Computer graphics and product modelling

ME735 – Course project

IIT BOMBAY

Improving machining efficiency by identifying and eliminating non-machinable region with the help of K-means clustering

Submitted by

1. Abhinandan Kumbhar (193109007)
2. Harishankar (193109008)
3. Mujahed Patil (193109009)
4. Rahul Pandey (193100074)

ABSTRACT

In recent decades, there is advancement in the automation field which helps in the manufacturing sector especially in machining. There are various machines which have multiple tool axis motion like 3-axis, 5-axis motion. In 5-axes, the tool has 3 linear motions and 2 rotational motions. While machining of freeform surfaces, tool orientation usually changes dynamically with 5 axis machine tool and kinematic performance of the rotary axes is usually weaker than that of linear axes, the optimum cutting speed is difficult or even impossible to achieve by the tool, which leads to lower machining efficiency. So, in this project, first, sub-division of surface is done. After that, surface optimization is performed to ensure the good machinability.

INTRODUCTION

Objective:

Identification of non-machinable region of freeform surface and re-clustering to improve machinability and efficiency with the help of K-means clustering

With advent of 5-axis machine tool, it has found stronghold in the field of machining of freeform surface parts in industries like automotive, aerospace and etc. 5-axis machines have some significant advantages over 3-axis machines. The two additional rotary axes of 5-axis machines greatly increase the accessibility and flexibility of machines which help in reducing machining time and increasing in machining efficiency. But these two benefits are not much utilized because due to geometric complexity, tool orientations usually change dynamically during freeform surface machining with 5-axis machine tool. As the kinematic performance (velocity, acceleration and jerk) of the rotary axes is usually weaker than that of linear axes, the real cutting speed is difficult or even impossible to reach the desired level, which further result in poor machining efficiency.

So, to reduce 5-axis machining time of freeform surfaces, 3+2 axis machining approach is used to machine freeform surfaces. 3+2 axis machining is in between 3-axis and 5-axis machining. It is used the 5-axis capability to orient the cutter and fix this tool orientation during the machining to obtain an optimal real cutting speed whiling machining complex surface machining. However, it is usually difficult or even impossible to determine such a fixed tool orientation optimally. Gray et al. [2] proposed a tool positioning strategy named the arc-intersect method to optimize the

tool orientation for given 3 + 2-axis machining toolpaths. Chen et al. [3] used fuzzy pattern clustering techniques and Voronoi diagram to divide the surface into several patches. Both toolpath and the set-up for each surface patch were planned for 3 + 2-axis machining. Roman et al. [4] classified the surface geometric properties into three categories, i.e. proximity parameters, orientation parameters and curvature parameters. Surface subdivision for 3 + 2-axis machining was then carried out by applying fuzzy C-means method with these parameters.

For this, an improved region-based 3+2 axis machining toolpath generation method for freeform surfaces using 5-axis machine tool is proposed. In this report, K-mean clustering algorithm is applied to classify all surface points into different categories for preliminary surface subdivision. Then a post processing procedure is used to optimize this subdivision result for ensuring good machinability i.e. finding non-machining areas.

ALGORITHM

Step 1: - To apply the K-mean clustering algorithm for surface subdivision, first the surface is meshed to extract discrete surface points. In this project, we use Bi-cubic Bezier Surface to demonstrate the algorithm. For Bi-cubic Bezier surface, 16 control points are required to draw the surface.

Let Cp_x, Cp_y, Cp_z are 4×4 matrix containing x, y, z coordinates of 16 control points respectively and blending functions for Bi-cubic Bezier surface are: -

$$Bu = [(1-u)^3, 3 \times u \times (1-u)^2, 3 \times u \times u \times (1-u), u^3]$$

$$Bv = [(1-v)^3, 3 \times v \times (1-v)^2, 3 \times v \times v \times (1-v), v^3]$$

So, surface points Px, Py, Pz can be obtained using vector valued parametric equation,

$$Px = Bu \times Cp_x \times Bv'$$

$$Py = Bu \times Cp_y \times Bv'$$

$$Pz = Bu \times Cp_z \times Bv'$$

Step 2: - Finding normal vector for each surface point and then applying feature normalising to all surface points and normal vector in order to keep the coordinates within 0 to 1.

Let V_i is a multi-dimensional vector containing coordinates of each surface point P_i and its normal vector N_i.

$$V_i = [P_i \quad N_i] = [Px_i \quad Py_i \quad Pz_i \quad Nx_i \quad Ny_i \quad Nz_i]$$

So, feature vector F_i for P_i

$$F_i = \frac{V_i - \min(V_i)}{\max(V)_i - \min(V_i))}$$

Step 3: - In K-means Clustering algorithm, 'K' is number of clusters whose value is generally in between [$1\sqrt{N}$] where N is number of surface points. So, success criteria for classification of points in a cluster is decided by NAI (normal aggregation index). NAI is the maximum angle made by normal at any point with the normal at centroid of cluster in which that point lies. If NAI $> NAI_0$, where NAI_0 is a predefined angle for measuring the similarity of the points inside a cluster, K should be plus 1 and the clustering procedures should be rerun with the new K. With the above method, surface points could be classified into different categories.

Step4: - Initiating with K=1 and locating centroid of a single cluster by writing code for K-means clustering in octave. Running K-means clustering for max iterations (50) over dataset and finding K centroids.

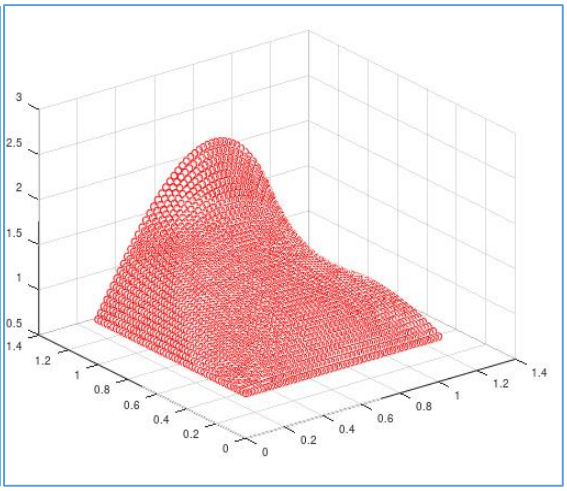
After completing one loop, the number of surface points that not fulfill the NAI_0 criteria are 1574 out of 2601 surface points. Ideally, all surface points should satisfy the NAI_0 criteria, but when surface is too complex, then it will be computationally more time consuming to form clusters. So, depending on surface and manufacturer requirement, the upper limit of the points that do not fulfill the NAI_0 can be relaxed a little bit.

Step 5: -

After iterating two more time. There are two more cluster formed as seen in figure and number of outlying points i.e. the surface points that do not fulfill the NAI_0

```
Running K means iteration

No of points lying outside the threshold NAI of c
outlying =  1574
Total number of clusters formed
K =  1
still some of the points have normals outside the
 taking more clusters
```

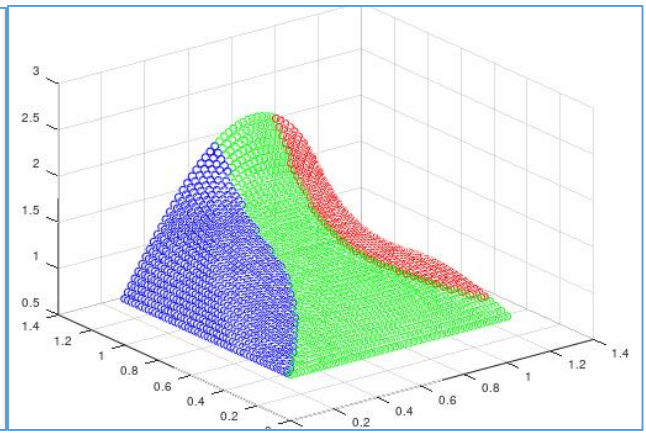


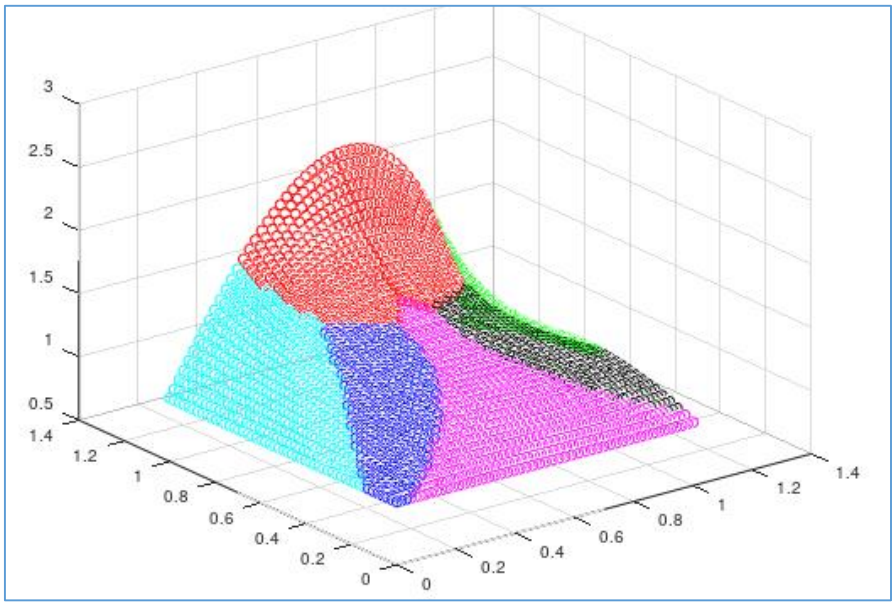
criteria are reduced to 148 out of total 2601 points.

Step 6: - For this surface, we set number of surface points that do not fulfill the NAI_0 criteria to 50. Then we get 6 clusters that fulfill all conditioned that are proposed.

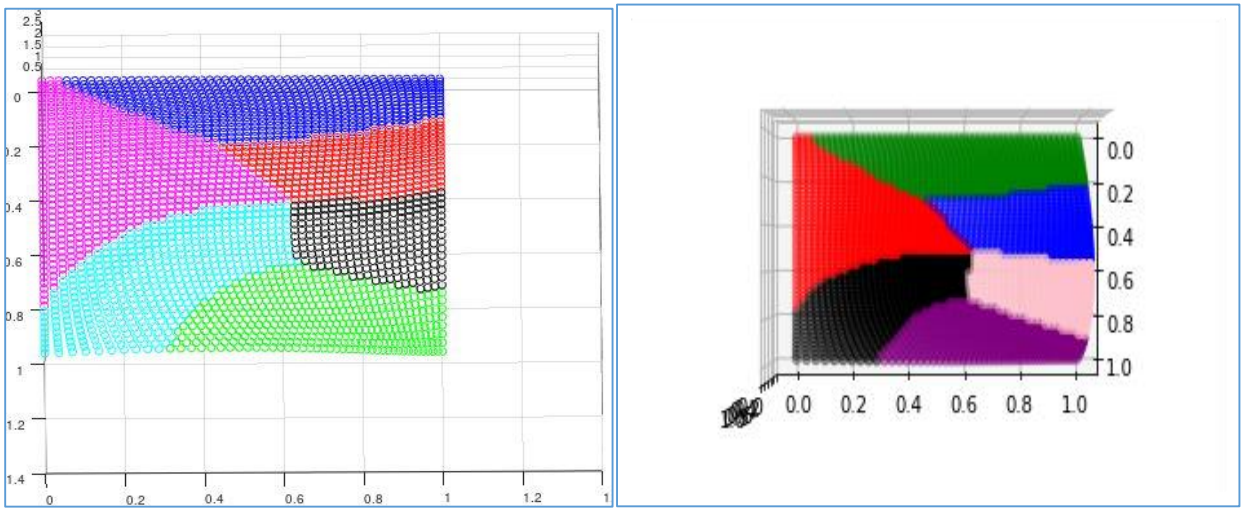
```
Running K means iteration

No of points lying outside the threshold NAI of c
outlying =  148
Total number of clusters formed
K =  3
still some of the points have normals outside the
 taking more clusters
```





Comparison in results from python and octave code for more clarification.



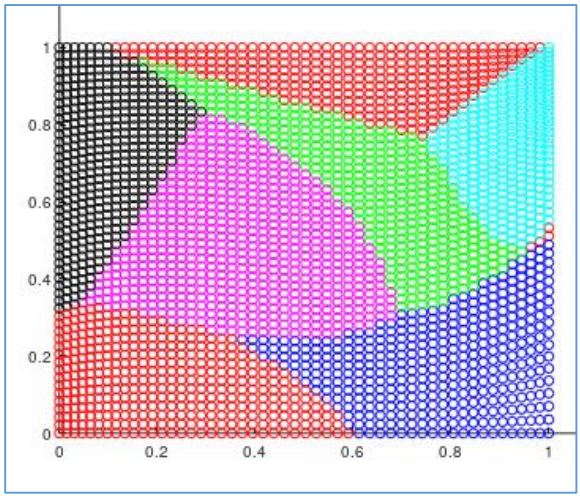
Octave result **Python result**

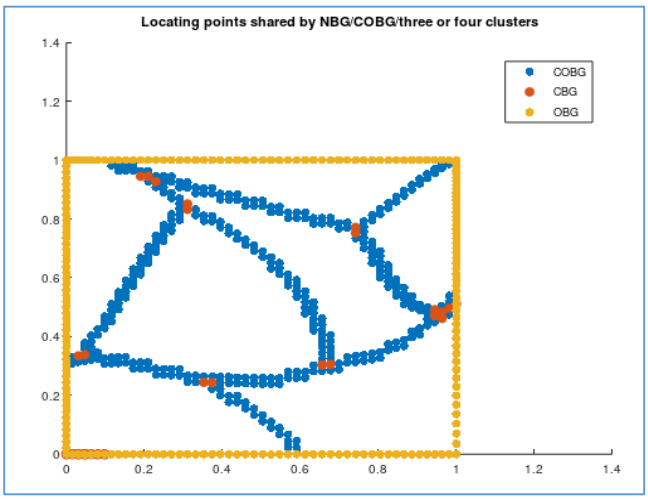
Here we can observe that both the algorithms shows similar results so we can verify our results obtained from the octave code and can continue the following steps of boundary points extraction and re-assignation of the region.

Step 7:- In this steps once the clustering is done, we will try to find the inside and outside boundaries. We first project the 3D surface on 2D surface.

1. Outside boundary points (OBG- Outside boundary points):
   By scanning through X and Y coordinates, we peak maximum and minimum of X and Y coordinates and whichever points, whose coordinates are close to this max or minimum values are added in OBG vector.
2. Inside Boundary points:
   a. Cross boundary grids (CBG): the BGs whose points are shared by more than two clusters. CBGs contain the intersections of inside boundaries.
   b. Corner boundary grids (CoBG): the BGs are made up of one point from a cluster and three from the other.
3. Normal boundary grids (NBG): the BGs except the above three classes.

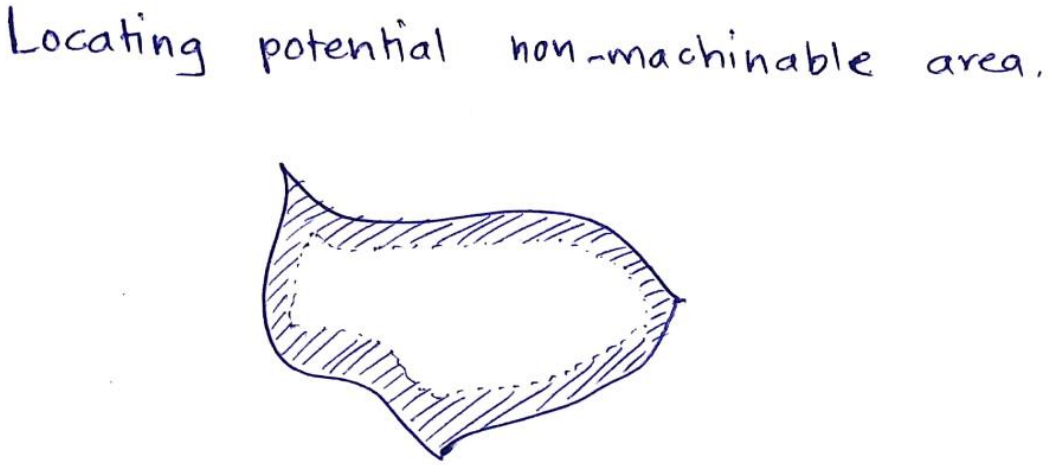
In order to find inside boundary points, for every point in domain we located closest 4 points and identified the cluster of this vicinity points. Then we used above criteria to divide them into different buckets.





Step 8:

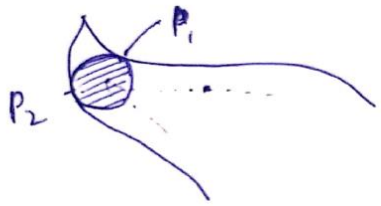
Now we try to identify the critical area which can be potential non-machinable area. This area will lie near the boundaries. For every point for chosen cluster we find the distance of boundary points from that point. This point will lie closer to the boundary only when there is at least one boundary point that lies within distance less than tool radius.



Step 9:

- The points which are tangent to the boundaries at more than two points will define the non-machinable areas, since the tool cannot proceed further due to interference.
- In order to locate such points, we take a point calculated in step 8 and two closest boundary points for that particular point are calculated. Then we measure the distance between these two closest boundary points. If this distance is more than predefined threshold L0 (given in paper), then we can say that the points lying beyond the line defined by this points will be non-machinable area.
- Essentially we are putting tool center at every point located by step 8 and we draw circle tangential to boundary points. And if for any point, circle is tangent to the boundary at more than two locations, we use this tangent points to locate non-machinable areas.

Finding actual non-machinable area



∵ In above case the tool is tangential at two points, and $|P_1 - P_1| \geq L_0$. This will given non-machinable area.

Step 10:

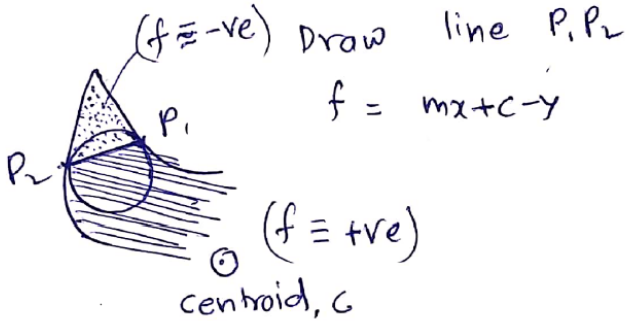
Locating the non machinable area from tangent points.

- Suppose we find that tool will be tangent to more than two points to boundaries, we can use this two tangent points to locate points in non-machinable areas.
- Now we will draw line between this tangent points. We can see that if centroid of cluster lies on one side of line then the area on the opposite side of the line will be non-machinable area.
- We will write equation of line which passes to this two points and define criteria for deciding whether any point lies on same side as the centroid of cluster or on the opposite side. For that we will put values of coordinates of centroid in equation of line and evaluate equation and whenever value of this equation gives opposite sign as that above. That point will lie in non-machinable region.

We can adjust the value of L0 to alter the how do we define the non-machinable areas. By using lower values of L0 we can capture detailed non-machinable areas.

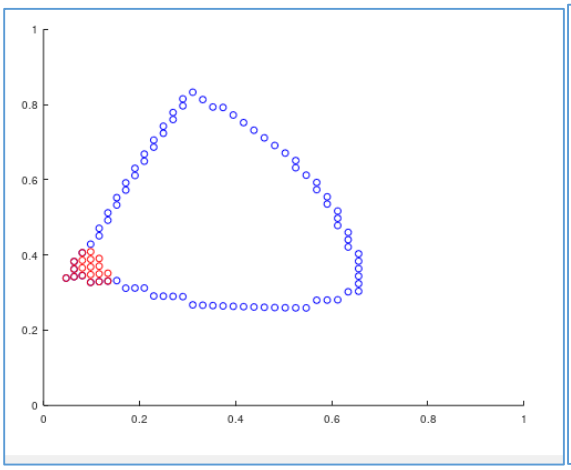
For L0=1.2*R (R is the tool radius)

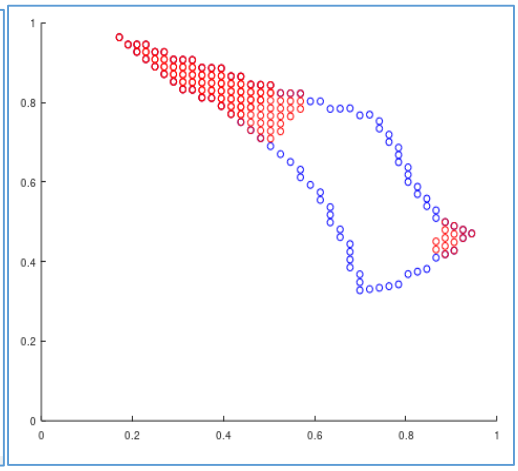
Locating points in non-machinable area

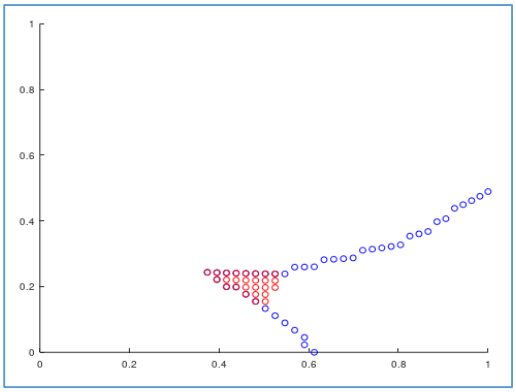


we evaluate f at centroid lets say it is positive then all the point beyond the line P_1P_2 will give -ve value of f and we can use this criteria to locate points in non-machinable region

Illustration:



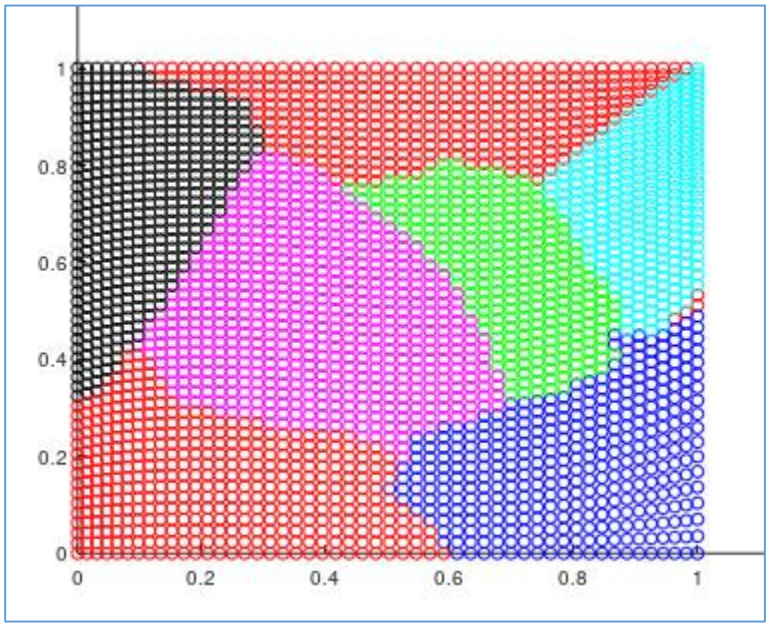




Step 11:

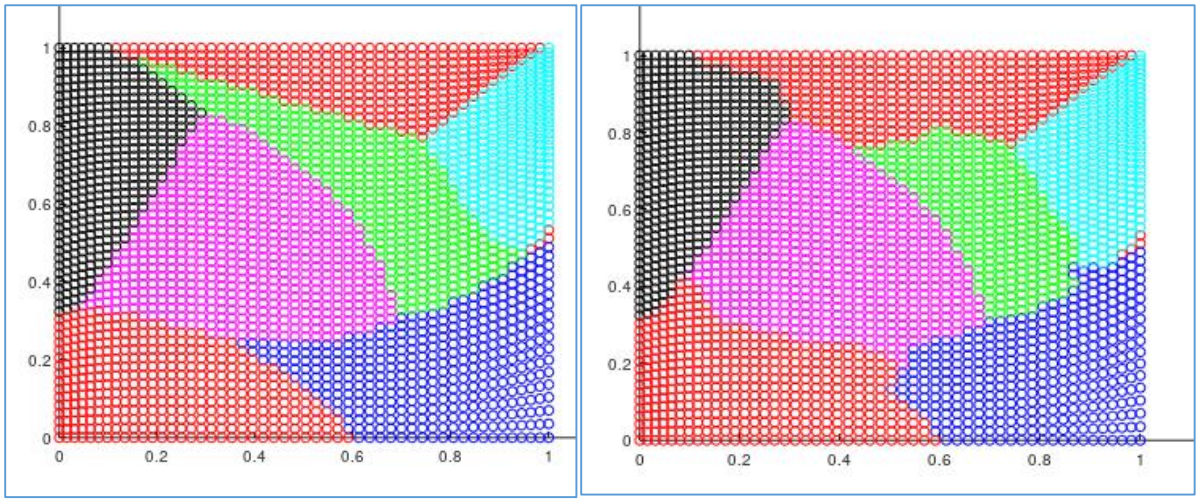
Reassigning new clusters:

Once we find the non-machinable region, we just have to see to which cluster centroid it is closest and change the tag of that particular point with new cluster index.



RESULTS

Following figures are the comparison between the region having non-machinable area before and after the implementation of algorithm. We are focusing on green shaded area as it contain the non-machinable area.



Before **After**

We can clearly observe that the non-machinable area in the green region is merged with the red and black region adjacent to it. And which is a machinable area.

Similarly, the non-machinable area in the blue and pink region is also merged with the adjacent red region.

CONCLUSIONS

From this Computer Graphics and Product Modelling project following conclusions are drawn:

1. We have created the Bi-cubic Bezier surface and collected the information of each point and the corresponding normal at that point.
2. Subdivision of the whole surface into the sub-surfaces is done by using k-means clustering algorithm and this subdivision is verified by using in-built sklearn library in python.
3. All the boundary points are extracted and non-machinable area is defined using the tool radius.
4. These sub-surfaces are successfully rearranged such that non-machinable area is merged with the adjacent clusters to improve the machinability.
5. With this, time will be saved as each region can be machined using fixed tool orientation.

Individual contributions

Name	Contribution
Abhinandan Kumbhar 193109007	1. Implementation of Bezier surface 2. Creation of dataset for K means. 3. Feature normalization of surface points and surface normals. 4. Written code for implementation of K means clustering algorithm. 5. Determination of outside boundary points, OBG 6. Determination of inside boundary points. CBG,COBG,NBG 7. Ideated algorithm for finding boundary points and non machinable areas. 8. Locating potential Non-machinable regions. 9. Identifying tangent points which define non machinable regions. 10. Implementing algorithm to locate actual non machinable areas. 11. Redistributing the points lying in non machinable areas. 12. Minor contribution in report writing.
Mujahed Alim Patil 193109009	1. Brainstorming and idea evaluations 2. Creation of dataset for k-means clustering using pandas. 3. Used python library (sklearn) to write a code for k-means clustering. 4. Compared and matched the results from python and octave code. 5. Tried for boundary point extraction in python. 6. Organized the points extracted from SGL file along the curve in python. 7. Curve fitting of the b-spline curve through boundary points using NURBS package in octave. (bspinterpcrv) 8. Result of above step is b-spline curve but with lot of irregularity. 9. Successfully created the boundaries on region using MATLAB. 10. Contribution in report writing. 11. Contribution in PPT creation for presentation.

Harishankar 193109008	1. Surface creation using solid works. 2. Point cloud generation from the surface. 3. Edited code for the extraction of Control points for the generation of Bezier surface. 4. Contribution in creating dataset for Bezier surface. 5. Contribution in implementing K-mean Clustering algorithm. 6. Major report writing for the project. 7. Minor contribution in PPT Creating for presentation.
Rahul Pandey 193100074	1. Review of the papers and discussion. 2. Leading the group in finalisation of project based on its feasibility 3. Actively participated in different stage of project 4. Contributed in dataset creation with K mean clustering. 5. Minor help in report making 6. Completed final presentation

References:

1. Xu Liu, Yingguang Li, Qiang Li A region-based 3 + 2-axis machining toolpath generation method for freeform surface.

2. Gray JP, Ismail F, Bedi S (2007) Arc-intersect method for $3\frac{1}{2}\frac{1}{2}$ –axis toolpaths on a 5-axis machine. Int J Machine Tools Manuf 47(1): 182–190.

3. Chen ZC, Dong ZM, Vickers GW (2003) Automated surface subdivision and toolpath generation for $3\frac{1}{2}\frac{1}{2}$ –axis CNC machining of sculptured parts. Comp Ind 50(3):319–331

4. Roman A, Bedi S, Ismail F (2006) Three-half and half-axis patchby-patch NC machining of sculptured surfaces. Int J Adv Manuf Technol 29(5):524–531