World Academy of Science, Engineering and Technology
International Journal of Aerospace and Mechanical Engineering
Vol:9, No:10, 2015

Determining the Width and Depths of Cut in Milling on the Basis of a Multi-Dexel Model

Jens Friedrich, Matthias A. Gebele, Armin Lechler, Alexander Verl

***Abstract*—**Chatter vibrations and process instabilities are the most important factors limiting the productivity of the milling process. Chatter can leads to damage of the tool, the part or the machine tool. Therefore, the estimation and prediction of the process stability is very important. The process stability depends on the spindle speed, the depth of cut and the width of cut. In milling, the process conditions are defined in the NC-program. While the spindle speed is directly coded in the NC-program, the depth and width of cut are unknown. This paper presents a new simulation based approach for the prediction of the depth and width of cut of a milling process. The prediction is based on a material removal simulation with an analytically represented tool shape and a multi-dexel approach for the workpiece. The new calculation method allows the direct estimation of the depth and width of cut, which are the influencing parameters of the process stability, instead of the removed volume as existing approaches do. The knowledge can be used to predict the stability of new, unknown parts. Moreover with an additional vibration sensor, the stability lobe diagram of a milling process can be estimated and improved based on the estimated depth and width of cut.

***Keywords*—**Dexel, process stability, material removal, milling.

I. Introduction

IN the modern machining industry, there exists an increasing need for flexible and customized production, as the customer is asking for individual products and mass customization. To tackle this need, in the European project Cassa Mobile [1] a modular production system in a container is realized. The production system consists of several production modules for additive manufacturing, milling, assembly and cleaning. The production system can be configured for different use-cases. Independent from the configuration and the use-case, the production of a part is fully automated. The costumer can provide CAD-data of the part via the human machine interface (HMI) and an integrated CAM-tool generates the command list for all modules. The production flow is supervised by a software workflow manager that coordinates the data flow and the production tasks for all modules. The interaction of the different software modules is described in [2]. The process from CAD-data to the final product is depicted in Fig. 1.

As the process planning and process execution is done without any user interaction, an integrated assessment of the command list and the resulting quality for each module is necessary. In this article, the focus is on the milling module. The input for the milling process is the tool path defined in a NC program. The NC program is generated using an automatic CAM system with parameter limits. Before the NC program is executed the resulting process conditions has to be estimated and evaluated. Improper process conditions will lead to an insufficient product quality or can even lead to damage of the tool, the workpiece or the milling module.

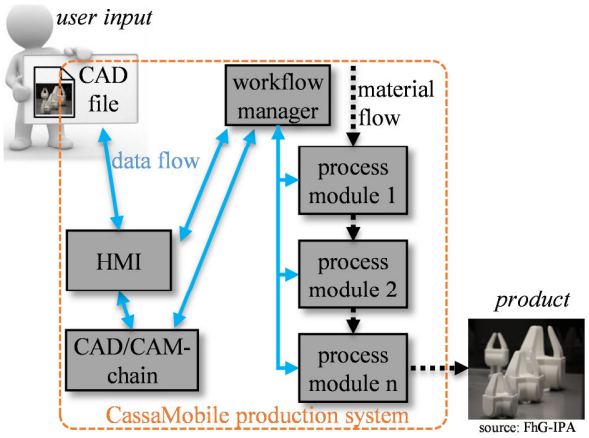

Fig. 1 CassaMobile production system

One of the most important factors limiting the productivity and quality in milling is the occurrence of chatter vibrations. Chatter vibrations are caused by the interaction of the current cutting edge with the wavy surface left by the last cutting edge. This leads to an excitation of the flexible machine tool structure. Therefor the effect can be described as a time-delayed system [3]. The rotation speed of the spindle and the number of teeth of the tool define the time delay. There exist several methods to analyze the stability of systems with time delay [4], [5]. For the milling process the stability is typically illustrated in a stability lobe diagram (SLD). In the SLD the critical depth of cut a_{cr}, separating stable and unstable cutting conditions, is drawn for different spindle speeds [6]. Fig. 2 shows a SLD calculated using the full discretization method as described in [5].

The SLD can be generated in several ways. Additional to the mathematical calculation of the stability of the time-delayed system as described above, there exist experimental approaches for the estimation of the SLD. Some additional sensors are necessary to assess the process behavior. Typically, accelerometers at the spindle housing are used to

J. Friedrich, M. A. Gebele and A. Lechler are with the Institute for Control Engineering of Machine Tools and Manufacturing Units (ISW), University of Stuttgart, Stuttgart, Germany (Corresponding author: J. Friedrich; Phone: 0049-711-68582416; Fax: 0049-711-68572416; e-mail: jens.friedrich@isw.uni-stuttgart.de).

A. Verl is with the Executive Board of Fraunhofer research organization, Munich, Germany.

identify the vibrations. Performing cuts with an increasing depth of cut for different spindle speeds allows the estimation of the maximum stable depth of cut for all spindle speeds [7]. A similar approach is to cut with constant depth of cut and changing spindle speed. The vibration signal can be analyzed and the stable spindle speeds for each depth of cut can be identified [8].

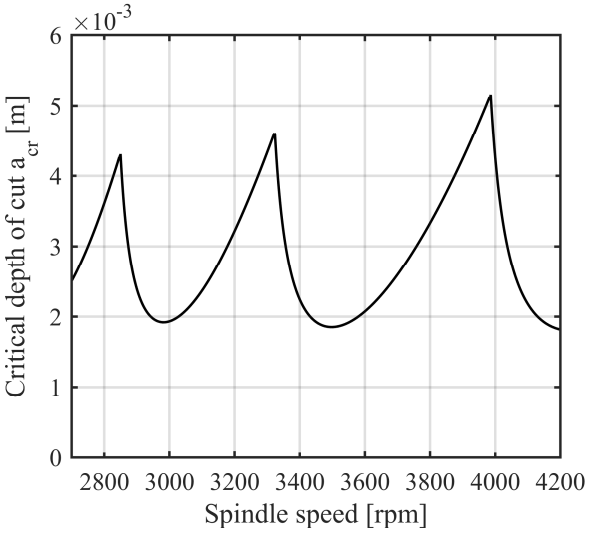

Fig. 2 Stability lobe diagram (SLD)

The SLD allows the selection of the spindle speed with the maximum depth of cut. The critical depth of cut a_{cr} also depends on the width of cut and therefore the SLD is changing for different width of cut. Budak [9] published an approach to reach maximum productivity based on finding optimal pairs of width and depth of cut [9]. Thus the two dimensional SLDs are only suitable to select the optimal depth of cut for one width of cut.

The process stability is depending on the cutting depth, cutting width and the spindle speed. So these three parameters are necessary to estimate the SLD from cutting tests. On the other hand, with an existing SLD the process stability can be predicted based on the cutting depth, the cutting width and the spindle speed. As mentioned above the typical input for a milling process is a NC program, which defines the tool path and the spindle speed. The spindle speed can directly extracted from the NC program, but the cutting depth and the cutting width are unknown, as they depend on the raw part geometry, the tool geometry and the path already travelled. Existing material removal simulations are developed for collision detection and visualization. The depth and width of cut are not calculated for each step. Due to the fact that the axial and radial cutting depths a_p and a_e are essential for the prediction of the process stability, this paper describes a new simulation based method for the computation of the cutting depths and cutting width.

II. Geometrical Modelling and Simulation

As in [10], material removal simulations can be used to review the tool path in the milling process, to determine unintended collisions and to predict process factors such as forces, temperatures etc. The simulation system presented in this paper concentrates on the geometrical aspect of material removal and therefore only includes models of the tool and the workpiece. Design and application of these models are further explained in the following sections. All other components of the milling system are assumed to be collision free.

For material removal simulation, the intersection of the tool and the workpiece has to be calculated and evaluated. Therefor a representation of the workpiece and the tool is necessary. Surmann, for example, analyzes the intersecting volume of cutter and workpiece that are represented by constructive solid geometries [11].

Moreover, research on the basis of discretized volume models has been done. The most frequently used discrete volume approximation methods are the voxel, the dexel and the octree models. Volumes represented by a voxel model consist of a three dimensional array of evenly sized cubes or cuboids which can either be inside or outside of the approximated volume. Simulations making use of these voxel models are described in [12] and [13]. In contrast to that, the octree model divides the cuboid around the volume that is represented into eight sub-cuboids. Every sub-cuboid is then similarly divided further if this increases the approximation quality [12], [14]. Dexel models make use of evenly spaced, parallel lines inside the volume.

In the approach described in this article an analytically described tool geometry intersects with a multi-dexel model of the workpiece.

A. Workpiece

The dexel model, as first mentioned by Van Hook [15], uses depth elements, so-called “dexels“, for the discretized representation of volume data. These dexels are parallel line segments, located over a regular grid. An approximation of the volume is given by the start and end points of these dexels, as they represent exact points on the volume surface. In milling simulation, workpiece models with high accuracy, good performance and acceptable memory requirements are mandatory. Given that the dexel model meets all these requirements and is furthermore easy to visualize [10], an enhanced type of dexel model is implemented. In order to further improve the approximation accuracy, a multi-dexel volume with three dexel volumes, perpendicular to each other, is used. This leads to significantly better results when intersecting with the tool envelope, as bodies of revolution are represented substantially better [16]. The workpiece geometry at the beginning of the n-th intersection is hereinafter referred to as $\mathbb{W}_n$.

B. Tool

While the recurring material removal makes it necessary to consider all previous cuts when calculating the workpiece shape, the cutter geometry only varies when the tool is changed. As described by Martellotti, it is furthermore assumed that the cutting edges follow a circular tooth path

[17]. On that account, the tool envelope can be represented by a combination of simple bodies of revolution. Therefore, the tool envelope can be described using analytical equations.

As depicted in Fig. 3, the envelope is parametrized using three of the seven general geometry parameters introduced in [18], namely the valid cutting edge height from tool tip H, tool diameter D and radius R. Depending on the parameter values, cylindrical, ball and bull nose end mill shapes can be defined.

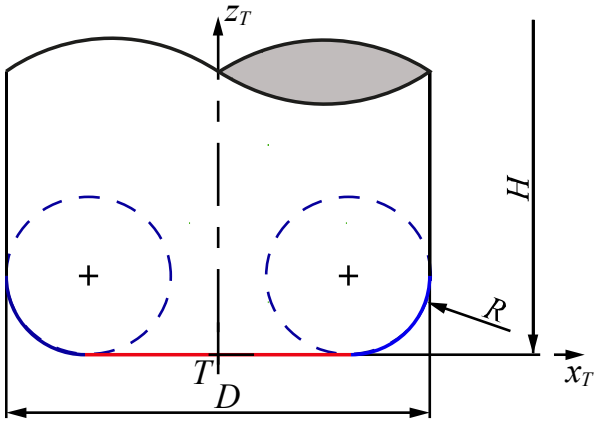

Fig. 3 Tool envelope geometry parameters

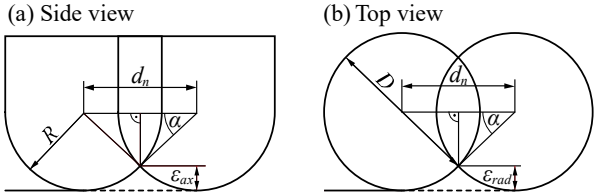

Fig. 4 Calculating the tool step as a function of the undercut error ε

The tool envelope $\mathbb{F}$ consists of a cylinder surface, a partial toroidal surface and two circular areas at the top and the bottom to close the volume. These areas are defined in the irrational tool coordinate system T using the three introduced parameters.

C. Material Removal

In order to determine the cutting depths, the simulation system requires information about the position of the tool relative to the workpiece. For this purpose, an import and parsing function for standardized NC programs generates the tool path data.

The material removal procedure consists of three steps:

1) Intersection of workpiece model and tool envelope at the current position
2) Calculation of the new workpiece shape
3) Relocation of the tool on the path by adding a set distance

Steps 1) to 3) are then repeated until the tool reaches its final position.

The intersection points of the workpiece dexels with the tool envelope calculated in the first step define the new workpiece shape that is valid after step two. In order to move the tool along the predefined tool path in the last step, a discretization step d_n needs to be set. Based on [16], this tool step is calculated considering a user-defined maximum value for the undercut error ε as well as the tool geometry. As depicted in Fig. 4, the undercut error occurs as a consequence of the tool path discretization.

For all tool geometries considered it holds that $R \leq D/2$, which leads to $\varepsilon_{ax} \leq \varepsilon_{rad}$. Based on the principles of trigonometry illustrated in Fig. 4, the tool step d_n is defined in (1) as a function of the tool diameter D and the undercut error ε.

$$d_n = D * \cos\left(\arcsin\left(1 - \frac{2\varepsilon_{max}}{D}\right)\right) \tag{1}$$

At the end of a path segment, the tool step is furthermore reduced such that the tool reaches the exact final position according to the NC code.

D. Calculating the Depths of Cut

The computation of the engagement conditions requires an examination of the particular tool envelope surface that is inside the workpiece volume. For this purpose, Fig. 5 (a) illustrates a typical scenario in the end milling process. The axial cutting depth a_p is calculated using two points $Q_1, Q_2 \in \mathbb{Q}_n$ that are on the envelope surface and inside the workpiece volume, such that $\mathbb{Q}_n = \mathbb{W}_n \cap \mathbb{F}_n$. It is furthermore essential that Q_1 is the one, or one of the particular points of $\mathbb{Q}_n$ with the lowest axial distance l_1 to the origin T of the tool coordinate system. Analogously, Q_2 is the point with the highest axial distance l_2. Distance l_i is obtained by projecting vector $\overrightarrow{TQ_i}$ on the vector $\vec{a}_{ax}$ that represents the tool axis. Consequently, the points are defined by (2) and (3):

$$Q_1 = Q_i \Leftrightarrow l_i = \min\left(\frac{\overrightarrow{TQ_i} \bullet \vec{a}_{ax}}{|\vec{a}_{ax}|}\right) \forall i \tag{2}$$

$$Q_2 = Q_i \Leftrightarrow l_i = \max\left(\frac{\overrightarrow{TQ_i} \bullet \vec{a}_{ax}}{|\vec{a}_{ax}|}\right) \forall i\,. \tag{3}$$

as in (4), the axial cutting depth is then computed as the difference between l_2 and l_1.

$$a_p = l_2 - l_1 \tag{4}$$

Fig. 5 (b) illustrates the same end milling scenario in top view. For further steps, another vector $\vec{r}_{proj}$, perpendicular to the tool axis $\vec{a}_{ax}$ and the feed direction $\vec{v}$ is introduced, such that $\vec{r}_{proj} = \vec{a}_{ax} \times \vec{v}$. The radial cutting depth a_e is calculated using two points P_1 and $P_2 \in \mathbb{Q}_n$. In analogy to the computation of a_p, the set $\mathbb{Q}_n$ is scanned for vectors $\overrightarrow{TP_i}$, whose projection p_i on $\vec{r}_{proj}$ takes on minimum and maximum values. Hence, the points are defined by (5) and (6):

$$P_1 = P_i \Leftrightarrow p_i = \min\left(\frac{\vec{r}_{proj} \bullet \overrightarrow{TP_i}}{|\vec{r}_{proj}|}\right) \forall i \tag{5}$$

$$P_2 = P_i \Leftrightarrow p_i = \max\left(\frac{\vec{r}_{proj} \bullet \overrightarrow{TP_i}}{|\vec{r}_{proj}|}\right) \forall i\,. \tag{6}$$

In order to avoid negative cutting depth values as a result of varying feed directions, the radial cutting depth is defined in

(7) as the absolute value of the difference between the lengths p_2 and p_1.

$$a_e = |p_2 - p_1| \tag{7}$$

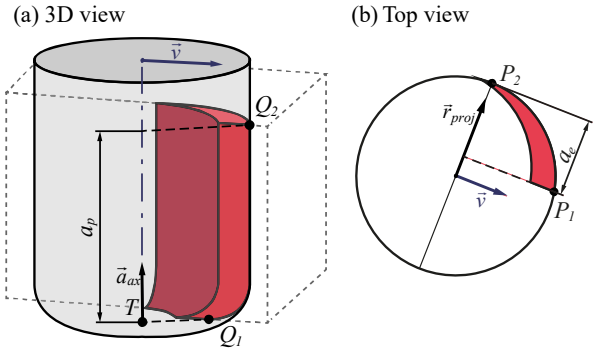

Fig. 5 Computing engagement conditions

III. Implementation

In this simulation system, material removal is done by subtracting the tool volume from the workpiece volume. In doing so, the system evaluates those parts of the tool envelope that are inside the workpiece dexel model.

A. Layout of the Workpiece Dexel Model

Prior to the simulation, all dexel volumes of the multi-dexel model must be initialized according to the workpiece dimensions and the discretization step δ. The layout of these dexel volumes is a further developed version of the dexel data set introduced in [19]. In this paper, the data layout described by König and Gröller was improved in order to make it applicable to multi-dexel volume models. As in [19], every dexel volume DF is defined as a quadruplet $(Res, L, M, \mathbb{D})$, such that:

- $Res = (Res_{x1}, Res_{x2}) \in \mathbb{N}^2$ defines the resolution of the dexel volume in x_1- and x_2-directions
- $L \in \mathbb{R}^3$ is the position of the dexel volume in world coordinates
- M defines the dexel direction
- $\mathbb{D}$ is initialized as a set of $Res_{x1} * Res_{x2}$ dexels $dex_{i,j}$

Direction M and location L of these dexel volumes are initialized, so that they overlap each other entirely and their dexel directions are perpendicular. As depicted in Fig. 6, one dexel $dex_{i,j}$ can be divided into two or more dexel elements $dex_{i,j,k}$. Therefore, every dexel element is defined as follows:

- beg specifies the M-value of the Cartesian distance from the dexel element beginning to the dexel volume origin
- end analogously specifies the M-distance from the dexel element end to the dexel volume origin
- Position Pos of a dexel element point in workpiece coordinates is defined in (8) and (9).

$$Pos.x_1 = i * \delta \quad Pos.x_2 = j * \delta \tag{8}$$

$$Pos.M = beg \quad \text{or} \quad Pos.M = end \tag{9}$$

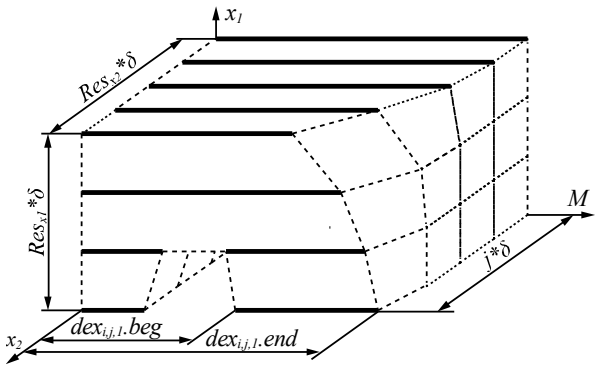

Fig. 6 Layout of a dexel data set

All dexels of a dexel volume are stored in a two dimensional array, such that the dexels can be located in workpiece coordinates using the array indices i, j and the discretization step δ. Since the number of dexel elements varies during the simulation, every array element of the dexel volume consists of a size-variable float-vector-container whose entries carry the beg- and end-values of the dexel elements. The use of float values guarantees a datatype precision of 7 to 8 significant digits, which is equivalent to a precision of 0.1 µm for workpiece dimensions smaller than 1000 mm. For ultra-high precision calculations it is recommended to use the double datatype which, however, doubles the memory requirements and may also decrease computation performance.

B. Finding Possibly Affected Dexels

While subtracting the tool from the workpiece, every dexel element must be checked for collision with the tool envelope and, if necessary, it has to be shortened, divided or deleted. This collision detection is based on the principle of ray casting. For every vertex of the dexel grid, an algorithm calculates all intersection points of a ray casted in dexel direction, with the tool envelope surface. Fig. 7 illustrates this procedure inside a transparent workpiece. Tool-workpiece collisions are determined by comparing the ray intersection points with the values of all dexel elements on this particular ray. Due to the fact, that in most end milling situations the tool volume is considerably smaller than the workpiece volume, the selection of rays to be casted is based on the projection of an axis aligned bounding box (AABB) around the tool on the $M = 0$ plane. These projection criteria, illustrated as grey shadows in Fig. 7, reduce the number of rays to be casted enormously and thus significantly decrease computation time.

C. Calculation of Ray-Tool Intersections

The intersection point calculation of the rays with the tool envelope is performed in the tool coordinate system. Therefore, all rays $\vec{s}$ must be transformed to tool coordinates. Due to the fact, that the tool coordinate system is not rotating, the transformation is limited to a translation, as performed in (10) using the example of rays in x direction.

$$\vec{s}_{x,T} = \delta \begin{bmatrix} 0 \\ i \\ j \end{bmatrix} + \begin{bmatrix} \lambda \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ T.y \\ T.z \end{bmatrix} \quad (10)$$

Both real intersection points P_{beg} and P_{end} are identified by inserting these ray-vectors into the equations describing the tool envelope and solving them for λ. In doing so, the algorithm has to make use of the appropriate surface equation, depending on the position and the direction of the rays.

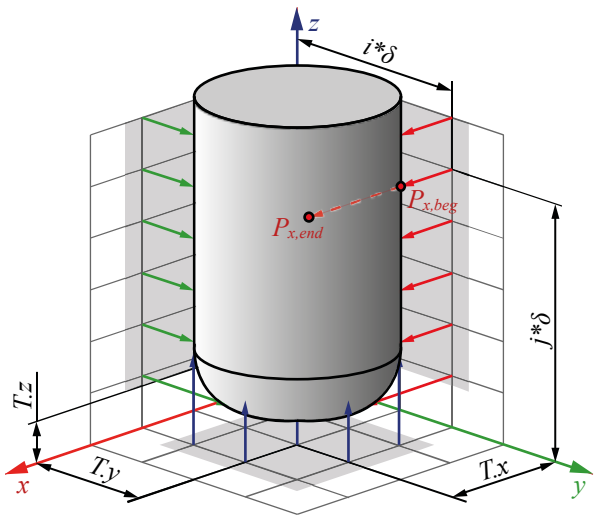

Fig. 7 Detecting relevant envelope points by casting rays in x-, y- and z-directions

Subsequently, the intersection points must be transformed back to workpiece coordinates in order to compare them with the correspondent dexel elements. The depth-values of these intersection points represent the tool in workpiece coordinates and are therefore hereinafter referred to as $tool_{i,j}.beg$ and $tool_{i,j}.end$.

D. Cutting the Dexel

Every valid tool value is now compared with the corresponding dexel element values so as to find all tool-workpiece intersections. The five possible intersections described in [19] are extended and adjusted to fit the requirements of the multi-dexel model used in this simulation system. The criteria for the six different intersection scenarios depicted in Fig. 8 are defined as follows:

a $dex_{i,j,k}.end \leq tool_{i,j}.beg$: No intersection found.

b $dex_{i,j,k}.beg < tool_{i,j}.beg < dex_{i,j,k}.end \leq tool_{i,j}.end$: The dexel element end must be shortened.

c $tool_{i,j}.beg \leq dex_{i,j,k}.beg < tool_{i,j}.end < dex_{i,j,k}.end$: The dexel element beginning must be shortened.

d $tool_{i,j}.end \leq dex_{i,j,k}.beg$: No intersection found.

e $dex_{i,j,k}.beg < tool_{i,j}.beg < tool_{i,j}.end < dex_{i,j,k}.end$: The tool cuts the dexel element into two pieces, which is why a new dexel element $dex_{i,j,k+1}$ must be generated. The newly created dexel element adopts the end-value of the original dexel element.

f $tool_{i,j}.beg \leq dex_{i,j,k}.beg < dex_{i,j,k}.end \leq tool_{i,j}.end$: Dexel element $dex_{i,j,k}$ must be deleted.

In case of a dexel element being shortened or cut into two pieces, the dexel element values adopt the correspondent tool values.

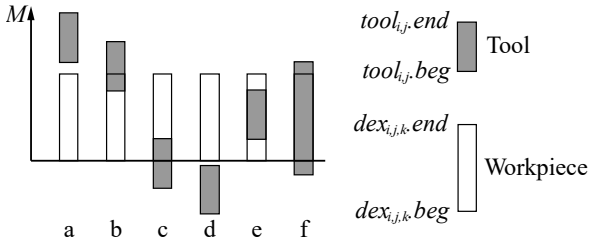

Fig. 8 Possible intersections of tool and workpiece dexels ([19], modified)

All dexel element points that are newly created or modified during the n-th cut must be part of the tool envelope as well as the workpiece and thus represent the set $\mathbb{Q}_n$. This set of points is now used to calculate the engagement conditions.

E. Axial and Radial Depths of Cut

As the simulation system presented focuses on 3-axis end milling, the computation of the axial cutting depth is reduced to a comparison of z-values, since the lengths l_i are equal to the z-value of the observed points Q_i. During the cutting procedure, l_1 and l_2 are identified by comparing and finding the highest respectively lowest value $Q_i.z$. These lengths are then used to determine the axial depth of cut, as defined in (4).

The computation of the radial cutting depth is realized as previously described. The lengths p_1 and p_2 are determined according to (5) and (6) during the cutting procedure and afterwards used to calculate the radial depth of cut, as defined in (7). When the cutting process is done, the determined cutting depths as well as the position and other NC code-related information are stored in a list for further use.

F. Software and Graphical User-Interface

The algorithm is implemented in a software tool with a user-friendly graphical user interface. The software tool is realized in C++ with Qt [20]. The software tool calculates the cutting depth and cutting width based on the NC-program. In the first step the NC-program is parsed and the programmed tool path is extracted. As described above the tool path is discretized and for each step the cutting depth and cutting width is calculated. For a fast analysis the result can be aggregated and for each NC-program line the maximum and average depth and width of cut can be displayed as shown in Fig. 9.

Moreover a graphical representation of the tool path and the resulting workpiece geometry is displayed using a splatting technology in OpenGL. The NC-program lines and the tool path are color coded and highlighted depending on the depth of cut. This allows a fast identification of critical cutting conditions in terms of process stability. Fig. 10 shows the 3D-view of the resulting workpiece and the color coded toolpath.

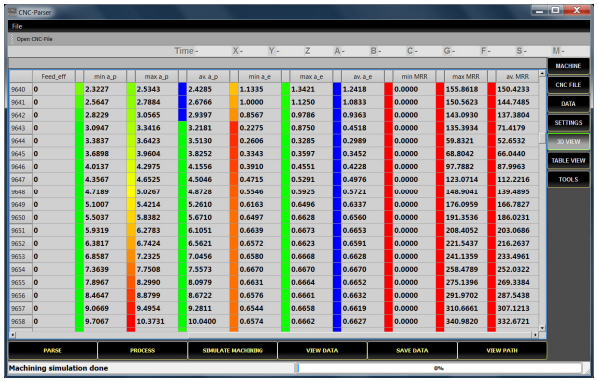

Fig. 9 List of Depth and width of cut of an NC-program

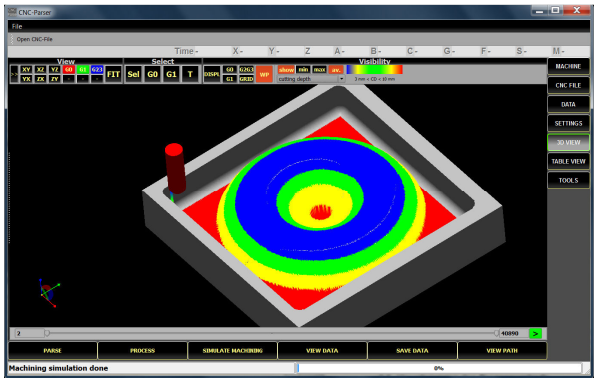

Fig. 10 3D-view of the workpiece geometry and the tool path

Even the algorithm was not developed for real-time application, test on a normal desktop PC showed, that the calculation and the visualization is performed faster than the program execution on the milling machine.

IV. Conclusion and Outlook

In this paper a new simulative approach for the calculation of the depth and width of cut in milling is presented. The cutting depth and width are, together with the spindle speed, the influencing parameters of the process stability in milling. The presented approach extracts the tool path from the NC-program. An analytically described tool shape is intersected with a multi-dexel workpiece model along this discretized tool path. For each step the depth and width of cut is calculated. A software tool with an intuitive user interface was developed to calculate and illustrates the result. The result is illustrated in a model of the resulting tool path and workpiece geometry. Moreover a table for all steps and for each NC-program line with the maximum depth and width of cut is generated. To allow a fast analysis and easy identification of the critical regions, the tool path and the program lines are color coded based on the calculated depth of cut.

The new approach allows the identification of stability lobe diagrams during milling, based on measured sensor signals. Moreover, with a known SLD, the process stability of new NC-program can be predicted before the production.

In future research the software tool will be connected to the milling machine; this will allow merging the measured sensor signals with the real axis positions. The automatic extraction of the SLD should be integrated in the software tool. Moreover an automatic optimization of the NC-program based on the predicted process stability is possible.

Acknowledgment

Authors thank to Niko Croon, for the support of the implementation. The research presented in this paper was done within the project Cassa Mobile (Project reference: 609146) funded by the seventh framework program of the European Commission.

References

[1] Flexible mini-factory for local and customized production in a container, www.CassaMobile.EU, 2013.
[2] J. Friedrich, S. Scheifele, A. Verl, and A. Lechler, “Flexible and Modular Control and Manufacturing System,” in Procedia CIRP: Elsevier, 2014.
[3] S. A. Tobias and W. Fishwick, “Theory of regenerative machine tool chatters,” The engineer, vol. 205, pp. 199–203, 1958.
[4] T. Insperger and G. Stépán, “Semi-discretization method for delayed systems,” International Journal for numerical methods in engineering, vol. 55, pp. 503–518, 2002.
[5] Y. Ding, L. Zhu, X. Zhang, and H. Ding, “A full-discretization method for prediction of milling stability,” International Journal of Machine Tools and Manufacture, vol. 50, pp. 502–509, 2010.
[6] Y. Altintaş and E. Budak, “Analytical prediction of stability lobes in milling,” CIRP Annals-Manufacturing Technology, vol. 44, pp. 357–362, 1995.
[7] G. Quintana, J. Ciurana, and D. Teixidor, “A new experimental methodology for identification of stability lobes diagram in milling operations,” International Journal of Machine Tools and Manufacture, vol. 48, pp. 1637–1645, 2008.
[8] C. A. Suprock, B. K. Fussell, R. B. Jerard, and R. Z. Hassan, “A low cost wireless tool tip vibration sensor for milling,” ASME MSEC, 2008.
[9] E. Budak and A. Tekeli, “Maximizing chatter free material removal rate in milling through optimal selection of axial and radial depth of cut pairs,” CIRP Annals-Manufacturing Technology, vol. 54, pp. 353–356, 2005.
[10] M. Stautner, Simulation und Optimierung der mehrachsigen Fräsbearbeitung. Essen. Vulkan-Verl., 2006.
[11] T. Surmann, Geometrisch-physikalische Simulation der Prozessdynamik für das fünfachsige Fräsen von Freiformflächen. Essen. Vulkan-Verl., 2006.
[12] Z. Hou, Beitrag zur voxelbasierten Simulation des fünfachsigen NC-Fräsens. Stuttgart. Fraunhofer-IRB-Verl., 2003.
[13] D. Jang, K. Kim, and J. Jung, “Voxel-Based Virtual Multi-Axis Machining,” The International Journal of Advanced Manufacturing Technology, vol. 16, pp. 709–713, 2000.
[14] P. Brunet, “Solid representation and operation using extended octrees,” ACM Trans. Graph., vol. 9, pp. 170–197, 1990.
[15] T. van Hook, “Real-time shaded NC milling display,” in the 13th annual conference. D. C. Evans, R. J. Athay, Eds., pp. 15–20.
[16] M. O. Benouamer and D. Michelucci, “Bridging the gap between CSG and Brep via a triple ray representation,” in the fourth ACM symposium. C. Hoffmann, W. Bronsvoort, G. Allen, M. Pratt, D. Rosen, Eds., pp. 68–79.
[17] M. E. Martellotti, “An Analysis of the Milling Process,” in Transactions of the A.S.M.E., vol. 63, pp. 677–700.
[18] S. Engin and Y. Altintas, “Mechanics and dynamics of general milling cutters,” International Journal of Machine Tools and Manufacture, vol. 41, pp. 2195–2212, 2001.
[19] A. H. Konig and E. Groller, “Real time simulation and visualization of NC milling processes for inhomogeneous materials on low-end graphics hardware,” in Computer Graphics International, pp. 338–349.
[20] Qt Cross-platform application & UI development framework, http://www.qt.io/, 2015.