A REPORT
ON

Modelling of Chatter Vibration in Thin-wall Milling and its Effect on Surface Roughness of Job

BY

Deep Patel	*2018A4PS0526P*
Rudhir Mehra	*2018A4PS0523P*
Utkarsh Anand	*2018A4PS0539P*
Saptarshi Das	*2018A4PS0535P*
R Harikrishna	*2018A4PS0560P*
Himanshu Sharma	*2018A4PS0556P*

Prepared in partial fulfillment of the
Production Techniques 2, Course No. **ME F313**
at

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI
(First Semester 2020-21)

ACKNOWLEDGEMENT

We would like to thank the instructor in-charge for Production techniques II, Prof T.C. Bera for giving us the opportunity to learn and work under his guidance on this interesting project and helping us throughout the course of this project without any hesitation.

We would also like to thank BITS Pilani administration and the BITS Pilani Library management for helping us during the course of this project.

In the end, we would like to thank our parents and close ones for helping and supporting us throughout the pandemic situation, ensuring we face no obstacles during this tough time.

TABLE OF CONTENTS

1. Introduction

The aim of this project is to model the chatter vibration on a thin wall milling and explore its effect on surface roughness for the same using the help of computer softwares such as MATLAB. Chatter vibration has been modelled here analogous to a spring-mass-damper system considering two degrees of freedom.

A lot of research papers were read and consulted to find out the relation between cutting parameters and vibration amplitudes, and the relation between vibration amplitude and surface roughness. The model we have chosen in our case and represented through MATLAB takes the axial depth of cut and the rotation speed (in rpm) as input and gives X-direction and Y-direction as output to the user.

In this report, we have described certain essential points related to this project and the working/ algorithm of our code and have attached the code in Appendix.

2. Chatter Vibration

Machining vibrations, technically known as chatter, refer to the relative movement between the workpiece and the cutting tool. These vibrations result in a wave-like pattern on the machined surface. This affects many of the typical machining processes, such as milling, turning and drilling, and some atypical machining processes, such as grinding. A chatter mark is an irregular surface flaw left by a wheel that is out of true in grinding or regular mark left when turning a long piece on a lathe, due to machining vibrations.

3. Chatter Modelling

Generally, the chatter is modelled by solving the equations of motion in the frequency and/or time domain where the cutting forces and some modal dynamic parameters at the cutting tool tip are used. The frequency domain solution is mainly used when the milling cutting forces show linear behaviour. This is very typical in case of conventional milling. Micro-milling is characterised by the use of cutting tools with small diameters (e.g. diameters between 10 mm and 1 mm) and small feed rates (a few microns per tooth). The micro milling cutting forces exhibit nonlinear behaviour at small uncut chip thicknesses and feed rates, respectively. This nonlinearity of the cutting forces can be justified with the help of the fact that at small uncut chip thicknesses the workpiece material is subjected to shearing and ploughing cutting with chip formation.

The boundary between the ploughing and chip formation phenomena is known as a minimum chip thickness, and it depends on the size of the cutting tool edge radius, feed rate, workpiece material, cutting velocity, tool-workpiece interaction, etc. The run-out effect and the velocity dependency of the cutting forces also introduce nonlinearities. Since the micro-milling cutting forces are nonlinear, the equation of motion must be solved in the time domain using numerical methods for integrating the ordinary differential equation of motion. By using the assumption that the helix angle of the micro-milling tool is negligible, the micro-milling system can be reduced to a 2-Degree of Freedom model.

3.1 Analogy with spring mass model

The flexible tool model we have considered is assumed to be a spring-mass system with dampers having 2 degrees of freedom. For the sake of simplicity the work-piece is considered to be rigid.The 2 DOF-model of flexible multi-teeth milling cutter works with simultaneous engagement. Simultaneous engagement refers to the engagement of more than one cutting edge of the milling cutter at a time. We assume that the cutting tool has straight flutes for simplicity in the modeling of cutting forces. The radial and tangential cutting forces (Fr, Ft) acting on a tooth can be expressed using the cutting coefficients (Kr,Kt) and chip width (h). The material dependent cutting coefficients can be found using the standard machining data handbooks .

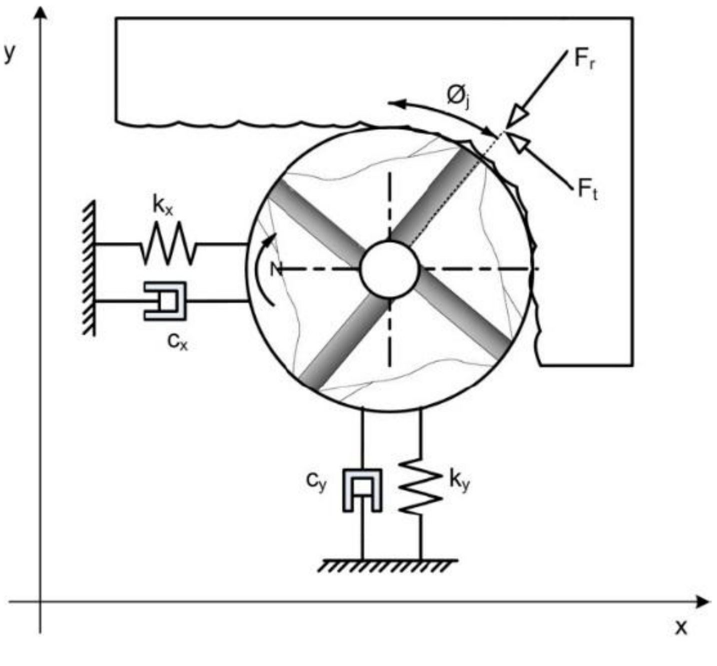


Fig 1: Milling analogous to spring-mass model

The model in discussion considers self-excited regenerative mechanism as the principal source of chatter vibrations. The regenerative chatter is a phenomenon associated with the modulation of chip thickness due to feedback between subsequent tool passes. Modulation of chip thickness results in oscillatory cutting force which in turn causes a periodic oscillatory motion of the workpiece tool system which can be characterized by different modes of vibration. It is generally considered that only the modes of vibration perpendicular to the cutter axis are dominant and hence chatter in milling can be expressed as 2-Degree of Freedom delay differential equation.

3.2 Formulation of problem

The flexible tool model is considered to be a spring mass-system with dampers having 2 degrees of freedom.We consider the workpiece to be rigid for simplicity of modelling. The 2-DOF-model of flexible multi-teeth milling cutter works with simultaneous engagement. Simultaneous engagement refers to engagement of more than one cutting edge of the milling cutter at any given time. We assume the tool to have straight flutes for simplicity in the modeling of cutting forces. The radial and tangential cutting forces (F_r, F_t) acting on the cutting tooth can be expressed using the cutting coefficients (K_t, K_r) and chip width (h). The material dependent cutting coefficients can be found using the standard machining data handbooks .

In local coordinates the forces (F_r, F_t) can be expressed as:

$$\begin{Bmatrix} F_t \\ F_r \end{Bmatrix}_j = \begin{bmatrix} K_t\, h(\theta)\, a \\ K_t\, k_r\, h(\theta)\, a \end{bmatrix}_j = \begin{bmatrix} K_t\, h(\theta)\, a \\ F_t\, k_r \end{bmatrix}_j \qquad (1)$$

where θ is the instantaneous angle made by the rotating end mill tooth (z-axis) referenced with respect to y axis and 'a' is the axial depth of cut (ADOC).

Transforming Eqn. 1 to global coordinates results in resolution of forces in the X and Y directions as shown below:

$$\begin{Bmatrix} F_x \\ F_y \end{Bmatrix}_j = \begin{bmatrix} -sin(\theta) & -cos(\theta) \\ -cos(\theta) & sin(\theta) \end{bmatrix}_j \begin{Bmatrix} F_r \\ F_t \end{Bmatrix}_j \qquad (2)$$

The parameters which affect the cutting tool dynamics of the end milling cutter are stiffness, mass and the damping factor. The periodical vibrating force is induced by the regenerative effect of the tool. The equations of motion of the flexible tool system consisting of Z cutting edges is given by-

$$\ddot{x} + 2\zeta_x \omega_{nx}\dot{x} + \omega_{nx}^2 x = \frac{1}{m_x}\sum_{j=0}^{Z-1} F_{xj} = \frac{1}{m_x} F_x(t)$$

$$\ddot{y} + 2\zeta_y \omega_{ny}\dot{y} + \omega_{ny}^2 y = \frac{1}{m_y}\sum_{j=0}^{Z-1} F_{xj} = \frac{1}{m_y} F_y(t)$$

(3) and (4)

where m is the modal mass, ω_n is the natural angular frequency, ξ is the damping ratio and F is the cutting force. Depending on the immersion angle of the cut, the workpiece may get engaged by multiple tooth of the Z toothed cutter at the same time. Therefore, the forces on every tooth have to be calculated and summed up to consider the effect of all the teeth engaged in the cut as shown in Eqns. 3 & 4.

Evaluation of instantaneous forces requires the calculation of chip thickness. The instantaneous chip thickness $h(\theta)$ due to regenerative effect is computed as follows. In case of a smooth surface, the chip thickness is given by :

$$h_s = c \sin(\theta) \qquad (5)$$

where c is the chip load or feed per tooth. Since the chip thickness is affected by the feedback effect of the previous pass (t-T), it can be expressed in local coordinates u-v (u: radial direction) as follows:

$$h_d = h_s + u(t - T) - u(t) \qquad (6)$$

where T = 60 / (NZ) is the tooth passing period and N is the cutter speed in minutes. Though it is possible to use the h_d directly to compute forces, for the sake of convenience it is expressed in the x-y global coordinate system using the transformation matrix.

$$u(t) = -x(t)\sin\theta - y(t)\cos(\theta) \qquad (7)$$

$$u(t - T) = -x(t - T)\sin\theta - y(t - T)\cos\theta \qquad (8)$$

Substituting in the Eqn. 6 for h_d

$$h_d = \{h_s + [x(t) - x(t-T)]\sin\theta_i + [y(t) - y(t-T)]\cos\theta_i\} \qquad (9)$$

In machining with end-mills, the tool edge need not necessarily engage with the workpiece at all times. The tool engagement time is dependent on the angle of entry and exit. This produces intermittent cutting operations. To consider the effect of intermittent cutting action of the tooth of the milling cutter, a unit step function $g(\theta_i)$ is multiplied with the above equation. We have to note that at any given point in time, depending on the number of tool edges and the entry and exit angle, more than one cutting edge could be involved in the cutting operation.

$$g(\theta) = \begin{cases} 1 & \theta_{entry} < \theta < \theta_{exit} \\ 0 & otherwise \end{cases} \qquad (10)$$

The angular position of the cutting edge at any given instant t can be found out with the relation:

$$\theta(t) = \frac{2\pi Nt}{60} + \frac{Z2\pi}{N} \qquad (11)$$

Using Eqn. 12, the instantaneous position of the tool edge can be calculated. If it lies within the entry and exit angle of the work, forces can be obtained using Eqn. 2. Otherwise forces experienced from that edge will be zero according to Eqn. 11. Substituting for h_d in the Eqn. 1, the forces F_x and F_y can be calculated using eqn.2. Afterwards, equations 3 & 4 can be solved numerically using the R-K method.

3.3 Regenerative effect

Machine tool chatter arises due to forced and self-excited vibrations. This effect arises from a periodic modulation of the uncut chip thickness within the frequencies of the eigenmodes, which results in a critical excitation in the consecutive cuts or tooth engagements.

The researcher Tlusty proved in one of his papers that the regenerative effect due to feedback effects of workpiece geometry from the previous tool pass is the major source of machine tool chatter. The associated change in chip thickness results in a varying dynamic cutting force resulting in a wavy surface accompanied with vibration which eventually leads to unstable machining process.

Due to high damping capacity of the system the variance of surface height and force per feed reduces, the variation in the depth of cut also reduces leading to lesser chatter.

In order to model a real milling process, a technique modelling the forced vibration and the regenerative effect was adopted. This model is made in order to show the mode coupling due to the regenerative effect.

4. MATLAB simulation

The system of nonlinear delayed differential Eq. (3 & 4), does not have an analytical solution; a numerical resolution was used. The numerical calculus is done with MATLAB software, using the dde23 routine (delay differential equations). The scalar relative error tolerance parameter is 1e-3 and the absolute error tolerance is 1e-6.

4.1 Solving of Delay Differential Equations

In order to solve a higher order differential equation, we have to first express it as first order simultaneous differential equations. A state space form which is easy to implement in MATLAB software is used here. For the chatter equations (Eqns. 3 & 4), the state variables can be expressed as given below:

$$x_1 = x$$
$$x_2 = \dot{x}_1 \qquad (12)$$

and the simultaneous differential equations can be expressed as:

$$\dot{x}_1 = x_2 \text{ and}$$
$$\dot{x}_2 = \frac{1}{m}(F - cx_2 - kx_1) \qquad (13 \text{ and } 14)$$

Runge-Kutta method is the most popular method which is used to solve DDEs. However, when used with constant step size it might lead to poor error control. Thus, modern solvers normally use variable step size, even though choosing a proper step size without increasing computation burden is a very important issue. Moreover it must be chosen to get an accurate approximation. As DDE with variable step size requires solution at previous times other than mesh points, interpolation of results is necessary. Cubic hermite interpolants are commonly used in DDE solvers to obtain the solution everywhere in the interval. Here, the solution for DDE is obtained using DDE23 which is proposed by Shampine [7]. However DDE23 is not suitable for solving variable time delay problems which occur as a result of varying speed and varying pitch.

In this section the implementation of chatter equations for DDE23 solver is explained using its standard interface. The dde23 function call is as follows

sol=dde23('ChtrddeFunc', [time delay], 'ChtrHistory', [tintial, tfinal], options, state); (15)

where,

1) *ChtrddeFunc* : a handle to function that defines the state space form of a delay differential equation.
2) *[Time delay]*: time delay which is equal to T according to Eqn. 7.
3) *ChtrHistory*: a handle to function for solution history
4) *[T_{intial} t_{final}]*: the time span of integration which is user defined (Generally 0-3 seconds)
5) *Options*: used for reentry
6) *State*: variable passed to trigger the start of a new integration.

The dde23 solver returns the solution 'sol' which is a structure consisting of time, solution 'y' and its derivative. The dde23 solver updates time 't' automatically and the adaptive increment to parameter 't' is chosen by the dde23.

4.2 Algorithm

The initial step in solving chatter equations consists of implementing the equations mentioned in section 2 as a function '*chtrddefunction*'. The function comprises the chatter equations (Eqns. 17-20) expressed in state space form. In order to calculate the forces at any given time, 't' the angular position of the tooth (θ) has to be found out first. Using equation.12 and looping through 1 to Z number of teeth, the instantaneous positions of all the teeth can be computed. Even though the initial position can be at any random angle, for the sake of simplicity the initial position is considered to coincide with the positive Y axis. When the location angle 'θ' lies within the 'θ_{entry}' and 'θ_{exit}' angles, the forces F_x, F_y are computed ,otherwise they are set to zero. This sub-procedure is made as a loop and all the forces are summed according to equations 3 & 4.

The chip thickness is calculated using equation 10. The history of previous values at (t-T) is provided by the YL lag values which are passed to this function as an input parameter according to the syntax.

Table 1: Data for MATLAB simulation

Natural freq. ω_{nx}, ω_{ny} (Hz)	600, 660
Damping ratio ξ_x, ξ_y	0.035, 0.035
Stiffness k_x, k_y (N/m)	$5600*10^3$
Number of teeth	4
K_t (MPa), K_r	600, 0.07
Milling type	Half immersion

Defining history of chatter

The next step is to define the history function. In most of the chatter problems the initial displacement profile is assumed to be ‘0’. Using this approach, it is possible to express the initial profile and its derivatives using the history function.

v = ChtrHistory (t, state) (16)

where ‘v’ is the vector of state variables computed for time ‘t'. We use the history definition only during the first iteration and for subsequent integration histories; the values from DDE solution (YL lag values) are used.

Restarting integration after each tooth pass

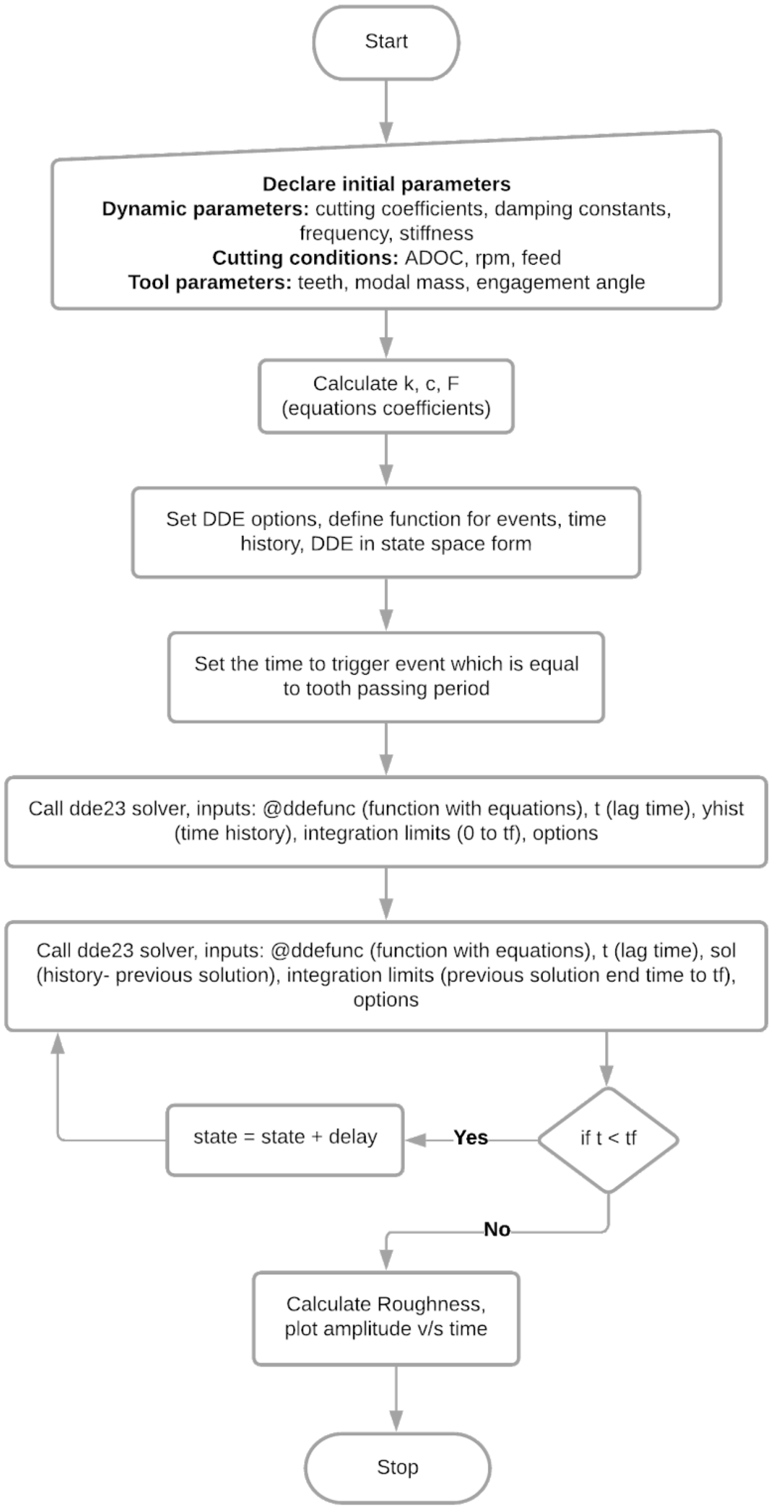
The next step focuses on the restarting of the integration. It is not possible to continue the integration of DDE indefinitely after every tooth passing period as the newly engaged tooth will encounter the profile left by the previous tooth. Therefore the solution for the current and newly engaged tooth has to take into account the displacement profile left by the previous tooth and its derivative. We can model this using the event options of DDE23 through the ddeset command.

options = ddeset(‘Event’,'ChtrEvents'); (17)

where ‘*ChtrEvents*’ is a handle to the function which is similar to ‘*Chtrddefunction*’ function. This function compares the state value with the instantaneous time ‘t’ and when they become equal, an event is triggered to stop the current integration.

In order to obtain a useful result, the numerical integration process has to be executed for a certain time span set by the user. During this time, the dde23 solver is called repetitively in a while loop at time intervals equal to the tooth passing period. In each iteration, the parameter ‘state’ is incremented in integer increments of time delay period. Thus the events function triggers the integration to stop when the state values equals the tooth passing period and the solution is placed in the sol variable. Comparing the time ‘t’ from the solution structure with the given max time span the execution of the while loop can be controlled. For all the subsequent calls made by dde23 solver, the solution returned by the previous call is passed as history. The initial value of the time span is updated with the final time value available at the end of solution structure. The summary is shown in figure 2.

4.3 Control flow diagram


Start
Declare initial parameters
Dynamic parameters: cutting coefficients, damping constants, frequency, stiffness
Cutting conditions: ADOC, rpm, feed
Tool parameters: teeth, modal mass, engagement angle
Calculate k, c, F (equations coefficients)
Set DDE options, define function for events, time history, DDE in state space form
Set the time to trigger event which is equal to tooth passing period
Call dde23 solver, inputs: @ddefunc (function with equations), t (lag time), yhist (time history), integration limits (0 to tf), options
Call dde23 solver, inputs: @ddefunc (function with equations), t (lag time), sol (history- previous solution), integration limits (previous solution end time to tf), options
if t < tf
Yes
state = state + delay
No
Calculate Roughness, plot amplitude v/s time
Stop

5. Surface Roughness

Surface roughness, often referred to as roughness, is an integral component of surface texture. It is quantified by the deviations in the direction of the normal vector of a real surface from its ideal form. The surface can be said to be smooth or rough depending on how large the deviations are.

Roughness plays a vital role in determining how a real object will interact with its environment. Roughness is often considered to be a good indicator of the performance of a mechanical component, since irregularities on the surface can lead to the formation of nucleation sites for corrosion or cracks.

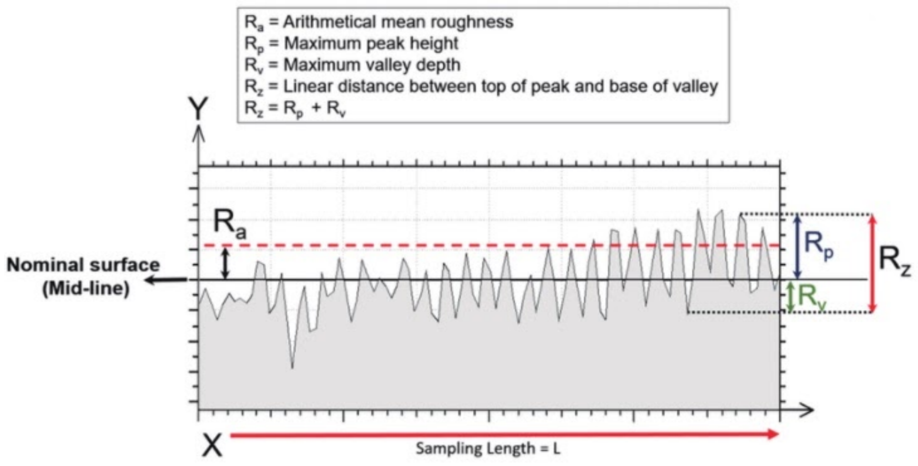

R_a = Arithmetical mean roughness
R_p = Maximum peak height
R_v = Maximum valley depth
R_z = Linear distance between top of peak and base of valley
$R_z = R_p + R_v$
Y
R_a
Nominal surface (Mid-line)
R_p
R_v
R_z
X
Sampling Length = L

Fig 2: Representation of different surface parameters

Surface Roughness parameters

R_a - It is the arithmetical average value of all absolute distances of the roughness profile from the centre line. It is known as arithmetic mean roughness.

R_t - It is the total height of the profile ie. the vertical distance between the highest peak and the deepest valley.

R_q - it is the root mean square variation of the profile deviations along the sampling length.

Surface roughness in our case was finally calculated using formula given below:

$$\mathrm{Ra} = \frac{1}{L}\int_0^L |Y(x)|\,\mathrm{d}x,$$

where,

Ra is the arithmetic average deviation from the mean line,

L is the sampling length, Y is the ordinate of the profile curve.

6. Results

Our results seem to agree with the results obtained in the theory part. The vibration profile shown below is for 3mm ADOC and 2000 rpm. The sudden valleys and peaks occurring in the graph are due to superposition of two different modes of vibration. These modes of vibration are obtained in seguy et.al. by Ansys simulation.

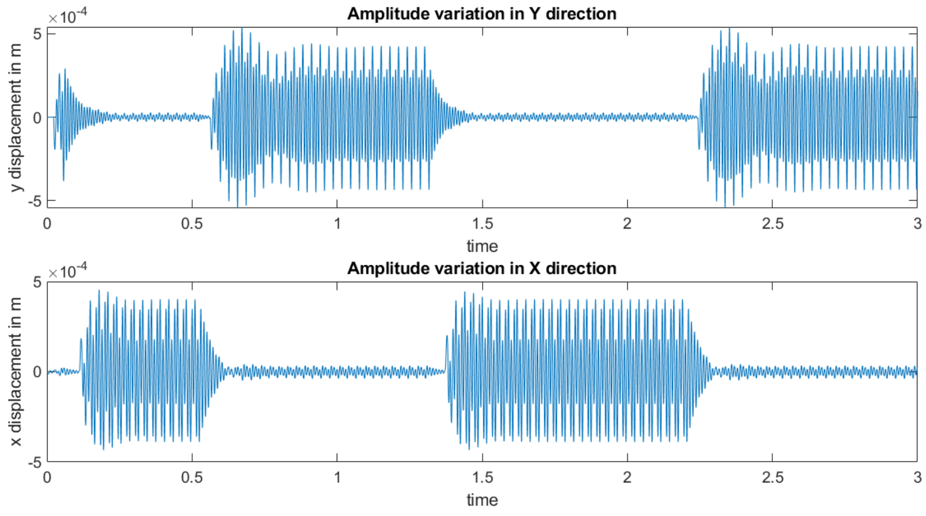


Fig 3: Amplitude profiles of chatter vibration

The proposed method is suitable to solve problems with an initial profile. The procedures developed in this study are helpful for the study of non-linear chatter problems in high speed machining. The tool displacement profiles along x and y provide inputs for constructing bifurcation diagrams, roughness values and the relevance of onset of chatter can be analyzed. Further, it is possible to analyze the chatter conditions according to frequency spectrum computed from the displacement profile. A clear peak frequency corresponding to the tool path frequency indicates chatter free machining process. With simple modification the proposed method can be extended to analyze chatter in the turning process.

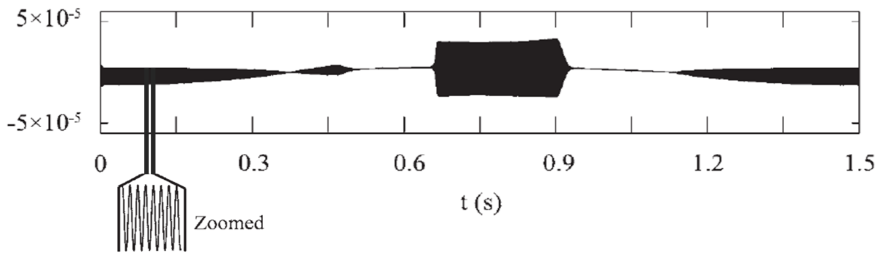


Fig 4: Plot of surface roughness vs. time

References

[1] Seguy, Sébastien, et al. ‘Surface Roughness Variation of Thin Wall Milling, Related to Modal Interactions’. *International Journal of Machine Tools and Manufacture*, vol. 48, no. 3–4, Jan. 2008, pp. 261–74. *HAL Archives Ouvertes*

[2] Shepard, Jonathan. ‘Chatter Simulation and Detection in CNC Milling’. *Master's Theses and Capstones*, Jan. 2017

[3] Insperger, Tamás, et al. ‘Machine Tool Chatter and Surface Location Error in Milling Processes’. *Journal of Manufacturing Science and Engineering*, vol. 128, no. 4, Nov. 2006, pp. 913–20

[4] Yue, Caixu, et al. ‘A Review of Chatter Vibration Research in Milling’. *Chinese Journal of Aeronautics*, vol. 32, no. 2, Feb. 2019, pp. 215–42

[5] Bolsunovskiy, S., et al. ‘Thin-Walled Part Machining Process Parameters Optimization Based on Finite-Element Modeling of Workpiece Vibrations’. *Procedia CIRP*, vol. 8, 2013, pp. 276–80

[6] Rusinek, Rafal, and Kazimierz Zaleski. ‘Dynamics of Thin-Walled Element Milling Expressed by Recurrence Analysis’. *Meccanica*, vol. 51, no. 6, June 2016, pp. 1275–86

[7] L.F. Shampine and S. Thomson“Solving DDEs in MATLAB,” Applied Numerical Mathematics, 37, pp. 441–458, 2001

[8] H. Schulz and T. Moriwaki, “High-speed Machining”, Annals of the ClRP, Vol. 41,pp.637-643,1992

Appendix

MATLAB code for chatter vibration simulation:

Variable Declaration

```
global omega_nx omega_ny xi_x xi_y Kt Kr kx Ky a N Z T theta time
theta_entry theta_exit mx my fz f
global f1 c1 k1 f2 c2 k2 f1_sum f2_sum state

Kt = 600; % radial cutting coefficient (from data book) in MPa
Kr = 0.07; % tangential cutting coefficient (from data book)
xi_x = 0.035; % damping ratio in x dirn
xi_y = 0.035; % damping ratio in y dirn
omega_nx = 600; % natural frequency in Hz (x component)
omega_ny = 660; % natural frequency in Hz (y component)
kx = 5600*10^3; % stiffness in N/m
Ky = 5600*10^3; % stiffness in N/m

mx = 10; % modal mass
my = 10;

a = 3; % axial depth of cut, ADOC in mm

N = 2000; % rpm
Z = 4; % no. of teeth on cutter
T = 60/(N*Z); % tooth passing period
f = 1; % feed
fz = f/Z; % feed per tooth

lags = [T T]; % time lags for DDE equal to tooth passing period

tf = 3; % total time of simulation

theta = 0; % instantaneous theta initialized
theta_entry = 0 *pi/180; % in rads
theta_exit = 90 *pi/180; % in rads
x=0;
y=0;
```

```
state = 0; % represents time in steps of tooth passing time
```

First calculation

```
f1_sum = 0;
f2_sum = 0;

for tooth = 1:Z

     theta = 2*pi*N*state/60 + Z*2*pi/N;

     engagement = g(theta); % accounts for tooth engagement
     f1 = ( Kt*Kr*a*(-sin(theta)) + Kt*a*(-cos(theta)) )*engagement;
     f2 = ( Kt*Kr*a*(-cos(theta)) + Kt*a*(sin(theta)) )*engagement;

     f1_sum = f1 + f1_sum;
     f2_sum = f2 + f2_sum;

end

c1 = -2*xi_x*omega_nx;
k1 = -omega_nx^2;

c2 = -2*xi_y*omega_ny;
k2 = -omega_ny^2;


options = ddeset('Event',@ChtrEvents); % custom trigger event is set
state = state + T;
sol = dde23(@ddefunc, lags, @yhist, [0 tf], options);
```

Simulation Calculations Till Final Time

```
while sol.x(end) < tf
     state = state + T;
     f1_sum = 0;
     f2_sum = 0;

     for tooth = 1:Z

     theta = 2*pi*N*state/60 + (tooth-1)*2*pi/Z;

     engagement = g(theta); % accounts for tooth engagement
     f1 = ( Kt*Kr*a*(-sin(theta)) + Kt*a*(-cos(theta)) )*engagement;
     f2 = ( Kt*Kr*a*(-cos(theta)) + Kt*a*(sin(theta)) )*engagement;

     f1_sum = f1 + f1_sum;
     f2_sum = f2 + f2_sum;

    end

     c1 = -2*xi_x*omega_nx;
     k1 = -omega_nx^2;

     c2 = -2*xi_y*omega_ny;
     k2 = -omega_ny^2;

     % calculating the solution
     sol = dde23(@ddefunc, lags, sol, [sol.x(end) tf], options);

end
```

Roughness Calculations

```
Ray = 0; % avg. roughness y dirn
for i = 1:length(sol.x)
     itr = i-1;
     if itr == 0
     prevTime = 0;
     else
     prevTime = sol.x(1,itr);
     end
     Ray = Ray + abs(sol.y(1,i))*( sol.x(1,i) - prevTime )/tf ;
end

fprintf('Average Roughness in Y direction = %f mm\n',Ray*1000);
```

```
Average Roughness in Y direction = 0.117743 mm
```

```

Rax = 0; % avg. roughness x dirn
for i = 1:length(sol.x)
     itr = i-1;
     if itr == 0
     prevTime = 0;
     else
     prevTime = sol.x(1,itr);
     end
     Rax = Rax + abs(sol.y(3,i))*( sol.x(1,i) - prevTime )/tf ;
end

fprintf('Average Roughness in X direction = %f mm\n',Rax*1000);
```

```
Average Roughness in X direction = 0.100177 mm
```

Amplitude plots- Separate graphs

```
figure
plot(sol.x,sol.y(1,:));
title('Vibration profile');
% set(gca,'FontSize', 12);
ylabel('y displacement in mm');
xlabel('time');

figure
plot(sol.x,sol.y(3,:));
title('Vibration profile');
% set(gca,'FontSize', 12);
ylabel('x displacement in mm');
xlabel('time');
```

Amplitude Plots- Combined Graphs

```
figure
subplot(2,1,1)
plot(sol.x,sol.y(1,:),'g');
title('Amplitude variation in Y direction')
ylabel('y displacement in mm');
xlabel('time');

subplot(2,1,2)
% plot(t, x, 'red');
plot(sol.x,sol.y(3,:),'r');
title('Amplitude variation in X direction')
ylabel('x displacement in mm');
xlabel('time');
```

Function declarations

```
function yp = ddefunc(t, y, YL)
     % DDE expression are written here
     global theta c1 k1 f1_sum fz mx c2 k2 f2_sum my

     yl1 = YL(:,1); % lag on y
     yl2 = YL(:,2); % lag on x
     y1 = y(1);
     y2 = y(2);
     x1 = y(3);
     x2 = y(4);

     yp = [ y2;
           c2*y2 + k2*y1 + f2_sum*fz*sin(theta)/my +
f2_sum*sin(theta)/my*(x1 - yl2(2)) + f2_sum*cos(theta)/my*(y1 -
yl1(1)) ;
           x2;
           c1*x2 + k1*x1 + f1_sum*fz*sin(theta)/mx +
f1_sum*sin(theta)/mx*(x1 - yl2(2)) + f1_sum*cos(theta)/mx*(y1 -
yl1(1)) ] ;
end

function y = yhist(t)
```

```
     % history values for first iteration only
     y = [0 0 0 0]';
end

function [value,isterminal,direction] = ChtrEvents(t,y,YL)
     % triggering event for integration to stop
     global state
     value = [state - t; state - t; state - t; state - t] ;
     isterminal = [1; 1; 1; 1];
     direction = [0; 0; 0; 0];

end

function intermittent_check = g(theta)
     % unit step function to account for tooth engagement
     global theta_entry theta_exit

     if (theta_entry < mod(theta,2*pi)) && (mod(theta,2*pi) <
theta_exit)
     intermittent_check = 1;
     else
     intermittent_check = 0;
     end
end
```