



46th SME North American Manufacturing Research Conference, NAMRC 46, Texas, USA

3D representation and CNC machining of 2D digital images

Sumit Sood^a, Ravinder Kumar Duvedi^{a,*}, Sanjeev Bedi^b, Stephen Mann^b

^aThapar Institute of Engineering and Technology, Patiala-147004, Punjab, INDIA

^bUniversity of Waterloo, Waterloo-N2M3G1, ON, CANADA

Abstract

In this paper a new paradigm for CNC carving of digital images in the form of lookalike three-dimensional (3D) surfaces on wooden or metallic plaques is discussed. This work is focused on development of a single page windows based console application for conversion of a two-dimensional (2D) digital image into a 3D freeform surface representation in the form of a point cloud data and STL format. Subsequently, the 3D surface data is then used for generating an efficient toolpath data for 3-axis CNC finish machining using a ball end mill tool. The results from the developed algorithm are validated using a machining simulation in the virtual environment of an Open-GL based 3D graphical simulator ToolSim [1].

© 2018 The Authors. Published by Elsevier B.V.

Peer-review under responsibility of the scientific committee of the 4th International Conference on System-Integrated Intelligence.

Keywords: 2D image; 3D surface; Point cloud data; STL; Toolpath; CNC Machining

1. Introduction

The advancements in the technologies like CNC machines, Computer Aided Design (CAD), Computer aided Manufacturing (CAM) have made it possible to create high quality intricate artifacts in less time. Market competition has diminished the price of CAD/CAM software packages, but they are still out of the reach of a hobbyist. Usually hobbyists do not require the accuracy provided by commercial CNC machines [2]. A number of economical CNC solutions are now available including CNC routers and single axis controlled CNC lathe milling machines for 3D sculpturing [2]. These machines do not require special expensive controllers; instead they make use of a personal computer that can perform the controlling work for these machines. The goal of the work presented in this paper is to provide economic CAD/CAM packages that may not provide all

features but can still fulfill the major needs of the hobbyist.

One need of the hobbyists is to sculpt images/human portraits in 3D form into solid materials like wood, metal or stone that can play a vital role in home or office décor. The same kind of fine craftsmanship can also be witnessed in various holy places like temples, mosques or churches, which is either highly human skill dependent or it is mechanized but then the designs and patterns are limited. To create human portraits, 3D scanning coupled with CAD and CAM software packages can be used to capture a 3D model of a face and then CNC machines can carve it. However, this process is expensive and the cost involved usually cannot be justified for the creation of only a single artifact. Moreover the digitization of the dark gray parts like hair on the head and face require special processing [10].

Rockwood and Winget [3] proposed a method in which a 3D model can be constructed using 2D images of the object and an adaptable mesh. The mesh is refined through iterations to approximate the shape of the object. Chuang et al. [4] presented an approach to generate 3D models from physical models by scanning a model using a 3D digitizing machine and integrating multiple

* Corresponding author. Tel.: +91-175-239-3086; fax: +91-175-239-3005.

E-mail address: rduvedi@thapar.edu

scans into a triangulated mesh. Zeng et al. [5] demonstrated a method for generating a 3D surface model of the object from the multiple images of the object taken by a common consumer camera at different angles. The whole surface was generated by stitching the surfaces patches together that can be scaled up and down as per the requirement. Published literature shows that no solution is available that can automatically generate physical 3D human portraits from a single 2D image. A great amount of research work has been done in the area of conversion of a normal two dimensional digital image into 3D data, but is limited to 3D digital displays which again need some additional tools to realize the desired effect [5–10].

In this paper an approach to generate toolpaths for carving 3D human portraits and images by utilizing the information available in a single digital image is presented. Freeform surfaces generated from digital images are stored in the form of point clouds and as faceted models in STL. These freeform surfaces are then used for the generation of a raster toolpath for carving the model into plaque using ball nosed end mill cutter. There are two keys ideas in our approach. The first idea is that we start with a front lit image, and use the pixel intensities as depth values. The second key idea is that at times this requires user intervention to specify the depth value of dark regions that should be adjusted to match those of lighter regions; such an adjustment is usually required for portrait images to adjust hair to be of similar depth to the face.

Two techniques are used to improve the efficiency of the technique. First, rather than generate a toolpath from the STL file, we can generate the toolpath directly from the point cloud data. Second, regular side-step and feed-forward step discretization for raster toolpath generation creates an excessive number of cutter location points. A CNC machine has to move the tool through all these points for which it has to accelerate and decelerate frequently. This increases the machining time drastically [11]. To achieve maximum possible feed rates, successive points must be as far apart as possible. A custom cutter data reduction algorithm is used to eliminate redundant points from densely created toolpath data to reduce the machining time [12].

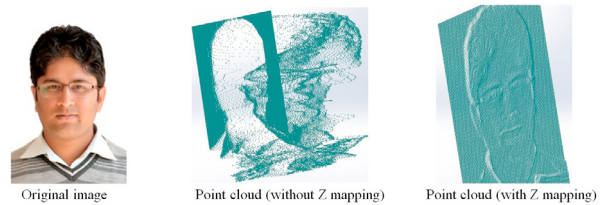


Fig. 1. Generating 3D point cloud data from a digital image.

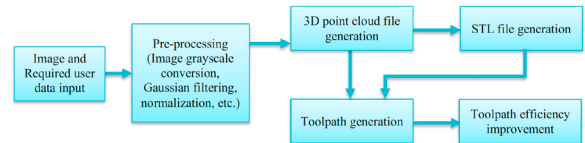


Fig. 2. Basic process flow of the steps used in the 2D image to 3D surface model development and toolpath generation for 3-axis vertical CNC machining.

2. Concepts used for 2D Image to 3D surface construction/ Background

2.1. Conversion of a digital image to freeform surface

An RGB (Red-Green-Blue) raster digital image contains a significant information in terms of color intensity at every pixel. The raster digital image is converted to a gray scale image, with each pixel converted to an n -bit scale using the following formula [13]

$$Y = 0.299 \times R + 0.587 \times G + 0.114 \times B. \quad (1)$$

where Y is the single intensity value of the pixel in grayscale, R is the intensity value of the red color for the pixel, G is the intensity value of the green color for the pixel and B is the intensity value of the blue color for the pixel.

Each pixel located at a given x - and y -axis position on a two dimensional digitized grayscale image can be then used to assign each pixel a Z -depth/height information using the following formula

$$Z = (I_c - Z_{min}) \times \frac{(Z_{max} - Z_{min})}{(I_{max} - I_{min})} + Z_{min}, \quad (2)$$

where Z is the mapped height of the pixel, I_c is the intensity value for the pixel, Z_{min} and Z_{max} are the minimum and maximum depth or height respectively (both user specified), I_{min} and I_{max} are the minimum and maximum values of intensity (among all pixels). Figure 1 shows the point cloud generated from an image without Z mapping and the point cloud generated when Z_{max} is 5.0mm and Z_{min} is 0.0mm.

This approach is used to develop a Microsoft Windows application in Visual C++ 2010 with a single page GUI. The OpenCV 2.4.10 computer vision library was used for image processing. OpenCV 2.4.10 is an open source computer vision library containing thousands of computer vision algorithms available under open source Berkeley Software Distribution license [13].

The process of the 2D grayscale image to a 3D surface consists of three steps:

1. Digital image to 3D freeform surface generation,
2. Toolpath generation, and
3. Toolpath efficiency improvement.

The Figure 2 depicts the overall process of the developed application.

The final output of the developed approach depends greatly upon the quality of the input image since the 3D model is generated using the information available in the image. There are several factors like image size/resolution, image brightness, etc., that need to be carefully considered for selecting an image for freeform surface generation to get the best possible results. These factors are briefly discussed below in the following section.

2.1.1. Image resolution

The resolution of a digital image can be defined in terms of number of rows and number of columns of pixels in an image. For example a resolution of 800×600 means that there are 600 rows and 800 columns of pixel in the image constituted of 480,000 pixels. In the proposed algorithm each pixel corresponds to a single point in a 3D space. Hence, the larger the number of pixels, the denser the point cloud data. The maximum size image that can be processed using this application is limited by the memory available for dynamic allocation, storage memory and constraints imposed by the OpenCV. Images of $10,000 \times 10,000$ pixels have been successfully tested with this application, which is much higher than the resolution usually required for the practical application of this methodology.

2.1.2. Image brightness

Image brightness or image intensity plays a key role in depth calculation. Brighter regions in the image are considered closer to camera while the darker regions are further away. Improper lighting and environmental conditions may cause unwanted shading and shadows in the image.

Figure 3(a) shows an image with improper lightening, causing a shadow on the right side of face, thus making it appear darker in comparison to the left region



Fig. 3. Images with different illumination: (a) image with improper lighting (illumination source towards one side of the face), (b) image with low lighting and (c) image with proper lighting (subject facing the source of illumination).

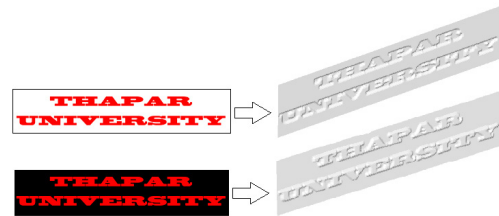


Fig. 4. Effects of colors of the input image on the developed 3D model.

of the image. Figure 3(b) represents an image taken in low lighting conditions which suppresses the sharp details required for construction of reasonably good 3D surface data. Figure 3(c) shows an image having a uniform illumination over the subject's face and such images are ideal for the developed application. For the developed methodology, properly illuminated images are the ideal input.

2.1.3. Image colors

Colors in the image plays a significant role in freeform surface generation since the color intensity at each pixel is used to compute the Z-height for the pixel. In case of an 8-bit grayscale image, the colors range from black (0) to white (255), with the other grayscale intensities having values between 0 and 255. Color selection in the image greatly affects the final output, based on which engraved or embossed text letters or patterns can be generated using the presented methodology. Figure 4 shows an example in which the colors of the fonts used in conjunction with the background color is used to decide whether the letters in the image will be engraved or embossed relative to the position of the subject in the input/base image.

2.1.4. Gaussian smoothing (blurring)

In a 2D digital image, the brightest as well as darkest region can exist at arbitrary locations and may be located adjacent to each other. Use of such an image for

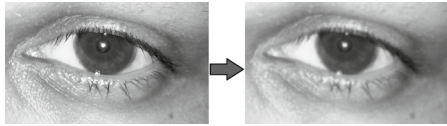


Fig. 5. Gaussian blurring effect: Original image (left) to the grayscale image after Gaussian blurring (right).

generating a 3D surface will result in abrupt changes in Z -height of the 3D points because the brightest region represents the highest Z -height value and the darkest region represents the lowest Z -height value. For generating a smooth 3D point cloud data or a smooth faceted 3D STL surface model for an image, the abrupt changes in Z -heights of the adjacent pixels are undesirable. To overcome this problem, Gaussian blurring was used to smooth the grayscale images using the following formula [13]:

$$G_0(x, y) = Ae^{-\frac{(x-\mu_x)^2}{2\sigma_x^2} - \frac{(y-\mu_y)^2}{2\sigma_y^2}}, \quad (3)$$

where $G_0(x, y)$ is the Gaussian function, A is the amplitude, μ_x and μ_y are the means, and the variables σ_x and σ_y are the standard deviation in x - and y -axes directions respectively. The Gaussian filter removes noise from the image and smooths the image by reducing the unnecessary details. Figure 5 shows an original image and the image obtained after applying Gaussian filtering. In the present work, OpenCV was used for performing Gaussian blurring on the input images.

3. Processing of a human portrait

In a human portrait, the human image is required to be identified so that the background can be either eliminated or can be processed separately so that a better 3D look can be given to the portrait. Also, the regions containing facial hairs are required to be identified so that they can be positioned at suitable Z -height/depth with respect to the whole portrait. For processing human portraits a separate option is included in the developed application. This method is not fully automatic and requires user intervention at various steps. Under this option, the application allows the user to select the human figure in the image to separate it from the background. This is done by using the polygon selection tool to select the image as shown in Figure 6. Once the selection is done, the application generates an image of the same size as that of input image where the human portrait area is marked as black and the rest of the area is marked as white. In addition, the user selects two other additional images. The first is for the selection of head

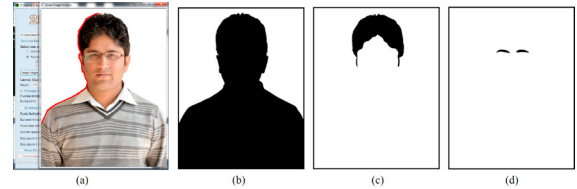


Fig. 6. Human portrait processing: (a) image region selection, (b) isolation of the image from the background, (c) identification of head hair (black colored) area and (d) identification of facial hairs areas (black colored).

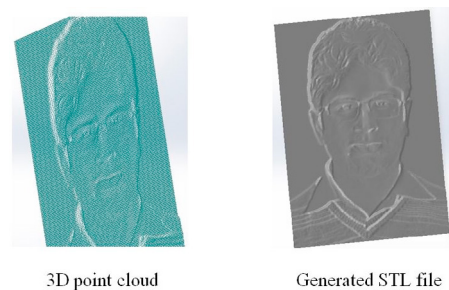


Fig. 7. Construction of the STL data file from point cloud data.

hairs and the second is for the selection of facial hairs and dark regions that need to be raised as compared to bright regions. Multiple objects/regions can be selected by drawing multiple selection polygons.

To elevate the dark region specified by the user, the intensities of the pixels in that area are modified. The proportion by which the intensity of pixels is modified is variable and can be adjusted by the user. After this modification, the image is processed using the same approach as that is used for normal image processing.

By implementing the methodology discussed so far, point cloud data can be generated for an input digital image. Once the point cloud data is generated (using the gray scale intensities of individual pixels), a 3D faceted/STL model can be crafted as all the points in the point cloud are evenly arranged in the xy -plane. A simple stitching algorithm is used to generate triangulated facets from the 3D point cloud and stored as an STL file. Figure 7 shows a sample 3D point cloud data and its corresponding STL file.

This complete process of converting a digital image into a 3D free form surface is summarized in a flowchart in Figure 8.

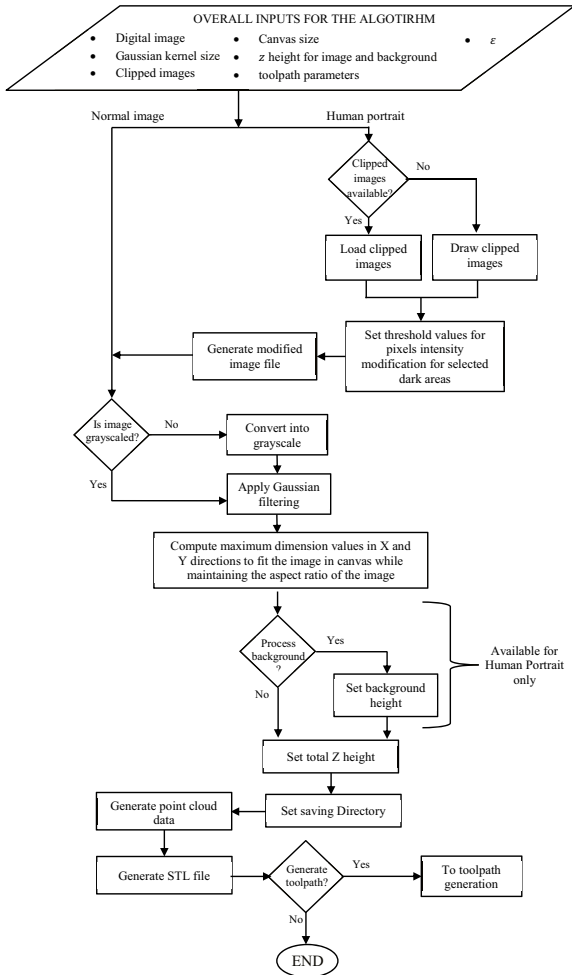


Fig. 8. Flowchart for the digital image to 3D surface data generation algorithm.

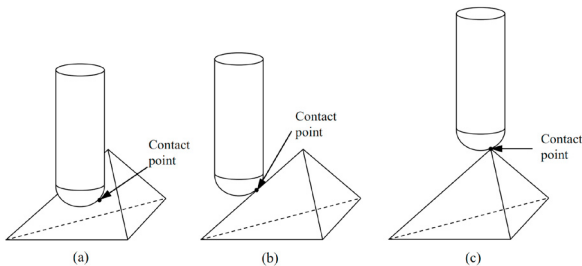


Fig. 9. Identifying the gouge free tool positioning of a ball end mill on the STL surface, when it is touching: (a) triangulated facet, (b) an edge of a facet, or (c) a raised vertex on the faceted surface.

4. Methodology for generating an efficient toolpath for 3-axis CNC machining

Toolpath planning is crucial for good results in CNC machining of freeform surfaces. In industry, ball end

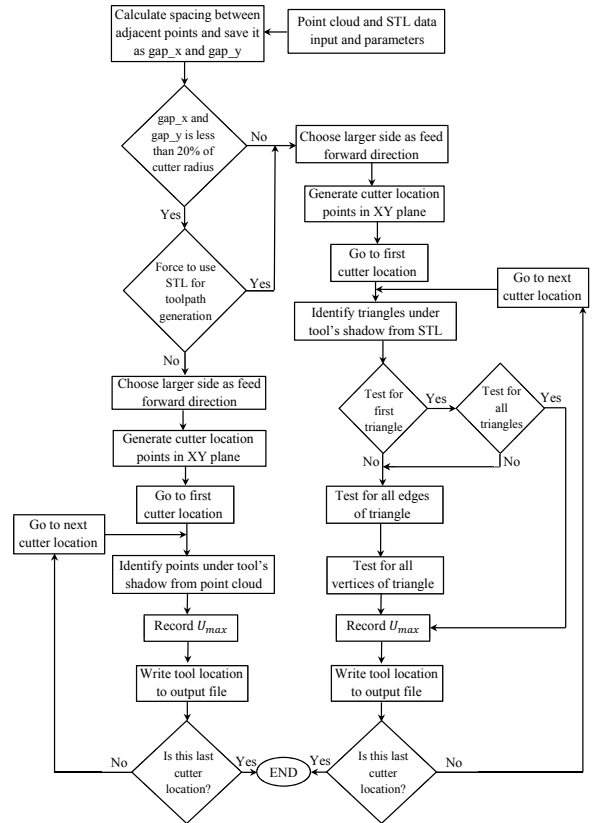


Fig. 10. Flowchart for toolpath generation algorithm from 3D point cloud and/or 3D faceted (STL) data.

mills are mostly used for the machining of complex freeform surfaces [14,15]. For the present work, a Cartesian toolpath with a zig-zag [17] tool footprint was implemented for generating toolpaths for a ball nosed end mill cutter. The “Ball drop” method of Manos et al. [2] is a robust method for calculating the tool locations that ensures gouge free toolpaths for STL surfaces. This method was initially developed for a Single Controlled Axis Lathe Mill (SCALM) that uses helical toolpaths, but later it was extended to other tool shapes and 3- and 5-axis machining [15–17]. To find cutter location points while using a ball nosed end mill, a sphere of radius equal to the radius of ball end mill is dropped along the tool axis at the tool location and its first contact with the surface is recorded. Based on this first contact with the surface, the tool center height is calculated to position the tool at that particular tool location. Similarly the tool height for each tool location is calculated. For STL models a tool can contact the surface in three ways as shown in Figure 9.

The tool might touch the face of any of the triangular facets, it might touch any edge shared by two triangles

or it might touch any vertex shared by two or more triangles. The mathematical algorithm for finding the tool's center height, called *cutter location* (CL) points, for 3-axis machining we used the “Tool Drop” algorithm for a generalized radiused end mill developed by Duvedi et al. [16,17]. The CL points are sequentially stored as the toolpath data.

A new shadow check method to identify the triangles lying under the tool at an instant was implemented for generating a toolpath for 3D surfaces using STL data. In this method the whole STL file is not considered at once. Instead of using the STL file as input, the point cloud file, from which the STL file was generated, is used. Positions of the points are stored in a 2D array in computer memory. Points lying under the shadow of the tool are identified using this array and the triangles are then generated using these identified points. These triangles are suitable candidates for the ball drop check for that position. Because of this, toolpath computation time decreases considerably since at every point we do not have to check all the triangles to identify the suitable candidates but instead, the triangles under the tool's shadow are generated and checked on demand. Once the triangles under the tool are identified, the three checks—namely the triangle check, the edge check and the vertex check—are performed to compute the tool center height that positions the ball end mill on the surface without gouging the neighboring faceted entities under the tool shadow [2,17].

The above method was also implemented for generating toolpaths using point cloud data by assuming that only the vertex check is performed on the points/vertices chosen from the input point cloud data. An additional criteria is needed that the distance between adjacent points (in x - and y -axis directions) in the point cloud data is such that the ball end mill of the given diameter does not dip between adjacent points. Figure 10 shows the algorithm for toolpath generation that was implemented in the present work.

4.1. Point cloud verses STL as input data for toolpath generation

It was observed during our tests that toolpath generation using point cloud data is much faster than toolpath generation using STL data (developed from same 3D point cloud data). The results for one of the test cases used to compare the toolpath generation metrics is shown in Figures 11. The input colored image used in this study is shown in Figure 11(a). Figures 11(b) shows the grayscale image after Gaussian smoothing. The gray scale image was mapped to the canvas size of

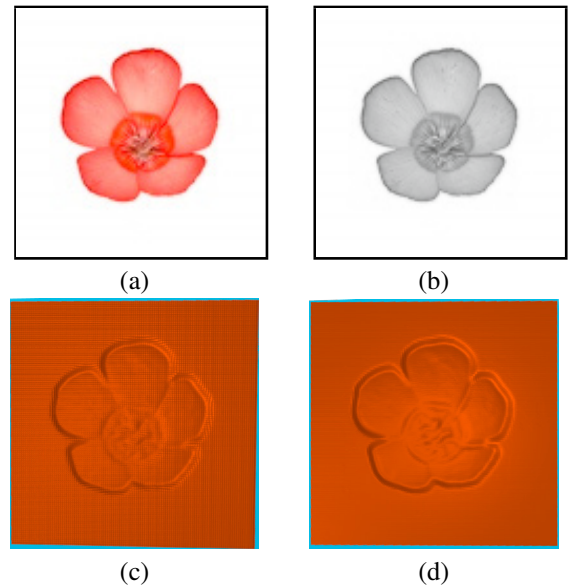


Fig. 11. Comparison of toolpath data generated for 3-axis vertical machining with a ball end mill tool: (a) input colored image—119×119 pixels [18], (b) raster gray scale image—119×119 pixels, (c) simulation results for the toolpath generated from 3D point cloud data with 14,161 spacial points, and (d) simulation results for the toolpath generated from 3D STL data file having 27,848 facets.

203mm × 203mm to generate the 3D point cloud data, where the spacial points in the 3D point cloud data had an equal spacing of 1.72mm along x - and y -axis directions.

The 3D point cloud data was also converted and saved as an STL file in ASCII. We separately generated toolpath data for the 3D point cloud and the STL data for 3-axis vertical machining for a ball end mill cutter of 3mm radius. The toolpath is generated for a side-step and feed-forward step size of 0.3mm.

The toolpath generation using the 3D point cloud data took 43 seconds while for the STL data, it took 196 seconds. The developed toolpaths were compared using the OpenGL based 3D graphical simulator ToolSim [1]. The simulation results for the toolpath generated using the 3D point cloud data and the STL data are shown in Figures 11(c) and (d), respectively.

In the case of the point cloud data (Figure 11(c)) the tool has a tendency to dip into the empty space between the adjacent points used for toolpath generation. Thus, the toolpath creates a grainy machined pattern that does not exist when machining using the toolpath generated from the STL file as shown in Figure 11(d).

Overall, the point cloud data can be used to generate a toolpath efficiently only if the point cloud is dense

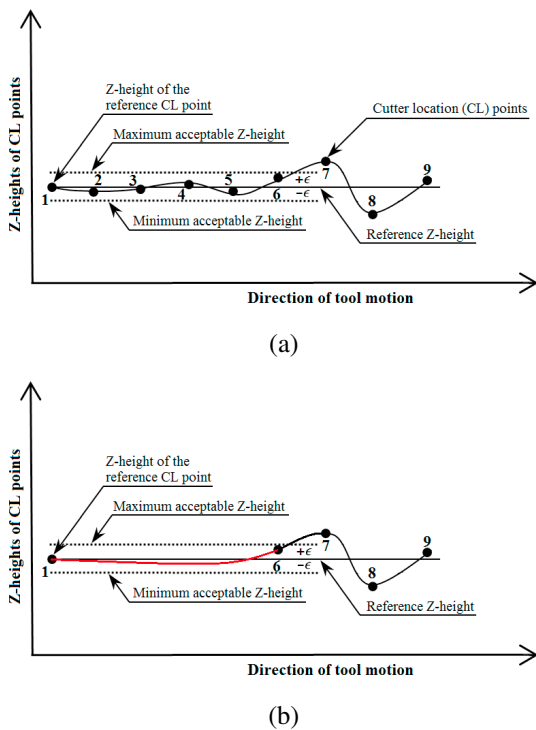


Fig. 12. Concept used for toolpath optimization using cutter location (CL) reduction algorithm: (a) CL points in an original input toolpath file (before implementing CL reduction algorithm), (b) CL points in toolpath file after implementing CL reduction algorithm (new toolpath shown in red color between point 1 to point 6).

enough so that tool does not dip in the space between the adjacent points beyond a required tolerance. No such issue exists if the STL data format is used for toolpath generation, but it is computationally slow.

Another issue identified from these studies is that the toolpaths generated using this methodology have an excessive number of tool location points. With a decrease in side-step value, feed-forward step value or both, the number of cutter location points increases further and vice versa. Therefore, to reduce the overall machining time using such toolpaths, we used a cutter location data reduction algorithm, which is discussed in the next section.

4.2. Cutter location data reduction algorithm

The total machining time for a surface machining operation on a CNC machine depends upon the number of cutter location (CL) points in the toolpath file. The number of CL points in a toolpath depends on: the size of the workpiece to be machined, the radius of the ball

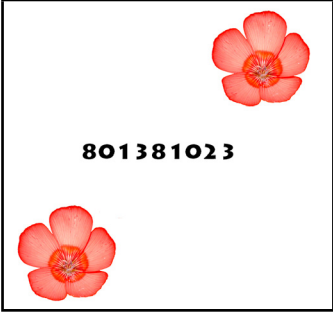


end mill used, the side-step and the feed-forward discretization of toolpath footprint [17] and the surface finish required. For an example, for a plaque having length and width of $203\text{mm} \times 203\text{mm}$ to be machined by a ball end mill tool of radius 3mm when the footprint discretization has an equal side-step and feed-forward discretization of 0.3mm , there will be more than 400,000 cutter location points in the finishing toolpath through which the tool has to move to machine the entire surface. Using a ball end mill cutter of smaller radius for finish machining can help in better surface quality, but it will be achieved at the cost of increased machining time; because of the smaller values of the side-step and the feed forward discretization, the resulting toolpath will have a larger number of closely spaced CL points. These closely spaced CL points do not let the CNC controller achieve the required machining feed-rate, further increasing the machining time since the machine has to interpolate through each CL point in the toolpath.

To achieve a good quality sculptured surface machining we cannot use higher values of side-step and feed-forward discretization for toolpath generation, which generally is equal to 25% to 40% of the tool diameter [15]. As a result the higher machining feed rates are not achievable while interpolating through these closely packed CL points.

Therefore, a cutter location points reduction algorithm was used in the toolpath generation algorithm to eliminate some of the identified CL points from the toolpath file size and thereby help in increasing the machining efficiency [12]. This algorithm takes advantage of the fact that in many cases of sculptured machining, very high surface accuracy is not desired. Thus some of the intermediate CL point can be omitted from the toolpath for which the difference between the Z-height, in comparison to its neighboring points, is within a given tolerance of $\epsilon \text{ mm}$. Thus, if the difference between the Z-heights of CL points along a line (along constant x - or y -axis position) is within a user specified tolerance ϵ , then all the CL points lying between the first and the last one (along the selected line) are omitted from the toolpath.

Figure 12(a) demonstrates the concept of cutter location reduction where numbers from 1 to 9 mark the position of nine CL points. Using the Z-height the CL point at position 1 as the reference, and for given tolerance of $\pm\epsilon$, the CL reduction algorithm identifies the last CL point in the given toolpath that lies in the given tolerance band of $(Z \pm \epsilon)$, which in case of Figure 12 is CL point at position 6. The algorithm eliminates CL points marked from positions 2 to 5, and retains the CL points at posi-

Table 1. Test results of cutter location (CL) points elimination algorithm. A ball end mill of radius 3mm, and a side-step size and feed-forward step size of 0.3mm was used in all example.

Input image [18]		
Part size	$203 \times 203 \times 5mm^3$	$203 \times 203 \times 5mm^3$
Number of CL points in original toolpath	456,976	456,976
Z-height tolerance (ϵ) used	0.0mm	0.02mm
Number of CL points in new toolpath	76,366	67,705
CL points eliminated	380,610 (83.28%)	389,271 (85.18%)
Processing time (cutter location data reduction)	19.906sec	18.938sec
Toolpath simulation without CL reduction		
Toolpath simulation after CL reduction		

tions 1 and 6. Thus, the tool will now move through CL points 1 and 6 only as shown in Figure 12(b). This step is repeated for all machining passes and eliminates unnecessary tool positions (CL points) from the toolpath file.





5. Results and discussion

The console application for 2D digital image to 3D surface model generation consists of three main components: a 2D to 3D surface data generation algorithm, a

toolpath generation algorithm and a CL data reduction algorithm. To validate the developed application, we tested it on several test cases. For the work presented in this paper the toolpaths were generated for a ball nosed end mill having radius of 3mm for a side-step and feed-forward step size of 0.3mm each.

Results from one of our test cases appears in Table 1. It is observed from the visual inspection of the toolpath simulation results that there is not any significantly noticeable distinction between the results obtained from the three different toolpaths for this test case. The mi-

Table 2. Toolpath generation results using STL data mode for an art image of Krishna and human portrait. A ball end mill of radius $3mm$, and a side-step and free-forward step size of $0.3mm$ was used in both examples.

Input image		
Image size and source	480×480 pixels, [19]	1104×1392 pixels Human Portrait with edited background
Image processing mode	Normal Image	
No. of point in point cloud	230,400	1,536,768
No. of triangulated facets in STL file	458,882	3,068,546
Toolpath generation mode	STL	STL
Initial number of CL points	456,976	363,549
Z-height tolerance (ϵ) used	$0.01mm$	$0.01mm$
No. of CL points omitted	27,089 (5.92%)	120,254 (33.07%)
CL points in updated toolpath file	429,887	243,295
Toolpath generation time	13min 7sec	66min 38sec
Total processing time	16min 14sec	72min 58sec
Toolpath simulation		




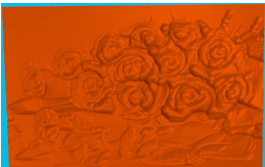

nor variations seen in the three machining simulation images, shown in Table 1, are because of the slight differences in the orientation and rendering while capturing the snapshots from the 3D graphical simulator Tool-Sim [1].

We tested our application on five other test images. Table 2 shows the results for the toolpath generated using STL data mode for two different test cases along with the 3D simulation results, while Table 3 shows the results for the toolpath generated using point cloud data mode for three different test cases (one of the source images in Table 3 has been omitted over copyright concerns; the missing image can be found at the URL cited in the table).

It can be observed from Tables 2 and 3 that for highly sculptured images there is less reduction in the number of CL points compared to the images with simpler back-

grounds or simple images like the human portrait images. The results for the toolpaths generated using the point cloud data for various images takes significantly less time compared to the cases when STL data format is used. Moreover, the CL data reduction algorithm works well for most of our test images, removing 32.91% to 85.18% of the CL points from the total toolpath data for Z-height tolerance of $\epsilon = 0.01mm$. The one exception is the Krishna image in Table 2, where only 5.92% of the CL points are eliminated. This reduction in data removal is because there is significant variation in the color intensities of most of the neighbouring pixels in the entire input image. To remove a larger number of intermediate CL points for such images, a higher value of ϵ needs to be used. The test results on which the CL reduction algorithm works effectively are images that have substantial regions with same color/depth of pix-

Table 3. Toolpath generation results using point cloud data mode. A ball end mill of radius 3mm with a step-size and feed-forward step size both of 0.3mm was used for all three examples.

Input image		Bouquet of roses [21]	
Image size and source	297 × 210 <i>pixels</i> , [20]	1024 × 640 <i>pixels</i> , [21]	1104 × 1392 <i>pixels</i> Human Portrait without background (Image region selection process given in Section 3 is implemented on input image to obtain a clipped image without background text similar to Figure 6(a) before computing 3D point cloud data)
Image processing mode	Normal Image	Normal Image	3D point cloud data)
Points in point cloud	202,704	655,360	1,536,768
Triangulated facets in STL	403,518	1,307,394	3,068,546
Toolpath generation mode	Point cloud	Point cloud	Point cloud
Initial number of CL points	242,684	286,371	363,549
Z-height tolerance (ϵ) used	0.01mm	0.01mm	0.01mm
No. of CL points omitted	165,444 (68.17%)	94,267 (32.91%)	151913 (41.78%)
CL points in updated toolpath	77,240	192,104	211,636
Toolpath generation time	52sec	1min 51sec	3min 34sec
Total processing time	2min 56sec	5min 32sec	10min 45sec
Toolpath simulation			

els. Regardless, even a reduction of 5% of the CL points from the toolpath data can significantly reduce the overall machining time for highly sculptured images such as the Krishna image. Further investigation is required to study the effect of variation in visual quality of the machined 3D surface by varying the scale factor for fitting the 3D point cloud data to a required canvas size and the ϵ values to be used for CL data reduction for a given image.

Overall, the developed application can generate accurate CNC machining toolpaths while significantly decreasing the total number of CL points (compared to when CL reduction step is not used), which in turn is helpful for achieving higher machining feed rates for precise machining of sculptured surfaces developed from the 2D digital images.

6. Conclusion

An application for converting a 2D digital image to a 3D machinable sculptured surface was presented in this work. The developed application was successfully implemented for home and office décor. The methodologies developed in this work were tested on general images as well as human portraits. It is found that the developed application is capable of handling these test cases as shown by the 3D simulation results of the developed toolpaths. Edge detection and other advanced image processing algorithms can be implemented for enhancing the developed application, which can be extended for identifying different objects in digital image so that a proper 3D model can be generated that will be equivalent to what our eyes perceive from the 2D image.

Acknowledgments

The authors acknowledge the indispensable help and resources provided by the State Initiated Design Centre, Thapar Institute of Engineering and Technology, India for compilation of this work. We also acknowledge the use of the 3D graphical machining simulation software “ToolSim” developed at the University of Waterloo, Canada for testing the machining toolpaths obtained from the developed 2D to 3D application.

References

- [1] S. Mann, S. Bedi, G. Israeli, X. L. Zhou, Machine models and tool motions for simulating five-axis machining. *Computer-Aided Design*, 42(3), (2010) 231–237.
- [2] N. P. Manos, S. Bedi, D. Miller, S. Mann, Single controlled axis lathe mill. *International Journal of Advanced Manufacturing Technology* 32(1–2), (2007) 55–65.
- [3] A. P. Rockwood, J. Winget, Three-dimensional object reconstruction from two-dimensional images. *Computer-Aided Design* 29(4), (1997) 279–285.
- [4] C. M. Chuang, C. Y. Chen, H.T. Yau, A reverse Engineering Approach to generating interference-free tool paths in three-axis machining from scanned data of physical models. *International Journal of Advanced Manufacturing Technology* 19(1), (2002) 23–31.
- [5] G. Zeng, S. Paris, L. Quan, F. Sillion, Accurate and scalable surface representation and reconstruction from images. *Pattern Analysis and Machine Intelligence*, IEEE Transactions 29(1), (2007) 141–158.
- [6] C. C. Cheng, C. T. Li, P. S. Huang, T. K. Lin, Y. M. Tsai, L. G. Chen, A block-based 2D-to-3D conversion system with bilateral filter. *Digest of Technical Papers of International Conference on Consumer Electronics*, IEEE 2009:1–2.
- [7] X. Huang, L. Wang, J. Huang, D. Li, M. Zhang, A depth extraction method based on motion and geometry for 2D to 3D conversion. *Third International Symposium on Intelligent Information Technology Application*, IEEE Computer Society 3 (2009) 294–298.
- [8] C. C. Cheng, C. T. Li, L. G. Chen, A 2D-to-3D conversion system using edge information. *Digest of Technical Papers of International Conference on Consumer Electronics*, IEEE 2010, 377–378.
- [9] S. K. T. Sheela, 2D TO 3D Conversion for images using Hough transform. *International Journal of Emerging Technology and Advanced Engineering* 3(1), (2013) 482–486.
- [10] N. E. Yang, J. W. Lee, R. H. Park, Depth map generation using local depth hypothesis for 2D-to-3D conversion. *International Journal of Computer Graphics and Animation* 3(1), (2013) 1–15.
- [11] B. H. Kim, B. K. Choi, Machining efficiency comparison direction-parallel tool path with contour-parallel tool path. *Computer-Aided Design* 34(2), (2002) 89–95.
- [12] S. Sood, Efficient NC Toolpath Generation for 3-axis Machining of 3-Dimensional Freeform Faceted Surfaces Developed from Digital Images. M.E. Thesis, Mechanical Engineering Department, Thapar Institute of Engineering and Technology, Patiala, India, 2015.
- [13] The OpenCV reference manual 2.4.9.0, 2014.
- [14] H. T. Yau, C. M. Chuanga, Y. S. Lee, Numerical control machining of triangulated sculptured surfaces in a stereo lithography format with a generalized cutter. *International Journal of Production Research* 42(13), (2004) 2573–2598.
- [15] K. Patel K, G.S. Bolaos, R. Bassi, S. Bedi, Optimal tool shape selection based on surface geometry for three-axis CNC machining. *The International Journal of Advanced Manufacturing Technology* 57(5-8), (2011) 655–670.
- [16] R. K. Duvedi, S. Bedi, A. Batish, S. Mann, A multipoint method for 5-axis machining of triangulated surface models. *Computer-Aided Design* 52 (2014):17–26.
- [17] R. K. Duvedi, S. Bedi, A. Batish, S. Mann, Numeric implementation of drop and tilt method of 5-axis tool positioning for machining of triangulated surfaces. *The International Journal of Advanced Manufacturing Technology* 78(9-12), (2015), 1677–1690.
- [18] <http://maxpixel.freegreatpicture.com/Red-Lein-Bloom-Translucent-Red-Flower-Blossom-61321>.
- [19] <https://in.pinterest.com/pin/560698222335779054/>.
- [20] http://all-free-download.com/free-photos/download/pelican-bird-sea-birds_219115.html.
- [21] <http://www.wallpapermania.eu/wallpaper/beautiful-pink-bouquet-of-roses-love-moments>