



JOAO VIVAS CISALPINO

**CONTROLE DE VELOCIDADE EM IMPRESSORAS 3D
UTILIZANDO O PRINCÍPIO DE FEEDFORWARD PARA
OTIMIZAR A VELOCIDADE E QUALIDADE DE IMPRESSÃO.**

LAVRAS - MG

2023

JOAO VIVAS CISALPINO

**CONTROLE DE VELOCIDADE EM IMPRESSORAS 3D UTILIZANDO O
PRINCÍPIO DE FEEDFORWARD PARA OTIMIZAR A VELOCIDADE E
QUALIDADE DE IMPRESSÃO.**

Monografia apresentada à Universidade Federal
de Lavras, como parte das exigências Curso de
Engenharia Mecânica, para a obtenção do título
de Bacharel.

Prof. Dr. Wander Gustavo Rocha Vieira
Orientador

LAVRAS - MG

2023

**Ficha catalográfica elaborada pela Coordenadoria de Processos Técnicos
da Biblioteca Universitária da UFLA**

Cisalpino, Joao Vivas Cisalpino

CONTROLE DE VELOCIDADE EM IMPRESSORAS
3D UTILIZANDO O PRINCÍPIO DE FEEDFORWARD
PARA OTIMIZAR A VELOCIDADE E QUALIDADE DE
IMPRESSÃO. / Joao Vivas Cisalpino. 1^a ed. rev., atual. e
ampl. – Lavras : UFLA, 2023.

34 p. : il.

Trabalho de conclusão de curso(Graduação)–Universidade
Federal de Lavras, 2023.

Orientador: Prof. Dr. Wander Gustavo Rocha Vieira.
Bibliografia.

1. TCC

CDD-808.066

JOAO VIVAS CISALPINO

**CONTROLE DE VELOCIDADE EM IMPRESSORAS 3D UTILIZANDO O
PRINCÍPIO DE FEEDFORWARD PARA OTIMIZAR A VELOCIDADE E
QUALIDADE DE IMPRESSÃO.**

Monografia apresentada à Universidade Federal
de Lavras, como parte das exigências Curso de
Engenharia Mecânica, para a obtenção do título
de Bacharel.

APROVADO em Data da defesa 2023.

Prof. Dr. Henrique UFLA
Prof. Dr. Belisario UFLA

Prof. Dr. Wander Gustavo Rocha Vieira
Orientador

**LAVRAS - MG
2023**

Dedicatoria

AGRADECIMENTOS

Agradecimentos

RESUMO

Palavras-chave: Palavras chave

ABSTRACT

Keywords: Keywords

LISTA DE FIGURAS

Figura 2.1 – Distribuição de uso de MA nas indústrias	13
Figura 2.2 – Número de artigos publicados sobre FDM ao longo do tempo	14
Figura 2.3 – Distribuição das pesquisas sobre FDM	15
Figura 2.4 – Comparação da resposta ao degrau e da resposta a escada	16
Figura 2.5 – Fluxograma FBF	17
Figura 2.6 – Resultados práticos da LPFBF	17
Figura 2.7 – Fluxograma RFBF	18
Figura 2.8 – Look ahead em uma grande mudança de direção	19
Figura 2.9 – Look ahead em uma pequena mudança de direção	20
Figura 2.10 – Curva de velocidades trapezoidal em uma sequência de movimentos	20
Figura 3.1 – Visualização das posições do sistema	27

LISTA DE TABELAS

Tabela 3.1 – Tabela de parâmetros opcionais da FMINCON	28
Tabela 3.2 – Tabela de parâmetros base das simulações	30
Tabela 3.3 – Tabela do parâmetro modificado das simulações	30
Tabela 3.4 – Tabela de especificações do computador	30

SUMÁRIO

1	INTRODUÇÃO	11
2	REFERENCIAL TEÓRICO	12
2.1	Manufatura Aditiva	12
2.2	<i>Fused Deposition Modeling</i>	13
2.2.1	Feedforward	15
2.2.1.1	<i>Input Shaper</i>	16
2.2.1.2	filtered basis function (FBF)	16
2.2.1.3	<i>Limited-preview filtered B-splines</i>	16
2.2.1.4	<i>Robust Filtered Basis Functions</i>	17
2.3	Geração de comando	18
2.3.1	<i>Look ahead</i>	19
2.3.2	Curvas de velocidade trapezoidal	19
2.3.3	Espaço de Estados	20
2.3.4	Integração implícita utilizando programação não linear e colocação	20
2.3.5	<i>Objective Function Optimization</i>	21
3	METODOLOGIA	23
3.1	Matlab	23
3.1.1	fmincon	23
3.2	Geração de Comando	23
3.2.1	Leitura Gcode	23
3.2.2	Velocidade de curva	24
3.2.3	Curva trapezoidal de velocidade	24
3.2.4	Interpolação	25
3.3	Modelagem dinâmica de uma impressora 3D	26
3.3.1	Espaço de estados	27
3.4	FMINCON	28
3.4.1	Configurações da função	28
3.4.2	Restrições lineares e limites de borda	28
3.4.3	Restrições não lineares	28
3.4.4	Função objetivo	28
3.5	Solução da trajetória da base	29

3.6	Descrição das simulações	29
4	RESULTADOS E DISCUSSÃO	31
4.1	Resultados	31
4.1.1	Correlações das variáveis de entrada	31
4.1.2	Influências do modelo dinâmico	31
4.1.3	Performance Computacional	31
4.2	Dificuldades	31
4.2.1	Função objetivo	31
4.2.2	Restrição de aceleração da base	31
4.2.3	Variáveis principais	31
4.2.4	Geração de Comando	31
4.3	Possíveis abordagens	31
4.3.1	Combinação com outros algoritmos	31
5	CONCLUSÃO	32
	REFERÊNCIAS	33

1 INTRODUÇÃO

texto

Entretanto, uma das grandes limitações da impressão 3D, principalmente do tipo *Fused Deposition Modeling*, é o tempo de impressão, que ainda limita muito o tamanho de peças impressas em um tempo razoável, geralmente sendo necessário reduzir muito a resolução da impressão.

Existe hoje, dentro da academia e das comunidades "faça você mesmo", uma busca por impressoras capazes de imprimir cada vez mais rápido mantendo a qualidade de impressão. Além da possível diminuição do tempo de impressão, além disso a capacidade de imprimir velozmente acaba proporcionando uma capacidade de aumentar a qualidade de impressão proporcional à diferença entre a velocidade máxima e a velocidade de impressão.

Portanto, vê-se relevante à procura por técnicas que permitem capacidades superiores de qualidade e velocidade de impressão, que flexibilizam a tecnologia e aumentam a capacidade da utilização comercial viável da tecnologia.

2 REFERENCIAL TEÓRICO

2.1 Manufatura Aditiva

O princípio básico da manufatura aditiva (MA) é a capacidade de fabricar um modelo tridimensional "diretamente", não sendo necessário o planejamento das operações de maneira individual e sim se preocupando com as configurações como um todo. O processo é calculado pelo fatiador com base em um modelo tridimensional digital, geralmente criado a partir de *Computer Aided Design* (CAD) e nas configurações do mesmo e resulta nas instruções necessárias para a máquina de manufatura aditiva construir o modelo físico. Uma das características principais da MA é a rapidez na qual é possível criar protótipo diretamente de modelos digitais, por conta disso, em um contexto de desenvolvimento de produto, o termo prototipagem rápida era utilizado. Entretanto, conforme a MA foi se aperfeiçoando era perceptível a capacidade dessas tecnologias não só se aterem à produção de protótipos, mas também de peças utilizadas em produtos finais. Além disso, o termo não considerava o princípio básico que unia essas tecnologias e assim o termo manufatura aditiva foi apresentado e adotado pela *American Society for Testing and Materials* (ASTM) GIBSON *et al.* (2015).

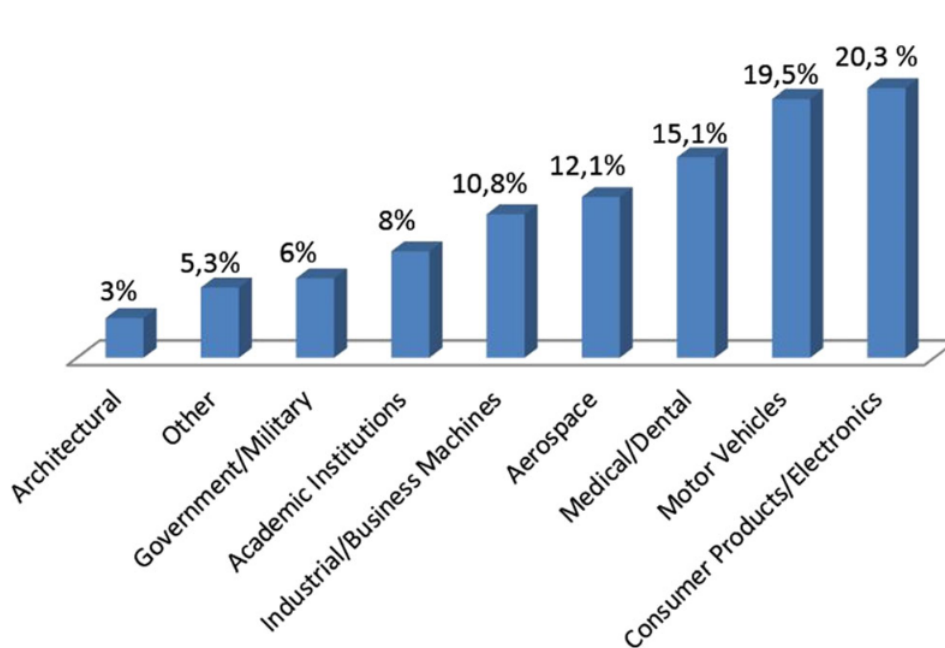
Apesar da manufatura aditiva ter sido criada a mais de 30 anos, apenas a partir de 2009, quando a última patente mais relevante de *Fused Deposition Modeling* (FDM) expirou. Com isso, vários entusiastas começaram a desenvolver essa tecnologia de uma maneira "caseira", com o forte movimento RepRap. Por conta dessa característica "caseira" e um senso de comunidade, os desenvolvimentos em sua maioria eram de caráter *Open Source* e com uma mentalidade de acessibilizar essa tecnologia para as pessoas. Com os avanços tecnológicos feitos pela comunidade, empresas, pessoas e a mídia começaram a se interessar cada vez mais, aumentando a popularidade das impressoras 3D e por consequência trazendo muita atenção para a manufatura aditiva, que a partir daí, mais pesquisas, mais empresas se interessavam em desenvolver esse tipo de tecnologia, não somente FDM (ATTARAN, 2017).

Atualmente, existe uma grande variedade de tecnologias e processos de manufatura aditiva. Estes variam na maneira com que depositam o material, nos princípios físicos que utilizam e nos materiais que podem ser utilizados. Como mencionado anteriormente, um dos métodos de manufatura aditiva mais populares é a tecnologia FDM, entretanto existem diversas outras tecnologias que tem crescido muito em popularidade como as tecnologias baseadas na cura seletiva de resinas, *stereolithography* (SLA) e *Masked stereolithography Apparatus* (MSLA),

alem de outras tecnologias menos acessíveis, mas com aplicações em diversas industrias, como por exemplo

selective laser melting (SLM) e *selective laser sintering* (SLS) (BIKAS; STAVROPOULOS; CHRYSSOLOURIS, 2016). Na figura 2.1 podemos observar a distribuição do uso de tecnologias MA por tipo de industria.

Figura 2.1 – Distribuição de uso de MA nas industrias



Fonte: BIKAS; STAVROPOULOS; CHRYSSOLOURIS, 2016

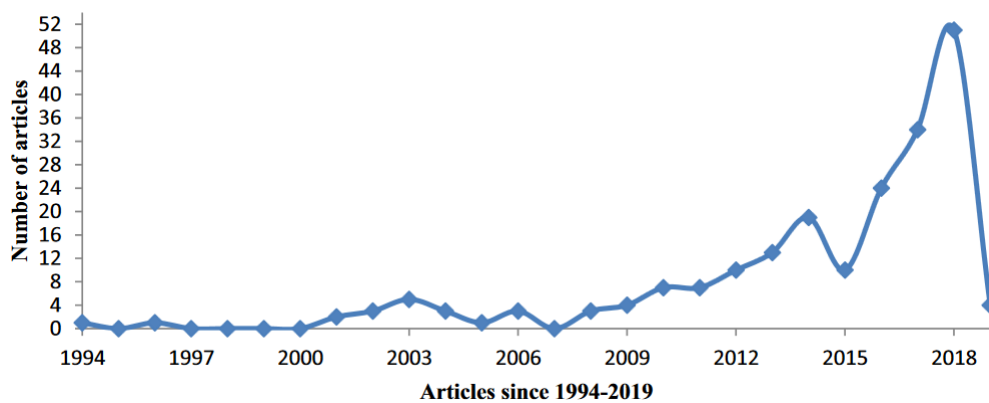
2.2 *Fused Deposition Modeling*

Fused Deposition Modeling (FDM) ou *Fused Filament Fabrication* (FFF) é uma das tecnologias MA mais populares como mencionado anteriormente. Ela se consiste por depositar material através de um processo onde um filamento de material é forçado dentro de uma câmara através, geralmente, de rolos dentados onde em uma região específica esse material é liquefeito. Por conta da pressão criada pelo filamento adentrando a câmara, ainda no estado sólido como um pistão, o material liquefeito é extrudado através de um bocal, comumente fabricado de bronze. Então, o filamento liquefeito é depositado em uma plataforma de forma a percorrer a trajetória desejada utilizando mecanismos movidos de forma controlada, geralmente por motores de passos. O processo é repetido camada por camada, de forma que elas estejam

apoiadas por camadas anteriores e a primeira camada continue fixa na plataforma ou cama, até que o processo finalize (TURNER et al., 2014) (TURNER; STRONG; GOLD, 2014).

O trabalho de (VYAVAHARE *et al.*, 2020) apresenta algumas características sobre o desenvolvimento científico sobre FDM ao longo dos anos, tendo como base 211 artigos diferentes de 1994 a 2020. É apresentado um grande salto no número de artigos publicados no tema em anos recentes (2015 a 2018) (figura 2.2), com 56% dos temas trabalhados em torno da otimização de parâmetros de impressão, acompanhado de 17% de trabalhos relacionados a aplicações utilizando o processo FDM, enquanto apenas x% são relacionados a outros temas, incluindo avanços tecnológicos relacionados a melhorias de *hardware* e *software* desses dispositivos. (Figura 2).

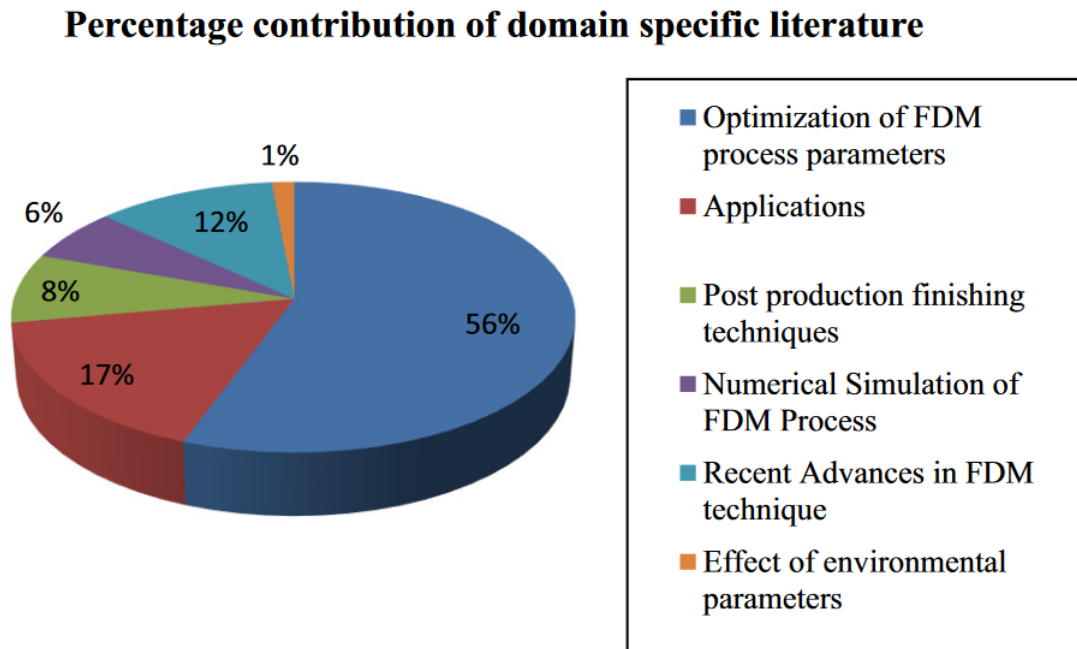
Figura 2.2 – Número de artigos publicados sobre FDM ao longo do tempo



Fonte: VYAVAHARE *et al.*, 2020

Podemos separar, de maneira simplificada, a porção de software de impressoras 3D FDM em três principais etapas: fatiamento (*slicing*), geração de comando e controle. A etapa de fatiamento envolve a topologia e a criação de instruções a partir do modelo, é nessa fase onde se decide a sequência de movimentos e outros eventos. Já na etapa de geração de comando, as instruções criadas pelo fatiador (*slicer*) na etapa anterior são interpretadas e os comandos detalhados são gerados, por exemplo as curvas de velocidade. Esses comandos são utilizados para movimentar os motores e outros equipamentos da impressora. Na etapa de controle, uma etapa relativamente nova nas impressoras 3D mais acessíveis, técnicas de controle são utilizadas para se diminuir vibrações e variações indesejadas em quaisquer parâmetros controlados, como a temperatura do bico. Um dos grandes avanços nessa etapa foi a implementação da técnica de *Input Shaping* por um *firmware Open Source* de impressora 3D chamado Klipper. Após a inclusão dessa etapa, principalmente no controle dinâmico da impressora, as capacidades

Figura 2.3 – Distribuição das pesquisas sobre FDM



Fonte: VYAVAHARE *et al.*, 2020

de velocidades e qualidade chegaram a outro patamar se comparados a impressoras que não implementam essa etapa (KLIPPER, 2017a).

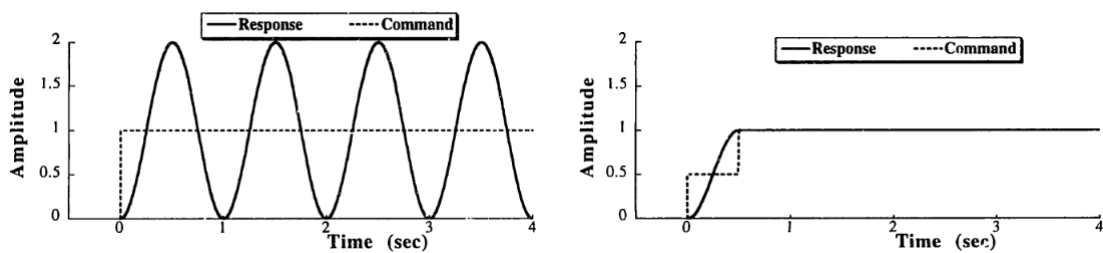
2.2.1 Feedforward

Dentre os métodos de controle em aplicações FDM o *Feedforward* é o mais eficiente dada as limitações de custo em impressoras 3D comuns e é capaz de ter um impacto maior em sistemas conhecidos e sensíveis ao erro, onde buscam corrigir o erro antes que ele aconteça. O método de *feedback* é mais eficiente em diminuir o impacto de excitações externas ou desconhecidas, entretanto não consegue prever os efeitos do sistema, se encaixando melhor em aplicações de usinagem utilizando CNCs (*Computer Numerical Control*), onde o valor dos equipamentos é maior e as forças envolvidas no corte influenciam mais do que as vibrações do próprio sistema. Já no caso das impressoras, quase 100% dos efeitos é causado pelo próprio sistema. As principais limitações da aplicação de técnicas Feedforward em impressoras 3D são a dificuldade de montar um modelo representativo, a exigência computacional elevada e por fim a necessidade da simulação se estender do início ao fim, pela dependência de se basear no estado inicial da impressão (RAMANI; EDOIMIOYA; OKWUDIRE, 2020; DUAN; YOON; OKWUDIRE, 2018).

2.2.1.1 *Input Shaper*

Conhecendo a trajetória desejada e conhecendo características do sistema é possível computar os comandos fornecidos para calcular uma série de comandos, levando em consideração as características do sistema para que o comando de referência seja modificado de forma à trajetória final ser o mais próximo possível do comando de referência. Entretanto, ao invés de computar todo o comando de referência, é possível obter um comando modificado em tempo real através de um filtro. Uma das abordagens desse tipo de filtro de comando é o *Input Shaper*, onde variados *Shapers* são construídos levando em consideração diferentes objetivos e restrições (SINGHOSE, 1997). Podemos ver na figura 2.4 um exemplo comparativo das respostas ao degrau e da função escada aplicada pelo shaper.

Figura 2.4 – Comparação da resposta ao degrau e da resposta a escada



Fonte: SINGHOSE, 1997

Essa abordagem vem sendo explorada na comunidade "faça você mesmo" a partir de 2017 quando a última patente desse método expirou, e tem aprimorado a área como um todo, empurrando os limites anteriores de velocidade e precisão, sendo popularizada pelo Klipper (KLIPPER, 2017b).

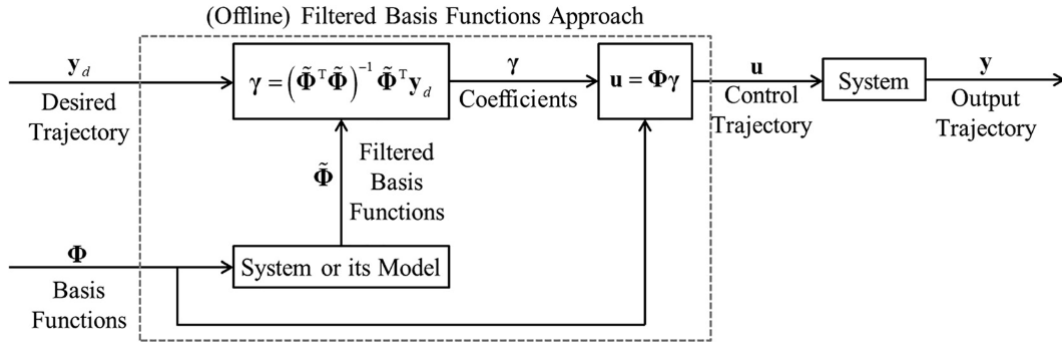
2.2.1.2 *filtered basis function (FBF)*

O método FBF necessita que a trajetória a ser rastreada seja totalmente conhecida e que a trajetória controlada possa ser expressa como uma combinação linear de funções base possuindo coeficientes desconhecidos. As funções base são utilizadas em um controle feedforward utilizando o modelo dinâmico do sistema e selecionando os coeficientes de maneira a minimizar os erros dada uma trajetória desejada (figura 2.5). (RAMANI *et al.*, 2017)

2.2.1.3 *Limited-preview filtered B-splines*

Uma das maiores dificuldades que os métodos avançados para o controle *feedforward* de trajetórias é a necessidade de se conhecer completamente a trajetória desejada, o que implica

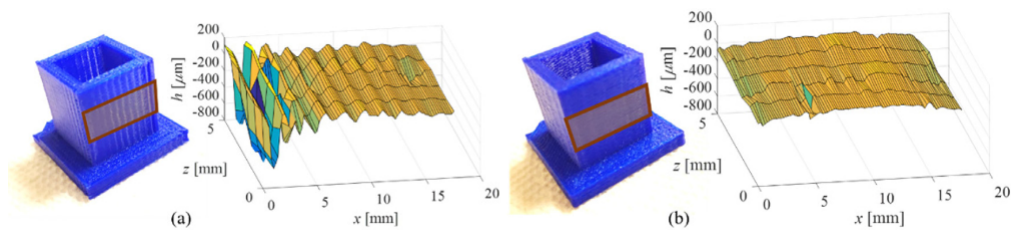
Figura 2.5 – Fluxograma FBF

Fonte: RAMANI *et al.*, 2017

em um grande custo computacional, principalmente em situações onde são necessárias uma grande quantidade de amostras da trajetória, por exemplo em casos de alta resolução e casos de longa duração. O *limited-preview filtered B-splines* divide a trajetória desejada em subgrupos com um número menor de amostras e utiliza um algoritmo de *receding horizon* para calcular recursivamente os coeficientes da função B-spline que minimizam os erros de trajetória (DUAN; YOON; OKWUDIRE, 2018).

A partir dessa otimização da divisão da trajetória em subgrupos, esse método conseguiu ser testado utilizando uma impressora 3D de verdade com modelos simples. Apresentando resultados promissores apresentados na figura 2.6.

Figura 2.6 – Resultados práticos da LPFBF



Fonte: DUAN; YOON; OKWUDIRE, 2018

2.2.1.4 Robust Filtered Basis Functions

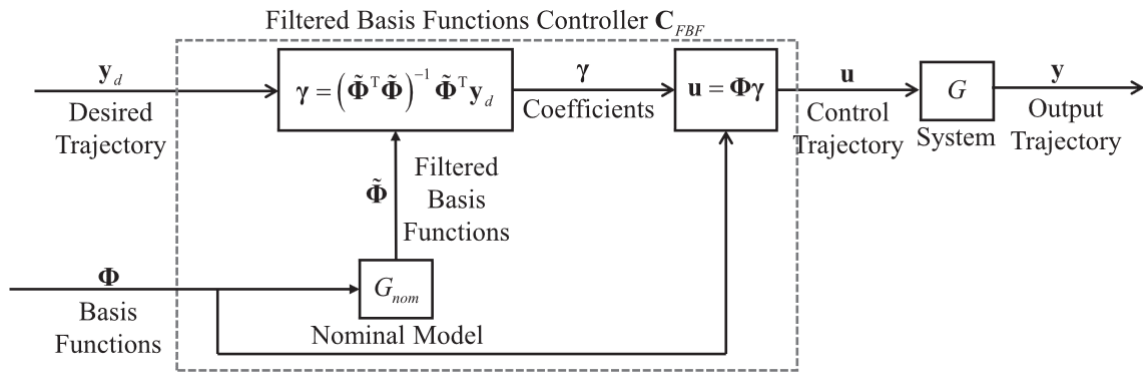
Com base nos desenvolvimentos nos trabalhos de (RAMANI *et al.*, 2017) e (DUAN; YOON; OKWUDIRE, 2018), comentados anteriormente, (RAMANI; EDOIMIOYA; OKWUDIRE, 2020) busca atacar um segundo desafio prático na implementação da FBF, sendo o primeiro desafio prático o custo computacional que foi endereçado pelo trabalho de (DUAN;

YOON; OKWUDIRE, 2018) através da LBFBF, permitindo a aplicação do algoritmo no mundo real.

Este segundo desafio se trata da degradação de precisão de rastreamento da abordagem FBF, causada por imprecisões no modelo ou incertezas na dinâmica da planta atrelado a característica do método de se utilizar puramente uma abordagem de *feedforward*. A não participação de *feedbacks* sensoriais do mundo real abre espaço para uma crescente divergência entre o modelo e a realidade.

Considerando o requisito de manter a performance computacional alcançada com o método LPFBF, (RAMANI; EDOIMIOYA; OKWUDIRE, 2020) propõe, também, a utilização de um filtro robusto em substituição à dinâmica nominal da planta para filtrar as funções de base. Esse filtro robusto é construído com base no inverso de um controlador feedforward ótimo, que minimiza uma função de custo de erro para lidar com a incerteza conhecida da planta como um filtro robusto. O esquema do método é apresentado na figura 2.7.

Figura 2.7 – Fluxograma RFBF



Fonte: RAMANI; EDOIMIOYA; OKWUDIRE, 2020

2.3 Geração de comando

A geração de comando é o processo que coordena a ativação dos atuadores, motores, dentre outros componentes de uma impressora. Ele recebe como base uma série de comandos que precisam ser interpretados e interpolados. Esse processo é responsável pelo controle de velocidade, aceleração dentre outras atividades que variam no tempo (YU *et al.*, 2020).

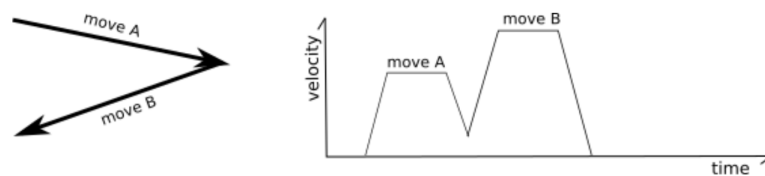
2.3.1 *Look ahead*

No processo de impressão 3D são fornecidos para a impressora uma sequência de pontos no espaço e limitações de velocidade entre os mesmos. A velocidade nos pontos é compartilhada entre trajetos em sequência, o que torna considerá-los independentemente ineficiente, introduzindo aceleração e desaceleração desnecessária impactando negativamente no tempo de impressão e na qualidade da peça impressa. O algoritmo *Look Ahead* procura manter o máximo de velocidade possível entre movimentos distintos, evitando acelerações e desacelerações desnecessárias, como podemos observar nas figuras 2.9 e 2.8, apesar de ser necessário um pré-processamento desses pontos que introduzem um custo computacional maior (YU et al. 2020) (YU *et al.*, 2020; KLIPPER, 2017b).

2.3.2 Curvas de velocidade trapezoidal

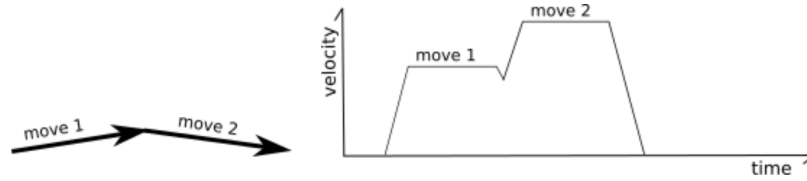
As impressoras 3D entre outros equipamentos, como máquinas CNC, necessitam de um planejamento de velocidade, pois o Gcode fornece apenas as velocidades desejadas de cada movimento, enquanto o algoritmo de lookahead calcula as velocidades de junção entre os movimentos, portanto ainda se faz necessário planejar o comportamento da velocidade ao longo do tempo do trajeto entre a velocidade inicial e final do movimento (figura 2.10). Uma das maneiras mais simples para a criação dessa curva de velocidade é a criação de uma curva trapezoidal, onde podemos separar o setor em 3 segmentos de aceleração constante. Em um primeiro momento uma crescente de velocidade até a velocidade desejada, seguido de um segmento de velocidade constante e por fim um segmento de desaceleração constante até a velocidade final. Alguns ajustes são feitos para as diferentes condições de velocidade inicial, final e velocidade máxima atingida com uma determinada aceleração máxima, que pode fazer com que se diminua a quantidade de segmentos (YU *et al.*, 2020; KLIPPER, 2017b).

Figura 2.8 – Look ahead em uma grande mudança de direção



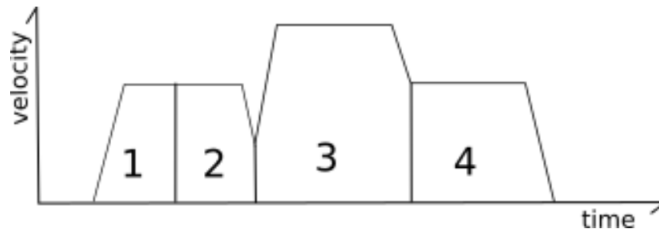
Fonte: KLIPPER, 2017b

Figura 2.9 – Look ahead em uma pequena mudança de direção



Fonte: KLIPPER, 2017b

Figura 2.10 – Curva de velocidades trapezoidal em uma sequência de movimentos



Fonte: KLIPPER, 2017b

2.3.3 Espaço de Estados

A maioria dos sistemas dinâmicos pode ser escritos através de uma formulação chamada de espaço de estados, que tem como objetivo expressar modelos de equações diferenciais parciais (EDP) ou ordinárias (EDO) de ordem superior como um conjunto de EDPs ou EDOs de primeira ordem. Essa formulação é construída a partir de um princípio de autoregressão das equações. Na equação 2.1 podemos observar uma EDO de segunda ordem representando um sistema massa mola simples, logo abaixo (2.2) a mesma equação representada na formulação de espaço de estados (HAMILTON, 1994).

$$m\ddot{x} + c\dot{x} + kx = f(t) \quad (2.1)$$

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ k/m & c/m \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} f(t) \quad (2.2)$$

2.3.4 Integração implícita utilizando programação não linear e colocação

HARGRAVES; PARIS (1987) descreve um algoritmo para a solução numérica direta de problemas de controle ótimo (HARGRAVES; PARIS, 1987). O método utiliza polinômios cúbicos para representar as variáveis de estado, interpolação linear para as variáveis de controle e colocação para satisfazer as equações diferenciais. Essa representação transforma o problema

de controle ótimo em um problema de programação matemática, que é resolvido por meio de programação quadrática sequencial. Esse método tem a vantagem de ser fácil de programar e pode lidar com problemas gerais de otimização de trajetória, incluindo restrições de caminho, estados descontínuos e desigualdades de controle.

O método aproxima a solução das equações diferenciais subdividindo cada estado da matriz de espaço de estados em segmentos, estes sendo representados por polinômios de 3º, onde são utilizados os valores do estado no contorno e a derivada no tempo do estado, como definido pelas equações, no contorno. Os valores de estado são então selecionados de forma que as derivativas no centro do segmento concordem com as equações diferenciais.

O procedimento base pode ser aplicado pelos seguintes passos.

A equação 2.3 avalia o estado no centro do segmento, onde x representa o estado, T representa o comprimento do segmento e f_i representa o valor da função avaliado em x_i . O subscrito c representa o centro do segmento.

$$x_c = \frac{x_1 + x_2}{2} + T \frac{f_1 - f_2}{8} \quad (2.3)$$

Da mesma maneira sua derivada é apresentada na equação 2.4.

$$\dot{x}_c = -3 \frac{x_1 + x_2}{2T} + \frac{f_1 + f_2}{4} \quad (2.4)$$

A equação 2.5 define então o valor do defeito no centro do segmento.

$$\Delta = f_c - \dot{x}_c \quad (2.5)$$

Considerando também que a entrada do sistema pode ser avaliada de forma aproximada no centro do segmento através da equação 2.6.

$$u_c = \frac{u_1 + u_2}{2} \quad (2.6)$$

Os valores de estado agora podem ser alterados de maneira que o defeito tenda a zero.

2.3.5 Objective Function Optimization

Funções objetivo de otimização são elementos importantes na execução de algoritmos de otimização. Alguns estudos apontam que a qualidade e o número de variáveis de projeto cru-

ciais para o resultado da otimização, além disso outros sub-parâmetros também são relevantes como restrições, limites e valores iniciais. A coerência entre estes fatores depende habilidade e de uma visão abrangente de seu criador (ALBAGHDADI; BAHAROM; SULAIMAN, 2021).

3 METODOLOGIA

3.1 Matlab

MATLAB é um software interativo para computação numérica que possui uma série de ferramentas, funções, visualisadores, ferramentas para debugging, estrutura de dados, entre outros auxílios que facilitam o desenvolvimento e o estudo de atividades que utilizam a computação numérica. Por conta desses facilitadores, o MATLAB é amplamente utilizado na indústria e no meio acadêmico.(HIGHAM; HIGHAM, 2016) Dada essa característica e a existência da função FMINCON a disposição no ambiente MATLAB, foi feita a escolha de se utilizá-lo para a construção do código presente neste trabalho.

3.1.1 fmincon

Como o modelo matemático a ser otimizado é multivariável e possuindo restrições não-lineares, a função FMINCON do ambiente do MATLAB é utilizada para otimizar as variações de velocidade de forma a diminuir o erro de trajetória associado às flexibilidades do sistema que causam perturbações e vibrações indesejadas. É uma função baseada em gradientes que busca por todos os mínimos locais de uma região que satisfaz outras restrições estipuladas (ALBAGHDADI; BAHAROM; SULAIMAN, 2021). Ela utiliza um conjunto de restrições superiores e inferiores para cada ponto e otimiza a função considerando as restrições estabelecidas pela função não linear, utilizando as equações de movimento para encontrar a solução da EDO de maneira a otimizar os parâmetros. Também permite uma série de configurações incluindo a escolha de diferentes algoritmos de otimização entre outros parâmetros sobre a função.

3.2 Geração de Comando

3.2.1 Leitura Gcode

Foi considerado no mapeamento do Gcode apenas comandos G1, extraíndo as informações dos eixos X, Y e do *feedrate* (F). Com base nesses valores uma matriz 3 por n é criada, n sendo o número de comandos lidos do arquivo Gcode. Em geral a unidade de F em arquivos Gcode gerados pelos fatiadores se dão em milímetros por minuto, mas é convertida para milímetros por segundo na construção da matriz de entrada.

3.2.2 Velocidade de curva

Por conta de alguns problemas de implementação de algoritmos de *lookahead*, a velocidade de junção, ou seja, a velocidade de transição desejada entre um movimento e outro, foi fixada como 0. Essa condição representa bem trajetórias que possuem ângulos entre os vetores de velocidade dos movimentos maiores ou iguais a 90°. Portanto, limitando os trajetos testados a esse tipo de trajetória o comando gerado se manteria próximo da realidade das impressoras 3D.

3.2.3 Curva rapezoidal de velocidade

A partir da matriz de entrada e das velocidades finais e iniciais dos movimentos estabelecida pelo *lookahead*, nos casos apresentados neste trabalho fixadas ambas em zero, utilizamos a função responsável por gerar a curva trapezoidal de velocidades.

Essa função separa o deslocamento total do movimento em 3 fases de aceleração constante. Na primeira fase a velocidade trazida da velocidade inicial até a velocidade desejada a aceleração constante, na segunda fase a velocidade é mantida constante na velocidade desejada e por fim na terceira fase a velocidade é levada da velocidade desejada até a velocidade final. Entretanto, em algumas situações pode não ser possível alcançar a velocidade desejada e o perfil se limitar a duas fases, outras condições onde alguma das velocidades é igual a velocidade desejada faz com que a quantidade de fases visíveis seja reduzida.

Para identificar se a velocidade desejada será alcançada é calculado a velocidade pico (v_p) que é obtida extrapolando retas com as inclinações da aceleração na velocidade inicial e final. É possível obter a velocidade de pico através da equação 3.1.

$$v_p = \sqrt{\frac{(v_i^2 + v_f^2)}{2} + acc * des_{tot}} \quad (3.1)$$

A partir da comparação da velocidade pico com a velocidade desejada, indicada pelo *feedrate* disponibilizado no Gcode, é possível determinar o padrão da curva de velocidade deste movimento. Caso a velocidade de pico for maior do que a velocidade desejada, temos 3 fases de deslocamento que podem ser calculadas pelas equações 3.2 e 3.3. Caso a velocidade de pico seja igual ou menor do que a velocidade desejada, teremos 2 fases de deslocamento que são calculadas a partir da equação 3.2.

$$des_{segment} = \frac{(v_f^2 - v_i^2)}{(2 * acc_{segment})} \quad (3.2)$$

$$des_{middle} = des_{total} - (des_{up} + des_{down}) \quad (3.3)$$

É possível calcular também os intervalos de tempo dessas fases, através das equações 3.4 e 3.5.

$$dt_{segment} = \frac{(v_f - v_i)}{acc_{segment}} \quad (3.4)$$

$$dt_d = \frac{des_d}{v_d} \quad (3.5)$$

Além disso é calculado a variação de velocidade nos intervalos pela equação 3.6.

$$\Delta vel = v_f - v_i \quad (3.6)$$

Esses passos resultam em uma nova matriz contendo informações sobre o a variação da posição, do tempo, da velocidade e sobre a aceleração e direção de deslocamento nos pontos iniciais e finais do Gcode e também nos pontos onde existe uma alteração na aceleração.

3.2.4 Interpolação

A partir dessa matriz, é utilizada uma função de interpolação para dividir cada intervalo dessa matriz em intervalos menores baseados em um passo de tempo definido para esta interpolação. Assim criando-se uma nova matriz dos dados interpolados. Para se dividir esses intervalos é possível utilizar a equação 3.7 calculando o número de passos neste intervalo, anexando à matriz os passos de tempo e por fim o restante do intervalo, calculado pela equação 3.8. Com base nestes passos de tempo, é possível calcular o deslocamento para cada um destes passos através da equação 3.9.

$$N_{steps} = \lceil \frac{\Delta t_i}{\Delta t_{stepsize}} - 1 \rceil \quad (3.7)$$

$$\Delta t_{laststep} = \Delta t_i - \Delta t_{stepsize} * N_{steps} \quad (3.8)$$

$$\Delta des_i = \Delta v_i * \Delta t_i + \frac{acc_{segment} * \Delta t_i^2}{2} \quad (3.9)$$

A partir dos vetores de direção e da função acumuladora que se consite em acumular os valores de um vetor. Obtemos uma matriz de posições, velocidades e tempo.

$$\begin{aligned} v_0 &= init_{value} + v_0 \\ v_k &= v_k + v_{k-1} \end{aligned} \quad (3.10)$$

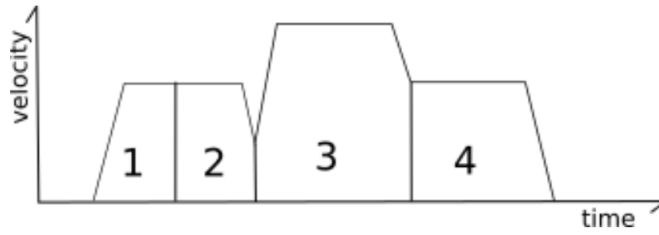
3.3 Modelagem dinâmica de uma impressora 3D

Para a modelagem do sistema mecânico são consideradas as seguintes simplificações:

- Não existe escorregamento nem perda de potência na interação entre a polia e a correia
- A correia apresenta um comportamento equivalente à uma mola e um amortecedor em paralelo
- O bico injetor é um corpo rígido uniforme de geometria simples
- A correia está acoplada nos dois lados da peça que se movimenta nos trilhos, entretanto como a correia só permite o tensionamento ela será considerada como um conjunto mola amortecedor simples

Para a modelagem dinâmica dos eixos X e Y da impressora 3D, é considerado que os eixos são completamente independentes, a flexibilidade da correia é aproximada utilizando um conjunto mola amortecedor e a transmissão de movimento e torque dos motores é considerada como ideal e não será abordada. Assim duas posições de estudo surgem para cada eixo, uma delas representa a posição ideal, caso o sistema não possuísse nenhuma flexibilidade ou perda, que também é a posição desejada pelo usuário (px_b). A segunda posição considera as forças inerciais e a flexibilidade introduzida pela correia, ou seja, a posição real simulada pelo modelo e no caso empírico a posição real (px) como na figura 3.1.

Figura 3.1 – Visualização das posições do sistema



$$m\ddot{p}x + c(\dot{p}x - \dot{p}x_b) + k(px - px_b) = 0$$

$$\ddot{p}x = -\frac{c}{m}(\dot{p}x - \dot{p}x_b) - \frac{k}{m}(px - px_b)$$

$$\ddot{p}x = -\frac{c}{m}\dot{p}x + \frac{c}{m}\dot{p}x_b - \frac{k}{m}px + \frac{k}{m}px_b$$

$$\ddot{p}x = -\frac{c}{m}\dot{p}x - \frac{k}{m}px + \frac{c}{m}\dot{p}x_b + \frac{k}{m}px_b \quad (3.11)$$

3.3.1 Espaço de estados

A formulação de espaço de estados foi utilizada com intuito de facilitar as operações e a a solução do sistema, dado sua característica de dividir uma equação diferencial de ordem superior em um sistema de equações diferenciais de ordem 1. O modelo dinâmico do sistema é apresentado na formulação de espaço de estados na equação 3.12, baseado na equação 3.11, utilizando a mesma equação para a base do eixo y.

$$\begin{bmatrix} \dot{p}x \\ \ddot{p}x \\ \dot{p}y \\ \ddot{p}y \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k_x}{m_x} & -\frac{c_x}{m_x} & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -\frac{k_x}{m_x} & -\frac{c_x}{m_x} \end{bmatrix} \begin{bmatrix} px \\ \dot{p}x \\ py \\ \dot{p}y \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{k_x}{m_x} & \frac{c_x}{m_x} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{k_x}{m_x} & \frac{c_x}{m_x} \end{bmatrix} \begin{bmatrix} px_b \\ \dot{p}x_b \\ py_b \\ \dot{p}y_b \end{bmatrix} \quad (3.12)$$

Em uma notação simplificada temos a equação 3.13

$$\dot{x} = A * x + B * u \quad (3.13)$$

3.4 FMINCON

3.4.1 Configurações da função

Foi utilizado o seguinte conjunto de configurações opcionais da função (tabela 3.1). As configurações não indicadas se mantem na configuração padrão, que pode ser encontrada nos manuais da mesma.

Tabela 3.1 – Tabela de parâmetros opcionais da FMINCON

Opção	Valor
TolFun	0.000000001
MaxIter	100000
Display	iter
DiffMinChange	0.0001
Algorithm	interior-point
StepTolerance	1e-12
MaxFunEvals	700000

3.4.2 Restrições lineares e limites de borda

Não foi utilizado nenhuma restrição linear nessa otimização. Os limites superiores (*upperbound*) e inferiores (*lowerbound*) foram definidos baseados nos limites físicos da impressora (0 a 200 milímetros) para as posições x_b e y_b .

3.4.3 Restrições não lineares

A função para as restrições não lineares foi implementada com base nas equações 2.3, 2.4, 2.5, 2.6 de maneira a popular a variável de restrições de igualdades com os valores do defeito dos segmentos (Δ) e também com os valores iniciais. Além disso, foi utilizado também como restrição de igualdade a arrancada, atrelado a um condicional para que se comporte como uma desigualdade. A variável de restrições de desigualdades não foi populada, os motivos são apresentados na seção de resultados e discussão.

3.4.4 Função objetivo

Foram realizados alguns testes com funções objetivo e seus resultados são apresentados na seção de resultados e discussão, entretanto a função objetiva foi definida como zero, ou seja, pra qualquer valor ela retornara zero nas chamadas da FMINCON.

3.5 Solução da trajetória da base

É considerada como trajetória desejada a trajetória obtida através da função de geração de comando apresentada anteriormente e são utilizados os vetores de tempo e posição como variáveis globais, para serem acessados dentro da função das restrições não lineares, chamada pela FMINCON. Além disso, esses mesmos vetores de posição também são considerados como o chute inicial e variável principal na FMINCON, sendo adaptados em forma de matriz.

A variável principal inserida na FMINCON é a matriz contendo os vetores de posição da base (x_b e y_b), enquanto os vetores de posição da ponta é fixado pela trajetória desejada presente na forma de variáveis globais.

Para conseguir realizar os cálculos necessários dentro da função de restrições não lineares é utilizado o seguinte conjunto de equações básicas (3.14) para se derivar a curva posição-tempo considerando aceleração constante, sendo dt a variação do tempo no intervalo, des_n o deslocamento final do intervalo, des_i o deslocamento inicial no intervalo, acc_n a aceleração do intervalo, vel_n a velocidade final do intervalo e vel_i a velocidade inicial do intervalo. Assim construindo o vetor de velocidade a partir da condição inicial de deslocamento e velocidade zero, para ambos os eixos (x e y).

$$acc_n = 2 \frac{\left(\frac{des_n - des_i}{dt} \right) - vel_i}{dt} \quad (3.14)$$

$$vel_n = vel_i + acc_n dt$$

3.6 Descrição das simulações

Foi realizado uma série de simulações utilizando como base uma sequência de dois movimentos em 90 graus, 10 milímetros ao longo do eixo x e depois 10 milímetros ao longo do eixo y, através de um arquivo Gcode de teste. Em cada uma dessas simulações um dos parâmetros é alterado, enquanto o restante permanece em uma configuração base, apresentada na tabela 3.2. Além disso, em cada teste foram separados algumas variações do mesmo parâmetro em questão e as versões do teste são identificadas por uma letra de A a C. Assim, as configurações para as diferentes simulações adicionais segue a tabela 3.3.

Além das curvas de posição, deslocamento e velocidade, algumas variáveis também foram explicitadas para análise. São elas o tamanho dos vetores, o tempo de simulação e a

Tabela 3.2 – Tabela de parâmetros base das simulações

Parâmetro	Valor	Unidade
Frequência	100	rad/s
Coefficiente de amortecimento	0,5	-
Aceleração base	5000	mm/s^2
Velocidade desejada	100	mm/s
Resolução de interpolação	0,005	s

Tabela 3.3 – Tabela do parâmetro modificado das simulações

Teste	Parâmetro	Valor A	Valor B	Valor C	Unidade
1	Frequência	50	200	500	rad/s
2	Coefficiente de amortecimento	0	1	2	-
3	Aceleração base	1000	10000	-	mm/s^2
4	Velocidade desejada	50	200	-	mm/s
5	Resolução de interpolação	0,01	0,001	0,0002	s

viabilidade ao longo . O tamanho dos vetores é definido na fase de geração de comando e tem influência direta da resolução de interpolação definida e do tempo necessário para percorrer o caminho dada as definições de velocidades na geração de comando. O tempo de simulação começa a ser contado logo depois da geração de comando e para quando a função FMINCON termina de ser executada. A viabilidade é um dos resultados da função FMINCON e este representa o valor da maior restrição não cumprida.

A máquina utilizada para a realização das simulações foi um notebook acer com as configurações apresentadas na tabela 3.4.

Tabela 3.4 – Tabela de especificações do computador

Processador	Intel I7-5500U 2.40GHz
Memoria	8,00 GB
Placa de vídeo	Nvide Geforce 920M
Sistema	64 bits

4 RESULTADOS E DISCUSSÃO

4.1 Resultados

4.1.1 Correlações das variáveis de entrada

4.1.2 Influências do modelo dinâmico

4.1.3 Performance Computacional

4.2 Dificuldades

4.2.1 Função objetivo

4.2.2 Restrição de aceleração da base

4.2.3 Variáveis principais

4.2.4 Geração de Comando

4.3 Possíveis abordagens

4.3.1 Combinação com outros algoritmos

5 CONCLUSÃO

O presente trabalho não conseguiu alcançar seus objetivos de maneira completa dada a limitação da implementação nos limites de aceleração. Entretanto, os resultados obtidos mostram o potencial da técnica iterativa de otimização, necessitando de uma execução melhor para ser capaz de atuar de forma completa. Outra dificuldade da técnica é o tempo de processamento, onde não é possível realizar o processamento em real-time sendo necessário utilizá-lo em conjunto com um pré processamento do Gcode.

REFERÊNCIAS

ALBAGHDADI, A. M.; BAHAROM, M. B.; SULAIMAN, S. A. bin. Parameter design optimization of the crank-rocker engine using the fmincon function in matlab. In: IOP PUBLISHING. **IOP Conference Series: Materials Science and Engineering**. [S.l.], 2021. v. 1088, n. 1, p. 012072.

ATTARAN, M. The rise of 3-d printing: The advantages of additive manufacturing over traditional manufacturing. **Business horizons**, Elsevier, v. 60, n. 5, p. 677–688, 2017.

BIKAS, H.; STAVROPOULOS, P.; CHRYSSOLOURIS, G. Additive manufacturing methods and modelling approaches: a critical review. **The International Journal of Advanced Manufacturing Technology**, Springer, v. 83, p. 389–405, 2016.

DUAN, M.; YOON, D.; OKWUDIRE, C. E. A limited-preview filtered b-spline approach to tracking control—with application to vibration-induced error compensation of a 3d printer. **Mechatronics**, Elsevier, v. 56, p. 287–296, 2018.

GIBSON, I. *et al.* Applications for additive manufacture. **Additive Manufacturing Technologies: 3D Printing, Rapid Prototyping, and Direct Digital Manufacturing**, Springer, p. 451–474, 2015.

HAMILTON, J. D. State-space models. **Handbook of econometrics**, Elsevier, v. 4, p. 3039–3080, 1994.

HARGRAVES, C. R.; PARIS, S. W. Direct trajectory optimization using nonlinear programming and collocation. **Journal of guidance, control, and dynamics**, v. 10, n. 4, p. 338–342, 1987.

HIGHAM, D. J.; HIGHAM, N. J. **MATLAB guide**. [S.l.]: SIAM, 2016.

KLIPPER. **Klipper Documentation**. 2017. Disponível em: <<https://www.klipper3d.org/>>.

KLIPPER. **Klipper Kinematics Documentation**. 2017. Disponível em: <<https://www.klipper3d.org/Kinematics.html>>.

RAMANI, K. S. *et al.* Tracking control of linear time-invariant nonminimum phase systems using filtered basis functions. **Journal of Dynamic Systems, Measurement, and Control**, American Society of Mechanical Engineers Digital Collection, v. 139, n. 1, 2017.

RAMANI, K. S.; EDOIMIOYA, N.; OKWUDIRE, C. E. A robust filtered basis functions approach for feedforward tracking control—with application to a vibration-prone 3-d printer. **IEEE/ASME Transactions on Mechatronics**, IEEE, v. 25, n. 5, p. 2556–2564, 2020.

SINGHOSE, W. E. **Command generation for flexible systems**. Tese (Doutorado) — Massachusetts Institute of Technology, 1997.

TURNER, B. N.; STRONG, R.; GOLD, S. A. A review of melt extrusion additive manufacturing processes: I. process design and modeling. **Rapid prototyping journal**, Emerald Group Publishing Limited, v. 20, n. 3, p. 192–204, 2014.

VYAVAHARE, S. *et al.* Fused deposition modelling: a review. **Rapid Prototyping Journal**, Emerald Publishing Limited, v. 26, n. 1, p. 176–201, 2020.

YU, K. *et al.* Application of the five-phase s-curve velocity model on fdm three-dimensional printer. In: IEEE. **2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC)**. [S.l.], 2020. p. 1365–1371.