Final Technical Report

Manipulative Virtual Tools for Tool Mark Characterization

S. Zhang and L. S. Chumbley
Ames Laboratory / Iowa State University
January 25, 2013

Report Title: Manipulative Virtual Tools for Tool Mark Characterization

Award Number: This work was conducted at Ames Laboratory, Iowa State University under the NIJ/DOE Interagency Agreement number 2009-DN-R-119, Manipulative Virtual Tools for Tool Mark Characterization

Author(s): S. Zhang and L. S. Chumbley

This report describes research conducted at Ames Laboratory / Iowa State University involving characterization of toolmarks using quantitative data analysis tools facilitated by advanced simulation/visualization techniques. Persons involved in the project include ISU faculty members L. Scott Chumbley, Max Morris, and Song Zhang. Graduate students involved in the research include Laura Ekstrand, Amy Hoeksema, Yuanzheng Gong, andTaylor Grieve. Undergraduate assistants employed during the course of the work include Barry King,VictorVillagomez, and Morgan Hoke. The project was also assisted by Mr. David Eisenmann, staff scientist at the Center for Nondestructive Evaluation at Iowa State University and Mr. Jim Kreiser, retired forensic examiner for the State of Illinois.

Abstract

The purpose of this research is to provide quantitative data analysis tools for characterization of tool marked surfaces to improve the scientific basis of toolmark identifications. The methodology used will be an integrated approach that combines surface characterization using an optical profilometer to enable development of advanced visualization techniques to analyze and simulate striated surfaces. Specifically, thisresearch was to develop a methodology whereby a three-dimensional (3-D) computer simulation of a tool tip is generated. This "virtual tool" can then be used to produce "virtual toolmarks" - a series of predicted markings where the applied force, twist of the tool, and angle of attack of the tool tip can be varied. Quantitative 3-D data from the suspected tool and evidence toolmarkwas acquired and a virtual reality program developed that takes this data and reconstructs a "virtual tool"for computer manipulation to create "virtual toolmarks". Since the "virtual tool" can be manipulated to produce a range of markings, the exact parameters required to obtain the best possible match to the actual tool mark can be found. Duplicate marks based on these results can then be statistically compared and ranked on the basis of quantitative measurements.

The developed software includes four major parts: (1) automated noise reduction for 3D data of both tools and real marks directly coming from an optical profilometer; (2) arbitrary virtual toolmark generation for any given tool; (3) easy to use software graphical user interface (GUI) for data visualization, manipulation, interaction; and (4) integrated computer-based objective comparison algorithms to provide statistical measure of a pair of toolmarks.

Initial experiments were based on statistically analyzing 6 different tools sampled from the 50 sequentially manufactured screwdriver tips, and 34 actual toolmarks made by a qualified toolmark examiner using a special jig. These scans were carefully cleaned to remove noise from the data acquisition process and assigned a coordinate system that mathematically defines angles and twists in a natural way. Using the virtual tookmark generation method, virtual marks were made at increments of 5 degrees and compared to a scan of the real tool mark. The previously developed statistical algorithm performs the comparison, comparing the similarity of the geometry of both marks to the similarity that would occur due to random chance. Finally, the method informs the forensics examiner of the angle(s) of the best matching virtual mark, allowing the examiner to focus his/her mark analysis on a smaller range of angles and twists.

Experimental results were very promising. In a preliminary study with both sides of 6 tips (and 6 x 2 x 13 = 156 virtual marks) and 34 real marks, the method can distinguish matches from non-matches with only a few false negatives reported (i.e. matches mistaken for non-matches). For matches, it can also generally distinguish between marks made at high and low angles with good prediction.The experimental data indicated that angle for the real mark predicted by the virtual mark could often be achieved to within five degrees of the actual angle.

In summary, the question posed as the goal of this study, "can a manipulative "virtual" tool be made to generate "virtual marks" for quantitative and objective toolmark characterization?" has been answered in the affirmative given the right conditions. Factors affecting the correct identification include the quality of the marking, suitable noise cleaning techniques, suitable virtual mark making approach, and the suitable statistical routine. Moreover, this method presents a unique opportunity to improve tool mark analysis by saving examiners' time and reducing the possible damage to evidence.

Table of Contents

Executive Summary

This project has sought to answer the following question: Can a manipulative "virtual" tool be made to generate "virtual marks" for quantitative and objective toolmark characterization? In other words, can it be said that toolmarks obtained from a "virtual tool" yields characteristics that are unique enough to that tool (and only that tool) to enable matching of the virtual toolmark to actual toolmarks to allow an objective identification algorithm to be employed for identification purposes? Providing answers to these questions based upon quantitative techniques rather than subjective analysis removes uncertainties raised by the 1993 Daubert decision, which created a higher standard for federal courts to accept expert witness testimony. The new standard calls for scientific knowledge with a basis in the scientific method to be the foundation for testimony of expert witnesses (in this field, toolmark examiners). Thus, development of a method of analysis that reduces the subjective nature of comparative evaluation and provides statistical confirmation of a match, with known error rates and confidence intervals, is desirable. The scientific method involving observation, hypothesis formation, hypothesis testing through experimentation, and result analysis was followed in this study. The research sought to directly link a tool to its resultant tool mark in a quantifiable way, providing information concerning the manner in which the mark was made. Additionally, the methodology employed allows a statistical analysis to be made of the nature of the match between the mark and the tool that provides probabilities and error rates for the best match as opposed to when other tools are substituted. Thus, the current research reinforces statistical efforts involving measurement of consecutive matching striations in placing toolmark identification on a sound scientific footing.

This study involves the development of a "virtual tool" to generate "virtual marks" for toolmark characterization. In this study, the following two hypotheses were tested:

Hypothesis 1: *A 3D simulation tool can be generated and manipulated to create "virtual marks" that match with "real marks" made by the same tool under the same conditions (e.g. angle, twist, and force).*

Hypothesis 2: *The virtual tool mark generation technique can be used to effectively and objectively characterize the tool marks.*

The experimental design consisted of (1) making marks with a series of ostensibly identical screwdriver tips acquired previously; (2) obtaining 3D topography information from both screwdriver tips and toolmarks using a non-contact optical profilometer; (3) preprocessing the raw scanned 3D data including automated noise reduction and edge trimming; (4) developing manipulative tools for virtual toolmark generation; and (4) comparison of virtual marks with real marks using a statistical algorithm.

The Ames Lab/Iowa State University team has fifty sequentially produced screwdriver tips that were obtained from Omega Company and used by Mr. Jim Kreiser, former firearm and toolmark examiner for the State of Illinois to create toolmarks. In this study, sides A and B of 6 screwdriver tips were used to mark lead samples at angles of 45°, 60°, and 85° using a jig to maintain the selected angle for the toolmark as closely as possible. The Alicona Infinite Focus Microscope (IFM), a non-contact optical profilometer based on focus variations, was used to measure surface roughness on all toolmarks at 10X spatial resolution. The spatial resolution is 1.6 μm per pixel. The edges of the actual screwdriver tips were scanned at 45° by the same optical profilometer to generate 3D representations of the tips.

A computer software algorithm takes the raw scanned files and cleans the noises coming from the scan for further analysis. A software algorithm takes the cleaned real marks, and generates real trace marks perpendicular to the striations. The trace data consisted of surface height (z) as a function of distance (x) along a linear trace. Another simulation algorithm takes cleaned screwdriver tips, manipulates them, and then produces virtual trace marks for specified parameters (etc. twists, angle, and force), and at exactly the same spatial resolution as the real marks. This is because the analysis algorithm we employed had the assumption that the values of z are reported at equal increments of distance along the trace and that the traces are taken as nearly perpendicular to the striations as possible. The algorithm then allows comparison of two such linear traces. All these algorithms are integrated into a graphical user interface (GUI) for interactive manipulation and visualization.

A statistical algorithm, developed previously by Max Morris of the Ames Lab/Iowa State team and funded by National Institute of Justice, was employed to validate the virtual tool mark characterization technique. Validation depends on the ability to provide some measure of how well the generated virtual mark duplicates an actual mark produced under a controlled set of conditions. The data examined in this analysis are two types of trace data: the virtual mark (VM) and the real mark (RM). The VM trace data are the type of data that are created virtually by the virtual tool at a specified condition.

Briefly, the algorithm determines matching along one-dimensional profilometer data traces (z vs. x) where the values of z are reported at equal increments of distance along the trace and the traces are taken as nearly perpendicular to the striations as possible. Such was the case for the data files available from the Virtual Tool (VT) and VM data files of this study.

The algorithm first goes through an Optimization step to identify a region of best agreement in each of the two data sets. The correlation coefficient (R-value) of this region is determined. The algorithm then conducts a second step in the comparison process called Validation, where corresponding windows of equal size are selected at randomly chosen, but common distances from the previously identified regions of best fit. The assumption behind the Validation step is that if a match truly does exist, correlations between these shifted window pairs will also be reasonably large because they will correspond to common sections of the tool surface. In other words, if a match exists at one point along the scan length (high R-value), there should be fairly large correlations between corresponding pairs of windows along their entire length. However, if a high R-value is found between the comparison windows of two nonmatch samples simply by accident, there is no reason to believe that the accidental match will hold up at other points along the scan length. In this case rigid-shift pairs of windows will likely not result in especially large correlation values.

The correlation values computed from these segment-pairs can be judged to be "large" or "small" only if a baseline can be established for each of the sample comparisons. This is achieved by identifying a second set of paired windows (i.e. data segments), again randomly selected along the length of each trace, but in this case, without the constraint that they represent equal rigid-shifts from their respective regions of best fit.

The Validation step concludes with a comparison of the two sets of correlation values just described, one set from windows of common random rigid-shifts from their respective regions of best agreement, and one set from the independently selected windows. If the assumption of similarity between corresponding points for a match is true, the correlation values of the first set of windows should tend to be larger than those in the second. In other words, the rigid-shift

window pairs should result in higher correlation values than the independently selected, totally random pairs. In the case of a nonmatch, since the identification of a region of best agreement is simply a random event and there truly is no similarity between corresponding points along the trace, the correlations in the two comparison sets should be very similar.

A nonparametric Mann-Whitney U-statistic is generated for the comparison. Where the correlation values of the two comparison sets are similar, T1 takes values near zero, supporting a null hypothesis of "no match". If the correlations from the first rigid-shift sample are systematically larger than the independently selected shifts, the resulting values of T1 are larger, supporting an alternative hypothesis of "match".

Analysis of the data, then involves simply looking at the resulting T1 values to see if any particular hypotheses tested is supported by the data. Values at or near 0 will support the null hypothesis, i.e., there is no relationship between the comparison pairs. A non-zero value says there is a relationship, the greater the separation from 0 the higher probability of a "match" actually existing.

Initial studies based on scanned data directly coming from the optical IFM system were found unsuccessful even with significant effort of post processing the raw data. The major reason found was noise due to the difficulty of optically scanning screwdriver tips because of their surface specularity, and the sharp angle involved between the two faces of interest (around 90 degrees between two faces). It is precisely these same characteristics that cause problems when using a laser confocal measurement method. Therefore, investigations were made to improve the scanning quality. Further studies showed that adding more lighting sources to properly illuminate the screwdriver tips is vital to the success of the research project. Yet, the raw tip scans still contained noise, though at a lower level more easily tolerated by the statistical software algorithm, especially after an automated routine was developed that cleaned the data before further analysis. In addition, due to the limitation of the optical IFM profilometer and the high angle around screwdriver tip edges, this research found that scanning the screwdriver tips at an angle of 45 degrees (i.e., both surfaces make a 45 degree angle with respect to the optic axis of the instrument) will result in the best quality data, which was then used for all tip scans. Optically scanning the actual real toolmark was found straightforward since the surface is close to being flat.

The basic principle behind this tool mark simulation is to take the projection of the tip geometry in the direction of tool travel and identify the highest points on that projection. The highest points will scrape the deepest into the plate material, so they are responsible for leaving the observed striae.

Clearly, this approach is quite simple and ignores several complexities of mark making including material properties of the tool and plate, forces, plastic vs elastic deformation, and the possibility of partial markings. These phenomena can be very difficult to control and account for in a simulation. Since this is a first approximation, the assumption of complete geometry transfer from the tip to the plate was made. Hence, the effects of the specific forces and deformations are treated as insignificant since the geometry seems likely to be the strongest signal in the mark.

Although this method is quite simple in principle, the scale of the geometry simulation required makes it complex to carry out on the computer. Each tip data set at the 10x magnification level used in this research contains more than $9000 \times 1000 = 9$ million points. Projecting and determining the edge of such a large amount of data on the CPU would take a long time.

Therefore, the Open Graphics Library (OpenGL) and its OpenGL Shading Language (GLSL) are used to build the simulation in order to take advantage of the parallel capabilities of the computer's Graphics Processing Unit (GPU). Dividing the geometric computations among theshaders, a computer program that runs on the GPU to do postprocessing, drastically reduces the computation time required and makes virtual mark generation feasible on a standard computer in a reasonable period of time.

To efficiently create virtual marks, a special "virtual camera"was created that looks straight down upon the flattened edge of the screwdriver tip data. This camera records the distance to the points that it sees rather than their color; it is a depth camera formed from an advanced computer graphics tool: an OpenGL object known as a depth buffer. This depth camera is easy to configure and control because they are virtually created. The virtual camera was configured as an orthographical projection (like a real camera attached with a telocentric lens) such that no geometry distortion was introduced; the pixel size of the camera was precisely chosen to be the same as the spatial resolution used for the real mark capture; the orientation of the camera was chosen to reflect the different twist and angle effects on mark making; and the field of depth was controlled to simulate the different force applied during the mark making procedure. Finally, the virtualmark was created by looking at each cross section of the virtual depth image along the direction that the virtual tool intends to move and to make marks; examining the deepest point along each line; and creating the depth as a function of spatial distance. Since all these process are done virtually and in parallel on GPU, the virtual marks can be created quickly.

A graphical user interface (GUI) was also developed to integrate all these tools into a package such that minimum computer skills are required to operate and evaluate the techniques developed. This GUI was designed using Qt. The window is divided into three widgets: tip (top), plate (middle), and statistical comparison (bottom). The tip and plate widgets feature 3D representations of the file geometry on the left side. For these views, the geometry is down-sampled by a factor of 6 to improve graphics speed and performance. Users can left click and drag on the geometry to translate it, and a Qt-provided trackball model allows users to intuitively rotate the geometry with right click and drag. The scroll wheel allows users to zoom in and out. Users can double-click on the realmark (or plate) view to interactively select a cross section of real mark data for comparison. Users can view the geometry in one of four modes by clicking the buttons immediately to the left of the geometry views: shaded, wireframe, textured, and height-mapped. Textured mode overlays the 2D texture from the Alicona onto the 3D geometry. A fifth viewing mode is provided for the tip widget which shows the tip geometry projected in the direction of tool travel; this mode helps users understand the mark generation process.

The right sides of the tip and plate widgets provide plots for profile data. The plate widget provides the name of plate file and a box for changing the selected column. The tip widget provides the name of the tip file and boxes for editing the desired tip rotation for mark generation. Users can click the adjacent button to create a virtual mark; when the mark is complete, users can click on the virtual mark tab to see the view. This trim view presents the recommended end points from the edge detection algorithm as the left and right sides of a purple box and allows users to interactively change these end points. Moreover, the trim view features a flip button that flips the virtual mark to compensate for a plate scanned backwards.

For both the tip and plate widgets, the statistics plot tab provides a view of the trimmed and de-trended mark. De-trending is the process of fitting a first-order line to the data with linear least squares and then subtracting this line from the data; it is an essential preparatory step for the

statistical comparison. Once the user clicks the calculate button in the statistics widget, purple windows pop up on both statistics plots denoting the locations of the search windows.

Initial experiments were based on statistically analyzing sides A and B of 6 different tools sampled from the 50 sequentially manufactured screwdriver tips and 34 actual toolmarks made by a qualified toolmark examiner using a special jig. These real toolmarks were made at angles of 45°, 60°, and 85° with the horizontal (6 x 2 x 3 = 36 real marks; two 45° marks were unavailable at the time of the study). The tip scans were carefully cleaned to remove noise from the data acquisition process and assigned a coordinate system that mathematically defines angles and twists in a natural way. Using the virtual tookmark generation method, virtual marks were made at angles with the horizontal of 30°, 35°, 40°, 45°, 50°, 55°, 60°, 65°, 70°, 75°, 80°, 85°, and 90° (6 x 2 x 13 = 156 virtual marks) and compared to scans of the real tool marks. The previously developed statistical algorithm performs the comparison, comparing the similarity of the geometry of both marks to the similarity that would occur due to random chance. Finally, the method informs the forensics examiner of the angle(s) of the best matching virtual mark, allowing the examiner to focus his/her mark analysis on a smaller range of angles and twists.

Experimental results are very promising. In this preliminary study, the method could distinguish matches from non-matches with only a couple false negatives (i.e. matches mistaken for non-matches). For matches, it can also generally distinguish between marks made at high and low angles with good prediction.The experimental data indicated that the matching angle might not be perfectly the same as angle as the mark was made at, but is very close. These slight discrepancies could be caused by the real toolmarks ostensibly being made at a particular angle actually being slightly different from the desired angle value due to slight deflection of the tip during the mark making process and/or slightly non-flat lead plate used for making the mark.

In summary, the questions posed as the goal of this study, "can a manipulative "virtual" tool be made to generate "virtual marks" and can those marks be used for quantitative and objective toolmark characterization?" have been answered in the affirmative given the right conditions. The results of this developed method support the conclusions of traditional tool mark analysis based upon observation and experience, giving them a firmer statistical background. Factors affecting the correct identification include the quality of the marking, suitable noise cleaning techniques, suitable virtual mark making approach, and the suitable statistical routine. The noise of the raw data directly obtained from an optical profilometry cannot be too large on the edges to ensure the success of the developed methodologies, and thus special care shall be given to this step. Moreover, this method presents a unique opportunity to improve tool mark analysis by saving examiners' time and reducing the possible damage to evidence.

The software package developed uses module-based strategies, making it easier to change some modules. The GUI was developed with Qt, and the programming language used is C++. The software was designed in such a way that it can operate on mobile computers (e.g., a laptop) with standard hardware configurations. A further study involves development of an all-mobile system for toolmark and firearm examinations is of considerable interest. The intention of making this software package free and open source once it matures could benefit the whole community at large.

I. Introduction

I.1: Statement of the problems

This project has sought to answer the following question: Can a manipulative "virtual" tool be made to generate "virtual marks" for quantitative and objective toolmark characterization? In other words, can it be said that toolmarks obtained from a "virtual tool" yields characteristics that are unique enough to that tool (and only that tool) to enable matching of the virtual toolmark to actual toolmarks to allow an objective identification algorithm to be employed for identification purposes? Providing answers to these questions based upon quantitative techniques rather than subjective analysis removes uncertainties raised by the 1993 Daubert decision, which created a higher standard for federal courts to accept expert witness testimony. The new standard calls for scientific knowledge with a basis in the scientific method to be the foundation for testimony of expert witnesses (in this field, toolmark examiners). Thus, development of a method of analysis that reduces the subjective nature of comparative evaluation and provides statistical confirmation of a match, with known error rates and confidence intervals, is desirable. The scientific method involving observation, hypothesis formation, hypothesis testing through experimentation, and result analysis was followed in this study. The research sought to directly link a tool to its resultant tool mark in a quantifiable way, providing information concerning the manner in which the mark was made. Additionally, the methodology employed allows a statistical analysis to be made of the nature of the match between the mark and the tool that provides probabilities and error rates for the best match as opposed to when other tools are substituted. Thus, the current research reinforces statistical efforts involving measurement of consecutive matching striations in placing toolmark identification on a sound scientific footing.

I.2: Literature citations and review

In the sixteen years since the 1993 Daubert vs. State of Florida decision, increasing attacks have been aimed at firearm and tool mark examiners by defense attorneys via motions to exclude evidence based on expert testimony. Such motions claim that the study of tool marks has no scientific basis, that error rates are unknown and incalculable, and that comparisons are subjective and prejudicial. These motions misstate the true nature of firearm and tool mark research in a number of ways. The claim that scientific evidence is lacking in tool mark examinations ignores the numerous studies that have been conducted, especially in the area of firearms [Biasotti, 1959; Bonfantis and DeKinder, 1999a, 199b; Biasotti and Murdock, 1984], to investigate the reproducibility and durability of markings. These studies have shown time and again that while matching of cartridges cannot be universally applied to all makes and models of guns using all types of ammunition, the characteristic markings produced are often quite durable and a high percentage can be successfully identified using optical microscopy, a fact supported by a recent report from the National Academy of Sciences [NAS Report, 2008]. While error rates are unknown, and that the statistical probability of different guns having identical markings has not been established, it must be understood that establishing error rates and probabilities in the area of tool marks is fundamentally different than in an area such as genetic matching involving DNA. When considering genetic matching, all the variables and parameters of a DNA strand are known and error rates can be calculated with a high degree of accuracy. This is not the case in tool marks where the variables of force, angle of attack, motion of the tool, surface finish of the tool, past history of use, etc. are not known or cannot be determined, and the possibility for variation is always increasing as the population under study continues to increase and change. For practical purposes, this may indeed mean that realistic error rates cannot be completely characterized. However experiments based on sequentially manufactured tools may

lead to useful approximations and/or bounds, and such studies are underway [Faden et al. 2007; Chumbley et al., 2010].

The proposition that tool marks must necessarily have a quantifiable basis is the principle upon which the Integrated Ballistics Imaging System (IBIS) developed and manufactured by Forensic Technology, Inc. for bullets and cartridge cases operates. IBIS uses fixed lighting and an image capture system to obtain a standard digital image file of the bullet or cartridge case. The contrast displayed in the image is reduced to a digital signal that can then be used for rapid comparisons to other files in a search mode. The latest version of IBIS uses the actual surface roughness as measured by a confocal microscope to generate a comparison file [Bachrach, 2002]. The results are displayed in a manner analogous to a web search engine, where possibilities are listed in order with numbers associated with each possibility. An experienced tool mark examiner must then review the list of possibilities to make a judgment as to whether a match does, in fact, exist. In instances where a match is declared, it is quite common for the match not to be the first possibility displayed by IBIS, but to be further down the list. In other words, while the analysis/algorithm employed by FTI produces the numbers associated with each match, these numbers carry no clear statistical relevance or interpretation related to the quality or probability-of-match of any given comparison [NAS Report, 2008].

In recent studies researchers at Iowa State University (ISU) have developed a computer-based data analysis technique that allows rapid comparison of large numbers of data files of the type that might be produced when studying striated tool marks [Faden et al., 2007; Chumbley et al., 2010; Baldwin et al., 2004]. In these studies sequentially manufactured tools have been used to produce tool marks, which are then characterized using various means to yield a data file consisting of quantitative measurements. A major aim of the research has been to construct well-defined numerical indices, based upon the information contained within the tool mark itself, that are useful in establishing error rates for objective tool mark matching. While this error rate may only be practically achievable for a particular set of experimental conditions, it should serve as a benchmark error rate for subsequent studies. Experiments involving comparisons of samples obtained from a single tool to each other, and to samples produced from other similar sequentially manufactured tools, show that the analysis can fairly reliably separate sample pairs that are known matches from the same tool from pairs obtained from different tools [Faden et al., 2007; Chumbley et al., 2010]. Additionally, the index provides a means of calculating estimates of error rates within the specific setting of the tools studied. It is interesting to note that while the computer algorithm developed at ISU works very well, it still falls short of the proficiency of an experienced tool mark examiner [Chumbley et al., 2010].

The studies cited above involve comparisons of marks produced from either sequentially made tools or marked bullets fired from similar guns. In all instances the comparisons are of marks to marks, i.e. a connection is inferred to exist between the marked surfaces and the tool that produced them. While this assumption is believed to be a valid one, in the current era of court challenges and newspaper reports it would seem advisable that work be undertaken to provide a more direct link between a tool marked surface and the tool which produced it. This approach has a significant advantage in that it establishes a direct link between the mark and the suspect tool. Additionally, if a connection between a specific tool or weapon can be established in quantifiable terms, error rates and probabilities can be established between the suspect tool and similar tools.

The research described below outlines a program that seeks to directly link a tool to its resultant tool mark in a quantifiable way, providing information concerning the manner in which the mark

was made. Additionally, we anticipate that the methodology employed will allow a statistical analysis to be made of the nature of the match between the mark and the tool that will provide probabilities an error rates for the best match as opposed to when other tools are substituted. This proposal was based on initial experiments that have already been conducted, and many potential problems have already been encountered and overcome.

I.3: Statement of hypothesis and rationale for this research

This study involves the development of a "virtual tool" to generate "virtual marks" for toolmark characterization. In this study, the following two hypotheses were tested:

Hypothesis 1: *A 3D simulation tool can be generated and manipulated to create "virtual marks" that match with "real marks" made by the same tool under the same conditions (e.g. angle, twist, and force).*

Hypothesis 2: *The virtual tool mark generation technique can be used to effectively and objectively characterize the tool marks.*

These hypotheses were tested using the procedures outlined below.

II. Methods

II.1: Non-contact 3D optical profilometer

The Alicona Infinite Focus Microscope (IFM), shown in Figure 1, was used to digitize the 3D geometry of both the tips and the tool marks. The IFM captures the surface topology using an operating principle known as focus variation or depth from focus/defocus [Alicona, 2012; Bolton-King, 2010]. This technique is based on the focal plane inherent in any optical system. Objects at the focal distance from the optical detector will be in sharp focus in the resulting image, whereas objects closer or farther away from the detector will be slightly blurred in the image. In the Alicona IFM Model G3, the sample is placed on a precision stage that mechanically moves it up and down (in and out of focus) relative to the detector. Tracking the sharpness of the features and the displacement of the stage allows the IFM to measure the height of the features along with their 2D color texture.

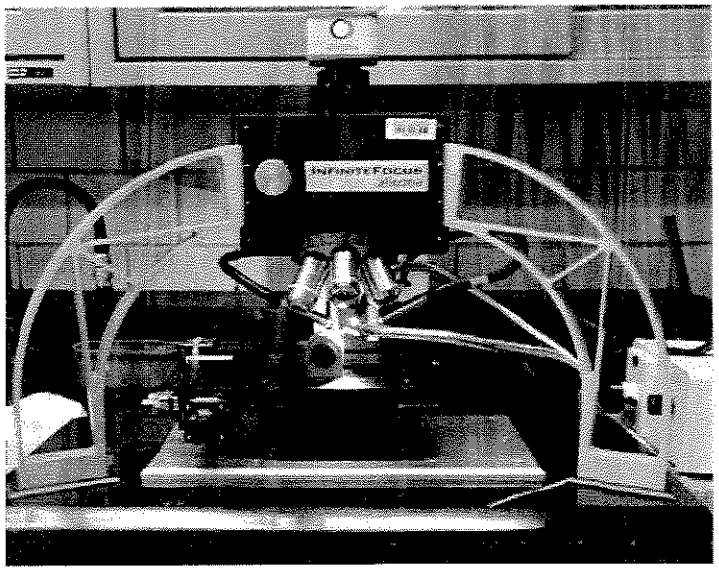


Figure 1: *Photo of the AliconaInfiniteFocus Microscope (IFM)*

The IFM was chosen for investigating tool tips early in 2008. For her master's thesis research, Kidd [2007] tested the ability of a wide variety of 3D measurement systems to capture the geometry of screwdriver tips. Initially, she used the AliconaMeX triangulation software to compute the depth from pairs of Scanning Electron Microscope (SEM) images of the tips taken at varying angles. However, these measurements contained too much noise for forensic analysis. The stylus profilometer had difficulty measuring along the edge of the tip and required physical contact with the tip for measurements (which is undesirable since it potentially alters the evidence). The laser optical interferometer had difficulty staying appropriately focused on the sample, and therefore its measurements also contained a large amount of noise. Penetrating dye, confocal florescence, and x-ray tomography all failed due to inability to resolve the tool tip striations due to its steep angles (close to 90 degrees). Representatives at Alicona were able to successfully capture the surface patterns on one of Kidd's screwdriver tips using the IFM, which played a large role in its selection.

Moreover, at that time the IFM outperformed the 3D confocal microscope in its ability to measure steep angles. Bolton-King et al. [2010] demonstrated that the confocal microscope, unlike the focus-variation technique, could not detect the transitions between the land-engraved

areas and groove-engraved areas on the NIST standard bullet. The confocal technique at the time was limited to a maximum surface angle of 70 degrees due to its numerical aperture, whereas the focus-variation technique could achieve measurements of surfaces at close to 90 angles to the detector. Similarly, the confocal technique would have had difficulty measuring the approximately 90 degree angle of the screwdriver tip. For this reason, the IFM was chosen to perform this research. In the future, as Bolton-King et al. [2010] inferred, the confocal technique may improve and become a better choice for this type of research.

II.2: Noise reduction

3D data analysis starts with acquiring 3D data from physical samples. Often, the 3D raw data coming from an optical profilometer contain noises, making it difficult for further analysis. Specifically, we are processing and analyzing 3D data of screwdriver tools and marks scanned by the IFM. Since the screwdriver tips are shiny with steep angles (close to be 90 degrees), the measurement results are full of noise that must be significantly reduced [Zhang et al., 2009].

In this study, the screwdriver tips were measured at 45-deg. by the IFM, which was found to give the best quality and the most reliable scans; and the marks were also scanned by the same IFM, both at 10x resolution. The software provided by Alicona requires an extremely time-consuming procedure to clean up the noise manually; and due to the huge size of data produced by the IFM, e.g., approximately 5 GB per frame in *.Obj file format, commercially available geometry processing tools such as Maya, GSI (Geometry System Inc), Solidworks, failed to handle such data. Thus, a new means is needed for efficient processing of such 3D geometry data.

Two methods were investigated in this research: (1) develop software algorithms to automatically clean the noises on the raw data; and (2) optimize the IFM setup such that highest possible data can be acquired.

II.2.1: Automatic noise reduction software algorithm

Figure 2 shows the raw data of a screwdriver tip scanned by the Alicona IFM, which clearly shows that the noise (spikes) is substantial and the desired signal (the profile of the screw driver tip) is hidden in the noisy surroundings. To process those extremely noisy data,we developed the following automated algorithms. For any measurement point (i, j) of a 3D scan, it contains x_{ij}, y_{ij}, z_{ij} coordinates and the texture (color) information. Where, $i$is the row number, and j the column number. For a given row, $i = i_0$, the relationship between the z coordinates and the x coordinates is approximated as an n-th order polynomial, that is,

$$\bar{z}(x) = \sum_{k}^{n} c_k x^k, \tag{1.1}$$

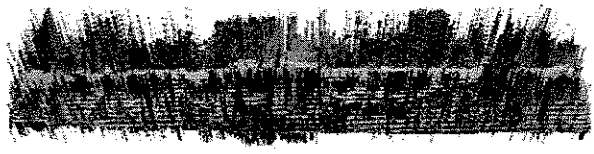
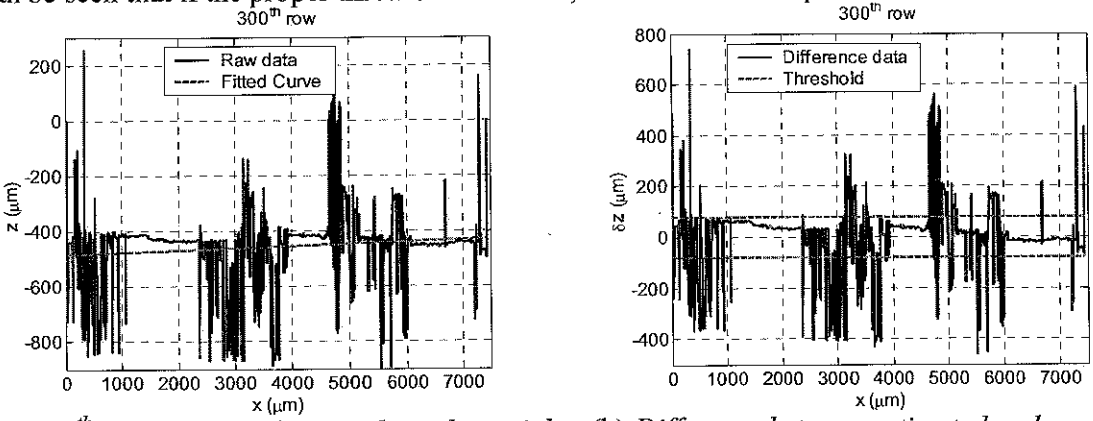


Figure 2:*Raw measurement result of a screwdriver tip without additional lighting*

here, from the measurement points $x_{i,j}$ and $z_{i,j}$ $(j = 1, 2, \ldots N)$, c_k is estimated using the least square algorithm. Once the polynomial function is obtained, the difference between the approximated data and the actual data is calculated, if the difference is beyond a given threshold (th), this point is marked as bad point and eliminated, namely,

$$|\bar{z}(x_{i_0 j}) - z_{i_0 j}| > th \xrightarrow{yields} (i_0, j)\, is\, bad. \tag{1.2}$$

Figure 3(a) shows a typical row data with significant amount of bad points. Because the horizontally the screw driver tip is relatively flat, a 1st –order polynomial is used to approximate the data points, which is shown in the red curve. Figure 3(b) shows the difference between the measured data and the approximated result. If a threshold of 80 μm is used, the data points below the lower red line or above the topper red line will be treated as bad and should be eliminated. It can be seen that if the proper threshold is chosen, almost all the bad points can be removed.

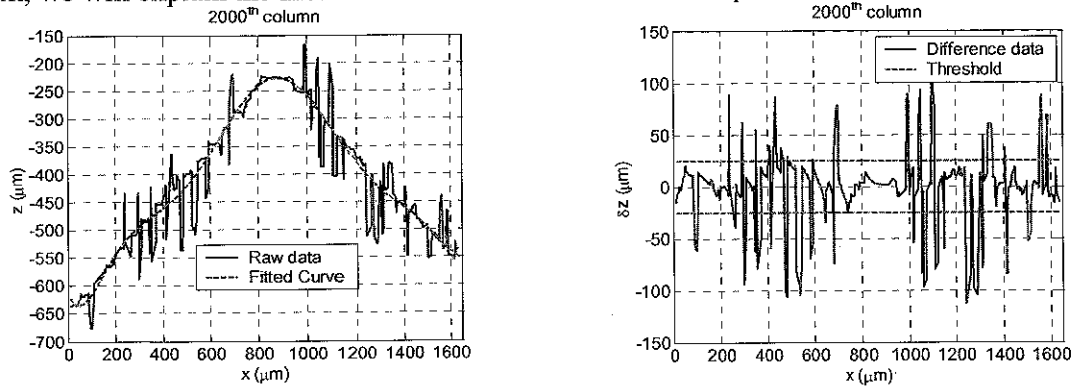


(a) *300th row fitting with 1st order polynomial* (b) *Difference between estimated and raw data*

Figure 3:*Fit the data line-by-line by polynomials horizontally*

Similar operations are applied to the remaining good points vertically line by line. Because the vertically the screwdriver tip is not flat, a 9th-order polynomial was found to be appropriate to fit the curve. Figure 4 shows the corresponding to the plots of 2000th row. It clearly shows that the 9th order polynomial well represents the curve, and a threshold of 25 μm is sufficient to eliminate most of the bad points. After this operation, more bad points are removed and the data is further cleaned.

Figure 5(a) shows the result after removing all bad points. In this figure, removed bad points are depicted as black areas. This figure clearly shows that most of the bad points (spikes) in Figure 2 are successfully removed. It should be noted that the same threshold is used for all horizontal lines (vertical lines), and the same order of polynomial is used for all horizontal lines (vertical lines). However, leaving the holes in the geometry is not desirable for future data processing. Next, we will explain the method used to fill in those removed points.



(a) 300th row fitting with 9th order polynomial (b) *Difference between estimated and raw data*

Figure 4:*Fit the remaining good data point line-by-line by polynomials vertically*

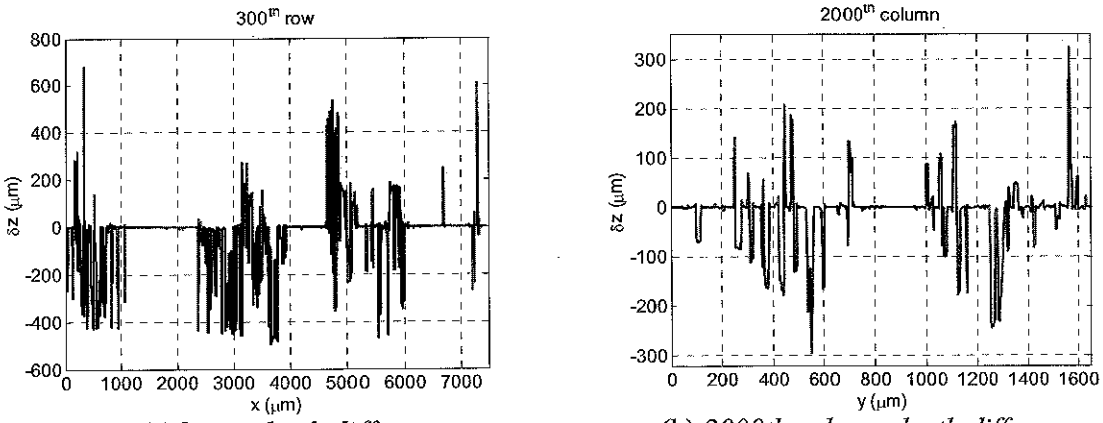
(a) Cleaned data before hole filling *(b) Cleaned data with hole filling*

Figure 5:*Final results after applying the noise reduction algorithm*

To fill in those removed bad points, the remaining good points are used to generate polynomials line-by-line vertically and horizontally. The bad points are then replaced with the value calculated by the polynomial functions without touching the good points. Figure 5(b) shows the result after processing. All spiky bad points are removed, and the profile of the screwdriver tip is well represented.

To verify the influence of this processing algorithm, the difference image between the raw 3D data and the resultant 3D data is calculated point by point. Figure 6 shows the plot of a typical cross sections, 300th row and 2000th column, respectively. It clearly shows that the difference image contains mostly the spiky noisy points while remains the good points untouched.

More experiments demonstrated that by properly selecting the order of polynomials and the thresholds, almost 100% bad points are found and eliminated without significantly affecting good points. In addition, this proposed algorithm is very fast, it only takes approximately 6 seconds to process 4656×1028 data points with a dual core 2.66GHz Pentium 4 CPU Dell Optiplex 330 desktop computer. Based upon experiments, a conservative threshold of 100 μm was chosen for both the rows and the columns and used for all of the screwdriver tips in this study.



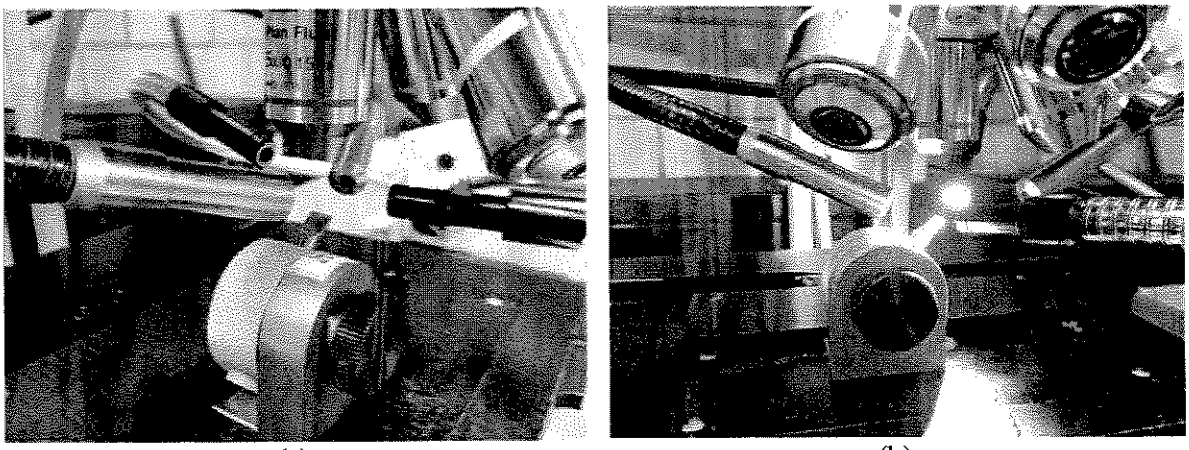
(a) *300th row depth difference* (b) *2000th column depth difference*

Figure 6:*The cross section of the difference between the raw data and the final data.*

II.2.2: Noise reduction by improving raw scans

However, our study showed that even after cleaning up the noise from the original scans using the aforementioned noise reduction algorithm, it was difficult to use a virtual tool for toolmark characterization because the noisy points are artificially filled in, which may not truthfully represent the physical characteristics of the screwdriver tips. Therefore, a significant effort has been devoted to improving the scanning quality itself before any post processing.

To improve the scan quality, in addition to the lighting coming through the lens of the IFM, four fiber optic lighting cables, each pair lit by a 150 W white light source, are equally positioned around the screwdriver tip during scanning to provide more neutral lighting with fewer specular highlights. Figures 1 and 7show these additional lights.The gooseneck lights were not on in the Figure 7(a) so one can see what the setup is, as the lights are a bit blinding when the scan is set up to run as shown in Figure 7(b). Basically, we had the tip set up in the desired angle in the holder and had the holder sitting flush with the back of the stage. The tip was centered underneath the objective lens chosen (in these pictures it's 5x). The lighting then comes from the lens (above), left and right from the gooseneck lights with the silver tips and top and bottom from the black tipped gooseneck lights. All of these lights were adjusted for intensity and placement (for gooseneck lights) to ensure the best illumination of the tip.



(a) (b)

Figure 7: *Close-ups of the tip lighting in the IFM, showing the four fiber optic cables. (a) Additional light positions; (b) When all additional lights are ON.*

The capturing software provided by Alicona provides an automatic color correct tool and an adjustable gamma value to get to proper setting for the tip. The scan is then run at 5x low resolution to begin with to see if there are any large spikes that need to be corrected. We keep adjusting the lighting if there are spikes until we come out with a smooth scan. Then we move onto either a higher resolution 5x scan or a 10x low resolution scan. There really are no special tricks with this scan setup, all it is lighting adjustment. When we make an adjustment, usually it's a small change, either the intensity on one of the gooseneck light setups or a slight repositioning of the lights to get more or less illumination where it's needed on the tip. The scanned data are then visualized with the software GUI we developed to double check their quality; if the quality is good, the scan is kept, otherwise, we have to rescan that sample.

Figure 8 shows a representative data with a much higher quality, comparing with the initial data as shown in Figure 2. This clearly shows that the additional lighting indeed drastically improved the data quality. Yet, the data still contains noises, albeit they not as severe as before.

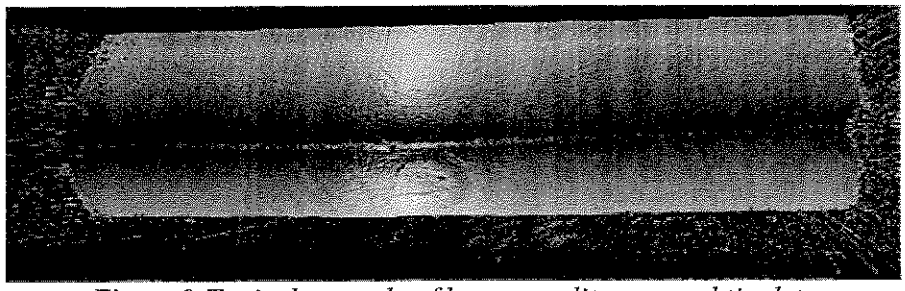
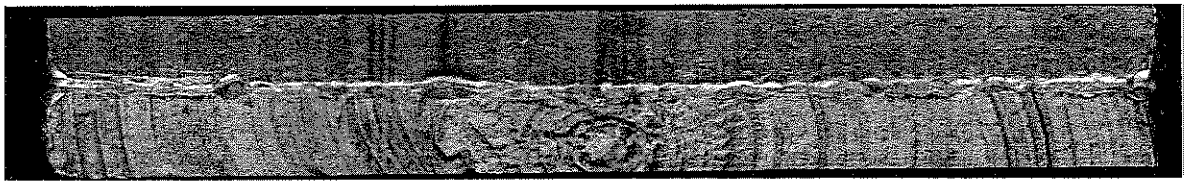


Figure 8:*Typical example of better-quality scanned tip data*

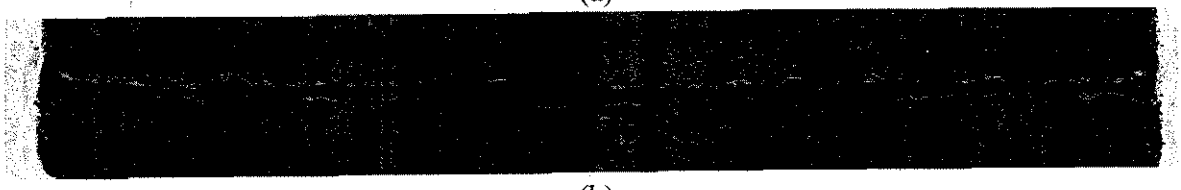
II.2.3: Tip noise reduction

Although with the added lighting, the scan quality was drastically improved. Yet, two types of noise remain on screwdriver tips that still interfere with virtual tool mark making. First, spike noise appears at the ends of the screwdriver tip. Figure 8 shows a representative example. With reference to the screwdriver texture in Figures 9(a), we can see that this type of noise occurs where there is no real screwdriver geometry. In particular, the spike noise in Figure 8 is very dark in comparison to the light texture of the screwdriver, and from visual inspection of the 2D texture in Figure 9(a), it appears that this dark area is beyond the ends of the screwdriver tip.

To remove this end noise, we can simply compute a mask (a memory buffer indicating which data pixels to ignore in the analysis) based on the texture and the quality map. Figure 9(b) shows the Alicona IFM's quality map, a metric displaying the quality of the measurement; high quality regions are dark and low quality regions are light in this image. Comparing this to the 2D texture, it is evident that the quality map also well defines the end of the screwdriver. Pixels with 8-bit grayscale values in the quality map of over 200 and pixels where the maximum of the red, green, and blue 8-bit values was less than 20 were turned off in the mask (thresholded). A standard connected components algorithm was then used to find the largest single mass of on values in the mask; all other on values were turned off to prevent little islands of noise from appearing around the edge of the screwdriver. Finally, a 20-pixel border was removed from the mask to ensure that no leftover edges (and their accompanying spikes) remained.



(a)



(b)

Figure 9:*Texture and quality map. (a) 2D texture of scan in Figure 8. (b) Alicona-computed quality map*

Figure 10 presents the results of this noise removal process. These results are representative of end noise removal. As the figure demonstrates, the tip ends have become acceptably clean for forensic analysis.

The second type of noise encountered is the presence of spikes in the middle of the dataset. Figure 11 presents two examples of this type of noise. Figure11(a)shows an extreme example of these spikes; Figure11(b) shows a more representative example of these spikes. All of these spikes show up as extremely white and bright. Therefore, it seems readily apparent that the source of these artifacts is saturation in the image sensor due to specular highlights on the metal surface.

A previously addressed: two approaches have been used to reduce this type of noise. First, the tip scanning procedure has been radically altered to improve the quality by adopting the method discussed in Section II.2.2. Comparing Figures 2, representative of the screwdriver data taken before the new lighting procedure, and all data taken after the new lighting procedure, we can see that the lighting procedure has drastically reduced the noise.

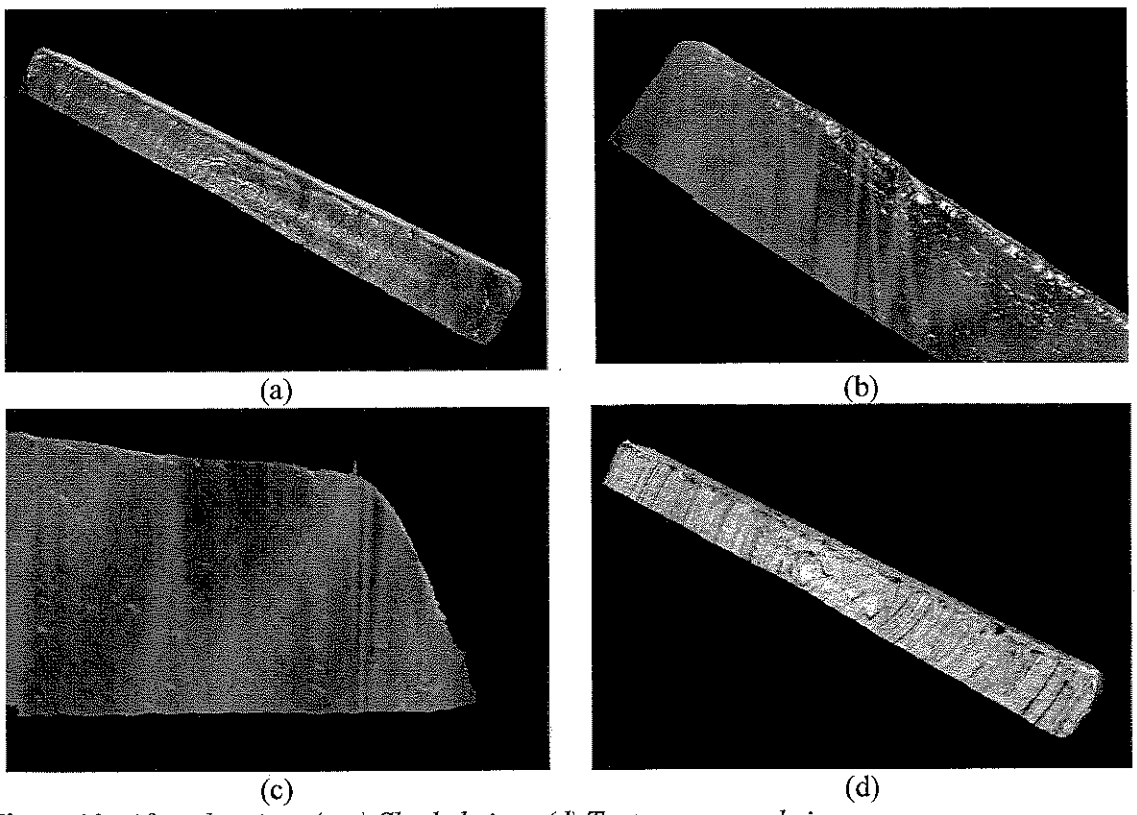
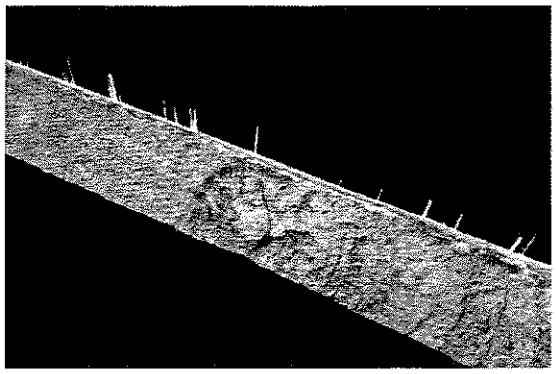
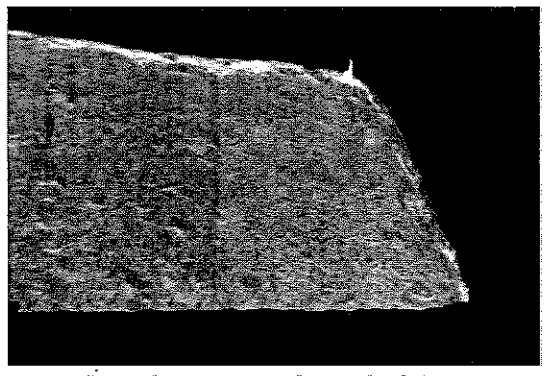


Figure 10: *After cleaning. (a-c) Shaded view. (d) Texture-mapped view.*



(c) *Spiky noise on the edge*



(d) *Spiky noise at the end of the tip*

Figure 11: *Spike noise in the middle of the dataset.*

The second approach used to reduce the spike noise in the middle of the dataset is a processing algorithm discussed in Section II.2.1. The results of this second algorithmic noise reduction approach are mixed. Figure 12 presents the result after cleaning the noise shown in Figure 11(a). The noise is indeed dramatically reduced, but it is not completely eliminated. Clearly, the parameters of the algorithm could be tweaked to reduce more noise, although the hole filling size would have to be raised carefully to avoid introducing too much interpolated data. However, the data captured under the new lighting technique could be further improved to minimize the influence of this type of noise. For such scans, we typically rescan those tips before further analysis. Figure 13 shows the high-quality results after noise cleaning with a better quality scan. The data is so clear that the few remaining should have an insignificant effect on the statistical matching algorithm. Therefore, all of the data used for this research was taken with the new lighting approach to obtain high quality scans to start with.

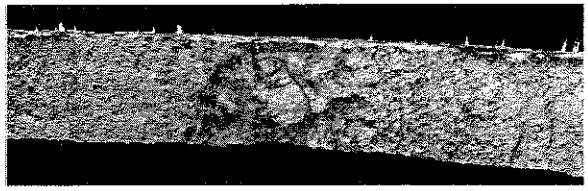


Figure 12: *Result after algorithmic cleaning for example show in Fig. 11(a).*

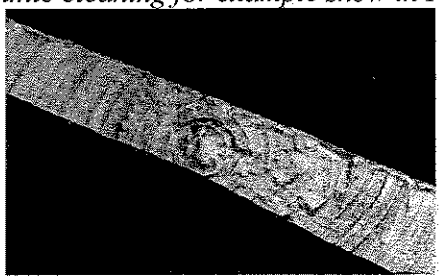


Figure 13: *Result after algorithmic cleaning for example show in Fig. 11(b)*

In summary, the current cleaning techniques proved largely successful at mitigating significant noise. The remaining few small spikes should not have a significant impact on the results.

II.2.4: Mark plate noise cleaning

Mark plate (real mark) noise cleaning is quite easy comparing with the tip scan because plates are rather flat and the scanning quality is usually much higher. Figure 14shows examples of

originally scanned before applying the noise reduction algorithms and those after applying noise cleaning algorithm we developed. It clearly shows that these results are of high-quality after noise cleaning.

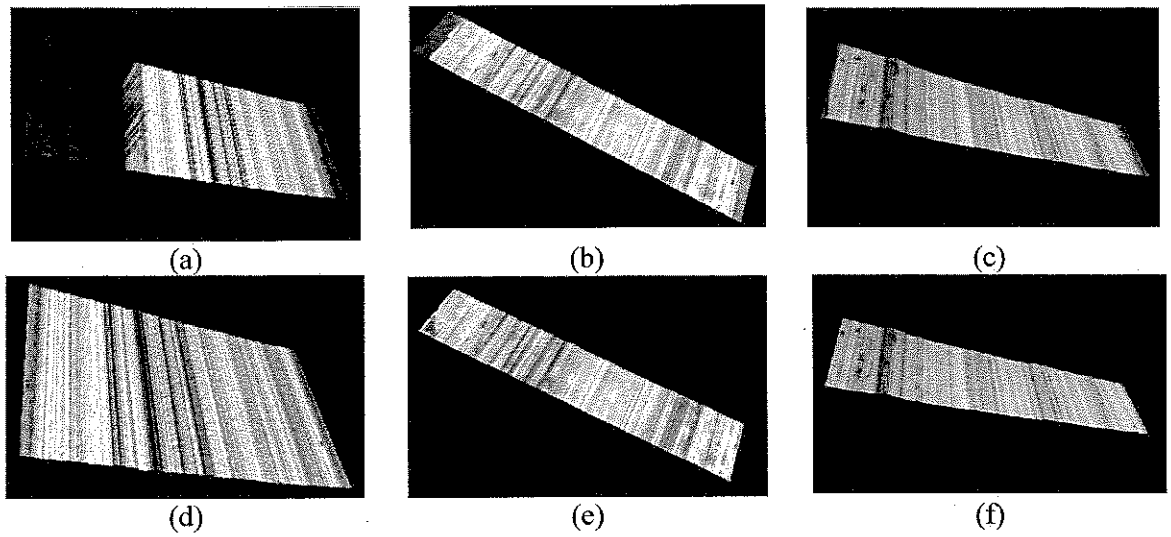


Figure 14: *Plates beforeafter noise reduction. (a-c) Plate scans before noise cleaning. (d-f) Correspondingresults after noise cleaning.*

II.4: Coordinate system assignment

At this point, the cleaned data of the tip measurement is defined in terms of the coordinate system of the IFM. In order to meaningfully compare virtual marks made with this tip to marks generated in the real world, we need to give the virtual tip a new coordinate system derived from the real-world marking coordinate system. Figure 15 illustrates the process used to generate tool marks in lead. The screwdriver is placed in the mark-making jig shown in Figure 15, which constrains the screwdriver to move in the x direction of the world coordinate system. Care is taken when tightening the black thumbscrew to align the tip edge with the direction of the y axis. The design of the jig ensures that the screwdriver shaft is held at angle α about the y axis during the marking process. The jig in Figure 15 defines an angle $\alpha = 45°$; other similar jigs are also used that define the angles $\alpha = 30°$, 60°, and 85°.

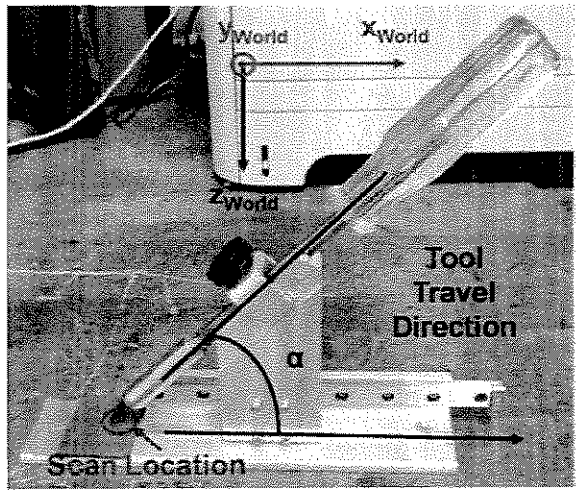
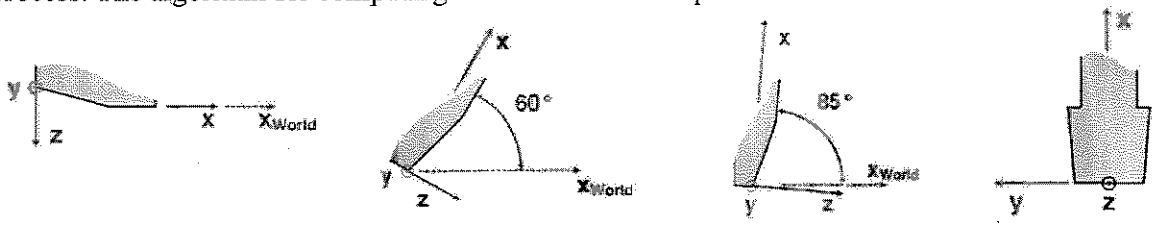


Figure 15: *Jig for making screwdriver marks in lead at $\alpha = 45°$.*

To define the position of the tip relative to the world coordinate system, we rigidly assign the coordinate system shown in Figure 16 to the tip geometry. Figure 16(a) shows the orientation of the tip when the angle α is 0°. Figures 16(b) and (c) show the tip's orientation at $\alpha = 60°$ and 85°, respectively. It can be seen from these figures that the y axis of the tip coordinate system lies along the edge of screwdriver tip so that a rotation about the y axis is a rotation about the edge. This makes sense since the edge is what will contact the lead plate. The x axis is made to point along the shaft of the screwdriver; this corresponds to the fact that the jig holds the shaft (and not the flat side of the screwdriver) at the angle α. Finally, since the angle between the shaft and the flat bottom of the plane is approximately 90°, the z axis will point along this flat bottom. Figure 16(d) reveals that the coordinate system is centered along the screwdriver edge. This was chosen mainly out of convenience, since for now only variation in y angle rotation is under study. At this time, it is suspected that the exact origin of coordinates is not as important as the orientation of the axes, since translations in the mark location can be identified in the computer and removed.

To make virtual marks that correspond to the real marks, we must affix the coordinate system of Figure 16 to the measured tool geometry from the IFM, redefining the geometry in terms of this new coordinate system. This is easily done by computing a linear transformation between the two coordinate systems and applying this to the geometry in OpenGL during the marking process. The algorithm for computing this transformation proceeds as follows:



(a) (b) (c) (d)

Figure 16: *Tip coordinate system. (a-c) View of tip side. (d) View looking at the flat side of the screwdriver*

1. *Identify the screwdriver edge*. This is accomplished by sampling approximately 250-300 rows evenly spaced throughout the height of the tip data and recording the points in those rows with the maximum height. Figure 17 illustrates this process, depicting the selected rows with red arrows and the recorded points with red x's. Only 250-300 rows are selected since this is enough to reasonably characterize the edge and taking more only slows the process down without increasing the accuracy significantly. This process assumes that the data set is indeed that of a standard screwdriver head and that the head was fixed relative to the IFM, at an angle of 45° to the detector. To avoid having noise along the boundaries of the data set mistaken as the tip edge, any points within 10 pixels of the edge were not recorded.

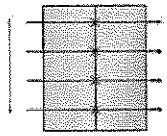


Figure 17: *Process of locating the edge points*

2. *Find the center of coordinates.* The points identified in the previous step are averaged to compute the geometric centroid of the edge, $\bar{v} = [\bar{x}, \bar{y}, \bar{z}]^T$. The translation matrix to this point, M_T, shown in below, is recorded for later.

$$M_T = \begin{bmatrix} 1 & 0 & 0 & -\bar{x} \\ 0 & 1 & 0 & -\bar{y} \\ 0 & 0 & 1 & -\bar{z} \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (1.3)$$

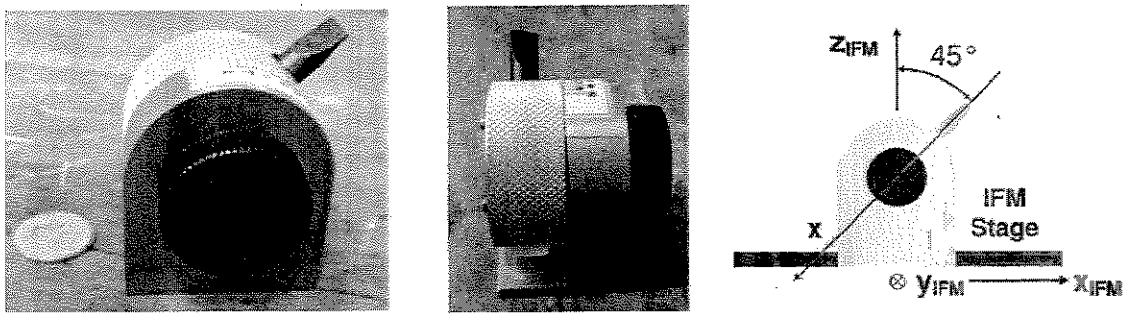
3. *Determine the direction of the y axis.* In this step, a line is fit to the previously identified edge points using linear regression. The result is a y axis vector, which is normalized to unit length and saved for later.
4. *Determine the direction of the x axis.* Figure 16 illustrates the derivation of the x axis direction. During scanning by the IFM, the tip is placed in a special measurement jig shown in Figure 16. This consists of an adjustable knob with a hole in it for the screwdriver tip shaft. The angle of rotation is labeled on the main body of the jig, and this body fits snugly against a ledge on the IFM stage during measurement. For all of the tip data, the tip angle used for measurement was 45°. As shown in Figure 18(c), the IFM defines its z axis as straight up into the detector; therefore, if the measurement jig is snug against the ledge, the IFM z axis will be coplanar with the screwdriver shaft. When this is satisfied (which it is for the tip scans used in this research), the x axis can be computed by rotating the vector $[0\ 0\ 1]^T$ through a 45° angle about the direction of the IFM y axis (that is, $[0\ 1\ 0]^T$) and then negating it.
5. *Determine the direction of the z axis.* The z vector is computed as the normalized cross product of the x axis and the y axis.
6. *Ensure orthonormality.* Because of the cross product, the z vector is guaranteed to be normal to the xy plane. However, the x and y vectors are not guaranteed to be normal to each other. In particular, the edge of the screwdriver may not be exactly perpendicular to the shaft due to wear and the limitations on the accuracy of the manufacturing process. In this case, it is assumed that the screwdriver marking motion will be more constrained by the edge of the screwdriver dragging along the plate than the direction of the shaft in the person's hand. Therefore, the x axis direction is corrected to be normal to the y axis direction by re-computing it as the normalized cross product of the y axis and the z axis.
7. *Compute the basis change matrix.* The basis change matrix M_B is constructed as shown in Eq. (1.4). Here, $[x_x x_y x_z]^T$, $[y_x y_y y_z]^T$, and $[z_x z_y z_z]^T$ are the x, y, and z axis vectors computed above, respectively.

$$M_B = \begin{bmatrix} x_x & y_x & z_x & 0 \\ x_y & y_y & z_y & 0 \\ x_z & y_z & z_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (1.4)$$

8. *Compute the coordinate system matrix.* The coordinate system M_C is computed according to the following equation

$$M_C = (M_B)^{-1} M_T. \quad (1.5)$$

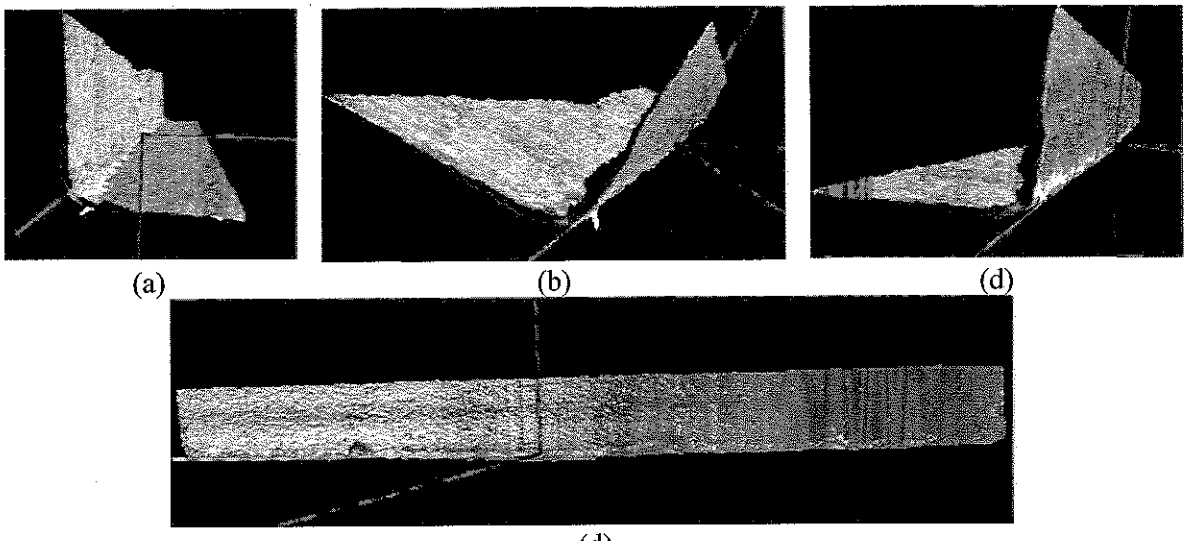
Note that M_B must be inverted so that M_C converts coordinates from the IFM coordinate system to the tip coordinate system rather than the reverse.



(a) (b) (c)

Figure 18: *x axis determination. (a-b) The jig used for screwdriver tip measurement in the IFM. (c) Diagram of the jig during IFM measurement*

Therefore, M_C as computed using the algorithm above is a linear transformation from the original IFM coordinate system used in the data file to the new tip marking coordinate system. This is utilized by directing OpenGL to always multiply the tip data first by M_C. The resulting coordinate system applied to actual representative data is shown in Figure 19. The light-colored solid lines denote the tip coordinate system, and the darker dashed lines denote a local world coordinate system. For both of these, red denotes the x axis, green denotes the y axis, and blue denotes the z axis. Overall, the coordinate system looks correct.



(a) (b) (d)

(d)

Figure 19: *The coordinate system transformation applied to the data. (a-c) The side view with y axis rotations of (a) 0°, (b) 60°, and (c) 85°. (d) View of the flat side of the screwdriver with a rotation of 0°*

II.5: Creation of virtual depth camera

The ultimate goal of this project is to utilize the cleaned data to characterize the screwdriver tips for forensic applications where the toolmarks need to be generated at any arbitrary angle and at any arbitrary force. Since the existing statistical software algorithms developed at Iowa State University/Ames Laboratory [Chumbley et al., 2010] only takes in 1D trace data with equal distances between sampling points, this usually requires a very time-consuming re-sampling procedure should a conventional geometry analysis technique is adopted due to the large data size (typically 9k x 1k points per high-resolution scan). Moreover, the problem becomes more complex if the originally scanned surface is manipulated since the (x, y) coordinates of surface

points may become irregular after manipulation. Therefore, a new efficient tool is needed for converting 3D geometry to surface height image for toolmark trace data generation.

In this research, we developed a *virtual depth camera* with the advanced computer graphics tools to address this challenge. In this technique, the OpenGL rendering pipeline was utilized to re-sample the data rapidly and precisely into regular grids. In computer graphics, to render 3D geometry into 2D images on a computer screen, the computer graphics rendering pipeline provides a means to sample 3D geometry into 2D grid points. Moreover, the advanced computer graphics tools also provide a way to obtain the surface height (z) for each sampled pixel. Therefore, if the OpenGL pipeline is properly setup, 3D geometry can be re-sampled uniformly as depth map, which is desirable for further data processing, and also useful for 3D geometry data storage efficiently [Zhang, 2012; Zhang et al., 2012].

Figure 20 illustrates a typical 3D computer graphics rendering pipeline (CGRP) [Shreiner, 2010]. The rendering process starts with geometry processing that culls the back faces (e.g., remove those vertices that face away from the viewer); The next step is to project 3D coordinates of each vertex onto 2D image plane by applying the projection matrix; Then, the polygons are filled in through 2D interpolation using the projected vertices, and hidden points are removed using depth buffering. This step is called the rasterization; finally, the 2D images are displayed on 2D computer screen pixel by pixel.

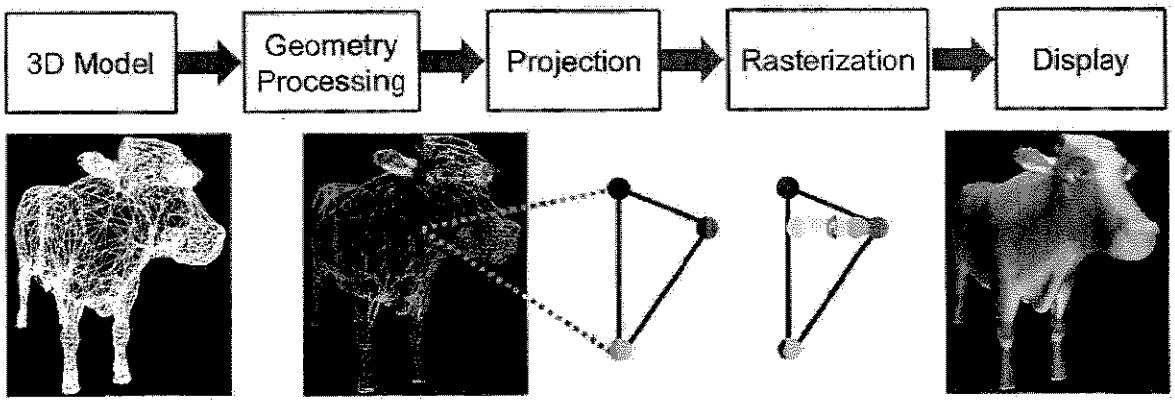


Figure 20: *Computer graphics rendering pipeline (CGRP)*

If the projection is chosen as orthographical, the screen coordinates (i, j) are naturally proportional to the original (x, y) in a 3D object space because most computer screens contain squared pixels. This means that the CGRP provides a means to sample 3D shape uniformly along x and y directions. Since the advanced computer graphics tools can perform high-resolution, real-time 3D rendering, it thus also provides a very efficient way for this procedure. Moreover, the advanced computer graphics rending techniques provide a mean called*render to texture* to retrieve depth (z) per pixel. By rendering the scene to texture instead of computer screen, the depth z can be recovered pixel by pixel through unprojection. This research is to use this technique for re-sample 3D data at any arbitrary orientation accurately.

To verify the accuracy of the proposed re-sampling framework, we firstly tested an ideal unit sphere generated, and re-sampled with the proposed pipeline using an image resolution of 512 x 512. Figure 21(a) shows the 3D plot of the re-sampled data. One cross section of the re-sampled 3D sphere and the ideal one were plotted in Figure 21(b): they are almost identical. To better visualize the difference between these two, Figure 21(c) plots the difference. One may notice

that the difference becomes significantly larger on both ends of the plot. This is due to sampling of the system: it is well know that when the sampling direction is approximately parallel to the surface tangent plane of an object, the sampling error becomes extremely large. Even with those sampling error, we found that the difference between the re-sampled one and the ideal one is very small: the root-mean-square (rms) error is approximately 0.006%.

Sphere is a representative 3D shape with smooth features, the re-sampling technique works quite well. To further demonstrate the capability of the proposed technique to sample sharp features: mathematically extreme points on 3D surfaces. We tested a trapezoidal 3D shape and a pyramid, as shown in Figs. 21(d) and (g). Figures 21(f) and (i) show that cross sections of the difference between re-sampled ones and the ideal ones. As expected, the extreme points result in significantly larger error, with trapezoids having smaller error because the surface change is not as sharply as the pyramid. Nevertheless, the rms errors for both cases are still very small (less than 0.1%).

Screwdriver data were also resampled with the depth camera. The cleaned screwdriver tip shown in Figure 8 was used for this test. Figure 22(a) shows the re-sampled result with its natural scanned orientation; and Figure 22(b) graphically shows the re-sampled result and the original 3D shape. In this image, the golden color represents the original 3D data, while the green color represents the re-sampled result. This experimental data shows that they are well aligned before and after re-sampling.

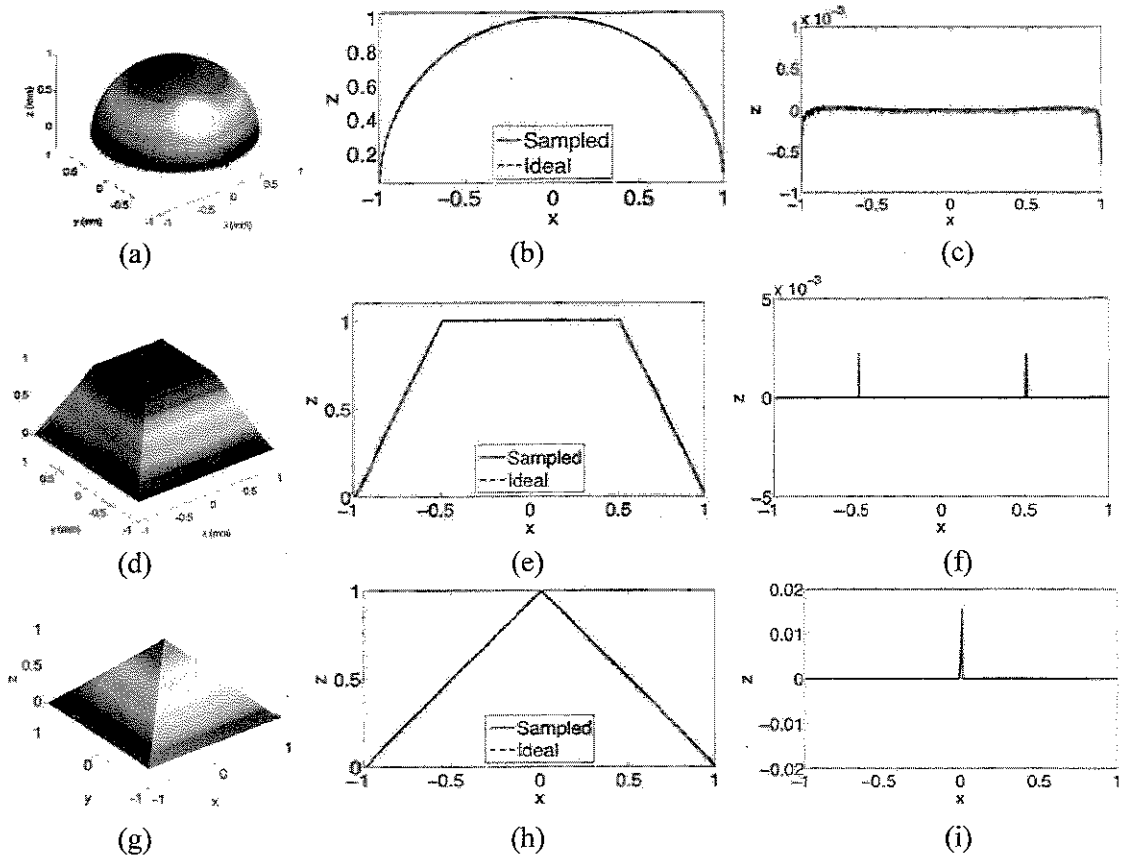


Figure 21: *Re-sampling results of ideal 3D objects. (a) Resampled 3D unit sphere; (b) Cross sections of the ideal sphere and the re-sampled one; (c) Difference of the cross sections shown in*

(b) with rms error of 0.006%; (d) Resampled 3D trapezoid; (e) Cross sections of the ideal trapezoid and the re-sampled one; (f) Difference of the cross sections shown in (e) with rms error of 0.018%; (f) Resampled 3D pyramid; (g) Cross sections of the ideal pyramid and the re-sampled one; (h) Difference of the cross sections shown in (h) with rms error of 0.095%

To further verify the flexibility of the proposed technique, the tip was rotated 15 degrees about its edge (or y axis), and then re-sampled by using the proposed technique. Figures 22(c) and (d) show the results. It once again shows that they are well re-sampled.

II.6: Creation of "virtual tool marks" on GPU

The basic principle behind this tool mark simulation is to take the projection of the tip geometry in the direction of tool travel and identify the highest points on that projection. The highest points will scrape the deepest into the plate material, so they are responsible for leaving the observed striae. Figure 23gives an illustration of this idea. In this diagram, the screwdriver is held at an angle of about 45° with respect to the plate surface (a y rotation) and twisted about its shaft (x axis) by some angle β. The inset shows the measured tip geometry, resembling an extruded chevron, making the mark. The black wavy line down the middle represents the tip geometry closest to the plate. These points dig into the plate material, imparting the striae. Because the tip is at an angle β to the direction of motion, the depth profile of the mark is the projection of the black wavy line onto a line at the angle β. From the diagram, we see that we could have found this mark profile by squishing the screwdriver tip onto a plane perpendicular to the direction of tool travel. The lowest points on this squished tip would be the ones that generate the mark and the same ones participating in the black wavy line. This type of collapsing process will be used to make virtual mark profiles.

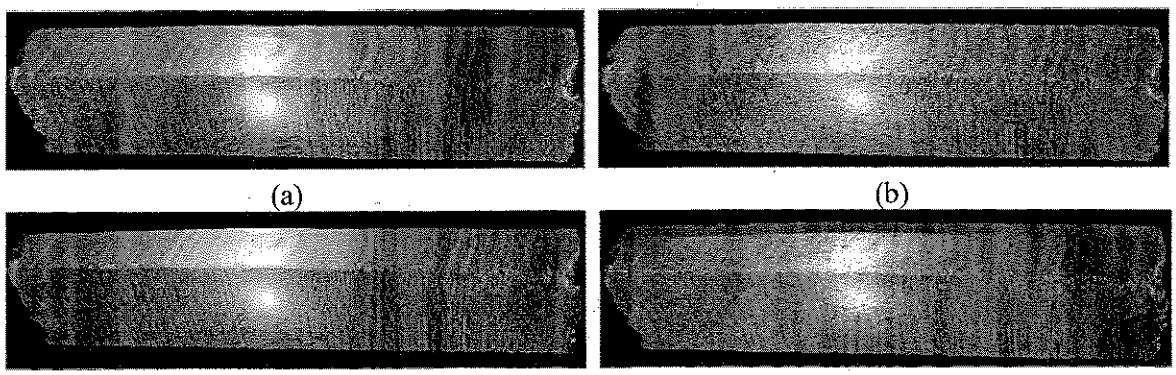


Figure 22: *Arbitrary orientation re-sampling of real scanned 3D range data. (a) 3D result of re-sampled data at natural orientation (the same as scanner); (b) Overlapping the original 3D data with the re-sampled one shown in (a); (c) 3D re-sampled result after rotating the tip by 15-degrees around its edge; (d) Overlapping the original 3D data with the re-sampled one shown in (c)*

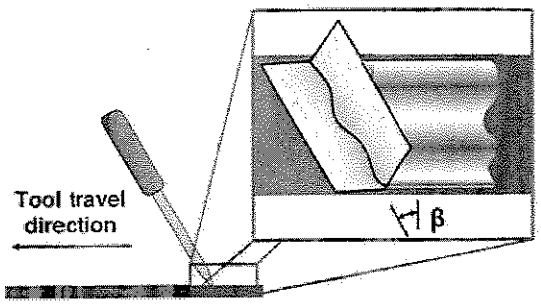


Figure 23: *The mark as a projection of tool tip geometry in the direction of tool travel*

Clearly, this approach is quite simple and ignores several complexities of mark making including material properties of the tool and plate, forces, deformations, and the possibility of partial markings. These phenomena can be very difficult to control and account for in a simulation. Since this is a first approximation, the assumption of complete geometry transfer from the tip to the plate is made. Hence, the effects of the specific forces and deformations are treated as insignificant since the geometry seems likely to be the strongest signal in the mark.

Although this method is quite simple in principle, the scale of the geometry simulation required makes it complex to carry out on the computer. Each tip data set at the 10x magnification level used in this research contains more than 9000×1000 = 9 million points. Projecting and determining the edge of such a large amount of data on the CPU would take a long time. Therefore, the Open Graphics Library (OpenGL) and its OpenGL Shading Language (GLSL) are used to build the simulation in order to take advantage of the parallel capabilities of the computer's Graphics Processing Unit (GPU). Dividing the geometric computations among the shaders, a computer program running on GPU to do postprocessing, drastically reduces the computation time required and makes virtual mark generation feasible.

The following basic process was used to implement this simulation in OpenGL:

1. A rotation matrix M_R is formed to describe the desired angular position of the tip relative to the local world coordinate system. The user specifies a set of three angles (α, β, γ) in degrees which are the desired rotations about the x, y, and z axes, respectively. Equation (1.6) is used to compute M_R. Note that since the rightmost matrix is the x rotation, it will get applied first to each vertex. The y rotation will be applied next, and the z rotation will be applied last.

$$M_R = R_z R_y R_x$$

$$= \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 & 0 \\ \sin\gamma & \cos\gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & -\sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ \sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.6)$$

2. The tip's bounding box is rotated into the desired position using M_R. The "squish" matrix, M_S in Equation (1.7) below, is then applied to the bounding box to project it onto the plane normal to the local world x axis. This matrix sets the x coordinates of each vertex to zero. The eight points of the bounding box are then iterated over to find the maximum and minimum y values taken up by the squished bounding box. This is a quick way to estimate the height of the tip projection in the scene, which will tell us how tall to make the scene window and the depth camera.

$$M_T = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.7}$$

3. A one-dimensional depth camera is created, which we will use to find maximum points along the flattened edge of the screwdriver tip as illustrated in Figure 24. The height of this camera is carefully chosen to achieve the desired sampling resolution for the edge.

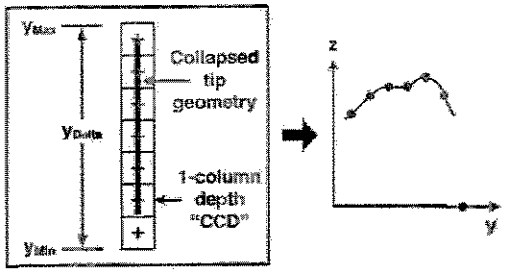


Figure 24: *The depth camera concept*

4. OpenGL is directed to apply the coordinate system matrix M_C, M_R, and M_S, in that order, to the tip data when it is drawn. The camera is also positioned at 5000 μm from the center of coordinates so the whole tip can be imaged, and it is set up to use orthographic projection (meaning that it becomes a perfect non-pinhole camera that never views things with the distortion of perspective)
5. The tip geometry is drawn in the scene.
6. The sampled tip geometry is pulled out of the camera. It is converted from OpenGL camera coordinates back to μm and rotated 180° to become an impression. The rotation is performed by multiplying the mark by -1 and then flipping it from left to right.
7. An edge detection algorithm is used to clip off the parts of the mark due to the sides of the screwdriver tip. This is performed since the sides of the screwdriver can sometimes confuse the statistical comparison algorithm. The sides are not in general part of the striae used for matching.

II.6.1: GPU implementation of virtual depth camera

Using techniques discussed in Section III.2, a special virtual camera is created that looks straight down upon the flattened edge of the screwdriver tip data. This camera records the distance to the points that it sees rather than their color; it is a depth camera formed from an OpenGL object known as a depth buffer. In depth test mode, OpenGL uses an array called the depth buffer to keep track of whether or not a certain point in a graphics scene is obscured by another one. In order to get drawn by OpenGL, each pixel-sized element sampled from the scene (referred to as a fragment) must survive the "depth test." OpenGL starts with the depth buffer initialized to the back limit of the camera's view (the far clip plane). As it receives the fragments, it records their distance from the camera in the depth buffer. If a new fragment is received that belongs in the same pixel as an older one, it checks the depth of the newcomer. If the depth value is less than that of the old one (meaning that the new fragment is closer to the camera), that new fragment replaces the old one. Its depth value is written into the depth buffer for the next round of the depth test. There are several ways to direct OpenGL to return this depth buffer instead of a color image. Therefore, it can become a depth camera, returning the depth values of those points closest to it.

Since the data samples needed are one-dimensional, the depth camera is created as single column. A two-dimensional depth camera would simply record extra scene background points that would have to be filtered out. The number of pixels h needed to sample the tip geometry at the desired resolution δ is computed according to Equation (1.8), where y_δ is the height computed from the bounding box in Step 2 of the overall procedure above.

$$h = floor\left(\frac{y_\delta}{\delta}\right) + 1. \tag{1.8}$$

Equation (1.9) readjusts y_δ so that δ is assured to be the desired value.

$$y_\delta = h\delta. \tag{1.9}$$

To create this special one-dimensional depth buffer of the correct size without interfering with the on-screen window, an OpenGL frame buffer object (FBO) is used. A frame buffer object is like a bundle to collect various images for OpenGL to draw into instead of the standard window. The FBO in this research contained only one image: the 1 × h depth buffer for the depth camera. Originally, the depth buffer was created as a renderbuffer object; however, ATI graphics cards would not accept an FBO with only a renderbuffer in it. Therefore, the depth camera had to be a 2D texture object.

The maximum allowable height of the depth camera texture in the FBO depends upon the capability of the computer's GPU. For a 10x resolution data set, δ = 0.803568 μm, which causes h to be about 8000 when the tip is at the angles (0, 0, 0). Many typical graphics cards cannot handle a texture this tall. Therefore, the flattened screwdriver scene is automatically divided into equal vertical partitions as needed to satisfy the demands of the graphics card. The depth camera then takes an image for each one of these partitions, and these images are later stitched together into a whole mark. Equation 3.8 is the equation used to compute the height of the depth buffer for the case of multiple partitions, where n is the number of partitions and y_{max} and y_{min} are the extreme values of the collapsed bounding box in Step 2. The y_δ parameter is still computed with Equation (1.10), but instead of being equivalent to $y_{max} - y_{min}$ as shown in Figure 24, it is now only equal to the height of the depth camera (which only covers afraction of the mark) in μm.

$$h = floor\left(\frac{y_{max} - y_{min}}{\delta n}\right) + 1. \tag{1.10}$$

Finally, although the FBO with the depth texture is the major component of the depth camera, the OpenGL viewport and projection matrix need to be correctly set. The viewport (the size of scene on the computer screen) is set to 1 × h. The orthographic projection matrix is set such that the visualized part of the scene is y_δ μm tall. In the first scene partition, the lower clip plane is set to y_{min} and the upper clip plane to $y_{min}+ y_\delta$. For subsequent partitions, y_δ is added to these clip planes to advance the part of the scene that the depth camera captures.

II.6.2: Drawing the tip

As mentioned above, the average digitized tip in this research contains slightly more than 9000×1000 = 9 million points. For OpenGL to draw these points, it must transfer at least threefloats (floating point numbers) representing x, y, and z to the graphics card for each vertex.This amount of information takes up

$$\left(4\frac{bytes}{float}\right)\left(3\frac{floats}{vertex}\right)\left(9\times10^6\ vertices\right)\left(\frac{1kB}{1024\,bytes}\right)\left(\frac{1MB}{1024\,kB}\right);\ 100\ MB$$

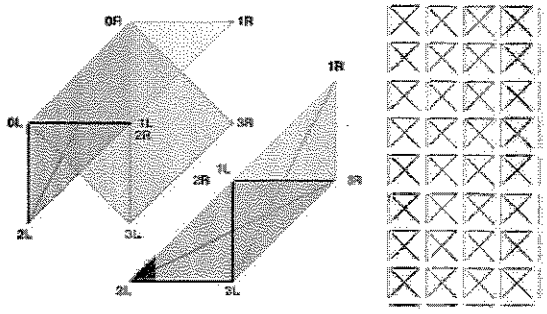
of memory. There is a limit on the rate of transfer from the CPU memory to the GPU, so in the ideal case, we would store this vertex data in a vertex buffer object (VBO) on the GPU to avoid transferring it each time the mark is drawn. However, the GPU only possesses a certain amount of RAM. While higher-end GPUs have gigabytes of RAM, many existing consumer- level GPUs do not. Therefore, we decided to conservatively limit the video RAM usage to 128 MB, a minimum system recommendation for modern video games. With approximately 60 MB of the video RAM already devoted to showing down-sampled 3D representations of the tip and mark in the user interface, the mark vertex data needed to be streamed to the graphics card rather than stored.

At first, the OpenGL glVertex command was used for streaming. However, most implementations of OpenGL could not handle this much streaming data and reserved several gigabytes of CPU RAM as a buffer for the data during streaming. This inhibited operations of other programs during mark generation. The mark-making process also ran slowly. Therefore, we created a special object called a StreamBuffer to stream the vertices to the GPU instead of OpenGL. Therefore, we created a special object called a StreamBuffer to stream the vertices to the GPU instead of OpenGL. The StreamBuffer allocates a pointer to 30 MB of CPU RAM and creates a 30 MB VBO on the GPU. The mark generation program hands its vertices to the StreamBuffer object, which stores them in the CPU pointer. When the CPU pointer becomes full, the StreamBuffer copies them to the VBO and instructs OpenGL to draw them. The data is initially stored in the CPU memory as a backup for the video RAM; the process of editing video RAM can sometimes fail and need to be repeated. A dynamically allocated CPU array was used to avoid the function overhead associated with vector objects. For normal operations, this function overhead is not noticeable, but for the intense streaming operation, it slowed mark generation by several seconds overall on a desktop PC.

Moreover, in order to generate the virtual mark at the desired resolution δ, we will need to interpolate between the data points. OpenGL will automatically perform this interpolation if the data is appropriately formatted in a triangle mesh. Two problems arise from this requirement for a triangle mesh. First, we can make a simple meshing algorithm to determine the proper relationship between the vertices. If each vertex is assigned a unique number, then we can store the results of meshing as a vector of integers and re-use the solution each time the tip is drawn. However, each vertex number needs to be a full-sized integer. A short integer can only hold at most the value 65535, not enough to count all 9 million vertices. Therefore, assuming that each vertex participates in only three triangles, the required index array would take up roughly 100 MB. Again, this is too much to store on the GPU. Since meshing is a relatively quick operation while transferring data is relatively slow, the mark generator simply re-meshes the data each time.

Another bigger problem arises from the way in which OpenGL samples the geometry for the depth camera. This sampling process is known as rasterization. In the interpolated mode, OpenGL draws only those fragments which lie inside of the established triangles in the mesh [Segal and Akeley, 2006]. Since the meshed geometry is collapsed into an infinitely thin line using the squish matrix, none of the points reside inside of a triangle. Therefore, the depth camera cannot "see" these points and record them. Therefore, the projected edge needs to be extruded to a finite thickness in the camera's field of view. To do this, we adopt a meshing algorithm that meshes between two instances of the tip geometry as shown in Figure 25. For each rectangle of four points in the tip data (labeled 0, 1, 2, and 3 in the diagram), the algorithm forms triangles on six surfaces between a left and a right copy of the points: bottom, top, left, right, top-left to bottom-right diagonal, and top-right to bottom-left diagonal. Triangles are not

formed out of any masked-off points. To save processing and drawing time, the right and bottom surfaces are only formed at the right and bottom borders of the tip data, respectively, to avoid drawing redundant surfaces. As a valid triangle is formed, its vertices are sent to the GPU with w, the fourth vertex coordinate which is typically equal to 1, set to one of two pre-determined values indicating whether the vertex belongs to the left or right data instance.



(a) (b)

Figure 25: *Meshing between a left and right copy of the vertex data. (a) Surfaces formed. (b) Meshing pattern that avoids redundant surfaces.*

At the GPU, we process each vertex with a GLSL vertex shader. This shader retrieves the left/right indicator in the w coordinate and temporarily stores it. It sets the w coordinate back to 1 and applies the model view matrix stack from OpenGL (the coordinate system, the desired rotations, and the squish operation). Then, it pulls the two squished instances of data apart by setting the vertex's x coordinate to the left/right indicator value. For the current software version, these indicators are -2 and 2. Finally, the shader applies the projection matrix to the data.

The net effect of this shader program is something analogous to a game of cat's cradle. The shader rotates and then squishes two identical copies of the tip that exist in the exact same space. Then it moves each copy to an opposite side of the screen. When this happens, the triangles formed between the two data sets "stretch" to form an extrusion of the projected tip. The depth camera can then image the edge of the tip projection, taking samples in a vertical line from the center of these triangles.

II.6.3: Converting from projected coordinates back to μm

Figure 26 presents a mathematical model of the graphics pipeline. At the far right, a generic vertex enters the pipeline; at the far left, it has been transformed into x and y window coordinates wx and wy and a value of depth for the depth buffer z_{db}. Throughout the bracketed section, the data remains in μm. The projection matrix normalizes the data such that x, y, and z all range between -1 and 1. The glViewport command, provided that the window positioning arguments are both 0 as they are in this research, behaves as a pair of matrices that normalizes x to range between 0 and the window width, y to range between 0 and the window height, and z to range between 0 and 1.

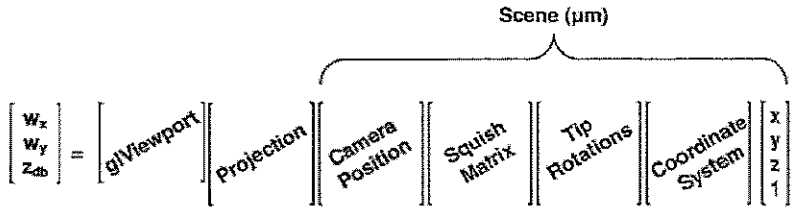


Figure 26: *Mathematical model of the graphics pipeline for this research*

Therefore, to convert the depth buffer values back into μm values for the virtual mark, we must undo the action of the glViewport command and the projection matrix. If the z coordinate after the projection matrix is designated z_{ndc}, then Equation (1.11) converts the depth buffer value back into projected (or normalized device) coordinates.

$$z_{ndc} = 2z_{db} - 1. \tag{1.11}$$

The virtual mark software uses an orthographic projection matrix, which has the form

$$M_p = \begin{bmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & -\frac{2}{f-n} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{1.12}$$

where l, r, b, t, n, and f are the locations of the left, right, bottom, top, near, and far clip planes, respectively. Therefore, the virtual mark data can be unprojected using Equation (1.13) to yield a mark in μm.

$$z_{\max} = -\frac{1}{2}\left[(f-n)z_{ndc} + f + n\right]. \tag{1.13}$$

Combining Equations (1.11) and (1.12) yields a simple conversion (Equation (1.13)) from depth buffer values to mark values in μm.

$$z_{\max} = z_{db}(n-f) - n. \tag{1.14}$$

II.6.4: Detecting the edges

The edge detection algorithm was developed through trial and error by analyzing mark data with MATLAB. The finalized C++ version follows these steps:

1. The derivative is approximated using partial differences. The resulting vector has elements $z_i - z_{i-1}$ and is one member shorter than the original mark data because it starts with $z_1 - z_0$.
2. The derivative vector is filtered with a size 15 Gaussian 1D filter to remove the effects of noise.
3. The mean and standard deviation of the filtered derivative are computed.
4. The algorithm begins looking in from both ends of the mark for a new start and end point with these criteria: the new points must be at least 20 points in from either edge of the mark and must be the first points beyond this border to satisfy $|z'(i) - \mu_{z'}| \leq \sigma_{z'}$. In the above condition, $z'(i)$ is the derivative at the i-th point and $\mu_{z'}$ and $\sigma_{z'}$ are the mean and standard deviation of the derivative, respectively. If the search fails, the original start and end points are used.

5. If the search is successful, the mean and standard deviation are recomputed for the portion of the derivative within the new start and end points. Step 4 is then repeated once using this updated mean and standard deviation along with the original, full-sized derivative vector.
6. The final values of the start and end index are returned as a suggestion to the user for later trimming.

Figure 27 shows a representative result from this trimming algorithm. The red line is the complete virtual mark retrieved at a y-rotation of 60°. The left and right sides of the purple box indicate the trimming suggestion from the edge detection algorithm, and the trim controls boxes show the corresponding values of the new start and end indices. This trimming seems very reasonable.

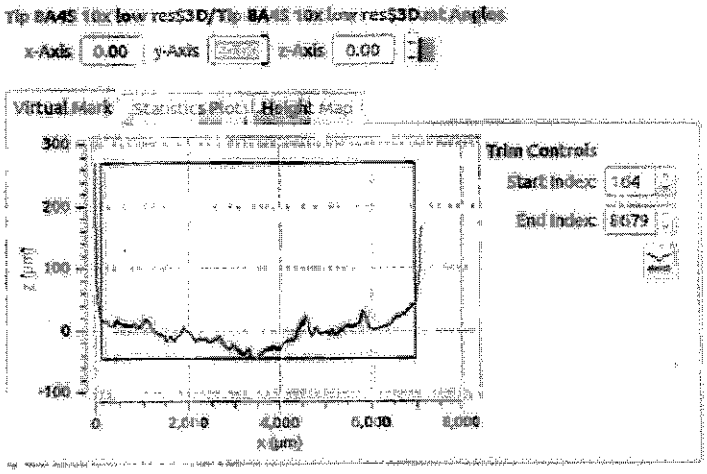


Figure 27: Trimmed mark

II.7: Statistics

We employed the statistical algorithm that was developed previously by our team to validate the virtual tool mark characterization technique [Chumbley et al., 2010]. The attainment depends on the ability to provide some measure of how well the generated VM duplicates an actual mark produced under a controlled set of conditions. The data examined in this analysis are two types of trace data: the virtual mark (VM) and the real mark (RM). The RM data are the type of data that are collected by an optical profilometer that records surface height (z) as a function of discrete (x, y) coordinates, and further takes a linear trace perpendicular to the striations present in a typical tool mark. The VM trace data are the type of data that are created virtually by the VT at a specified condition following the aforementioned procedures. The virtual mark is also a trace data that record mark depth z as a function of discrete x position. Important assumptions in the analysis are that the values of z are reported at equal increments of distance along the trace and that the traces are taken as nearly perpendicular to the striations as possible. The algorithm then allows comparison of two such linear traces. This section details the algorithm we employed.

A small experimental jig for making physical tool marks was designed and constructed where angle, pressure, and twist of the tool tip can be rigidly controlled. This part of the project will be done in consultation with certified tool mark examiner Jim Kreiser, who has worked with us on previous studies [Faden et al. 2007;Chumbley et al., 2010]. Mr. Kreiser oversaw the production both of samples using the jig. The actual comparisons employed a variant of an algorithm developed for the comparison of two tool marks. This algorithm is described fully in [Chumbley

et al., 2010], so it will not be described in detail here. Briefly, the algorithm determines matching along one-dimensional profilometer data traces (z vs. x) where the values of z are reported at equal increments of distance along the trace and the traces are taken as nearly perpendicular to the striations as possible. Such will be the case for the data available from the VT and VM of this study.

The algorithm first goes through an Optimization step to identify a region of best agreement in each of the two data sets, as shown in Figure 28. The R-value of this region is determined [Faden et al. 2007]. The algorithm then conducts a second step in the comparison process called Validation, where corresponding windows of equal size are selected at randomly chosen, but common distances from the previously identified regions of best fit. The assumption behind the Validation step is that if a match truly does exist, correlations between these shifted window pairs will also be reasonably large because they will correspond to common sections of the tool surface. In other words, if a match exists at one point along the scan length (high R-value), there should be fairly large correlations between corresponding pairs of windows along their entire length. However, if a high R-value is found between the comparison windows of two nonmatch samples simply by accident, there is no reason to believe that the accidental match will hold up at other points along the scan length. In this case rigid-shift pairs of windows will likely not result in especially large correlation values.

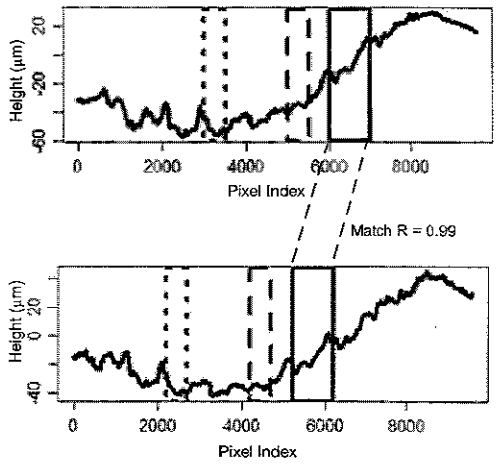


Figure 28: *Comparison of two marks showing best fit (solid rectangle) and Validation windows (dotted rectangles).*

The correlation values computed from these segment-pairs can be judged to be "large" or "small" only if a baseline can be established for each of the sample comparisons. This is achieved by identifying a second set of paired windows (i.e. data segments), again randomly selected along the length of each trace, but in this case, without the constraint that they represent equal rigid-shifts from their respective regions of best fit, Figure 29(b).

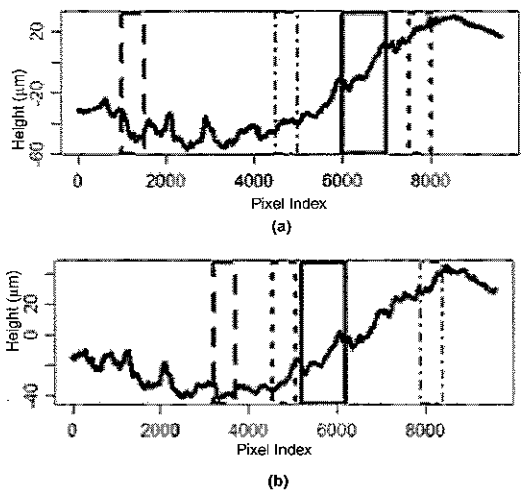


Figure 29: *(a) Comparison pair showing a true match. Best region of fit shown in solid rectangle. Note the similarity of the regions within the two possible sets of validation windows (dashed and dotted rectangles). (b) Validation windows (dashed, dotted, and dot-and-dash rectangles) selected at random for the comparison pair shown in (a) to establish a baseline value*

The Validation step concludes with a comparison of the two sets of correlation values just described, one set from windows of common random rigid-shifts from their respective regions of best agreement, and one set from the independently selected windows. If the assumption of similarity between corresponding points for a match is true, the correlation values of the first set of windows should tend to be larger than those in the second. In other words, the rigid-shift window pairs should result in higher correlation values than the independently selected, totally random pairs. In the case of a nonmatch, since the identification of a region of best agreement is simply a random event and there truly is no similarity between corresponding points along the trace, the correlations in the two comparison sets should be very similar. Using this statistical algorithm we previously developed had shown that tool marks can be quantified and error rates established [Chumbley et al., 2010].

II.8: Software graphical user interface (GUI) design

Figure 30 presents the main window of the graphical user interface (GUI) for the virtual mark software. This GUI was designed using Qt. The window is divided into three widgets: tip (top), plate (middle), and statistical comparison (bottom). The tip and plate widgets feature 3D representations of the file geometry on the left side. For these views, the geometry is down-sampled by a factor of 6 to improve graphics speed and performance. Users can left click and drag on the geometry to translate it, and a Qt-provided trackball model allows users to intuitively rotate the geometry with right click and drag. The scroll wheel allows users to zoom in and out. Users can double-click on the plate view to interactively select a column of plate data for comparison. This selected column appears in the plate view as a red plane as shown in Figure 30. Users can view the geometry in one of four modes by clicking the buttons immediately to the left of the geometry views: shaded, wireframe, textured, and height- mapped. Textured mode overlays the 2D texture from the Alicona onto the 3D geometry. A fifth viewing mode is provided for the tip widget which shows the tip geometry projected in the direction of tool travel; this mode helps users understand the mark generation process.

The right sides of the tip and plate widgets provide plots for profile data. The plate widget provides the name of plate file and a box for changing the selected column. The tip widget

provides the name of the tip file and boxes for editing the desired tip rotation for mark generation. Users can click the adjacent button to create a virtual mark; when the mark is complete, users can click on the virtual mark tab to see the view in Figure 30. This trim view presents the recommended end points from the edge detection algorithm as the left and right sides of a purple box and allows users to interactively change these end points. Moreover, the trim view features a flip button that flips the virtual mark to compensate for a plate scanned backwards.

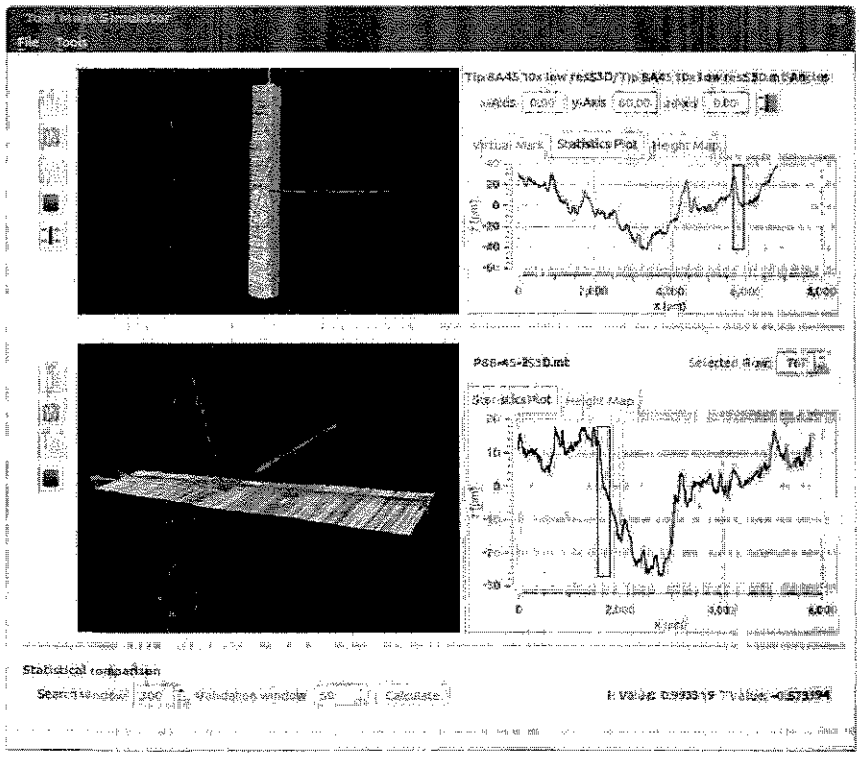


Figure 30: *The graphical user interface* (GUI)

For both the tip and plate widgets, the statistics plot tab provides a view of the trimmed and de-trended mark as shown in Figure 30. De-trending is the process of fitting a first-order line to the data with linear least squares and then subtracting this line from the data; it is an essential preparatory step for the statistical comparison. Once the user clicks the calculate button in the statistics widget, purple windows pop up on both statistics plots denoting the locations of the search windows.

Figure 31 presents the third type of plot view, the height map. This is an alternative view of the profile data that represents height as a color between black and white. The color bar on the right shows the scale of the data. This plot is designed to resemble the physical mark views on the 2D comparison microscopes used by practicing forensics examiners.

The statistical comparison widget provides an interface to the algorithm validated by Chumbley et al. (2010). Users can use the boxes to edit the size of the search window and the validation windows. The calculate button is enabled when there is both a virtual mark and a selected plate row to allow users to perform the comparison. The R and T values from the comparison appear on the right side of the window after the comparison. The R value represents the maximum correlation statistic, which corresponds to the goodness of match between the contents of the purple windows. The purple windows denote the regions of the two marks which match the best.

The T value represents the average of 200 comparisons between rigid- and random-shift windows. The T1 statistics that comprise this T value are computed by com- paring the correlation between windows a set distance from the purple maximum correlation windows to the correlation between randomly shifted windows.

Finally, the file menu allows users to open tip and mark files. Users can import the data from an Alicona .al3d file and save it after cleaning as a Qt-based .mt file for later use. The tools menu allows users to bring up the "Automator," a dialog that sets up the GUI to automatically perform comparisons between several plate files and a tip at multiple angles. This automated process saves the T values to a comma-separated list file for later viewing and takes a screenshot of each comparison.

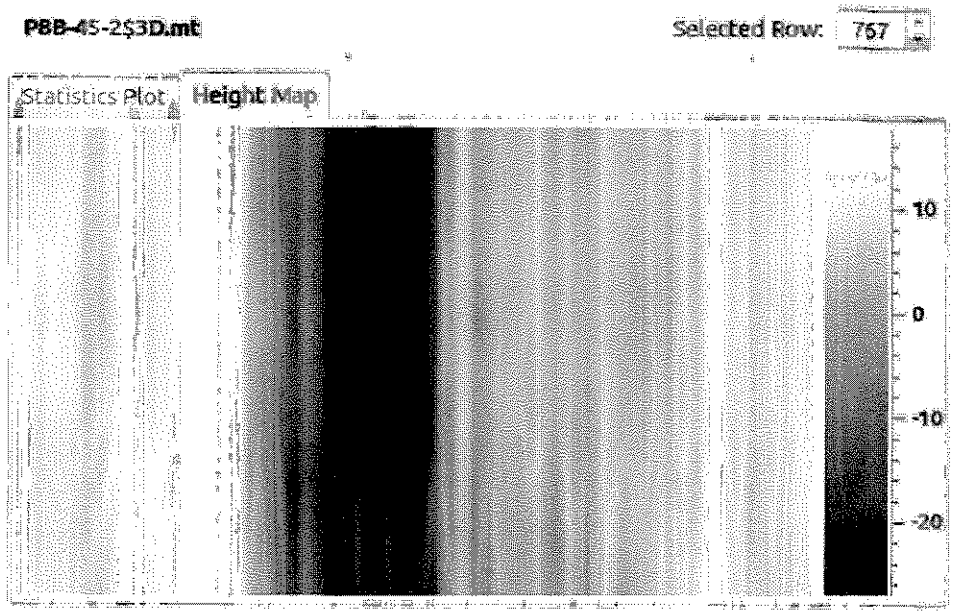


Figure 31: *Height map plot*

III. Results

This study was to test the following two hypotheses:

Hypothesis 1: *A 3D simulation tool can be generated and manipulated to create "virtual marks" that match with "real marks" made by the same tool under the same conditions (e.g. angle, twist, and force).*

Hypothesis 2: *The virtual tool mark generation technique can be used to effectively and objectively characterize the tool marks.*

The first and most fundamental assumption that all tool marks are unique had been supported by the prior study conducted by Chumbley et al. [2010]. This research was to use the virtual tool mark generation technique to further test this assumption by a comparison of real marks made by different screwdriver tips at the angles of 45, 60 and 85 degrees with respect to horizontal (6 x 2 x 3 = 36 real marks; two 45° marks were unavailable at the time of the study); and of virtual marks made by the virtual screwdriver tool at every 5 degrees starting from 30 degrees with respect to horizontal (30, 35, 40, ..., 85, 90 degrees; 6 x 2 x 13 = 156 virtual marks).

Figures 32, 33, and 34 present sample known-match comparisons. "Known match" for this study was defined as using the same tip side to generate both real and virtual marks. Figures 30, 35, and 36 present sample known non-match comparisons; known non-matches for this study included marks made by different screwdrivers and different sides of the same screwdrivers. Different sides of the same tip were included because the work of Chumbley et al. [2010] supports the conclusion that different sides have unique marks. The known non-matches for this study also included both non-matches and matches where the virtual mark was flipped relative to the scan direction of the real mark; these were included to ensure that the algorithm was not matching based upon identical class and/or subclass characteristics of the marks. In these figures, the red upper mark is a virtual mark, and the blue lower mark is a cross-section from a real mark.

The T1 statistic values from the known matches in the study are shown in Figure 37 as a function of angular comparison. The data is plotted as box plots, the boxes indicating where 50% of the data falls with the spread of the outlying 25% at each end of the distribution shown as dashed lines. As stated previously, when using a T1 statistic a value relatively close to 0 indicates that there is essentially no evidence in the data to support a relationship between markings. Figure 37 shows that for pairs made from the same screwdriver tip edge, the virtual and real marks produce the largest T1 values when they were made from an angle within 5 degrees. The T1 values drop to 0 close to zero if the angle made the virtual marks and the angle made the real mark are far apart. When the angles are at the matching angles, the T1 values are mostly larger than 2.5, except a few outliers with T1 value below 2.5 where the match cases become non-matches.

In comparison, for known non-matches (Figure 38), the majority of the index T1 values produced by the algorithm are near the 0 value (within ±2.5), indicating that there is essentially no evidence in the data to support a relationship between markings.

Our preliminary studies found that for non-match cases and T1 values less than -2.5, most of them have similar characteristics as shown in Figure 36. There was a large unusual structure at the end of the virtual mark, making the Optimization step result in an incorrect "best fit" location. We are still investigating the cause of these outliers and hopefully develop an automated routine to handle the issue like the one shown in Figure 36.

We also noticed that the higher the angle, the larger the variance of the T1 values for matches, whilst no obvious disparities for non-matches. This might be caused by the fact that it is more difficult to make the higher real marks by an examiner. Further study will also be conducted to determine the causes of the differences between high and low angle marks for matching cases.

Combining all preliminary studies together, two hypotheses were clearly supported, although there are minor issues worth further investigation.

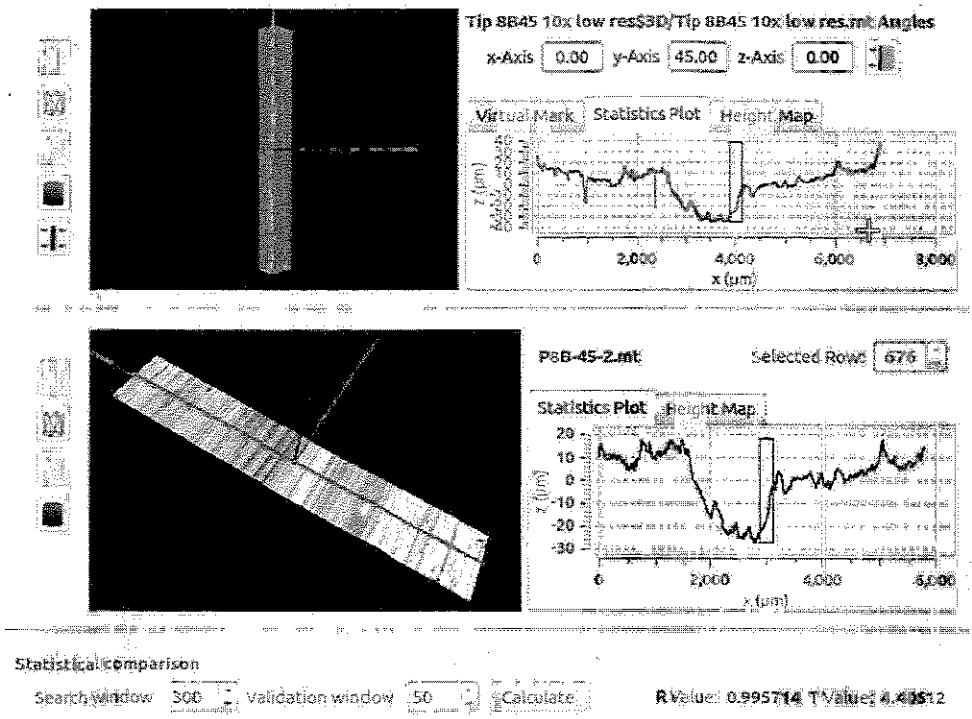


Figure 32: *A known-match comparison between Tip 8B at 45 degrees and Plate 8B-45.*

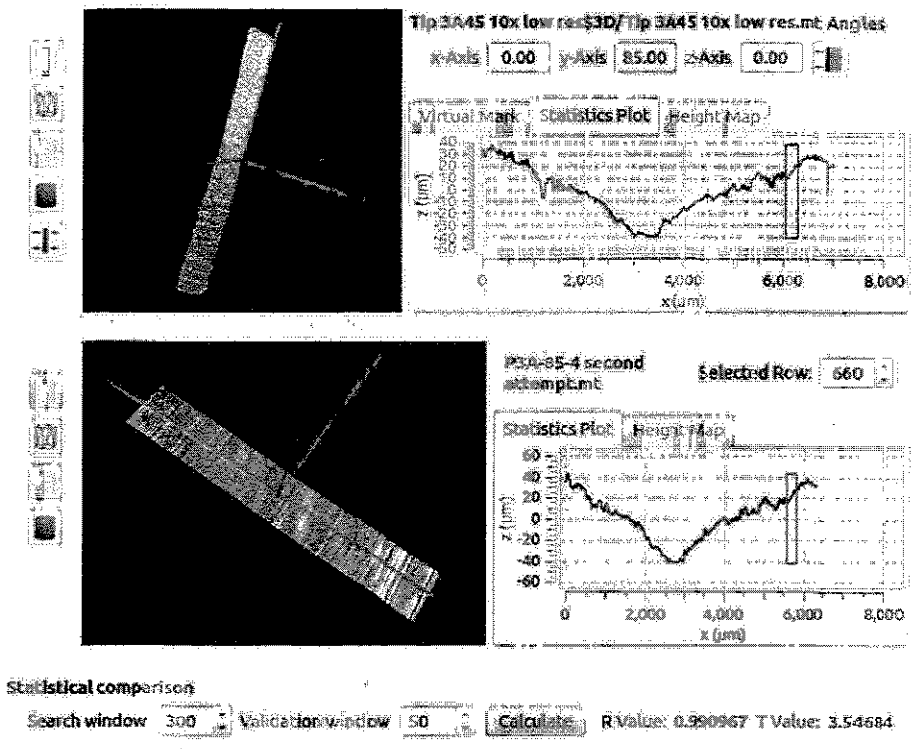


Figure 33: *A known-match comparison between Tip 3A at 85 degrees and Plate 3A-85.*

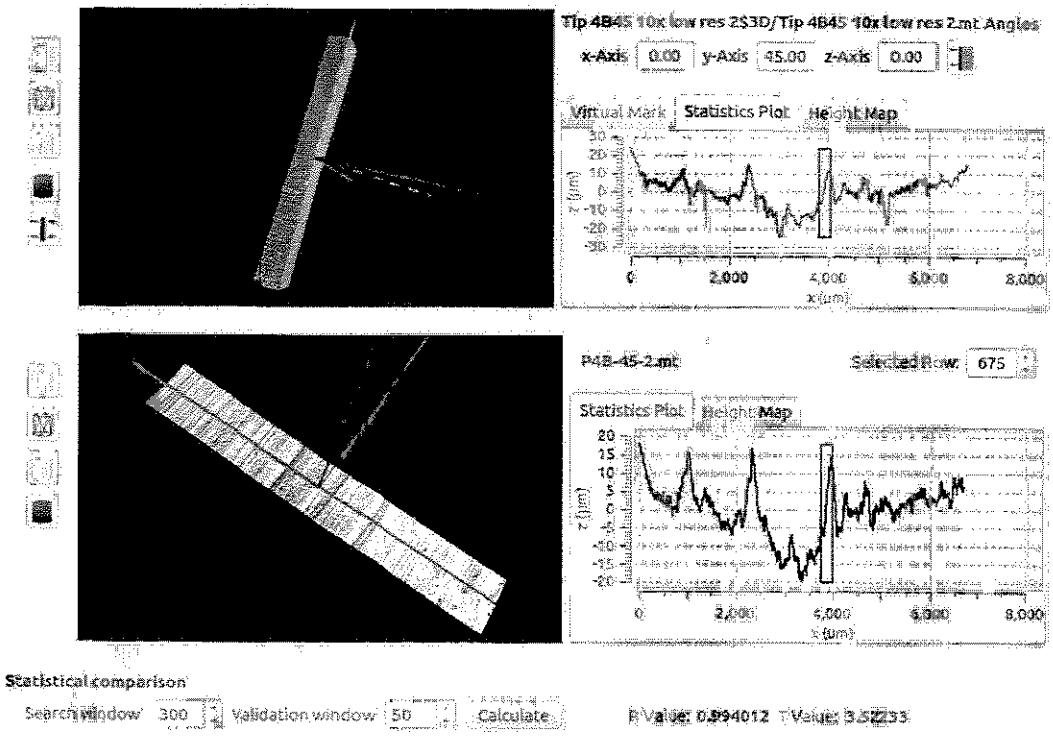


Figure 34: *A known-match comparison between Tip 4B at 45 degrees and Plate 4B-45.*

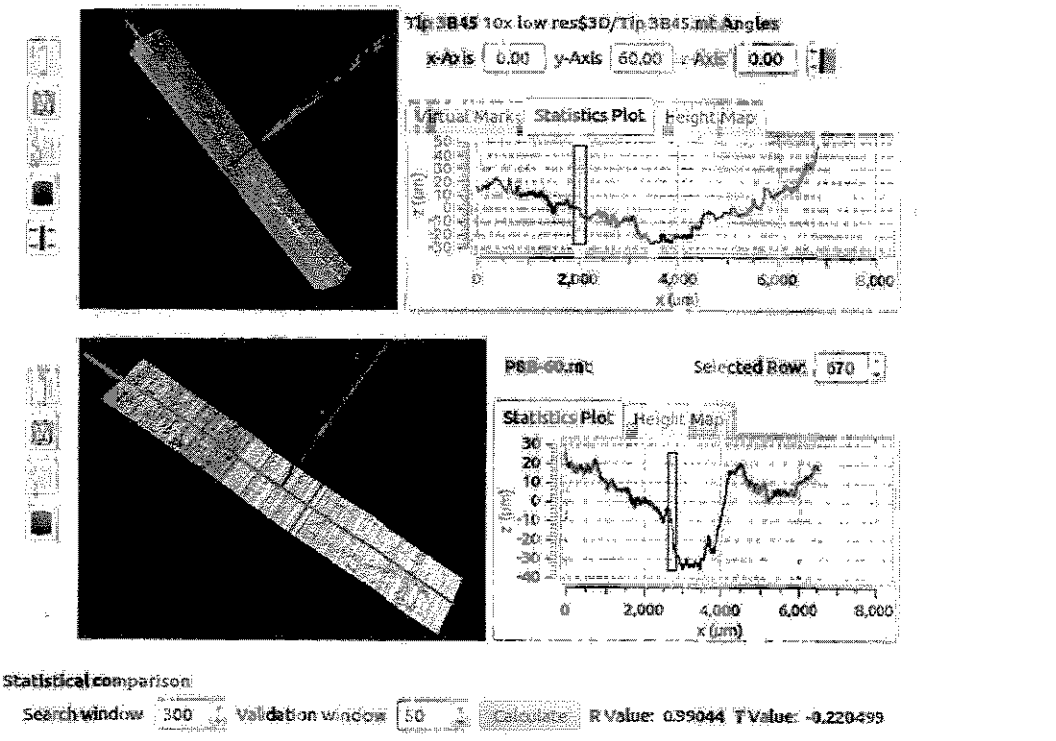


Figure 35: *A known non-match comparison between Tip 3B at 60 degrees and Plate 8B-60.*

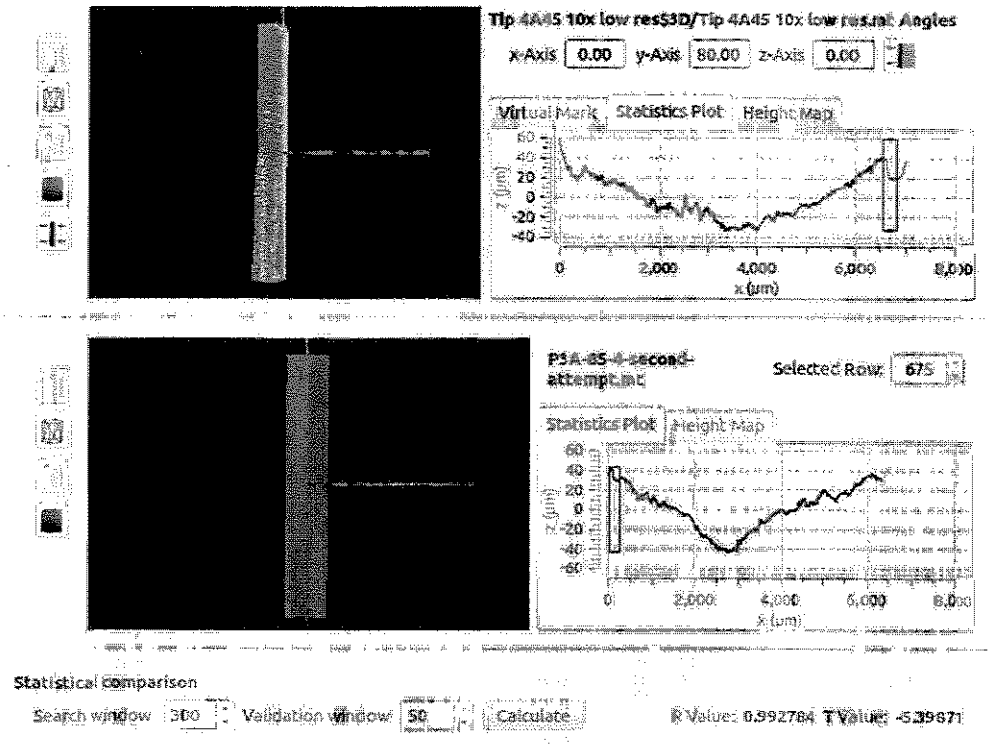
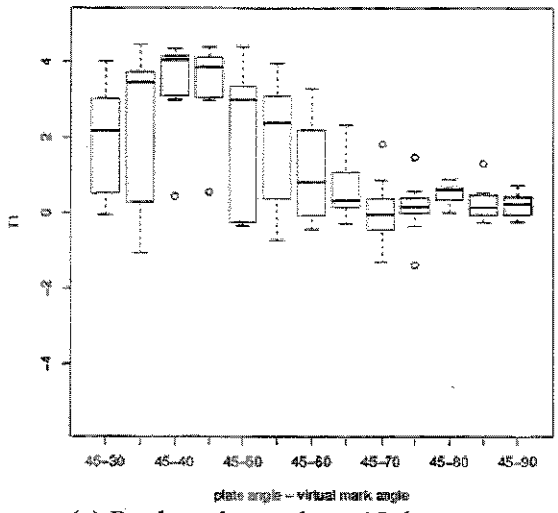
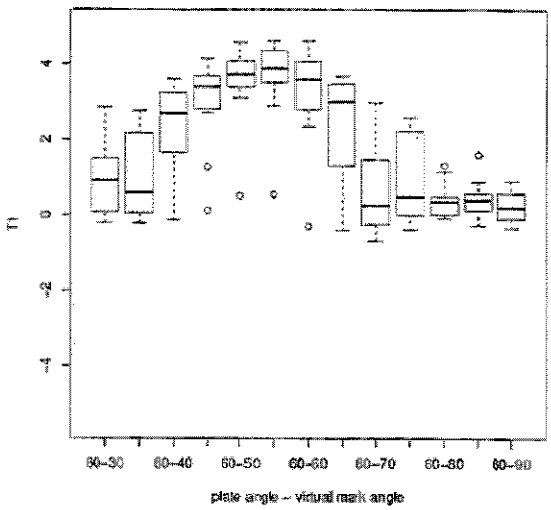


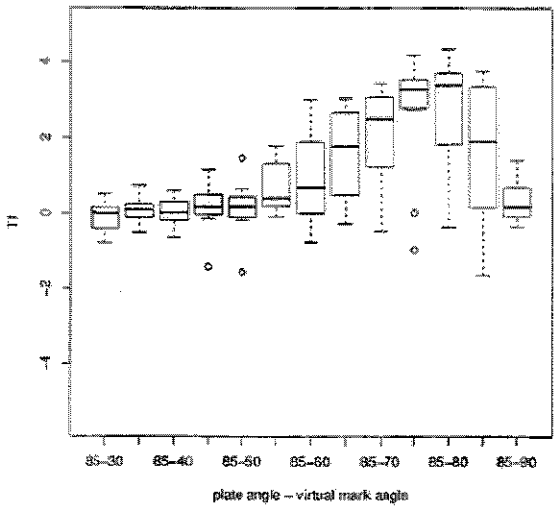
Figure 36: *Possible causes of large negative index T1 values for non-match cases. There was a large unusual structure at the end of the virtual mark, making the Optimization step result in an incorrect "best fit" location.*



(a) Real marks made at 45 degrees

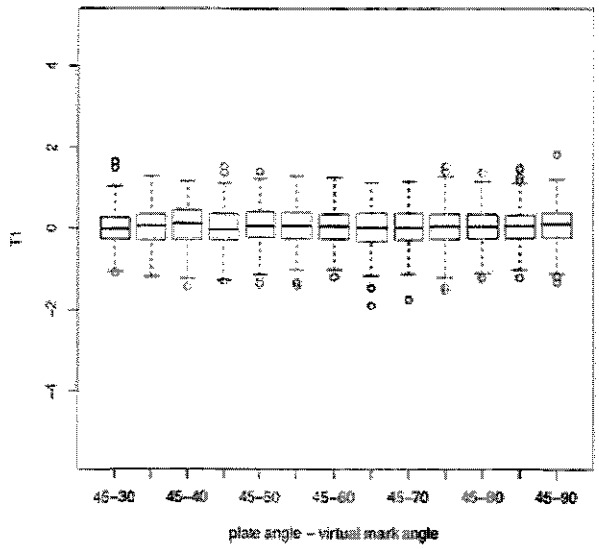


(b) Real marks made at 60 degrees

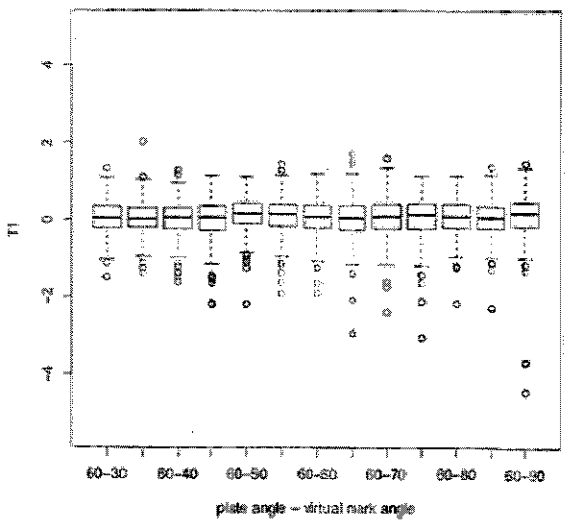


(c) Real marks made at 85 degrees

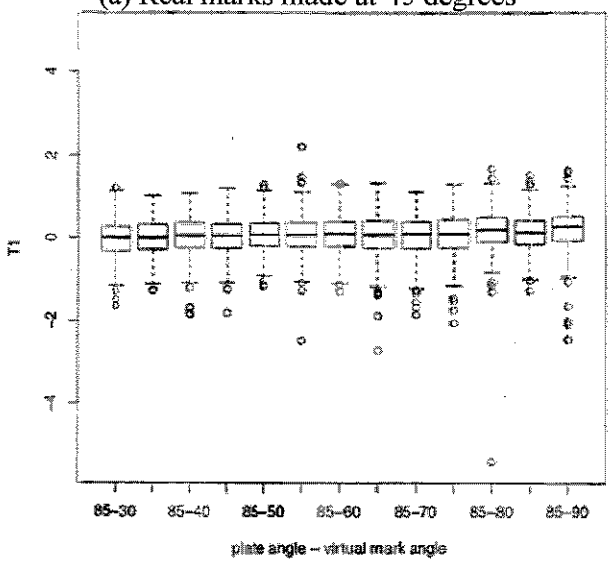
Figure 37: *Box plots showing T1 results when comparing virtual marks with real marks made from known angleswith the same screwdrivers*



(a) Real marks made at 45 degrees



(b) Real marks made at 60 degrees



(c) Real marks made at 85 degrees

Figure 38: *Box plots showing T1 results when comparing real and virtual marks from different screwdrivers (different sides of the tip or different tips)*

IV. Conclusions

IV.1: Discussion of findings

This project has sought to answer the following question: Can a manipulative "virtual" tool be made to generate "virtual marks" for quantitative and objective toolmark characterization? In other words, can it be said that toolmarks obtained from a "virtual tool" yields characteristics that are unique enough to that tool (and only that tool) to enable matching of the virtual toolmark to actual toolmarks to allow an objective identification algorithm to be employed for identification purposes? This question has definitely been answered in the affirmative for the population of tools examined. The ability to discriminate between markings is based on a number of factors, including:

1. The quality of the marking itself.
2. The ability to quantify the marking that exists.
3. The ability to truthfully generate 3D digital representation of the tools.
4. The ability to manage noise from an optical profilometer.
5. The ability to correctly generate the virtual markings from the virtual tool.
6. The manner in which the objective, automated routine is designed to operate.

The importance of having high quality markings and truthful 3D digital representation of tooltips is supported by the objective results of the study. If high quality markings do exist and the virtual markings are correctly produced from truthful 3D virtual tools, using suitable methods to quantify the results and incorporating contextual information into the analysis of the data greatly increase the ability of an automated routine to separate marks made from ostensibly identical tools.

The virtual toolmark characterization technique heavily relies on truthfully scanning screwdriver tips. Yet, optically scanning 3D screwdriver tips with high quality turned out to be a non-trivial task due to high angles as well as surface specularity. Special care should be given to this step to ensure "sufficient" quality 3D data obtained before using any software routines.

IV.2: Implications for policy and practice

Given that the tools examined in this study should have been as identical as possible to one another implies that unique markers do exist for every tool manufactured by this company using the tools currently employed for their manufacture. The question then becomes, of the factors listed above, what elements must be addressed to yield a fully automated, objective result? If a poor quality marking exists an unambiguous determination may be impossible. The level to which the toolmark(s) must be examined then becomes a matter of question and this level was not determined in this study. Certainly the level used in this study would appear sufficient if contextual information is included. Finally, the exact manner in which the algorithm operates, and the manner in which data is acquired, becomes a critical question.

IV.3: Implications for further research

Testing of the algorithm on other types of tool marks would be appropriate to determine the applicability of the program to other marks. Currently, the algorithm is set up to evaluate striated marks; whether this can be generalized to other marks is unknown.

Another area of research is the question concerning the cut-off values used to qualify a comparison as either a match or nonmatch. The T1 values identified for the angles tested were seen to vary. This is most likely related to the quality of the marking. Some quantitative measure of mark quality could possibly be developed that would give an indication of when the

results of a comparison are likely to be valid. Related to this, a study concerning the variance in data would be of value. This would involve having multiple persons acquire data from the same samples then employ the algorithm to see how the results compare between operators. This study would yield valuable statistical information concerning algorithm performance and robustness.

Unlike a stylus profilometer that only generates one trace per scan, the optical profilometer usually provides hundreds even thousands of traces per scan, making the size of data enormously larger. For example, the size of each 3D scan used in this research is approximately 100 MB. How to manage and store such 3D metadata would be a vital issue to handle. Therefore, developing compression techniques (similar to 2D image compression techniques) for 3D data would be of significant interest.

Finally, the software package developed uses module-based strategies, making it easier to change some modules. The GUI was developed with Qt, and the programming language used was C++. The software was developed and primarily tested on Ubuntu 12.04 LTS. (Ubuntu is a very popular Linux flavor.) It was also tested on Windows XP, Windows 7, and Mac OS X Snow Leopard. Further testing on Mac OS is currently underway. The software was designed in such a way that it can operate on mobile computers (e.g., a laptop) with standard hardware configurations. A further study involving development of an all-mobile system for toolmark and firearm examinations is of considerable interest. The intention of making this software package free and open source once it matures could benefit the whole community at large.

V. References

Alicona Imaging GmbH (2012), "Focus Variation".English edition. Retrieved from http://issuu.com/alicona/docs/alicona_focusvariation3_e_high_resolution.

D. Baldwin, M. Morris, S. Bajic, Z. Zhou, M. J. Kreiser (2004),"Statistical Tools for Forensic Analysis of Tool marks Ames (IA)," Ames Laboratory Technical Report IS-5160.

B. Bachrach (2002), "Development of a 3-D-based Automated Firearms Evidence Comparison System", Journal of Forensic Science, **47**, 1253-1264.

R. S. Bolton-King, J. P. O. Evans, C. L.Smith, J. D. Painter, D. F.Allsop, and W. M. Cranton, (2010), "What are the prospects of 3D profiling systems applied to firearms and toolmark identification?" AFTE Journal, 42(1) 23-33.

M.S. Bonfanti, J. DeKinder (1999a), "The Influence of the Use of Firearms on their Characteristic Marks,"AFTE Journal, 31(3), 318-323.

M.S. Bonafanti, J. De Kinder (1999b),"The Influence of Manufacturing Processes on the Identification of Bullets and cartridge cases – A Review of the Literature," Science and Justice 39, 3-10.

A. Biasotti (1959), "A Statistical Study of the Individual Characteristics of Fired Bullets," Journal of Forensic Science 4(1), 34-50.

A. Biasotti, J. Murdock (1984), "Criteria for Identification or State of the Art of Firearm and Tool mark Identification," AFTE Journal, 4, 16-24.

L.S. Chumbley, M. Morris , J. Kreiser, C. Fisher, J. Craft, L. Genalo, S. Davis, D. Faden, and J. Kidd (2010), "Validation of Toolmark Comparisons Obtained Using a Quantitative, Comparative, Statistical Algorithm," Journal of Forensic Science, 54(4) 953-961.

J. Kidd (2007), "Comparison of Screwdriver Tips to the Resultant Toolmarks", Master's thesis,Iowa State University, Ames, Iowa.

D. Faden, J. Kidd, J. Craft, L.S. Chumbley, M. Morris, L. Genalo, J. Kreiser, and S. Davis (2007), "Statistical Confirmation of Empirical Observations Concerning ToolmarkStriae," AFTE Journal, 39(3) 205-214, 2007.

NAS Report (2008), "Report by a committee for the National Research Council for the National Academy of Sciences," Ballistic Imaging, March.

D.Shreiner (2010), [OpenGL Programming Guide], Addison Wesley, 7thed.

S. Zhang (2012), "Three-dimensional range data compression using computer graphics rendering pipeline," Appl. Opt. **51**(18), 4058-4064.

S. Zhang, D.Eisenmannand L. S.Chumbley, "Automatic 3-D Shape Measurement Noise Reduction for an Optical Profilometer", OSA Topical Meetings on Digital Holography and Three-Dimensional Imaging (DH), Vancouver, Canada, 2009.

S. Zhang, L. Ekstrand, T. Grieve, L. S. Chumbley, and M. Morris (2012), "Three-dimensional data processing with advanced computer graphics tools," in SPIE Optics and Photonics, (San Diego, California).

VI. Dissemination of Research Findings

A. Refereed Publications

L. Ekstrand, S. Zhang, T. Grieve, L. S. Chumbley, M. J. Kreiser, "Virtual toolmark generation for efficient striation analysis," Journal of Forensic Science (to submit)

B. Theses Written

L. Ekstrand, "Virtual tool mark generation for efficient striation analysis in forensic science," Master of Science Thesis, Iowa State University, Ames, Iowa, December 2012 (defended on Nov. 6, 2012).

C. Non-Refereed Publications

S. Zhang, L. Ekstrand, T. Grieve, L. S. Chumbley, and M. Morris, "Three-dimensional data processing with advanced computer graphics tools," in SPIE Optics and Photonics, (San Diego, California) (2012).

D. Presentations

L. Ekstrand, S. Zhang, L. S. Chumbley, T. Grieve, and D. J. Eisenmann, "Virtual tool mark generation for efficient tool mark analysis," in American Academy of Forensic Sciences, (Washington, DC) (2013) (accepted for oral presentation).

S. Zhang, L. Ekstrand, T. Grieve, L. S. Chumbley, and M. Morris, "Three-dimensional data processing with advanced computer graphics tools," in SPIE Optics and Photonics, (San Diego, California) (2012).

L. Ekstrand, S. Zhang,L.S.Chumbley,T.Grieve,E.Villagomez,D.J.Eisenmann,M.Morris,andJ.Kreiser, "Recent study on virtual manipulative tools for tool mark characterization," in 39th Annual Symposium Communication in a Laboratory Setting, (Denver, Colorado) (2011) (poster presentation).

S. Zhang, L. S. Chumbley, M. Morris, B. W. King, A. B. Hoeksema, T. Grieve, Y. Gong, V. Villagomez, D. J. Eisenmann, and J. Kreiser, "Virtual manipulative tools for tool mark characterization," in AFTE 2011 Training Seminar Program, (Chicago, IL) (2011) (oral presentation).