figure 4. Characterization of two sets of points selected for validation. Diamonds represent points of best match. Closed dots represent points with the same random perturbation in each image. Open dots represent points selected by unrestricted sampling across each image.

populations (e.g. the "match" group is significantly larger because the match is real). The results of our algorithm test (Section C) indicate that this theoretical "critical value" might be adequate for some kinds of surfaces, but not for others. For example, for 15X images of non-matching ground surfaces, more than half the calculated index values were larger than 0.8 (Figure 9). This indicates that the "classical" assumptions used in the asymptotic analysis of U-statistics are not entirely appropriate here. For example, standard theory requires that each data value be statistically independently sampled, but the local random perturbations we use cannot be *entirely* independent because they are selected within a limited spatial section of the image. This "failure" of the classical analysis is more striking for some surfaces than for others, but the asymptotic values it suggests still provide useful "benchmarks" for what might be expected in idealized circumstances.

C. ALGORITHM IMPLEMENTATION AND TESTING

For purposes of testing, the algorithm described above was implemented in Matlab©.[10] Listings of the four routines are provided in the Appendix. These programs are "experimental" and have not been optimized with respect to execution time or memory management. Their primary function has been in empirical testing of the algorithm concept described herein.

There are a number of adjustable parameters that must be specified to completely define the algorithm, but which can be varied as needed. The following table summarizes these, listing a description of the parameter, the symbol used for it in this text (where appropriate), and the test value or setting we have used in the evaluations to be reported next.

Algorithm Parameter	**Symbol in Text**	**Test Value**
Dimension of images		Approx. 3200 by 2300 pixels
Initial grid of points in each image		25 in alternating rows of 4 and 3
Region of image used to calculate each variogram		201 by 201 pixel block, reduced to 21 by 21 pixels for calculation
Radial separation of variograms		1°
Dimension of strips used to construct linear data series for comparisons		201 by 60 pixel block, with the long side parallel to the local direction of greatest change
Number of data values in each series		201, corresponding to 1-pixel spacing
Size of "perpendicular" step	Δ	Initially 50 pixels, reduced by a factor of 2 with each augmentation
Number of largest correlations used to expand the grid at each stage		3
Number of correlations computed in each group in the calibration step	N	100
Distribution of perturbation distances used in generating first calibration set	D	Fixed value of 200 pixels
Distribution of perturbation angles used in generating first calibration set	θ	Uniform over the complete circle

1. Algorithm Testing

The potential effectiveness of the algorithm described here has been examined in a preliminary experimental assessment using the library of images collected as a part of this project. The assessment was performed on a subset of all the images acquired during the project. This subset consisted of the following surface types: ground, milled, broached, filed (each produced under controlled conditions in our laboratory), and cold chisels (produced by a commercial manufacturer). The reason for using only a subset of images lies in the length of time required to perform a comparison on a pair of images. As mentioned earlier, the algorithm was developed in MatLab, which is a non-compiled language. Currently, under MatLab, performing a comparison of two images takes approximately 15 minutes. A more fully developed form of the current algorithm could be recoded in a compiled language, such as C++, for substantially faster execution, once various parameters within the algorithm are determined for each manufacturing process.

Sets of three replicate images were used for each specimen from a collection of ten samples representing the same surface type. Multiple sets of replicate images for the same surface type allow the algorithm to be tested in a controlled study in which images of the same object should be classified as "matches", while images of similar objects of the same surface type should not. The resulting 30 images, compared two at a time, provide 30 image pairs that should match (those taken from the same replicate set) and a very large number of possible pairs that should not match. Again, given the time required for each digital comparison, not all pairs of images of different specimens were compared; instead, a subset of 27 of these pairs was assessed, based on the labeling of objects and replicate images of each object (see Figure 5). (While 10 commercially produced cold chisels were examined, two surfaces were photographed of each tool, and in this case 60 pairs of images should match, while 54 pairs should not.) Index values for same-object pairs should be larger than those for different-object pairs; the degree to which these samples are well separated (good) or overlap (bad) is an indication of how well the algorithm performs.

It should be noted that these groups of 30 and 27 comparisons should not really be interpreted as "independent samples" in the statistical sense, since each image is involved in more than one comparison. However, they should nevertheless serve as a useful preliminary assessment of the algorithm's potential effectiveness for separating unique patterns from class features in the four surface types tested.

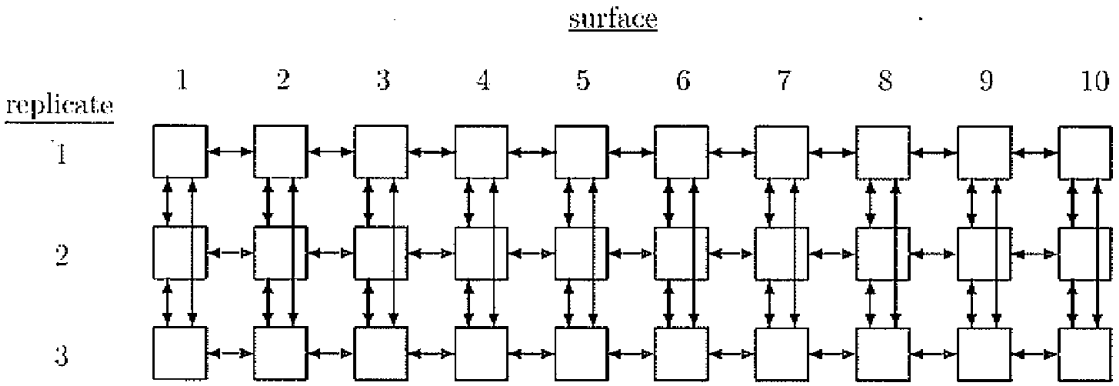

Figure 5. Schematic of how groups of replicate images are used to assess the performance of the algorithm. Boxes represent 3 replicate images of each of 10 surfaces. Vertical arrows represent 30 comparisons between replicate images of a common surface. Horizontal arrows represent 27 comparisons between images of different surfaces.

2. *Results*

In discussing these results, it should be noted that in the development of these algorithms, we primarily used images of ground surfaces in the selection and optimization of various parameters. The algorithms are in a preliminary stage and still require further development and optimization for different manufacturing processes. Nevertheless, numerical validation studies were performed to demonstrate the validation of the toolmark uniqueness proposition.

Results of the test comparisons are displayed in Figures 6-10. Figures 6-9 contain the index values resulting from 30 pairs of images from the identical surfaces (which therefore should indicate a match) and 27 pairs of images of similar surfaces (which therefore should not indicate a match) for in-house produced broached, filed, milled, and ground samples. Figure 10 contains the index values resulting from 60 pairs of images from identical surfaces and 54 pairs of images of different surfaces of the commercially produced cold chisels. Each figure presents the index values for one surface type at either 15x or 25x magnification.

As described above, the index used is a scaled value of a U-statistic, and theory suggests that for perfectly independent data values drawn from identical distributions, the mean and standard deviation of the index values should be approximately 0.500 and 0.041 for the sample image sets used here. This distribution would result in "mean-plus-three-standard-deviations" upper value of 0.622, which can be used as an informal guide to the index size required before judging a comparison to be a positive match. However, the patterns displayed in the figures suggest that this model is not always appropriate for the values generated from images that do not match (closed dots at the bottom of each panel in Figures 6 - 10). While these samples do often have means and ranges similar to what the theory suggests, there are also cases (e.g. for ground surfaces) where the index values are skewed to larger values. All figures taken together suggest that a value of approximately 0.8 is effective at separating "matches" from "non-matches" in many cases. The following table summarizes the number of false positive and false negative conclusions that would be reached from these data, if 0.8 were used as a decision value.

Surface type	Magnification	False Negatives	False Positives
Broached	25x	0/30	0/27
Broached	15x	6/30	0/27
Filed	25x	1/30	0/27
Filed	15x	10/30	0/27
Milled	25x	19/30	0/27
Milled	15x	16/30	0/27
Ground	25x	0/30	3/27
Ground	15x	0/30	16/27
Cold chisels	25x	5/60	0/54
Cold chisels	15x	3/60	0/54

Although false negatives and positives are relatively few in most situations, two exceptions are especially marked. A large number of false negative results (in fact, more than 50%) are observed for milled surfaces at either level of magnification, but no false positives are seen in these cases. A large number of false positive results, especially for the lower level of magnification, are observed for comparisons involving in-house produced ground surfaces; here, no false negative results were observed.

The large number of false negatives observed for milled surfaces is probably due to the nature of the marks for this manufacturing process. Milled surfaces typically have long and regular repeating striation patterns, reflecting class characteristics. Recall that the lag optimization step involves a fixed linear series of pixels. Although the algorithms can be "tuned" to improve their effectiveness by judicious selection of the controlling parameters, the dimension of this series of pixels was the same for all the processes and images analyzed. It is quite probable that size of the series of pixels used was unsuitable (i.e., on the order of the pattern width) in the analysis of this particular milling process, leading to indistinguishable or high (i.e., nearly unit value) lag correlations for the points compared. Hence, the validation of the "highly correlated" points on the images leads to index values indicative of just random comparisons of image positions when images of the same specimen are analyzed (Figure 8). The striation patterns of ground surfaces are qualitatively quite different and appear more random and locally variable. We speculate that in order for the algorithms to work well for surfaces as different as these (i.e., milled versus ground), further developments are necessary to "calibrate" the algorithm's parameters to the kinds of images being compared in a given application.

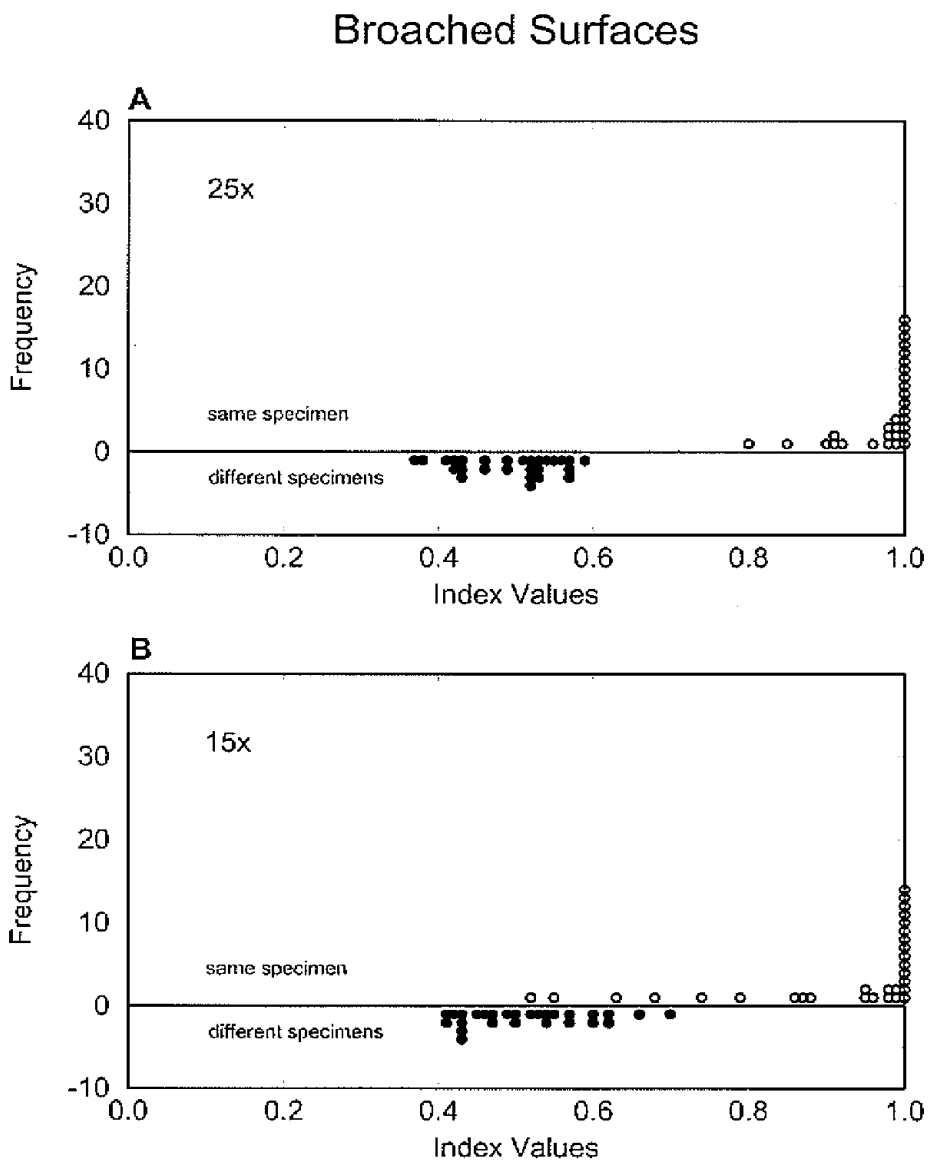

Figure 6. Study results for broached specimens. A: Index values for comparisons made at 25x magnification. B: Index values for comparisons made at 15x magnification.

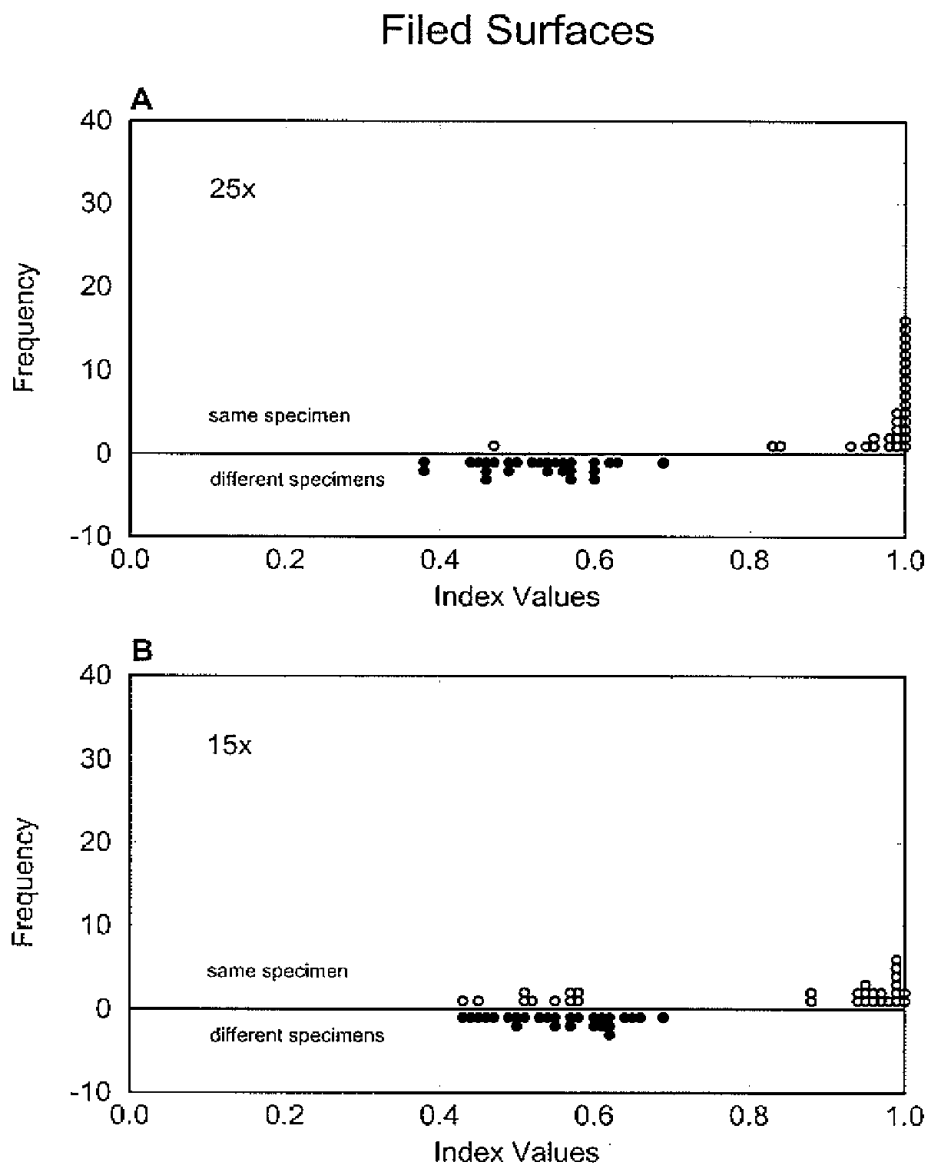

Figure 7. Study results for filed specimens. A: Index values for comparisons made at 25x magnification. B: Index values for comparisons made at 15x magnification.

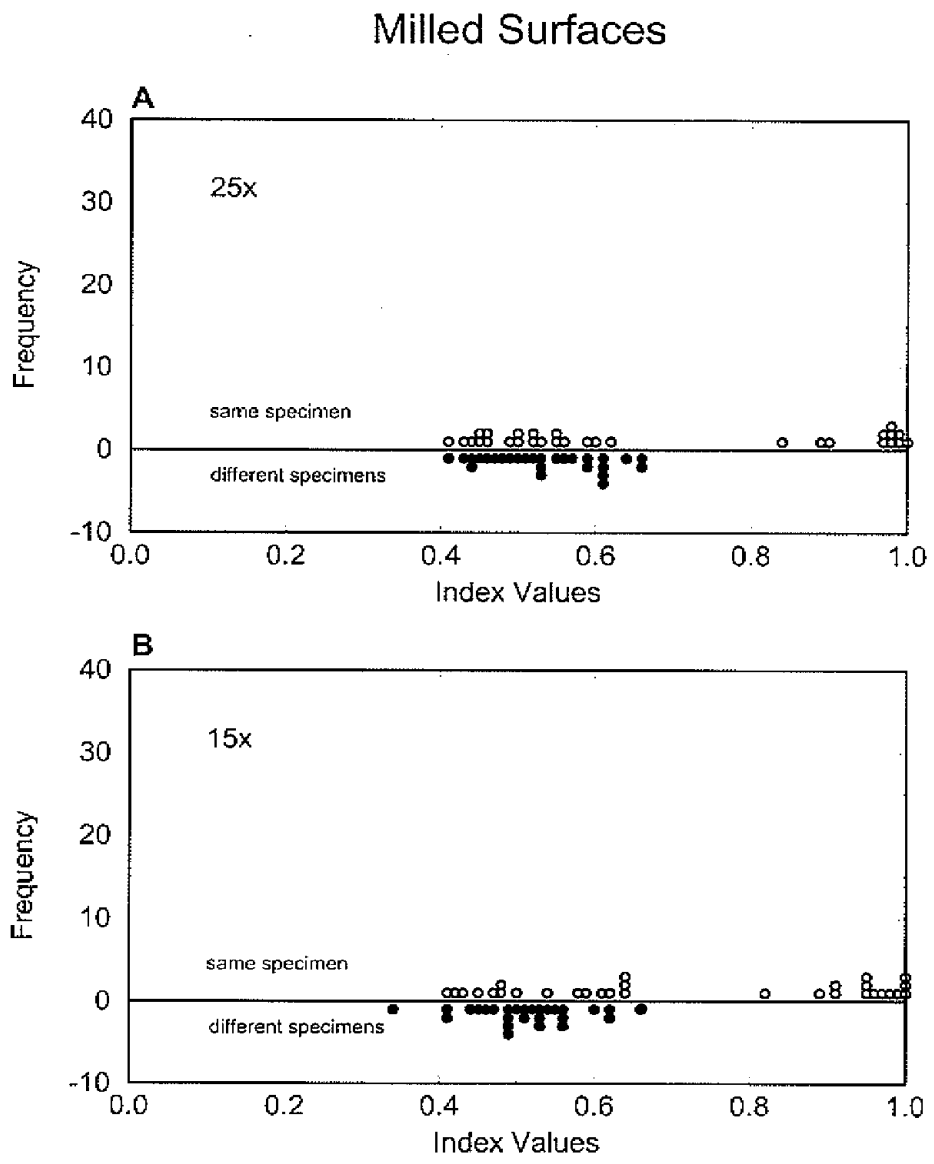

Figure 8. Study results for milled specimens. A: Index values for comparisons made at 25x magnification. B: Index values for comparisons made at 15x magnification.

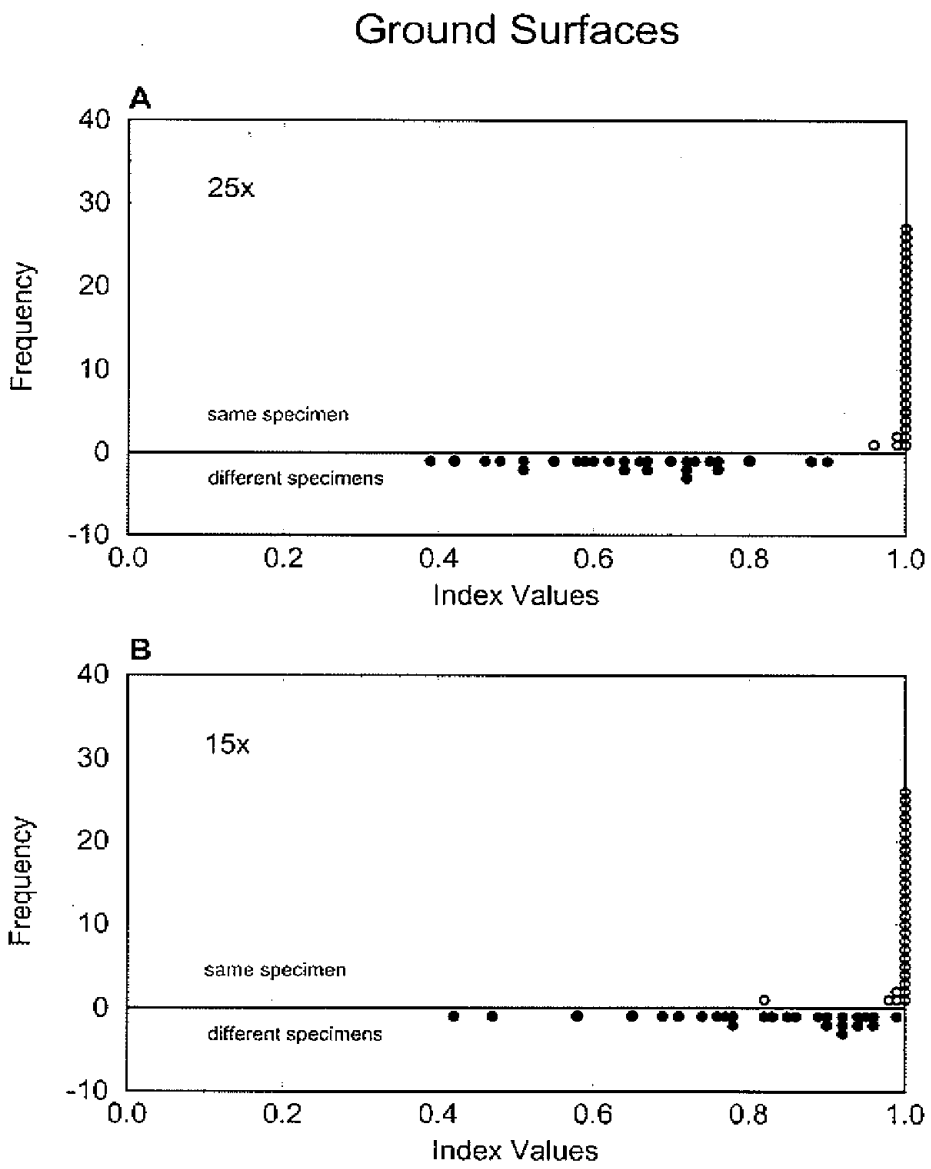

Figure 9. Study results for ground specimens. A: Index values for comparisons made at 25x magnification. B: Index values for comparisons made at 15x magnification.

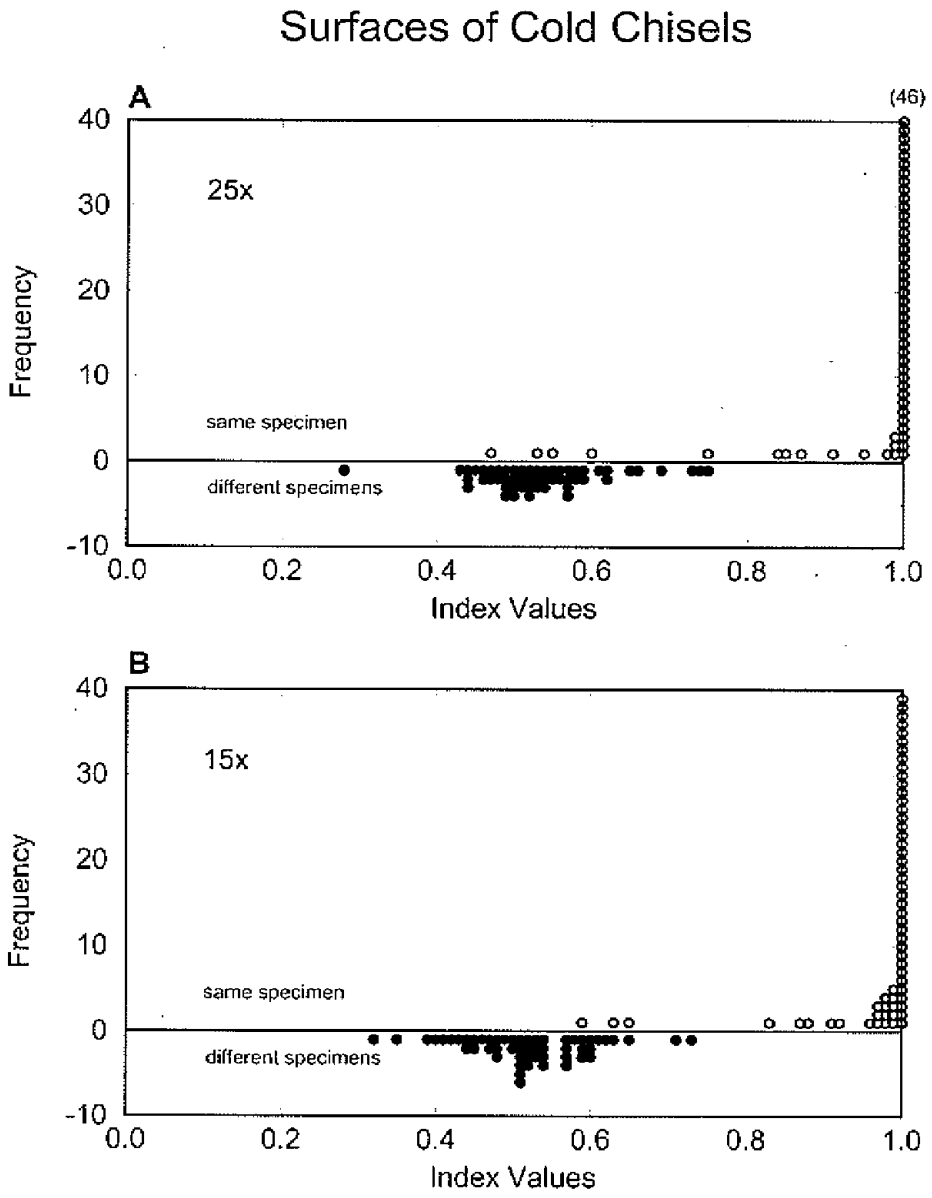

Figure 10. Study results for commercially produced cold chisels. A: Index values for comparisons made at 25x magnification. B: Index values for comparisons made at 15x magnification.

Interestingly, visual inspection of the consecutively made in-house ground samples indicated that there are areas on successive samples that do closely match. These "coincidences" could possibly lead to the number of false positive identifications shown in Figure 9, as well as the observed skewed distribution of index values. Figure 11 is an example that illustrates coincident areas on successively made samples. The physical distance between the areas displayed in the figure is approximately 1.2 cm. These coincidences likely arise due to the way the in-house samples were produced. These samples were produced by feeding a three-foot length of tool steel blank at a constant rate past a grinding wheel. Although the surface of a grinding wheel is friable, the wear on this machining surface is apparently slow enough that recurring patterns do occur under precisely controlled conditions. Similar behavior may not be likely for poorly controlled conditions, such as encountered during hand-held grinding.

The difference in the index value distributions for the two examples of ground surfaces may be related to the nature and quality of the striations. Although in both cases the striations are random and locally variable, there still are qualitative differences (although slight). The commercially produced cold chisels exhibit striations that are somewhat deeper and wider in their appearance. Furthermore, there are less likely to be areas of coincidental matches among the chisels. This is likely because acquiring commercial items results in a very random sampling and the probability of having consecutively made items is quite low. Moreover, the way commercial chisels are produced is quite different than the grinding process used in-house. Commercial chisels are normally ground by hand using a belt grinding wheel.

One additional point of interest is the pattern of index values seen when the algorithm mistakenly classifies a true match (a false negative). These relatively small index values are distributed in approximately the same pattern as those calculated from images that are not true matches (i.e. the solid dots below the line in each plot Figures 6 - 10). Our speculation is that there are cases in which the heuristic search algorithm simply failed to find locations in the two images that correspond to the same physical point on the object. While the numerically optimized correlation values may still be large in these cases, those generated in the first calibration set are much smaller (because the match region isn't correct), and so are similar in value to the unrestricted random correlations calculated in the second set. Under these circumstances, we should expect to see index values very similar to those calculated from images that do not really match.

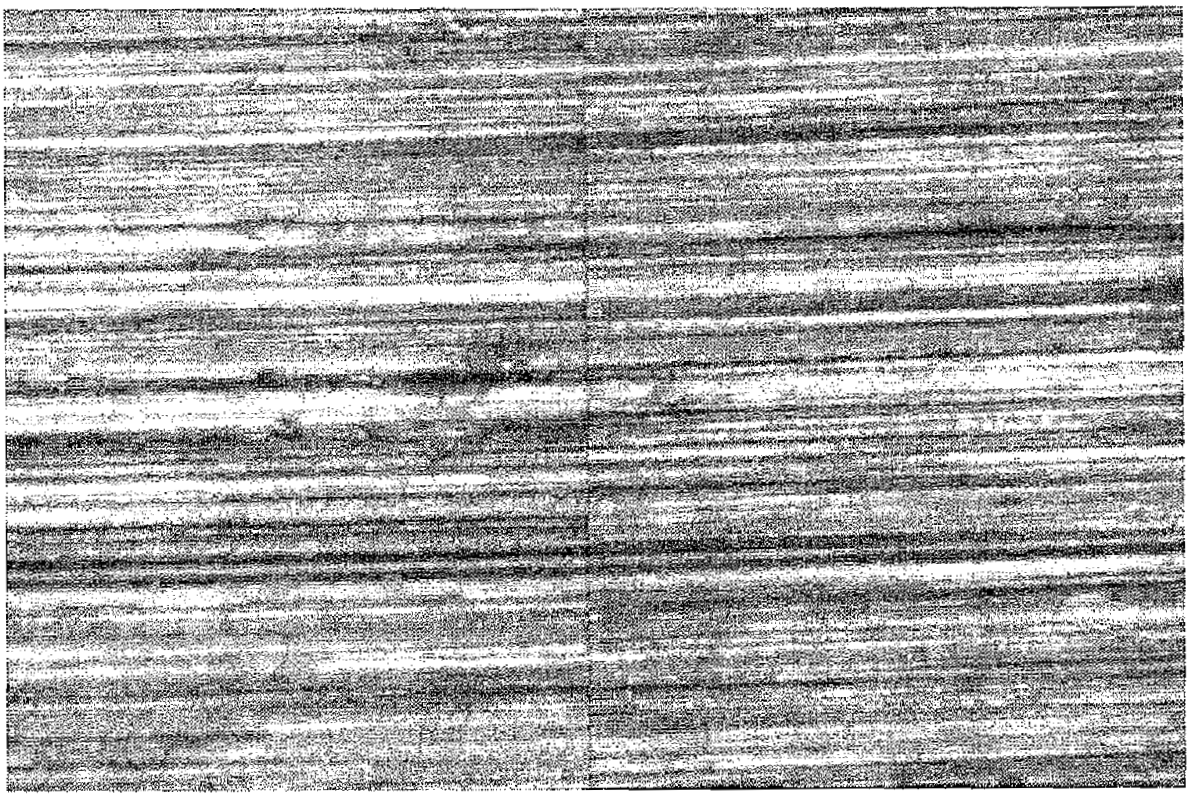

Figure 11. Digital image from comparison microscope of two consecutively made-in-house ground samples illustrating "coincident" matching striated areas.

IV. CONCLUSION / FUTURE WORK

In this report we present preliminary results demonstrating a numerical validation study of digital images to study the uniqueness of toolmarks from different manufacturing processes. The described algorithms were developed to mimic the comparison process used by a forensic scientist. These algorithms were applied to digital grayscale images. The developed algorithms, although preliminary, lay the groundwork for a more detailed and comprehensive study. Initial testing on limited (in number) image databases indicate that a "degree of association" or degree of similarity can be obtained that is indicative for a particular manufacturing process.

When applied to images of ground surfaces, the algorithm works quite well. However, when applied to images of other types of manufacturing processes, for example milling, the number of false negatives increase. This comes as no surprise since the algorithms were developed and select parameters were chosen using images of ground surfaces. It should be noted that the algorithms can be "tuned" for different surfaces, and further developments are necessary to calibrate algorithm parameters for those surfaces.

Although the algorithm and selected parameters may be used to study images of ground surfaces, further refinements and testing are needed. First, the algorithm needs to be translated into a

compiled language, such as C, to speed image comparison. Faster analysis will allow more images to be compared, leading to more refined determinations of error rates and index breakpoints for various surfaces. Additionally, further study is required to determine whether different surfaces need qualitatively different treatment (for example, milled versus ground). We also need to develop a better understanding of how class characteristics affect analyses and whether dominant class characteristics actually make effective matching fundamentally more difficult.

ACKNOWLEDGEMENTS

The authors thank Amber Umble and Molly Schiel, undergraduate students at Iowa State University, for collecting the digital images used in this study. This work was performed at Ames Laboratory, Iowa State University for the United States Department of Energy under Contract W-7405-ENG-82. Funding for this work was provided by the Federal Bureau of Investigation under AL-WFO-2002-11 (FBI Contract A2I206283, A2).

REFERENCES

1. Frye v. United States, **294** F. 1013(D.C. Cir.1923).
2. Nichols, R.G., "Firearm and Toolmark Identification Criteria: A Review of the Literature," *Journal of Forensic Sciences*, **42**, 466-474 (1997).
3. Vandiver, J., "New Screwdrivers Production and Identification," *AFTE J.*, **8**, 29-52 (1976).
4. Watson, D., "The Identification of Toolmarks Produced from Consecutively Produced Knife Blades in Soft Plastics," *AFTE J.*, **10**, 43-45 (1978).
5. Cassidy, F., "Examination of Toolmarks from Sequentially Manufactured Tongue and Groove Pliers," *Journal of Forensic Sciences*, *25*, 796-809 (1980).
6. Tuira, S., "Tire Stabbing with Consecutively Manufactured Knives," *AFTE J.*, **14**, 50-52 (1982).
7. Cressie, N. (1993). Statistics for Spatial Data (revised). Wiley, NY.
8. Gibbons, J.D. (1971). Nonparametric Statistical Inference, McGraw-Hill, NY.
9. Mann, H.B., and D.R. Whitney (1947). "On a Test Whether One of Two Random Variables is Stochastically Larger than the Other," *Annals of Mathematical Statistics* **18**, 50-60.
10. MatLab, The MathWorks, Inc.,1984-2001.

Appendix
(written for MatLab)

validation_simple:

```
function
result=validation_simple(pict_1,gridcen_1,pict_2,gridcen_2,delta,n_check,dist,h,n_best,width,len)
  % Here, gridcen_1 and gridcen_2 are two sets of grid centers we prelocate in images "pict_1"
  % and "pict_2". After running "movetwo" function over pict_1 and pict_2, we get two best
  % (largest correlation) matching centers "center_1" and "center_2", in other words, center_1
and
  % center_2 are very similar. Is this a coincidence? This function validation is trying to
answer
  % this question. If ld1 and ld2 are the least changing direction for center_1 and center_2,
  % random pick a degree theta, for a given distance "dist", we have a new center point
center_new_1
  % by rotating ld1 by theta anticlockwise for pict_1 and center_new_2 for pict_2. Computing the
  % correlation of center_new_1 and center_new_2. If center_1 and center_2 are similar, then this
  % new correlation should be large. Repeat the above step for n_check times, and compute the
  % average of new correlations. Next, we pick two random points, one from the inner pict_1 and
the
  % other from the inner pict_2, compute their correlation. Repeat this step for n_check times.

  % Now, we have two times n_check correlations, the first n_check correlations, we called them
  % rotated correlation, the second n_check correlations, called random correlations; Comparing
  % rotated correlations with random correlations, let index be the ratio of rotated correlations
  % greater than random correlations;

  % The final result will be a length fourteen vector, the first one is the validation index;
  % the second is the old correltaion of center_1 and center_2; the third and fourth are
center_1;
  % the fifth and sixth are center_2; from seventh to tenth are the average,max,min and sd of
  % the n_check rotating correlation; from eleventh to fourteenth are average,max,min and sd of
  % the n_check random correlation.

    % run movetwo over pict_1 and and pict_2;
  temp=movetwo_simple(pict_1,gridcen_1,pict_2,gridcen_2,delta,h,len,width,n_best);

  center_1=temp(8:9);
  ld_1=temp(10);
  center_2=temp(11:12);
  ld_2=temp(13);

  % compute corrlag_fliptell for center_1 and center_2;
    stripe_test=zeros(len,2);
    nn=(len-1)/2+width;
    mat_outer=neighbor_radius(pict_1,center_1(1,1),center_1(1,2),nn);
    stripe_test(:,1)=smooth(ld_1,mat_outer,h,len,width);
    mat_outer=neighbor_radius(pict_2,center_2(1,1),center_2(1,2),nn);
    stripe_test(:,2)=smooth(ld_2,mat_outer,h,len,width);

    temp=corrlag_fliptell(stripe_test(:,1),stripe_test(:,2));
    corr_old=temp(1,1);
    flip=temp(1,4);

    % compute the results associated with rotated correlations;
    corr_rotate_new=zeros(1,n_check);
    center_new_1=zeros(1,2);
    center_new_2=zeros(1,2);
    radian_ld1=(ld_1-1)*pi/180;
    radian_ld2=(ld_2-1)*pi/180;
    for i=1:n_check
        theta=2*pi*rand(1);
        theta_new_1=radian_ld1+theta;
        center_new_1(1,1)=center_1(1,1)-dist*sin(theta_new_1);
        center_new_1(1,2)=center_1(1,2)+dist*cos(theta_new_1);
```

```
    theta_new_2=radian_ld2+flip*pi+theta;
    center_new_2(1,1)=center_2(1,1)-dist*sin(theta_new_2);
    center_new_2(1,2)=center_2(1,2)+dist*cos(theta_new_2);
    center_new_1=round(center_new_1);
    center_new_2=round(center_new_2);

    mat_100=neighbor_radius(pict_1,center_new_1(1,1),center_new_1(1,2),100);
    center_varg=fastvargram_100_tol2(mat_100);
    ld_new_1=avglen_ld(center_varg);

    mat_100=neighbor_radius(pict_2,center_new_2(1,1),center_new_2(1,2),100);
    center_varg=fastvargram_100_tol2(mat_100);
    ld_new_2=avglen_ld(center_varg);

    stripe_test=zeros(len,2);
    mat_outer=neighbor_radius(pict_1,center_new_1(1,1),center_new_1(1,2),nn);
    stripe_test(:,1)=smooth(ld_new_1,mat_outer,h,len,width);
    mat_outer=neighbor_radius(pict_2,center_new_2(1,1),center_new_2(1,2),nn);
    stripe_test(:,2)=smooth(ld_new_2,mat_outer,h,len,width);

    temp=corrlag_flip(stripe_test(:,1),stripe_test(:,2));
    corr_rotate_new(1,i)=temp(1,1);
end;
corr_rotate_mean=mean(corr_rotate_new);
corr_rotate_max=max(corr_rotate_new);
corr_rotate_min=min(corr_rotate_new);
corr_rotate_std=std(corr_rotate_new);

% compute the results associated with random correlations;
corr_rand_new=zeros(1,n_check);
center_rand_1=zeros(1,2);
center_rand_2=zeros(1,2);
nn_dbl=2*nn;
stripe_test=zeros(len,2);
n_row_col=size(pict_1);
n_row=n_row_col(1);
n_col=n_row_col(2);
for i=1:n_check
    center_rand_1(1,1)=round(nn+(n_row-nn_dbl)*rand(1));
    center_rand_1(1,2)=round(nn+(n_col-nn_dbl)*rand(1));
    center_rand_2(1,1)=round(nn+(n_row-nn_dbl)*rand(1));
    center_rand_2(1,2)=round(nn+(n_col-nn_dbl)*rand(1));

    mat_100=neighbor_radius(pict_1,center_rand_1(1,1),center_rand_1(1,2),100);
    center_varg=fastvargram_100_tol2(mat_100);
    ld_rand_1=avglen_ld(center_varg);

    mat_100=neighbor_radius(pict_2,center_rand_2(1,1),center_rand_2(1,2),100);
    center_varg=fastvargram_100_tol2(mat_100);
    ld_rand_2=avglen_ld(center_varg);

    mat_outer=neighbor_radius(pict_1,center_rand_1(1,1),center_rand_1(1,2),nn);
    stripe_test(:,1)=smooth(ld_rand_1,mat_outer,h,len,width);
    mat_outer=neighbor_radius(pict_2,center_rand_2(1,1),center_rand_2(1,2),nn);
    stripe_test(:,2)=smooth(ld_rand_2,mat_outer,h,len,width);

    temp=corrlag_flip(stripe_test(:,1),stripe_test(:,2));
    corr_rand_new(1,i)=temp(1,1);
end;
corr_rand_mean=mean(corr_rand_new);
corr_rand_max=max(corr_rand_new);
corr_rand_min=min(corr_rand_new);
corr_rand_std=std(corr_rand_new);

% compare corr_rotate_new and corr_rand_new;
index=zeros(1);
for i=1:n_check
    temp=corr_rotate_new(i)-corr_rand_new;
```

```
        index=index+sum(temp>=0);
    end;
    index=index/(n_check^2);

    result=[index,corr_old,center_1,center_2,corr_rotate_mean,corr_rotate_max,...
            corr_rotate_min,corr_rotate_std,corr_rand_mean,corr_rand_max,...
            corr_rand_min,corr_rand_std];
```

movetwo_simple:

```
function result_want=movetwo_simple(pict_1,gridcen_1,pict_2,gridcen_2,delta,h,len,width,n_best)
% gridcen_1 is a 25 center points located in the inner pict_1 according some designs.
% Similar for gridcen_2 in pict_2. Pair centers in gridcen_1 with those in gridcen_2, totally
% we got 625 pairs. For these 50 centers, compute their "ld's using aveslopelen. Then apply
% function moveone to the 625 pairs, we'll have 625 correlations;
% For the largest three, now move the corresponding centers by "delta" units along their ld's
% to get more centers for pict_1 or pict_2; Now repeat the above step, to see if the largest
% correlations change, if there is any improvement, repeat steps, if not, then step. Delta
% decrease by half for every new step.
% this function movetwo is to perform the above task. The result will be the largest three
% correlations by move the original center points along two direction, first the fast
% changing direction( through moveone ), then the least changing dirction.

numpoint=size(gridcen_1,1);
ld1=zeros(numpoint,1);
ld2=zeros(numpoint,1);

for i=1:numpoint
    mat_100=neighbor_radius(pict_1,gridcen_1(i,1),gridcen_1(i,2),100);
    center_varg=fastvargram_100_tol2(mat_100);
    ld1(i,1)=avglen_ld(center_varg);
    mat_100=neighbor_radius(pict_2,gridcen_2(i,1),gridcen_2(i,2),100);
    center_varg=fastvargram_100_tol2(mat_100);
    ld2(i,1)=avglen_ld(center_varg);
end;


result=zeros(numpoint^2,13);

oldcen_pict1=gridcen_1;
oldcen_pict2=gridcen_2;

for i=1:numpoint
   for j=1:numpoint
     result(numpoint*(i-1)+j,:)=moveone(pict_1,pict_2,oldcen_pict1(i,:), oldcen_pict2(j,:),...
                                        ld1(i,1),ld2(j,1),len,width,h);
   end;
end;

[result_sort,index_sort]=sort(-result(:,1));
result_sort=result(index_sort,:);

result_old_nbest=result_sort(1:n_best,:);
  % result_old_nbest is an nbest by 13 matrix.
oldcen_pict1=result_old_nbest(:,2:4);
oldcen_pict2=result_old_nbest(:,5:7);


corr_dif=repmat(-1,n_best,1);
step=0;
result_want=result_old_nbest;


newcen_pict1=cell(n_best,1);
```

```
newcen_pict2=cell(n_best,1);
n1_new=zeros(n_best,1);
n2_new=zeros(n_best,1);
ld1_new=cell(n_best,1);
ld2_new=cell(n_best,1);
result_1new=cell(n_best,1);
result_new2=cell(n_best,1);
result_newnew=cell(n_best,1);

n_best_dbl=2*n_best;


while any(corr_dif<0)
  step=step+1;
  delta=delta/(2^(step-1));

  % construct new points from center_3 and from center_4
  newcen_3=result_old_nbest(:,8:9);
  newcen_4=result_old_nbest(:,11:12);


  % construct new points from center_1 and center_2,each has six new center points;
  % also new points newcen_3 and newcen_4,each has six new center points;

   newcen_11=zeros(n_best_dbl,2);
   newcen_22=zeros(n_best_dbl,2);
   newcen_33=zeros(n_best_dbl,2);
   newcen_44=zeros(n_best_dbl,2);

   temp1=delta*sin((result_old_nbest(:,4)-1)/180*pi);
   temp2=delta*cos((result_old_nbest(:,4)-1)/180*pi);
   newcen_11(1:n_best,1)=result_old_nbest(:,2)-temp1;
   newcen_11(1:n_best,2)=result_old_nbest(:,3)+temp2;
   newcen_11((n_best+1):n_best_dbl,1)=result_old_nbest(:,2)+temp1;
   newcen_11((n_best+1):n_best_dbl,2)=result_old_nbest(:,3)-temp2;
   newcen_11=round(newcen_11);

   temp1=delta*sin((result_old_nbest(:,10)-1)/180*pi);
   temp2=delta*cos((result_old_nbest(:,10)-1)/180*pi);
   newcen_33(1:n_best,1)=newcen_3(:,1)-temp1;
   newcen_33(1:n_best,2)=newcen_3(:,2)+temp2;
   newcen_33((n_best+1):n_best_dbl,1)=newcen_3(:,1)+temp1;
   newcen_33((n_best+1):n_best_dbl,2)=newcen_3(:,2)-temp2;
   newcen_33=round(newcen_33);

   temp1=delta*sin((result_old_nbest(:,7)-1)/180*pi);
   temp2=delta*cos((result_old_nbest(:,7)-1)/180*pi);
   newcen_22(1:n_best,1)=result_old_nbest(:,5)-temp1;
   newcen_22(1:n_best,2)=result_old_nbest(:,6)+temp2;
   newcen_22((n_best+1):n_best_dbl,1)=result_old_nbest(:,5)+temp1;
   newcen_22((n_best+1):n_best_dbl,2)=result_old_nbest(:,6)-temp2;
   newcen_22=round(newcen_22);

   temp1=delta*sin((result_old_nbest(:,13)-1)/180*pi);
   temp2=delta*cos((result_old_nbest(:,13)-1)/180*pi);
   newcen_44(1:n_best,1)=newcen_4(:,1)-temp1;
   newcen_44(1:n_best,2)=newcen_4(:,2)+temp2;
   newcen_44((n_best+1):n_best_dbl,1)=newcen_4(:,1)+temp1;
   newcen_44((n_best+1):n_best_dbl,2)=newcen_4(:,2)-temp2;
   newcen_44=round(newcen_44);

   % combine new center for pict1 and pict2;
   for i=1:n_best
     newcen_pict1[i]=[newcen_3(i,:);newcen_11([i,i+n_best],:);newcen_33([i,i+n_best],:)];
     newcen_pict2[i]=[newcen_4(i,:);newcen_22([i,i+n_best],:);newcen_44([i,i+n_best],:)];

     % unduplicate newcen_pict1 and newcen_pict2 from themselves and oldcen_pict1,
     % oldcen_pict2 using the following subfunction "unduplicate";
```

```
    [n1_new(i),newcen_pict1{i}]=unduplicate(newcen_pict1{i},oldcen_pict1(i,:));
    [n2_new(i),newcen_pict2{i}]=unduplicate(newcen_pict2{i},oldcen_pict2(i,:));
  end;

  % compute ld for newcen_pict1 and newcen_pict2;
  for j=1:n_best
    ld1_new{j}=zeros(n1_new(j),1);
    for i=1:n1_new(j)
        mat_100=neighbor_radius(pict_1,newcen_pict1{j}(i,1),newcen_pict1{j}(i,2),100);
        center_varg=fastvargram_100_tol2(mat_100);
        ld1_new{j}(i,1)=avglen_ld(center_varg);
    end;
    newcen_pict1{j}=[newcen_pict1{j},ld1_new{j}];

    ld2_new{j}=zeros(n2_new(j),1);
    for i=1:n2_new(j)
        mat_100=neighbor_radius(pict_2,newcen_pict2{j}(i,1),newcen_pict2{j}(i,2),100);
        center_varg=fastvargram_100_tol2(mat_100);
        ld2_new{j}(i,1)=avglen_ld(center_varg);
    end;
    newcen_pict2{j}=[newcen_pict2{j},ld2_new{j}];
  end;

  result_new=result_old_nbest;
  % compute new correlation by moveone for the new center points;
  for jj=1:n_best
      result_1new{jj}=zeros(n2_new(jj),13);
      result_new2{jj}=zeros(n1_new(jj),13);
      result_newnew{jj}=zeros(n1_new(jj)*n2_new(jj),13);

      for j=1:n2_new(jj)
         result_1new{jj}(j,:)=moveone(pict_1,pict_2,oldcen_pict1(jj,1:2),...
               newcen_pict2{jj}(j,1:2),oldcen_pict1(jj,3),newcen_pict2{jj}(j,3),len,width,h);
      end;

      for i=1:n1_new(jj)
         result_new2{jj}(i,:)=moveone(pict_1,pict_2,newcen_pict1{jj}(i,1:2),...
                oldcen_pict2(jj,1:2),newcen_pict1{jj}(i,3),oldcen_pict2(jj,3),len,width,h);
      end;

      for i=1:n1_new(jj)
         for j=1:n2_new(jj)
         result_newnew{jj}(n2_new(jj)*(i-
1)+j,:)=moveone(pict_1,pict_2,newcen_pict1[jj](i,1:2),...
              newcen_pict2{jj}(j,1:2),newcen_pict1[jj](i,3),newcen_pict2{jj}(j,3),len,width,h];
         end;
      end;

      result_new=[result_new;result_1new{jj};result_new2{jj};result_newnew{jj}];
  end;


 % result_new=[result_old_nbest;result_1new;result_new2;result_newnew];
 [trash,index_sort]=sort(-result_new(:,1));
 result_new=result_new(index_sort,:);
 result_new_nbest=result_new(1:n_best,:);

 corr_dif=result_old_nbest(:,1)-result_new_nbest(:,1);
 result_want=result_new_nbest;

 result_old_nbest=result_new_nbest;
 oldcen_pict1=result_old_nbest(:,2:4);
 oldcen_pict2=result_old_nbest(:,5:7);

end;

result_want=result_want(1,:);
```

```
function [n_new,result]=unduplicate(newcent,oldcent)
   % this subfunction is to unduplicate "newcent" from itself and from "oldcent"
   % here all "newcent" and "oldcent" are three-column matrices, the first two
   % represent the coordinates of centers, the third the offspring.

temp1=newcent(1,:);
temp2=newcent(2:end,:);
result=temp1;
n2=size(temp2,1);
while n2>=1
    temp=temp2(:,1:2)-repmat(temp1(:,1:2),n2,1);
    index=(temp~=0);
    index=sum(index,2);
    temp=temp2(find(index),:);
    if isempty(temp)
        break
    else
      temp1=temp(1,:);
      result=[result;temp1];
      temp2=temp(2:end,:);
      n2=size(temp2,1);
   end;
end;

n_new=size(result,1);
n_old=size(oldcent,1);

temp=result;
for i=1:n_old
    index=((temp(:,1:2)-repmat(oldcent(i,1:2),n_new,1))~=0);
    index=sum(index,2);
    temp=temp(find(index),:);
    n_new=size(temp,1);
end;

result=temp;
```

moveone:

```
function result=moveone(pict_1,pict_2,center_1,center_2,ld1,ld2,len,width,h)
 % this function moveone is to compute the maximal correlation by moving
 % centers by half lag.

halflen=(len-1)/2;
n=halflen+width;

stripe_test=zeros(len,2);
mat_n=neighbor_radius(pict_1,center_1(1,1),center_1(1,2),n);
stripe_test(:,1)=smooth(ld1,mat_n,h,len,width);
mat_n=neighbor_radius(pict_2,center_2(1,1),center_2(1,2),n);
stripe_test(:,2)=smooth(ld2,mat_n,h,len,width);


temp1=corrlag_flip(stripe_test(:,1),stripe_test(:,2));
corr_1=temp1(1);
lag=temp1(2:3);

center_3(1,1)=center_1(1,1)-.5*lag(1)*cos((ld1-1)/180*pi);
center_3(1,2)=center_1(1,2)-.5*lag(1)*sin((ld1-1)/180*pi);
center_3=round(center_3);
center_4(1,1)=center_2(1,1)-.5*lag(2)*cos((ld2-1)/180*pi);
center_4(1,2)=center_2(1,2)-.5*lag(2)*sin((ld2-1)/180*pi);
center_4=round(center_4);
```

```
% compute ld's and stripes for new center_3 and center_4;
mat_100=neighbor_radius(pict_1,center_3(1,1),center_3(1,2),100);
center_varg=fastvargram_100_tol2(mat_100);
ld3=avglen_ld(center_varg);
mat_n=neighbor_radius(pict_1,center_3(1,1),center_3(1,2),n);
stripe_test(:,1)=smooth(ld3,mat_n,h,len,width);

mat_100=neighbor_radius(pict_2,center_4(1,1),center_4(1,2),100);
center_varg=fastvargram_100_tol2(mat_100);
ld4=avglen_ld(center_varg);
mat_n=neighbor_radius(pict_2,center_4(1,1),center_4(1,2),n);
stripe_test(:,2)=smooth(ld4,mat_n,h,len,width);

% compute the maximal correlation for center_3 and center_4
temp2=corrlag_flip(stripe_test(:,1),stripe_test(:,2));
corr_2=temp2(1);
lag=temp2(2:3);


corr_dif=corr_1-corr_2;
corr_want=corr_1;
center_3_want=center_1(1:2);
center_4_want=center_2(1:2);
ld3_want=ld1;
ld4_want=ld2;

stripe_test=zeros(len,2);
while corr_dif<0


     center_3_want=center_3;
     center_4_want=center_4;
     ld3_want=ld3;
     ld4_want=ld4;
     corr_want=corr_2;

     center_3(1,1)=center_3_want(1,1)-.5*lag(1)*cos((ld3-1)/180*pi);
     center_3(1,2)=center_3_want(1,2)-.5*lag(1)*sin((ld3-1)/180*pi);
     center_3=round(center_3);
     center_4(1,1)=center_4_want(1,1)-.5*lag(2)*cos((ld4-1)/180*pi);
     center_4(1,2)=center_4_want(1,2)-.5*lag(2)*sin((ld4-1)/180*pi);
     center_4=round(center_4);

     % compute ld's and stripes for new center_3 and center_4;
     mat_100=neighbor_radius(pict_1,center_3(1,1),center_3(1,2),100);
     center_varg=fastvargram_100_tol2(mat_100);
     ld3=avglen_ld(center_varg);
     mat_n=neighbor_radius(pict_1,center_3(1,1),center_3(1,2),n);
     stripe_test(:,1)=smooth(ld3,mat_n,h,len,width);

     mat_100=neighbor_radius(pict_2,center_4(1,1),center_4(1,2),100);
     center_varg=fastvargram_100_tol2(mat_100);
     ld4=avglen_ld(center_varg);
     mat_n=neighbor_radius(pict_2,center_4(1,1),center_4(1,2),n);
     stripe_test(:,2)=smooth(ld4,mat_n,h,len,width);

     temp2=corrlag_flip(stripe_test(:,1),stripe_test(:,2));
     corr_2=temp2(1);
     lag=temp2(2:3);

     corr_dif=corr_want-corr_2;
end;

result=[corr_want,center_1,ld1,center_2,ld2,center_3_want,ld3_want,center_4_want,ld4_want];
```

fastvariogram_100_tol2:

```
function result=fastvargram_100_tol2(neighbor)
  % this funcion give a variogram of the neighbor by using the already computed
  % "distangl_100_tol2" and "pair_100_tol2". This computation will be very fast,that's
  % why it is called "fastvargram". The "result" will be 24 by 2 by 180 array.

  % load distangl.mat;
  global distangl_100_tol2;
  global pair_100_tol2;
  global pair_100_tol2_noempty_subandsize;

  result=zeros(20,2,180);

  result(:,1,:)=distangl_100_tol2;

  for i=1:2310
       % temp=pair_100_tol2_noempty_sub(i,:);
       % z=neighbor(pair_100_tol2(pair_100_tol2_noempty_subandsize(i,1),...
         %                        pair_100_tol2_noempty_subandsize(i,2)));
       %result(temp(1),2,temp(2))=sum((z(:,1)-z(:,2)).^2)/size(z,1);
       %result(pair_100_tol2_noempty_sub(i,1),2,pair_100_tol2_noempty_sub(i,2))=sum((z(:,1)-
z(:,2)).^2)/size(z,1);
       result(pair_100_tol2_noempty_subandsize(i,1),2,pair_100_tol2_noempty_subandsize(i,2))=...
           sum(diff(neighbor(pair_100_tol2(pair_100_tol2_noempty_subandsize(i,1),...

pair_100_tol2_noempty_subandsize(i,2))),1,2).^2)/pair_100_tol2_noempty_subandsize(i,3);
  end;
```