Universidade de Brasília
Faculdade de Tecnologia

Aplicação do conceito de Digital Twin: implementação de um sistema de manutenção preditiva para impressoras 3D

João Vitor Ferreira da Silva

TRABALHO DE GRADUAÇÃO

ENGENHARIA DE CONTROLE E AUTOMAÇÃO

Brasilía

2022

Universidade de Brasília
Faculdade de Tecnologia

Aplicação do conceito de Digital Twin: implementação de um sistema de manutenção preditiva para impressoras 3D

João Vitor Ferreira da Silva

Trabalho de Graduação submetido como requisito parcial para obtenção do grau de Engenheiro de Controle e Automação.

Orientador: Prof. Dr. Geovany Araujo Borges
Coorientador: Profª. Dra. Andréa Cristina dos Santos

Brasilía
2022

Ferreira da Silva, João Vitor.
F383a Aplicação do conceito de Digital Twin: implementação de um sistema de manutenção preditiva para impressoras 3D / João Vitor Ferreira da Silva; orientador Geovany Araujo Borges; coorientador Andréa Cristina dos Santos. -- Brasilía, 2022.
89 p.

Trabalho de Graduação em Engenharia de Controle e Automação -- Universidade de Brasília, 2022.

1. Gêmeo Digital. 2. Manufatura inteligente. 3. Impressão 3D. 4. Manufatura aditiva. I. Araujo Borges, Geovany, orient. II. dos Santos, Andréa Cristina, coorient. III. Título

Universidade de Brasília
Faculdade de Tecnologia

Aplicação do conceito de Digital Twin: implementação de um sistema de manutenção preditiva para impressoras 3D

João Vitor Ferreira da Silva

Trabalho de Graduação submetido como requisito parcial para obtenção do grau de Engenheiro de Controle e Automação.

Trabalho aprovado. Brasilía, 3 de maio de 2022:

Prof. Dr. Geovany Araujo Borges,
UnB/FT/ENE
Orientador

Profª. Dra. Andréa Cristina dos Santos,
UnB/FT/EPR
Coorientador

Roberto de Souza Baptista
Examinador interno

Flavia Maria Guerra de Sousa Aranha Oliveira
Examinador interno

Brasilía
2022

Agradecimentos

Agradeço primeiramente a Deus, pois me deu força para continuar em meio as dificuldades do dia a dia. Por mais que o mundo tenha passado por tempos difíceis de pandemia e guerra, não devemos desanimar, pois, se tivermos fé no Senhor, sairemos vitoriosos. ˝No mundo tereis aflições, mas tende bom ânimo. Eu venci o mundo.” (João, 16:33). Tudo que conquistei e que irei conquistar é para a honra e glória do Senhor.

Agradeço também a minha família que me apoia nas minhas decisões e estão sempre dispostos a ajudar quando preciso, principalmente minha mãe e minha irmã, que são o que tenho de mais preciosas na minha vida. Quando temos alguém para nos motivar, para estar ao nosso lado, os momentos bons se tornam ainda melhores e os maus momentos se tornam algo passageiro e conseguimos lidar mais facilmente.

Agradeço também a professora Andréa Cristina dos Santos, que me permitiu fazer parte do Laboratório Aberto de Brasília (LAB), possibilitando viver momentos importantes de aprendizado que serão de extrema importância na minha vida profissional e pessoal. Além do aprendizado, também pude conhecer muitas pessoas, fazer novas amizades, fazer parte de momentos e projetos incríveis e fazer parte do crescimento do LAB. Tenho certeza que ela conquistará ainda mais sucesso com o projeto do LAB e irá impactar possitivamente a vida de muitos alunos da Universidade de Brasília. Deixo uma agradecimento especial também á Jéssica Mendes (antiga gestora do LAB), ao Alexandre Crepory (antigo gestor do LAB) e ao João Vitor Quintiliano (colaborador do LAB) que me ajudaram muito durante o tempo que fiz parte do LAB.

Agradeço também aos professores da Universidade de Brasília que me ensinaram com excelência o conhecimento necessário para ser um bom engenheiro, em especial o professor Geovany Araujo Borges que além de ser um excelente professor, me auxiliou durante a elaboração do trabalho de graduação.

Resumo

Nos últimos anos a quarta revolução industrial, também chamada de Indústria 4.0, surgiu graças às tecnologias habilitadoras como Cyber-Physical Systems (CPS), Internet of Things (IoT), Cloud computing, Blockchain, etc. A Industria 4.0 é considerada uma estrutura tecnológica que possibilita a integração e a extensão dos processos de fabricação afetando desta forma todo o ciclo de vida de um produto, desde sua concepção até seu declinio.

Para realizar a implementação da industria 4.0 é necessário dois elementos, dados do sistema e um mecanismo que possibilite a convergência entre o modelo físico e o modelo virtual. Um desses mecanismos é o Digital Twin, que oferece não apenas o monitoramento e o gerenciamento sistema, mas também permite realizar simulações no modelo virtual e atuar diretamente no sistema.

A utilização do Digital Twin pode ser feita em diversar áreas da industria manufatureira, uma dessas áreas é a area da área manutenção, em especial a manutenção preditiva. A manutenção preditiva pode ser voltada tanto para realizar o diagnóstico ou o proagnóstico do sistema.

Nesse trabalho foi desenvolvido um sistema de manutenção preditiva na manufatura aditiva, também conhecida como impressão 3D. Foi utiliza o conceito de Digital Twin para realizar o diagnóstico de uma impressora 3D do tipo Fused Deposition Modeling (FDM), possibilitando, desta forma, o monitoramento do estado da impressora e a atuação direta na operação.

O monitoramento da impressora foi feita a partir de dados coletados de sensores que foram instalados nela. Já atuação na impressora 3D utilizada foi feita de duas forma diferentes, sendo o cancelamento da operação ou então a pausa da operação.

Para auxiliar no desenvolvimento do sistema, foi utilizado o Diagrama em V que é uma ferramenta que auxilia no desenvolvimento de sistemas complexos, possuindo seu foco principal nas partes interessadas no sistema que foi criado. Portanto, foi necessário mapear quem são os stakeholders do sistema de manutenção preditiva e realizar o levantamento de quais a necessidades que o sistema construído deve suprir, para que dessa forma essas necessidades fossem transformadas em requisitos técnicos para que fosse possivél aa implementação do sistema.

Entre os vários requisitos levantados foi escolhido realizar a parte de sensoriamento da impressora que consistiu em detectar falhas de extrusão de filamento, falhas no nivelamento

da mesa e falhas de perda de passo, além de realizar um levantamento inicial sobre a relação entre a vibração e a tensão da correia da impressora.

Para detecção de falhas de extrusão foi utilizado um sensor de filamento inteligente que, além de detectar se existe filamento no sistema, também é capaz de detectar a existência de fluxo de filamento, no qual, quando não há fluxo, significa que poblemas de extrusão estão ocorrendo e, desta forma, ocorre uma atuação no sistema pausando a impressão.

Para detectar falhas no nivelamento da mesa foi utilizado um BLTouch que mede a distância entre o bico da impressora e a mesa, no qual, caso o valor medido sejá maior que um limiar previamente definido, significa a mesa de impressão está desnivelada, precisando assim de manutenção. Logo, é feita uma atuação no sistema para cancelar a impressão.

Para realizar a detecção de falhas decorrentes de perda de passo no motor que realiza a movimentação do extrusor, foi utilizada uma câmera que captura frames da peça sendo produzida que são utilizados por uma inteligência artificial capaz de indentificar falhas na peça ocassionadas pela perda de passo no motor. Nesse caso, quando uma falha é detectada, a atuação no sistema depende da gravidade dessa falha.

Já em relação ao levantamento da relação entre a vibração do extrusor e a tensão na correia, foi utilizado um acelerômetro de 3 eixos para mensurar a vibração do extrusor e nos dados obtidos foi aplicado uma FFT, de forma a obter os espectros que compõem a forma de onda de cada eixo, permitindo realizar uma análise visual para verificar se existe a relação.

A partir do trabalho desenvolvido foi possível realizar o levantamento de trabalhos futuros que podem ser desenvolvidos como etapas para a implementação de um sistema de Digital Twin completo, com todas as suas funcionalidades.

Palavras-chave: Gêmeo Digital. Manufatura inteligente. Impressão 3D. Manufatura aditiva. Industria 4.0. Manutenção preditiva. Diagrama em V. Internet das coisas. Sistema ciber físico.

Abstract

In recent years, the fourth industrial revolution, also called Industry 4.0, has emerged thanks to enabling technologies such as Cyber-Physical Systems (CPS), Internet of Things (IoT), Cloud computing, Blockchain, etc. Industry 4.0 is considered a technological structure that enables the integration and extension of manufacturing processes, thus affecting the entire life cycle of a product, from its conception to its decline.

To implement Industry 4.0, two elements are needed, system data and a mechanism that enables convergence between the physical model and the virtual model. One of these mechanisms is the Digital Twin, which offers not only system monitoring and management, but also allows simulations to be carried out in the virtual model and to act directly on the system.

The use of the Digital Twin can be made in several areas of the manufacturing industry, one of these areas is the area of maintenance, especially predictive maintenance. Predictive maintenance can be aimed either at diagnosing or prognosticating the system.

In this work, a predictive maintenance system was developed in additive manufacturing, also known as 3D printing. The Digital Twin concept was used to perform the diagnosis of a Fused Deposition Modeling (FDM) 3D printer, thus enabling the monitoring of the printer's status and direct action in the operation.

The monitoring of the printer was made from data collected from sensors that were installed on it. The operation on the 3D printer used was done in two different ways, being the cancellation of the operation or the pause of the operation.

To assist in the development of the system, the V Diagram was used, which is a tool that helps in the development of complex systems, having its main focus on the interested parties in the system that was created. Therefore, it was necessary to map who are the stakeholders of the predictive maintenance system and carry out a survey of what needs the built system must meet, so that these needs could be transformed into technical requirements so that the implementation of the system was possible.

Among the various requirements raised, it was chosen to carry out the sensing part of the printer, which consisted of detecting filament extrusion failures, table leveling failures and loss of pitch failures, in addition to carrying out an initial survey on the relationship between vibration and printer belt tension.

To detect extrusion failures, an intelligent filament sensor was used that, in addition to detecting if there is filament in the system, is also capable of detecting the existence of filament flow, in which, when there is no flow, it means that extrusion problems are present. occurring and, in this way, an action occurs in the system, pausing the printing.

To detect failures in the leveling of the table, a BLTouch was used, which measures the distance between the printer nozzle and the table, in which, if the measured value is greater than a previously defined threshold, it means the printing table is unleveled, thus requiring maintenance. Then, an action is made in the system to cancel the printing.

In order to detect failures resulting from loss of step in the motor that moves the extruder, a camera was used that captures frames of the part being produced, which are used by an artificial intelligence capable of identifying failures in the part caused by the loss of step in the motor. In this case, when a fault is detected, the performance in the system depends on the severity of the fault.

Regarding the survey of the relationship between the vibration of the extruder and the tension in the belt, a 3-axis accelerometer was used to measure the vibration of the extruder and an FFT was applied to the data obtained, in order to obtain the spectra that make up the shape. waveform of each axis, allowing to perform a visual analysis to verify if the relationship exists.

From the work developed, it was possible to carry out a survey of future works that can be developed as steps for the implementation of a complete Digital Twin system, with all its functionalities.

Keywords: Digital Twin. Smart manufacturing. 3D printer. Additive manufacturing. Industry 4.0. Predictive maintenance. V Diagram. Internet of things. cyber-physical system.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

CNC	Comando numérico computadorizado	25
CPS	Cyber-physical systems	16
DT	Digital Twin	16
EPI	Equipamento de proteção individual	15
ES	Engenharia de Sistemas	28
INCOSE	International Council on Systems Engineering	28
IoT	Internet of things	16
LAB	Laboratório Aberto de Brasília	15
MA	Manufatura aditiva	25
MP	Manutenção preditiva	17
RFID	Identificação por rádio frequência	20
RSSF	Redes de sensores sem fio	21
TIC	Tecnologias da informação e comunicação	19
ULEG	Unidade de Laboratórios de Graduação	15
UnB	Universidade de Brasília	15
WSN	Wireless sensor networks	21

Sumário

1 Introdução

No ano de 2020, com a ocorrência da pandemia causada pelo vírus SARS-CoV-2, que causa a Covid-19, ocorreu uma mobilização mundial para que os efeitos do vírus fossem amenizados e o menor número possível de pessoas fossem afetadas pelo vírus. Pesquisas sobre vacinas para imunização contra o vírus, maquinários para tratamento de pessoas afetadas, equipamentos para reduzir a probabilidade de contágio, entre outras tecnologias, foram criadas e impulsionadas com o surgimento do vírus.

Neste contexto, o Laboratório Aberto de Brasília (LAB), observando a necessidade de amenizar a taxa de contágio do vírus, identificou a oportunidade de aplicar as tecnologias que os colaboradores do LAB dominam para produzir um modelo de equipamento de proteção individual (EPI) chamado protetor fácil, que é produzido de diversas formas diferente. Uma delas é utilizando impressão 3D, sendo uma das tecnológias dominadas pelos colaboratores do LAB.

Por conta da urgência da pandemia, decidiu-se transformar o prédio de Unidade de Laboratórios de Graduação (ULEG),pertecente a Universidade de Brasília (UnB), no campus Darcy Ribeiro, que foi fechado para os alunos, em uma fábrica para criar os protetores faciais em uma escala maior utilizando impressoras 3D, com o objetivo de realizar doações dos protetores para os profissionais que atuavam na linha de frente do combate ao Covid-19. O projeto realizado recebeu o nome de Projeto Vida 2020. A impressão 3D era utilizada para fabricar a peça principal que é utilizada para fixar o visor transparente e o eslástico. A Figura 1 é uma demonstração de como é o produto final.

A fabricação dos protetores ocorreu durante todo o ano de 2020 até início de 2021. Durante esse tempo, dados sobre a produção foram adquiridos para a realização de estudos posteriores que possibilitaram a elaboração de pesquisas a fim de desenvolver novas tecnologias no LAB.

Figura 1 – Ilustração do protetor facial.

Fonte: (PRUSA 3D, 2020)

Com base nas operações desenvolvidas durante o Projeto Vida 2020 e nos dados de produção adquiridos, foi possível identificar que a ocorrência de manutenção causava uma queda significativa na produção. Também foi possível identificar a necessidade de otimizar todo o processo de produção.

Logo, desenvolver um sistema que possibilite otimizar a produção e ter um controle maior sobre todos as operações era necessário. Desta forma, foram feitos estudos para realizar o levantamento de tecnologias disponíveis que possibilitasse desenvolver esse sistema, destacando aquelas que possuem um maior potêncial. Como resultado das pesquisas foi proposta a do Digital Twin (DT) nas impressoras 3D.

O DT tem como objetivo convergir o modelo físico (um elemento na vida real) de um sistema com o modelo virtual (um elemento computacional que simula um modelo físico) criado desse mesmo sistema, no qual o modelo virtual é extremamente fiel ao modelo físico. Dois diferenciais são aparentados pelo mecanismo de DT:

1°. diferencial: a possibilidade de realizar simulações no modelo virtual, pois, como é extremamente fiel ao modelo físico, o que for simulado será, na teoria, o que acontecerá no modelo físico quando for implementada a mudança (), o que possibilita, por exemplo, uma grande redução de custos do processo de fabricação.

2°. diferencial: a possibilidade de atuar no sistema, seja no modelo virtual ou no modelo físico, realizando otimizações nas operações que estão ocorrendo (FEI TAO; LIU, 2019).

Logo, neste trabalho será apresentada a Industria 4.0 (seção 2.1) que tem como sua principais tecnologia habilitadoras o cyber-physical systems (CPS) e a internet of things (IoT), na qual foi introduzido a tecnologia de DT como mecanismo capaz de implementá-la (seção 2.2). Como o trabalho é voltado para os serviços prestados no LAB, foi introduzido o conceito de manufatura aditiva focando principalmente na tecnologia de Fused Deposition Modeling (FDM) (seção 2.3).

Levando em consideração a complexidade de um mecanismo de DT, será também apresentada uma ferramenta que é utilizada no desenvolvimento de sistemas de engenharia complexos chamada de Diagrama em V (seção 9), que possui a finalidade de ajudar no desenvolvimento focado nas necessidades que as entidades interessadas no sistema, stakeholders, apresentaram. Um de seus diferenciais é a verificação do sistema em cada etapa concluida para saber se o que está sendo desenvolvido é justamente o que os Stakeholders desejam, caso não esteja de acordo a etapa é refeita.

E por fim, será implementado um sistema de manutenção preditiva (MP) (introduzindo brevemente sobre os tipos de manutenção existentes)(seção 2.4) baseando-se no conceito de DT, onde será utilizado o Diagrama em V para realizar o seu desenvolvimento. Em

concordância com o Diagrama em V os seguintes passos serão seguidos no desenvolvimento no sistema de MP:

1°. Mapeamento dos stakeholders; (seção 3.1)

2°. Levantamento das necessidades dos Stakeholders; (seção 3.2)

3°. Definição dos requisitos; (seção 3.3)

4°. Design do sistema; (seção 3.4)

5°. Acoplamento dos componentes; (seção 3.4.3)

6°. Testes e verificação dos componentes em separadamente; (seção 4.1)

7°. Integração dos código. (seção s4.2)

No desenvolvimento do sistema de MP foi focada no sensoriamento de partes importantes da impressora que possuem grande impacto na qualidade superficial e/ou mecânica da peça produzida, permitindo a aquisição dos dados necessários para realizar o prognóstico da impressora 3D, além de realizar a integração dos componentes em um único framework.

Então, a partir do desenvolvimento do trabalho, será possível verificar quais trabalhos futuros podem ser desenvolvidos para auxiliar na implementação do DT nas impressoras 3D

2 Fundamentos

2.1 Industria 4.0

A tecnologia tem se desenvolvido continuamente ao longo dos anos e ,nos últimos anos, tem se desenvolvido de forma acelerada e um curto período de tempo é o suficiente para novas tecnologias surgirem. Como consequência desse desenvolvimento tem-se novos processos e aplicações sendo criados possibilitando melhorias na vida cotidiana, seja direta ou indiretamente. Alguma dessas novas tecnologias são chamadas de tecnologias habilitadoras, como a exemplo, o CPS, IoT, cloud computing, blockchain, entre outras tecnologias relacionadas. Essas tecnologias possuem um papel fundamental pois possibilitam que uma grande quantidade de informações sejam adquiridas e processadas, além de permitir que novas operações, processos e aplicações possam ser criados, tornando possível o que já é declarada a quarta revolução industrial, a Industria 4.0, que representa a "atual tendência de tecnologias de automação na indústria de manufatura" como citado no Eric L. Xu Li Da Xu e Ling Li (2018).

A Industria 4.0 tem ganhado bastante destaque nos últimos anos, sendo um dos tópicos mais frequentes em eventos, conferências, fóruns e exposições voltadas para a manufatura (YONGXIN LIAO; RAMOS, 2017).

O termo indústria 4.0 simboliza o início da quarta revolução indústrial e foi introduzido em 2011 durante a Feira de Hannover (evento mundial voltado para área de tecnologia, inovação e automação), porém, apenas em 2013 foi oficialmente formalizado como uma estratégia alemã que possui o objetivo de tornar a Alemanha pioneira na revolução da industria manufatureira (LI DA XU, E. L. X.; LI, L., 2018).

Para chegar até o momento atual da indústria, ocorreram três revoluções industriais. A Figura 2 ilustra a revolução sofrida pela indústria até o momento atual, tal revolução é dividida em 4 estágios.

A primeira revolução industrial, com seu início datado por volta do final do século XVII, é reconhecida pela utilização de máquinas mecânicas que utilizavam o vapor e a água como sua fonte de energia. A segunda revolução industrial tem seu início no final do século XIX, sendo caracterizada pela utilização de energia elétrica nas cadeias de produção. A terceira revolução industrial começa por volta da metade do século XX introduzindo os computadores, a automatização da indústria e a microeletrônica, possibilitando, desta forma, avanços na área de tecnologias da informação e comunicação (TIC) (JEFF HORN; SMITH, 2010) que possui papel importante no surgimento da quarta revolução industrial, pois a quarta revolução necessita de uma grande quantidade de informações sobre os processos,

além de um sistema de comunicação eficiente.

Atualmente estamos situados na quarta revolução industrial, chamada de Industria 4.0 ou Smart Manufacturing, possuindo seu surgimento datado em menos de 10 anos. A Industria 4.0 é considerada uma estrutura tecnológica que possibilita a integração e a extensão dos processos de fabricação em todos os níveis organizacionais (LI DA XU, E. L. X.; LI, L., 2018), ou seja, em todo o ciclo de vida do produto, sendo sua base tecnológica o CPS e IoT, porém outras tecnologias também são utilizadas como cloud computing, service-oriented computing, artificial intelligence and data science (KUSIAK, 2017).

Vale destacar que tanto a terceira quanto a quarta revolução possuem o objetivo de automatizar máquinas e processos, a quarta revolução possui um foco maior na digitalização ponta a ponta e na integração de sistemas industriais digitais buscando soluções totalmente integradas (LI DA XU, E. L. X.; LI, L., 2018).

2.1.1 CPS

O termo CPS foi primeiramente utilizado no primeiro workshop em sistemas ciberfísicos realizado em Austin, Texas. Nesse Workshop foi definido como objetivo das pesquisas sobre os sistemas ciberfísicos a criação de uma ´´geração de sistemas projetados que são altamente confiáveis, produzidos com eficiência e capazes de desempenho avançado em

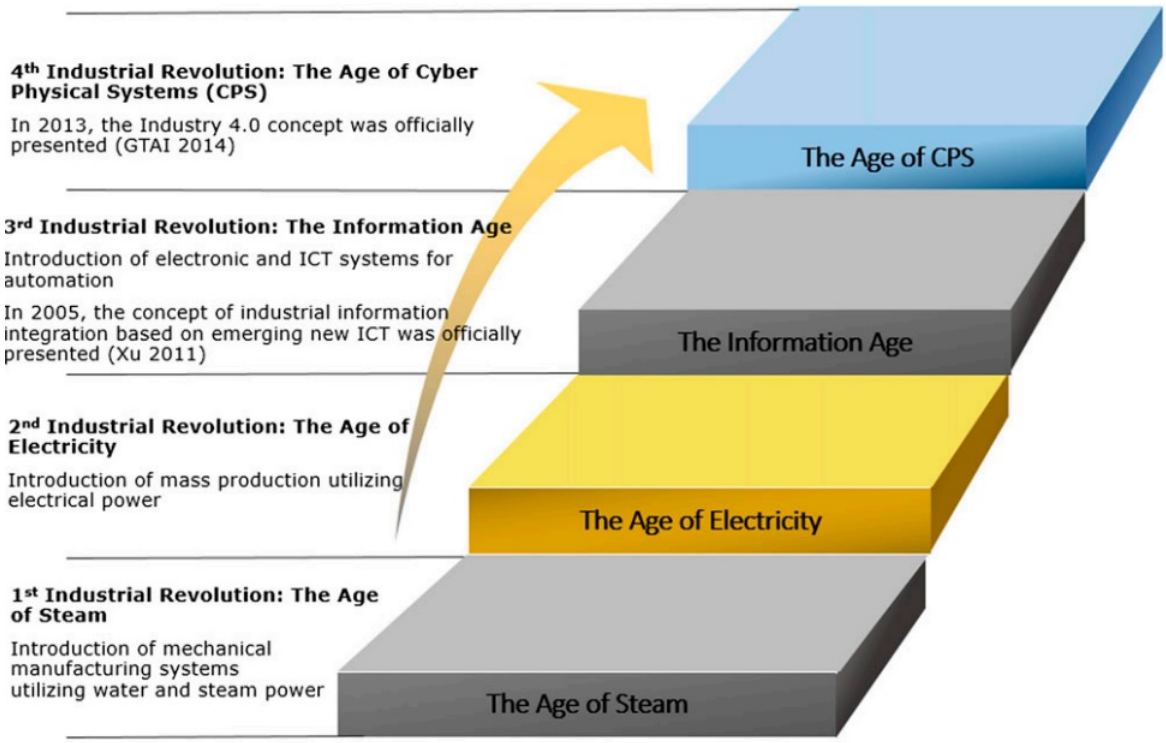


Figura 2 – Figura que ilustra a divisão da revolução ocorrida na Industria.

Fonte: (LI DA XU, E. L. X.; LI, L., 2018)

informação, computação, comunicação e controle", como foi apresentado em Foundation (2016). Logo, o objetivo era criar sistemas que possuem um monitoramento eficiente que possibilite o gerenciamento dos processos, de maneira a garantir seu correto funcionamento.

Os CPS podem ser definidos como sistemas computacionais colaborativos que são integrados com os sistemas físicos, possibilitando assim monitorar, controlar, gerenciar e integrar a outros sistemas, no qual os sistemas afetam um ao outro (RAGUNATHAN RAJKUMAR INSUP LEE; STANKOVIC, 2010). ″O CPS é sobre a interseção, não a união, do físico e do cibernético. Não é suficiente entender separadamente os componentes físicos e os componentes computacionais. Devemos entender sua interação", como citado no Edward Ashford Lee e Seshia (2017).

2.1.2 IoT

Inicialmente o termo IoT foi utilizado para referenciar dispositivos utilizando identificação por rádio frequência (RFID). Porém, com o desenvolvimentos de novas tecnologias como sensores, atuadores, dispositivos GPS e mobiles, mudanças na definição de IoT foram feitas, desta forma a sua mais atual definição é dada como:

> *a dynamic global network infrastructure with self-configuring capabilities based on standard and interoperable communication protocols where physical and virtual 'Things' have identities, physical attributes, and virtual personalities and use intelligent interfaces, and are seamlessly integrated into the information network. (XIAOLIN JIA1 QUANYUAN FENG; LEI, 2012)*

Em outras palavras, a IoT é uma rede global compostas por vários dispositivos interligados formando uma rede complexa que possuem um sistema de monitoramento, comunicação e processamento de informações (TAN; WANG, 2010), ″caracterizado por um alto grau de captura autônoma de dados, transferência de eventos, conectividade de rede e interoperabilidade", como foi apresentado em Xiaolin Jia1 Quanyuan Feng e Lei (2012).

A arquitetura de IoT pode ser dividida em três camadas: camada de percepção, camada de rede e camada de aplicação.

- Camada de percepção: é a camada principal de uma IoT que realiza a comunicação com o mundo físico, ou seja, são sensores, câmeras, marcadores, entre outros componentes responsáveis por coletar os dados (XIAOLIN JIA1 QUANYUAN FENG; LEI, 2012).
- Camada de rede: é a responsável por ligar a camada de percepção com a camada de aplicação. Nesta camada faz-se o uso de tecnologias de comunicação, como Wifi, Bluetooth, rede de rádio frequência, entre outras tecnologias que possibilitem o transporte dos dados obtidos na camada de percepção para a camada de aplicação (XIAOLIN JIA1 QUANYUAN FENG; LEI, 2012).

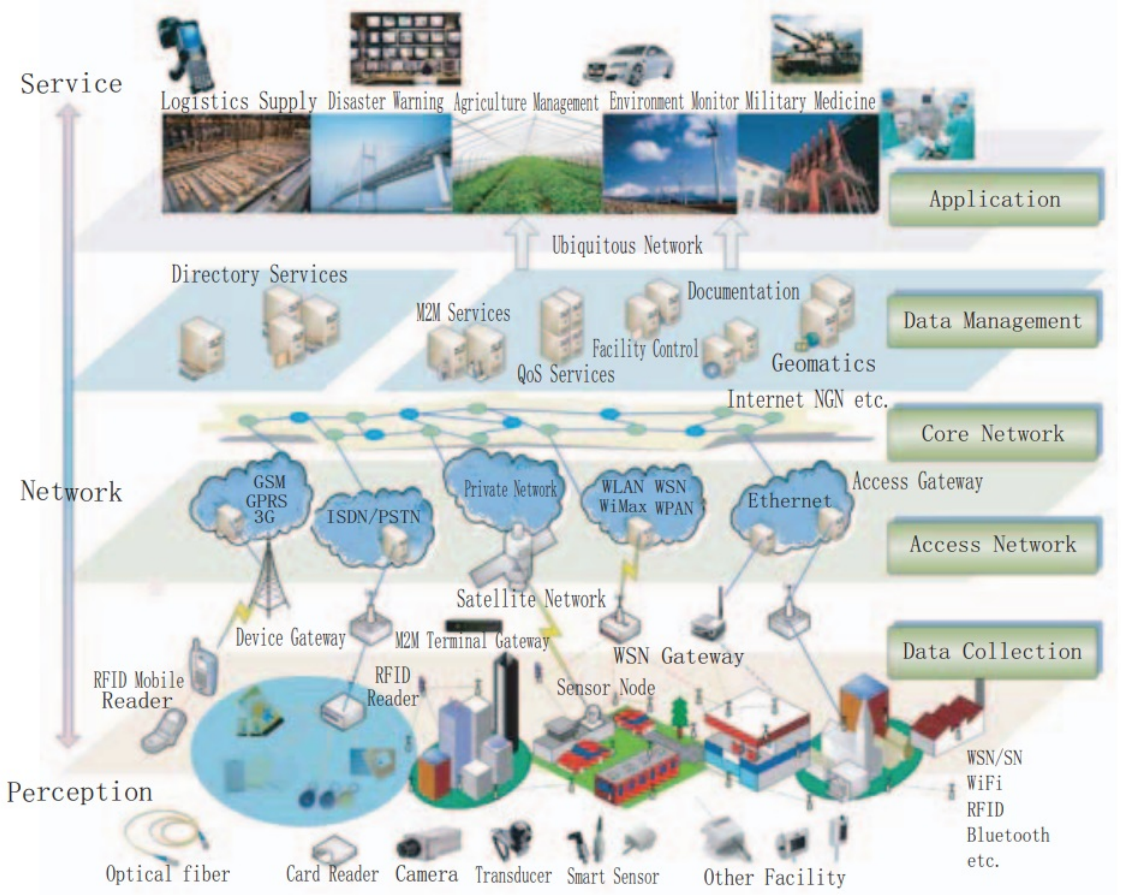


Figura 3 – A divisão da arquitetura de IoT.

(XIAOLIN JIA1 QUANYUAN FENG; LEI, 2012)

- Camada de aplicação: esta camada é dividida em duas subcamada, sendo elas a subcamada de gerenciamento, responsável pelo processamento de dados complexos e alguns tipos de informações, e a subcamada de serviço de aplicativo, que pega as informações obtidas e a transforma em dados úteis para o usuário final através de uma interface (XIAOLIN JIA1 QUANYUAN FENG; LEI, 2012).

A Figura 3 ilustra a divisão da arquitetura de IoT.

RFID e wireless sensor networks (WSN) são consideradas tecnologias principais para a existência de uma rede IoT (LI DA XU, E. L. X.; LI, L., 2018). A Figura 4 ilustra tecnologias e dispositivos usados que dão suporte à IoT, além do RFID e o WSN (LI DA XU, W. H.; LI, S., 2013) citados anteriormente.

A tecnologia de RFID possibilita automatizar a identificação, o rastreamento e o monitoramento de qualquer objeto que seja anexado a uma de seus marcadores (XIAOLIN JIA1 QUANYUAN FENG; LEI, 2012). Já as RSSF fazem o uso de sensores inteligentes que possibilitam a detecção e o monitoramento de sistemas físicos (LI DA XU, W. H.; LI, S., 2013).

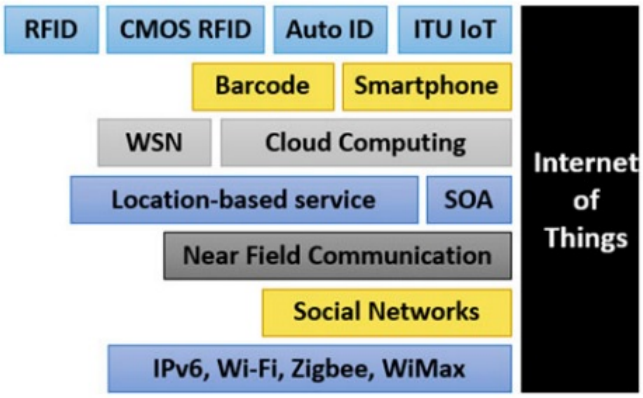


Figura 4 – Dispositivos que dão suporte a um sistema de IoT.

(LI DA XU, W. H.; LI, S., 2013)

A IoT vem ganhando bastante notoriedade e possibilitando avanços tecnológicos, como por exemplo na comunicação sem fio, permitindo assim ˝coisas" (objetos inteligentes) façam parte da IoT. Consequentemente, essas tecnologias relacionadas a IoT tiveram grande impacto nas novas tecnologia de TIC e CPS, possibilitando assim a implementação da Industria 4.0 (LI DA XU, W. H.; LI, S., 2013) capaz de desenvolver uma nova geração de sistemas de manufatura que integram e sincronizam dados em tempo real entre os modelos físicos e modelos virtuais (LI DA XU, E. L. X.; LI, L., 2018).

A Figura 5 apresenta os avanços em RFID, WSN e IoT, no qual podemos ver a partir de qual ano surgiu cada uma das tecnologias e exemplos de utilização.

2.2 Digital Twin

Para ser feita a implementação da Industria 4.0 é necessário o uso de dois elementos básicos, o primeiro elemento é o uso de dados do sistema para entender toda sua dinâmica de funcionamento e o segundo é um mecanismo que possibilite a convergência entre o modelo físico e o modelo virtual. Dessa forma, três problemas devem ser levados em consideração e solucionados para que seja possível a realização da convergência (TAO; ZHANG, M., 2017):

- Como construir um modelo virtual com alta fidelidade ao modelo físico?
- Como fazer o chão de fábrica e seu modelo virtual manter a consistência e sincronismo um com o outro e realizar uma otimização dupla?
- Como convergir os dados do modelo físico e do modelo virtual para gerar informações para a produção?

Logo, a implementação de uma Industria 4.0 necessita de mecanismos que possibilitem: a construção do modelo virtual altamente fiel ao modelo físico; a consistência e o

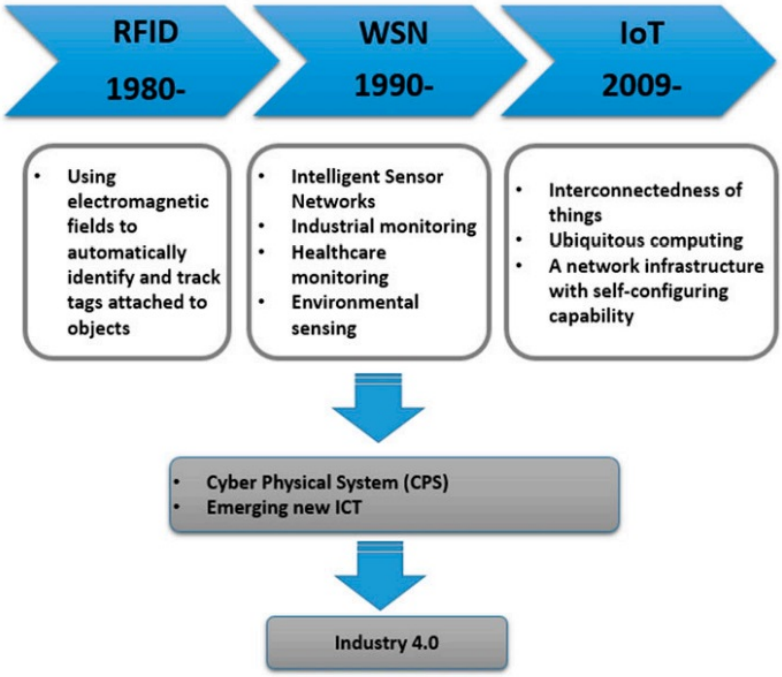


Figura 5 – O surgimento de novas tecnológias que possibilitaram o desenvolvimento do ICT e habilitam a indústria 4.0.

(LI DA XU, E. L. X.; LI, L., 2018)

sincronismo nos dados compartilhados e gerados por ambos modelos; e a convergência entre o modelo físico e o virtual. Um dos mecanismos criados é o DT (KAI DING; ZHANG, F., 2019), para realizar a convergência entre os modelos.

O termo DT foi primeiramente citado pelo pesquisador Grieves em 2003 (GRIEVES, 2015) na Universidade de Michigan, durante uma apresentação sobre Gerenciamento do Ciclo de Vida de Produto. Porém, com o passar dos anos a definição dada inicialmente para DT foi mudando até que em 2012 uma definição geral foi proposta por Glaessegen e Stargel onde definiram DT como:

> *A Digital Twin is an integrated multiphysics, multiscale, probabilistic simulation of an as-built vehicle or system that uses the best available physical models, sensor updates, fleet history, etc., to mirror the life of its corresponding flying twin (GLAESSGEN; STARGEL, 2013).*

O contexto dessa definição se deu no desenvolvimento de pesquisas voltadas para o futuro tecnológico dos veículos produzidos pela NASA e pela força aérea dos Estados Unidos da América. Porém, pode ser aplicado de forma geral a qualquer mecanismo de DT.

A Figura 6 ilustra a atuação do DT sendo utilizado no chão de fábrica, no qual o DT realiza a tarefa de gerenciamento dos três modelos: o modelo virtual, o modelo físico e o sistema de serviços. No modelo físico ele retorna feedback das operações que são realizadas nesse modelo, possibilitando que, as entidades que fazem parte do modelo, realizem ajustes

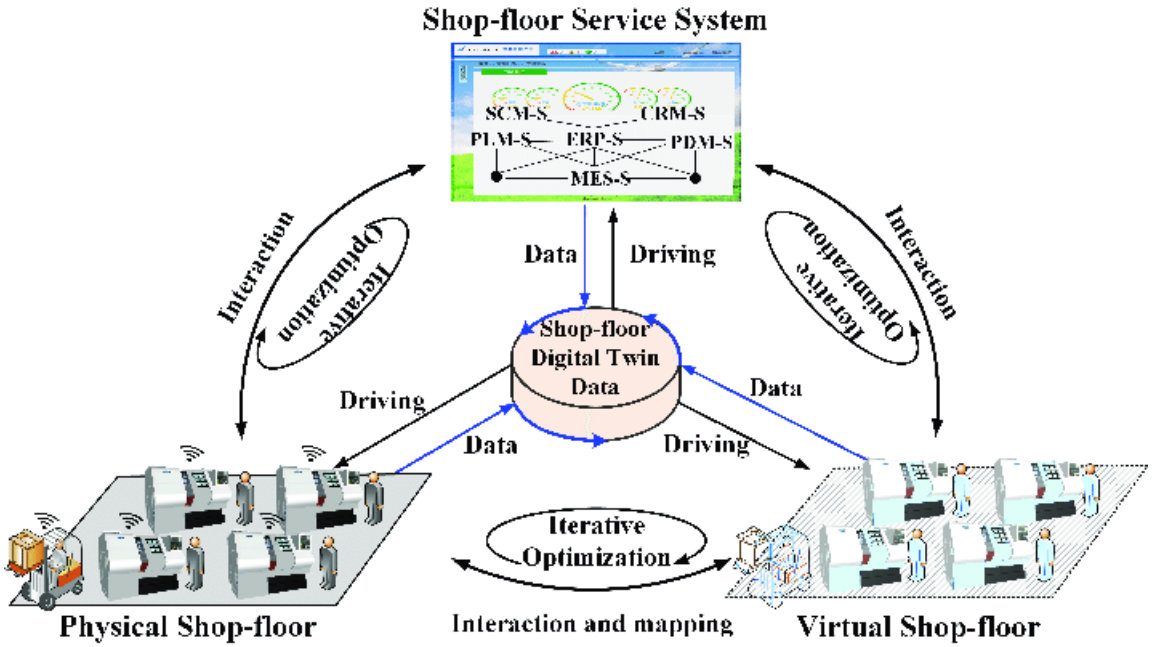


Figura 6 – Modelo conceitual de um DT no chão de fábrica.

(TAO; ZHANG, M., 2017)

nas operações que são realizadas. No modelo virtual é feita a modelagem do modelo físico e seus mecanismos de operação, sendo eles atualizados com base nas relações, restrições e regras definidas a partir dos dados obtidos do chão de fábrica. E por fim, no sistema de serviços os dados, algoritmos modelos, etc., obtidos do chão de fábrica, são ˝encapsulados" em sub-serviços para conduzir a composição do serviço e o processo de serviço subsequente (TAO; ZHANG, M., 2017).

De acordo com Tao (TAO; CHENG, 2017), a utilização de um DT possibilita nove aspectos para inovação de serviços no campo da manufatura:

a) Monitoramento do estado do sistema em tempo real;
b) análise e previsão do consumo de energia do sistema;
c) gerenciamento de usuário e análise de comportamento;
d) guia de operação do usuário;
e) otimização e atualização inteligentes;
f) análise e previsão de falha do produto;
g) estratégia de manutenção do produto;
h) manutenção virtual do produto;
i) operação virtual do produto.

A implementação de um DT em máquinas de comando numérico computadorizado (CNC) tem como objetivo possibilitar a utilização de ferramentas como auto-sensoriamento,

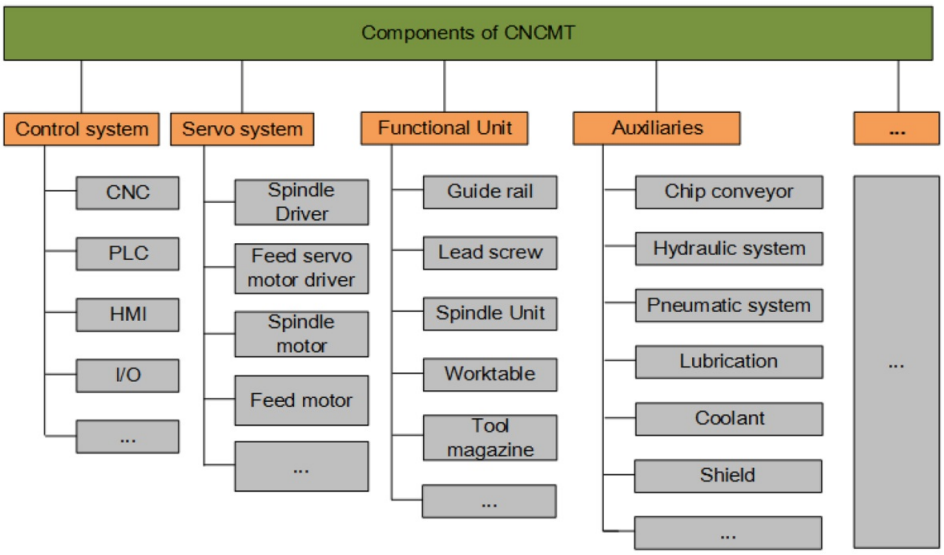


Figura 7 – Subsistemas que compõe um mecanismo de DT em uma máquina CNC.
(WEICHAO LUO; WEI, 2018)

auto-predição e automanutenção sem a necessidade de interferência humana, atuando como um cérebro do CNCMT possibilitando o monitoramento de falhas, análise de falhas e funções de tomada de decisão. O monitoramento de falhas monitora principalmente os parâmetros importantes do sistema, a análise de falhas visa localizar a fonte da falha e a tomada de decisão fornece previsão e solução de falhas (WEICHAO LUO; WEI, 2018).

Em uma estrutura de CNC é possível utilizar estratégias como dividir em vários subsistemas, pois cada elemento que compõem a máquina influência de certa forma o sistema como um todo, devendo assim possuir seu próprio DT para que sejam ligados em um só sistema posteriormente, contruindo um modelo virtual altamente fiel ao modelo físico (WEICHAO LUO; WEI, 2018).

2.3 Manufatura Aditiva

Manufatura aditiva (MA), também chamada de impressão 3D, tem ganhado um espaço importante na Industria graças a sua possibilidade de criar peças com geometrias complexas utilizando diversos tipos de materiais. Ela se baseia na manufatura de peças de camada em camada utilizando máquinas CNC chamadas impressoras 3D.

O interesse crescente nessa tecnologia tem feito com que avanços sejam realizados a fim de refiná-la e permitir que seja utilizada em diversas áreas, como por exemplo na medicina, na construção civil e no setor automobilístico (WEI GAO; ZAVATTIERI, 2015).

Entre as tecnologias de MA existentes, podemos destacar a tecnologia de FDM que utiliza como uma das suas matérias primas os termoplásticos. A FDM comparada a outras

tecnologias é mais barata e acessível, fazendo com que não só a industria se utilize dela, mas também o público em geral (ANDERSON, 2012). Sua aplicação vai desde peças de prototipagem até o produto final, porém seu uso profissional é mais voltado para prototipagem (CLAUS EMMELMANN JANNIS KRANZ; WYCISK, 2016) por, no estado atual, não oferecer uma qualidade no padrão que a Industria necessita e a escalabilidade (WEI GAO; ZAVATTIERI, 2015), porém estas desvantagens vem sendo atenuadas, sendo possível ver a industria utilizando peças feitas em FDM em seu produto final.

Como citado no Wei Gao e Zavattieri (2015), o principal impulsionador de mercado para tais sistemas tem sido consumidores e indústrias que dependem de prototipagem de baixa e média fidelidade nos estágios iniciais do projeto do produto."

Na MA faz-se o uso de impressoras 3D que são compostas por algum mecanismo de movimentação, um sitema de fundição e um sistema extrusão do material.

2.4 Manutenção

Manter o funcionamento correto dos equipamentos é essencial para garantir produtividade, qualidade e confiabilidade nos produtos que são manufaturados. E para ser possível garantir do funcionamento correto dos equipamentos algumas medidas podem ser tomadas, entre elas podemos destacar a manutenção dos equipamentos.

Realizar a manutenção de um equipamento pode ser definido como tarefas a serem realizadas a fim de restaurar o equipamento de tal forma que a função a qual é designado possa ser desempenhado novamente de forma eficiente (B.S. DHILLON, 2002). As estratégias de manutenção podem ser classificadas como Manutenção Corretiva, Manutenção Preventiva e Manutenção Preditiva (MP) (S.O. DUFFUAA M. BEN-DAYA; ANDIJANI, 2001).

2.4.1 Manutenção Corretiva

Manutenção Corretiva, também chamada de manutenção reativa, é uma estratégia que define a realização de atividades de manutenção após a ocorrência da falha, corrigindo o erro do equipamento ou substituindo-o (BENJAMIN S BLANCHARD; PETERSON, 1995), o que resulta em um alto tempo de inoperância do equipamento, diminuindo assim a produtividade, e ocasionando em altos gastos de manutenção (ROSMAINI AHMAD, 2012).

2.4.2 Manutenção Preventiva

Manutenção Preventiva é uma estratégia que define a realização de atividades de manutenção antes da falha do equipamento (WU, 2011), para que assim seja possível reduzir a frequência de falhas no equipamento, evitando a inoperabilidade do equipamento, diminuindo os gastos efetuados com reparo e/ou substituição (ROSMAINI AHMAD, 2012)

e possibilitando que a vida útil do equipamento aumente, além de garantir seu devido funcionamento. Ela é feita periodicamente, com o tempo entre as manutenções definido a partir das recomendações do fabricante ou a partir de dados coletados que permitam realizar um levantamento de vida útil do equipamento.

2.4.3 Manutenção Preditiva

Por volta de 1975 foi introduzida a MP, também chamada de manutenção baseada na condição, com a finalidade de maximizar a eficácia da tomada de decisão da manutenção preventiva (ROSMAINI AHMAD, 2012). ´´Manutenção Preditiva é um programa de manutenção que recomenda ações de manutenção (decisões) com base nas informações coletadas por meio do processo de mmonitoramento de condição", como foi apresentado em Andrew K.S. Jardine e Banjevic (2005). Ou seja, a partir do monitoramento é feito o levantamento das condições do equipamento, identificando quando acontecem diminuições de desempenho e peformance, além de indicar a causa dessa diminuição.

A MP é dividida em três etapas principais: aquisição de dados, processamento de dados e tomada de decisão (ANDREW K.S. JARDINE; BANJEVIC, 2005).

- Aquisição de dados: etapa responsável por obter dados sobre o estado do sistema. Essa aquisição pode ser feita de várias formas diferentes, utilizando uma vasta disponibilidade de sensores existentes.
- Processamento de dados: etapa responsável por processar e analisar os dados obtidos na primeira etapa a fim de detectar informações relevantes para realização da próxima etapa.
- Tomada de decisão: etapa responsável por direcionar a operação de manutenção a fim de tornar o processo o mais eficiente possível.

Um sitema de MP pode realizar diagnósticos, prognósticos ou ambos (ANDREW K.S. JARDINE; BANJEVIC, 2005).

- Diagnóstico: se refere a detecção, isolamento e identificação de falhas quando ocorrem. A detecção de falhas serve para identificar problemas que venham a ocorrer no

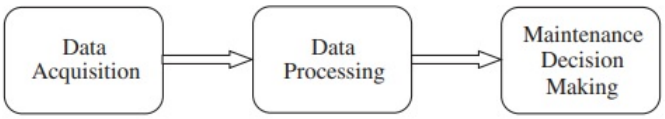


Figura 8 – As três etapas que compõem a MP.
(ANDREW K.S. JARDINE; BANJEVIC, 2005)

sistema; o isolamento de falhas serve para detectar o componente responsável pelo problema detectado; já a identificação de falhas serve para determinar qual a natureza do problema, por exemplo, se o componente queimou ou descalibrou.

- Proagnostico: se refere a previsão de falhas que possam vir a ocorrer, possibilitando prever quando a falha ocorrerá.

Logo, para ser possível a criação de um sistema de MP, é necessário a implementação de um sistema de monitoramento das condições do equipamento por meio de sensoriamento e/ou outros indicadores apropriados, possibilitando que a manutenção seja feita apenas quando necessário ou imediatamente antes da falha (ROSMAINI AHMAD, 2012).

Portanto, a implementação de um sistema de MP possibilitará que a máquina opere corretamente, garantindo qualidade e precisão nas peças manufaturadas. Porém, para a implementação desse sistema será necessário obter dados sobre o ambiente e sobre as peças que compõem o modelo físico.

Uma das formas de se aplicar o DT em um sistema de MP é atuando no sistema de acordo com o diagnóstico realizado, sendo o foco do sistema que foi desenvolvido, no qual as falhas são detectadas, o componente responsável pela falha é identificado e a natureza dessa falha é relatada, possibilitando assim que ocorra a atuação no sistema pausando ou cancelando a operação de impressão e notificando o operário sobre a falha que ocorreu.

2.5 Diagrama em V

Hoje em dia muito dos sistemas criados não são compostos por mais por apenas um sistema, mas por vários sistemas complexos integrados (ALEX GOROD; BOARDMAN, 2008). Logo, projetar esses sistemas e gerenciá-los se tornou uma tarefa com alta dificuldade, sendo assim necessário criar ferramentas que auxiliem o desenvolvimento desses sistemas conhecidos como Engenharia de Sistemas (ES).

Uma das definições para as ferramentas auxilia o desenvolvimento de um ES foi proposta no International Council on Systems Engineering (INCOSE), que descreveu o ES como:

> *a mean to enable the realization of successful systems" and "focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation while considering the complete problem". Moreover "it integrates all the disciplines and specially groups into a team effort forming a structured development process that proceeds from concept to production to operation". It "considers both the business and the technical needs of all customers with the goal of providing a quality product that meets the user needs (HASKINS, 2006).*

Em outras palavras, é uma ferramenta que auxilia no desenvolvimento de ES com foco nas necessidades dos Stakeholders.

Existem várias ferramentas que podem ser utilizadas para auxiliar o desenvolvimento desses sistemas, como por exemplo, o Modelo em Cascata, o Modelo em Espiral, o Modelo de Desenvolvimento Ágil, sendo que cada modelo possui suas particularidade e foco (FORSBERG; MOOZ, 1996). Uma das ferramentas criadas é o diagrama em V que tem a finalidade de auxiliar o planejamento, a execução, o monitoramento e controle das etapas de um ES(EUGENIO BRUSA; FERRETTO, 2017). O Diagrama em V, também chamado de Modelo Vee, foi desenvolvido pela NASA e apresentado no simpósio INCOSE de 1991 em Chattanooga, Tennessee (FORSBERG; MOOZ, 1996).

Baseado no Modelo em Cascata, ele é composto por duas partess de tal forma que seu formato se assemelha a letra ´´V". A partes esquerda da forma em V tem seu fluxo feito do topo da partes esquerda até a base do V, representando o processo de decomposição do sistema e definição, ou seja, representa a etapa de: mapeamento e levantamento de necessidades das partes interessadas, os Stakeholders; definição dos requisitos a partir das necessidades levantadas; definição e especificação do sistema de engenharia a ser implementado; decomposição em subsistemas; e definição e especificação dos componentes. Já a partes direita da forma em V representa a implementação, integração, veriicação e teste do componentes que compõem o sistema (RUPARELIA, 2010). A partes direta tem seu fluxo feito da base até o topo da partes direita, representando o processo de integração e verificação do sistema, onde em cada nível realizado é realizada a validação contínua com os Stakeholders e a avaliação contínua de riscos e oportunidades (FENGMING CUI; LI, L., 2018). A Figura 9 ilustra a composição do Diagrama em V.

Portanto, podemos definir as etapas a serem seguidas como:

1°. Mapeamento dos Stakeholders: nesta é feito o mapeamento dos Stakeholders que são definidos como ´´pessoas ou sistemas que tenham algum tipo de interesse com o sistema analisado", como foi apresentado em Haskins (2006), ou seja podem ser desde o cliente propriamente dito, tornando o processo de levantamento de necessidades mais eficaz; operadores; ou sistemas que serão acoplados ao sistema projetado.

2°. Levantamento de necessidades dos Stakeholders: a partir de pesquisas diretas ou indiretas com os Stakeholders, é feito o levantamento do que eles esperam do sistema, sejam operações que esperam que o sistema consiga efetuar, ou uma interface que possua as informações que os stakeholders necessitam, ou qualquer outro ´´desejo" que eles queiram que seja sanado.

3°. Definição dos requisitos: nesta etapa é feito uma análise das necessidades levantadas para que dessa forma seja possível converter as necessidades em um requisito do

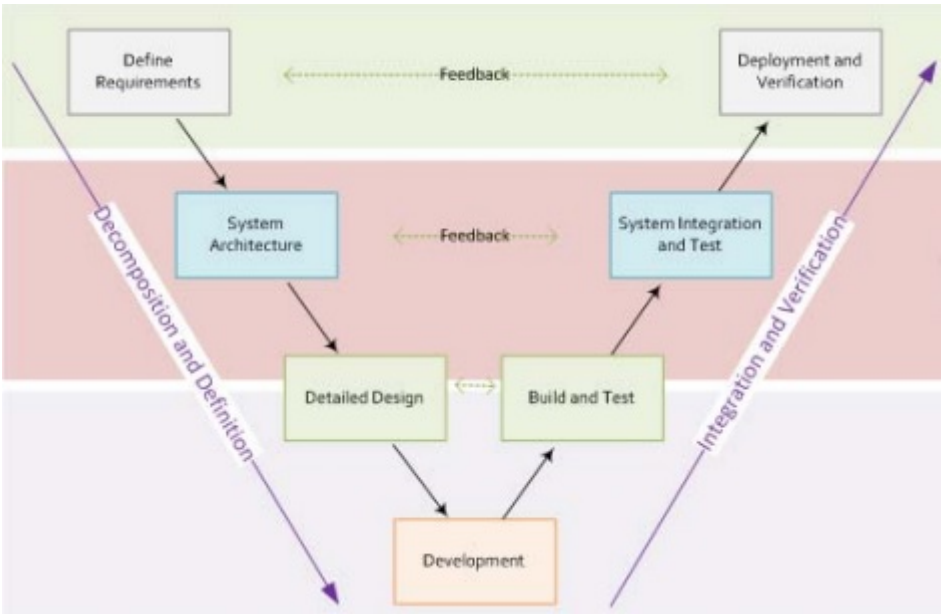


Figura 9 – Ilustração do Diagrama em V.

(RUPARELIA, 2010)

sistema claro e executável (EUGENIO BRUSA; FERRETTO, 2017), sendo a etapa necessária para ser seguir com as outras etapas.

Os requisitos do sistema são descritos como ´´a statement that identifies a product or process operational, functional or design characteristic or constraint which is unambiguous, testable or measurable and necessary for product or process acceptability.”, como foi apresentado em Forsberg e Mooz (1996)

4°. Modelagem do sistema: nesta etapa é feita a analise operacional, funcional, lógica e física do sistema para definir assim a estrutura que será utilizada, a lógica de funcionamento, os componentes que serão utilizados, a função de cada componente e como será feita a integração deles (FENGMING CUI; LI, L., 2018).

5°. Implementação: etapa no qual o sistema é construído, consistindo na programação, modelagem, fabricação e testes de cada parte que irá compor o sistema individualmente.

6°. Testes e verificação: Durante a modelagem do sistema, a implementação e a realização dos testes é feita a verificação para determinar se o que está sendo desenvolvido está de acordo com as necessidades obtidas. Cada componente escolhido para compor o sistema deve ser testado separadamente e feita a verificação com os stakeholders que retorão um feedback sobre o que foi implementado e, caso não esteja de acordo com as necessidades do cliente, são feitos ajustes para ficar de acordo com o desejado. Após todos os componentes serem testados separadamente, é feita a integração de todos os componentes e são realizados novos testes e a verificação com o stakeholder já com o

sistema completo, e, semelhante ao que acontece com os componentes separados, os stakeholders retorão um feedback sobre o que foi implementado e, caso não esteja de acordo com as necessidades do cliente, são feitos ajustes para ficar de acordo com o desejado (FENGMING CUI; LI, L., 2018).

3 Desenvolvimento

Após definir os conceitos que serão necessários para dar continuidade ao desenvolvimento, segue-se para a parte da aplicação. A aplicação consiste em implementar o sistema de manutenção preditiva com base nos conceitos de DT apresentados, focando principalmente no monitoramento e na atuação do sistema.

Para desenvolver o sistema de manutenção preditiva que irá compor o mecanismo de DT foi feito o uso do Diagrama em V para auxiliar no desenvolvimento das etapas do projeto. O Diagrama em V é voltado para projetos grandes que possuem várias equipes de desenvolvimento (RUPARELIA, 2010), foram feitas simplificações que permitem a utilização para desenvolvimento do sistema em questão, simplificando algumas etapas.

3.1 Mapeamento dos stakeholders

O primeiro passo do Diagrama em V seguido se referia ao mapeamento dos stakeholders. Os stakeholders desse projeto vai desde a comunidade maker até empresas de grande porte que produzem impressoras 3D, como é o caso da empresa Prusa, por ser um sistema que tem potencial de impactar a manufatura aditiva. Porém, a principal parte interessada no projeto é o LAB, desta forma foram definidos os stakeholders do sistema a ser desenvolvido que possuem ligação direta ao LAB. Portanto, os stakeholder foram mapeados e podem ser visualizados na Tabela 1, que contém qual o Stakeholder e sua descrição.

Tabela 1 – Stakeholders.

stakeholders	**Descrição**
Equipe de projetos do LAB	Membros do LAB voltados para a execução de projetos requisitados por clientes. Sua participação pode ir desde a concepção do projeto até o produto final, possuindo um relatório detalhado dos processos envolvidos durante o período em que o projeto esteve ativo.
Equipe de operação do LAB	Membros voltados para a execução das operações de manufatura e manutenção. Possuem acesso às máquinas disponíveis para a manufatura de demandas internas ou externas, sendo uma dessas máquina a impressora 3D. Desta forma dominam toda a parte técnica envolvendo o maquinário em questão.
Gestão de estoque do LAB	Membro responsável pelo controle de estoque, compras de materiais, levantamento de custos e gerenciamento de toda a parte financeira do LAB.
Gestão de operação do LAB	Membro responsável por gerenciar todo o processo realizado com um cliente, definindo a viabilidade dos projetos, todo o planejamento a ser seguido, o maquinário a ser utilizado, o orçamento do projeto, os membros que formarão a equipe de projetos e a equipe de operação, entre outras funções chaves para o funcionamento do LAB.
Cliente do LAB	Como cliente do LAB foi escolhido a ORIGEM que é voltada para aluguel de motos elétricas. O serviço prestado pelo LAB é a manufatura de peças em impressão 3D que são utilizadas nas motos.

Fonte: Produzido pelo autor

Após realizar o mapeamento dos stakeholders, é possível seguir para a próxima etapa, seguindo o fluxo de desenvolvimento proposto pelo diagrama em V.

3.2 Levantamento das necessidades

Nessa etapa é feito o levantamento das necessidades apresentadas pelos stakeholders, para que seja possível direcionar o projeto de forma que atenda a suas demandas, agregando valor ao produto final.

Existem várias estratégias para realizar o levantamento das necessidades dos clientes, porém foi escolhido nesse projeto realizar o levantamento das necessidades por meio de uma entrevista direta com um ou mais representantes de cada stakeholder mapeado, na qual perguntas foram feitas com a finalidade de auxiliar os entrevistados e adquirir informações para ser feito o desenvolvimento. A escolha dos representantes foi com base na experiência que a pessoa possui na sua área, sendo que apenas o cliente do LAB foi representado por uma pessoa, enquanto que a equipe de projeto e a equipe de operação foram representados por três pessoas e a gestão de operação e a gestão de estoque foram representados por duas pessoas.

As respostas dos entrevistados foram armazenadas em um formulário do Microsoft Teams que contêm as perguntas utilizadas para realizar a pesquisa. Vale ressaltar que a perguntas foram feitas tomando cuidado para não influenciar os entrevistados de forma alguma. É possível ter acesso as perguntas feitas acessando o site `https://forms.office.com/r/LkFuWBFN5r`.

A partir das respostas obtidas foi possível obter justificativas acerca da necessidade de criar um sistema de manutenção eficaz. Os motivos apresentados de forma geral foram:

a) A indisponibilidade de uma impressora 3D causada pela necessidade de manutenção tem a possibilidade de fazer com que alguns projetos, que necessitem de um protótipo que possibilite a realização ajustes e/ou validar a etapa em que o projeto está, seja inviabilizado até que a impressora 3D volta a operar normalmente.

b) A falta de manutenção impacta negativamente a qualidade mecânica das peças fabricadas, causando problemas, como por exemplo uma diminuição da resistência a forças de tração.

c) A falta de manutenção impacta negativamente a qualidade superficial das peças, possibilitando o surgimento de brechas entre as camadas, artefatos na superfície, etc.

d) A confiabilidade é afetada pela falta da manutenção das impressoras, pois as chances de erros ocorrerem durante a operação aumentam.

e) A repetibilidade impressões feitas são afetadas com a falta de manutenção.

Além de obter justificativas para a realização do projeto, também foram obtidas as necessidades que os stakeholders possuem para que sejam sanadas pelo sistema de manutenção preditiva. As necessidades apresentadas foram:

- Histórico de falhas: possibilitando acompanhar a quantidade de manutenções realizadas em cada impressora, os tipos de erros, quem realizou a manutenção, entre outras informações relevantes que possam ser utilizadas posteriormente para otimizar tanto o processo de impressão de peças quanto a operação de manutenção a ser realizada. Também possibilita a criação de relatórios periodicamente que são necessários para gerenciar toda de forma eficaz a produção e o estoque de material;
- Última manutenção feita na impressora 3D: a informação da última manutenção feita em cada impressora possibilita realizar o levantamento de informações como quanto tempo em média leva para realizar cada manutenção, as impressoras que apresentam maior ocorrência de falha, correlacionar o tipo de falha com a frequência que ela ocorre, etc.;
- Erro ocorrido: possibilitando o mapeamento dos erros que podem ocorrer de forma a criar posteriormente soluções que possam reduzir a ocorrência do erro ou até mesmo impedir com que aconteçam;
- Causa do erro ocorrido: mapear as causas dos erros e realizar uma correlação entre o erro e sua causa.
- Validação da manutenção feita na impressora 3D: realizar testes para poder validar a manutenção feita na impressora, evitando assim que impressoras com problemas voltem para a linha de produção.
- Tempo de inatividade por causa da manutenção da impressora 3D: medir o tempo que foi necessário para realizar a manutenção para assim poder levantar informações como eficiência na operação de manutenção, custos de manutenção, custos de inatividade, etc.;
- Custos para realização da manutenção: calcular os custos relacionados a manutenção que leva em conta os custos de material utilizado na operação, custo de mão de obra, custos de inatividade, entre outros custos que podem entrar no cálculo, possibilitando assim um gerenciamento melhor do LAB;
- Complexidade da manutenção: fazer o levantamento da complexidade das manutenções realizadas, para assim ser possível correlacionar o erro com a complexidade para solucioná-lo e os custos envolvidos.

- Check list de manutenção: um check list com as atividades que devem ser realizadas durante a operação de manutenção que possibilite um processo de manutenção mais eficaz;
- Estado da impressora 3D na área de manutenção (Esperando na fila, sendo feita manutenção ou liberada): monitorar o estado da impressora enquanto está em manutenção para um melhor gerenciamento da produção, possibilitando a alocação eficiente das impressoras;
- Quem está realizando a manutenção: informação necessária para acompanhar a performance de cada operador a fim de realizar treinamentos e outras atividades que melhorem a operação de manutenção.
- Periodicidade das manutenções: o levantamento da periodicidade das manutenções permite um controle maior da produção e do estado da impressora, aumentando sua vida útil.
- Quantidade de manutenção por impressora 3D: possibilitando saber quais impressoras possuem uma maior ocorrência de manutenções para que sejam tomadas providências;
- Peças utilizadas nas manutenções: necessário para o levantamento de custos e gerenciamento do estoque, pois peças que possuem uma taxa de falha maior devem ter uma maior quantidade em estoque;
- Extrusão de filamento: monitorar a extrusão de filamento possibilita monitorar falhas de extrusão de que possam vir a ocorrer. Esta informação é importante, pois, quando falhas ocorrem, na maioria das vezes uma grande quantidade de material e de tempo de operação são desperdiçados;
- Nivelamento da mesa de impressão: monitoramento de extrema importância, pois falhas na primeira camada de impressão podem ocasionar falhas superficiais ou mecânicas na peça impressa, ou problemas mais graves como descarte da peça, entupimento do bico ou quebra de alguns componentes que fazem parte do extrusor (sensor de temperatura, sensor de nivelamento, garganta, ventoinha, etc.);
- Perda de passo durante impressão: a perda de passo durante a impressão, na maioria dos casos, faz com que a peça impressa seja descartada, desperdiçando material e tempo. Sendo um problema comum e de difícil detecção;
- Tensão da correia: a tensão na correia influência diretamente na qualidade superficial, no erro dimensional e nas propriedades mecânicas da peça impressa. Portanto, possuir uma forma de monitorá-la possibilitaria um controle maior da qualidade das peças produzidas;

- Estrutura firme: a qualidade superficial das peças impressas sofre influência de muitas variáveis do sistema que compões a impressora 3D, uma delas é a vibração originada da estrutura utilizada na impressora. Logo, criar formas de garantir que a estrutura gere a menor quantidade de vibração possível afetaria a qualidade das peças produzidas;
- Interface gráfica para acompanhamento: a criação de uma interface gráfica amigável para facilitar a interação homem máquina;
- Planilha do Teams: utilizar o Teams para centralizar toda a informação é uma das demandas do LAB, pois é a plataforma que fazem o uso.

A partir do levantamento das necessidades, é necessário agora transformá-los em requisitos técnicos, possibilitando verificar a viabilidade e, aqueles que forem viáveis, as formas que podem ser desenvolvidos.

3.3 Definição dos requisitos

Possuindo as necessidades dos stakeholders, podemos agora definir os requisitos que o sistema necessita cumprir para estar de acordo tais necessidades.

- Banco de dados do histórico de manutenção: banco de dados com algumas informações importantes sobre a operação para auxiliar na gestão do LAB;
  - Última manutenção
  - Erro
  - Causa do erro
  - Validação da manutenção
  - Tempo de inatividade
  - Complexidade
  - Custos
- Banco de dados de peças utilizadas nas manutenções: pois possibilita uma gestão do estoque mais eficiente, possibilitando o levantamento de custos de manutenção;
  - Peças utilizadas nas manutenções
  - Custo de cada peça
  - Custo total
- Banco de dados do histórico da manutenção em execução: pois o registro do que foi feito na manutenção é um processo importante para poder mapear os erros, causas e as possíveis soluções;

  - Check list
    * O que deve ser verificado?
    * Validações para liberação
  - Status na área de manutenção (Aguardando, sendo feita manutenção ou liberada)
  - Quem está realizando a manutenção?

- Relatório de manutenções: a entrega de relatórios de manutenção periodicamente que possibilite ser feita uma análise para que possa ser realizada uma gestão melhor dos processos, da utilização das impressoras 3Ds, e também permitir otimizar as operações envolvendo a impressora 3D;
  - Histórico de falhas
  - Periodicidade das manutenções
  - Quantidade de manutenções por impressora 3D
  - Peças utilizadas nas manutenções (trocadas)
- Sensoriamento: sensoriamento da impressora 3D que possibilite adquirir informações sobre a situação em tempo real das impressoras 3Ds, além de possibilitar a construção do seu Digital Twin.
  - Extrusão de filamento
  - Nivelamento da mesa
  - Perda de passo
  - Tensão na correia
  - Problemas estruturais (parafuso necessitando de aperto, por exemplo)

Com a definição dos requisitos, é possível agora direcionar o desenvolvimento do sistema. Foram levantados um grande número de requisitos, contudo devido o tempo de desenvolvimento do projeto, priorizou-se o requisito do sensoriamento, pois será necessário para adquirir dados para dar prosseguimento na implementação do Digital Twin. Portanto para construção do sistema foi focado no sensoriamento da impressora 3D e aquisição de dados, permitindo assim o processamento inicial dos dados obtidos para construção do Digital Twin, possibilitando a implementação de um sistema de manutenção preditiva, além de ser um direcionamento para trabalhos futuros que irão continuar o desenvolvimento do Digital Twin no LAB.

3.4 Projeto do sistema

Com o foco definido a partir dos requisitos, é necessário ser feita a escolha dos componentes, da arquitetura e das API'S que irão compor o sistema.

Portanto, com a finalidade de criar um sistema de manutenção preditiva baseado no mecanismo de DT a ser desenvolvido no LAB, nesta seção serão abordadas os componentes que fazem parte do sistema, a escolha do framework utilizado para gerenciar o sistema, a forma com que os componentes foram acoplados ao sistema, o funcionamento de cada componente separadamente e o funcionamento como um todo.

3.4.1 Componentes

Vamos iniciar descrevendo cada componente que compões o sistema, na seguinte sequência:

- Impressora 3D: serão apresentadas informações sobre a impressora utilizada para o desenvolvimento do sistema;
- Raspberry pi: utilizado para possibilitar o uso do framework;
- Acelerômetro: foi utilizado para realizar um estudo para verificar a possibilidade de criar uma relação entre a vibração do extrusor que faz parte da impressora 3D com a tensão da correia;
- Sensor de filamento inteligente: foi utilizado para realizar a detecção da presença e da movimentação do filamento, possibilitando identificar erros de extrusão que possam vir a ocorrer;
- Câmera: foi utilizada para monitorar a impressão com o auxílio de um plugin fornecido pelo framework que possibilita a detecção de problemas na impressão durante sua produção utilizando inteligência artificial;
- BLTouch: sensor bastante utilizado para realizar o nivelamento da mesa, que foi utilizado não só para realizar o nivelamento, mas também para identificar a propagação do erro no nivelamento da mesa que venha a possibilitar identificar quando foi necessário realizar a manutenção para ajustar seu nivelamento;
- Planilha do Google: foi utilizada para armazenar algumas informações sobre a impressora e também armazenar os dados coletados pelo BLTouch;
- Pushbullet: ferramenta que foi utilizada para fazer uma ponte entre o framework e o programa que implementa o sistema de DT, como visto na Figura 6;

- The Spaghetti Detective: serviço disponibilizado pelo framework que faz uso da câmera para detectar problemas na impressão durante sua produção utilizando inteligência artificial;

Cada componentes foi abordado separadamente, explicando como funcionam, como são configurados e para qual finalidade.

3.4.1.1 Impressora 3D

Um dos objetivos desse trabalho é desenvolver um sistema de manutenção preditiva para manufatura aditiva baseado no Digital Twin.

Para o desenvolvimento do sistema foi utilizada uma impressora 3D baseada na tecnologia de Fused Deposition Modeling (FDM), que possui seu funcionamento e componentes semelhantes as impressoras 3D utilizadas no LAB. A escolha de utilizar uma impressora que não fosse do LAB ocorreu para evitar utilizar trasnporte público no período de pandemia, já que a impressora utilizada é do próprio autor do trabalho. A utilização da impressora em questão foi feita para que não fosse necessário a locomoção até a UnB com muita frequência, ficando menos exposto ao contágio do Covid-19.

A impressora 3D utilizada é de fabricação lobista (caseira) baseada no projeto de impressora 3D chamado Hypercube Evolution (Hevo), possuindo todas as funções necessárias para o projeto. O projeto da impressora 3D Hevo é open-source, possuindo uma vasta comunidade que realiza melhorias na impressora e as compartilham na internet. O projeto completo pode ser encontrado no site `https://www.thingiverse.com/thing:2254103`, que é um dos sites mais utilizados para compartilhamento de arquivos voltados a impressão 3D. No site é possível encontrar a lista completa de materiais que são necessários para a construção da impressora, como parafusos, perfis de alumínio, hotend, mesa aquecida, entre outras peças que a compõem, assim como instruções de montagem e arquivos com extensão ˝.stl” que podem ser impressas em outra impressora 3D.

A Figura 10 demonstra o modelo 3D da impressora utilizada.

O LAB utiliza impressora 3Ds do modelo Prusa I3 MK3S+ (Prusa Researc, Praga, República Tcheca) e do modelo Ender-5 Plus (Creality, Shenzhen, China). Ambas impressoras fazem o uso do mecanismo de movimentação chamado de ˝cartesiano”, enquanto que a utilizada no desenvolvimento faz o uso do mecanismo de movimentação chamado ˝coreXY” que possibilitar alcançar uma velocidade máxima maior sem afetar a qualidade e a precisão das peças produzidas também é maior.

A Figura 11 é uma ilustração da impressora Prusa I3 MK3S+ e a Figura 12 uma ilustração da impressora Ender-5 Plus. O sistema de manutenção desenvolvido foi feito de forma que possa ser facilmente implementado nesses dois modelos de impressora impressoras.

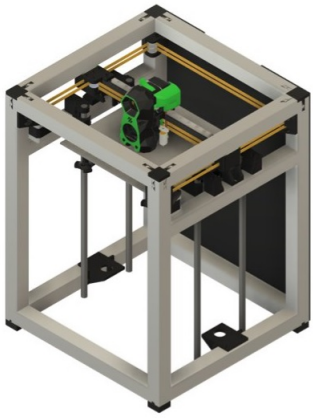


Figura 10 – Foto da impressora 3D Hevo utilizada no projeto.

Fonte: Produzido pelo autor

Na MA existem alguns mecanismos de movimentação que podem ser utilizados, porém a ″coreXY" e a ″cartesiano" são as mais comuns de serem encontradas. A seguir tem a explicação sobre cada um desses mecanismos.

a) CoreXY: O mecanismo de movimentação chamado ″coreXY" possui como principal característica o fato de que os motores responsáveis pela movimentação no plano XY trabalham em conjunto, tornando o torque mais estável (ZHU; LEE, J.-S., 2019), possibilitando também a redução de peso no plano XY. Utilizando esse mecanismo, no caso de uma impressora 3D, o eixo Z possui um ou mais motores dedicados para sua movimentação, não havendo correlação com a movimentação dos eixos X e Y.

b) Cartesiano: O mecanismo de movimentação chamado de ″cartesiano", também conhecido como retilíneo ou ortogonal, possui sua movimentação realizada por um ou mais motores dedicados a um único eixo, sendo eles X, Y ou Z, não existindo correlação na movimentação entre os eixos (WIKTOR, 2020).

Figura 11 – Figura ilustrativa da prusa i3 MK3s+.

Fonte: (PRUSA RESEARCH, s.d.)

Figura 12 – Figura ilustrativa da ender-5 Plus

Fonte: (CREALITY, s.d.)

Tendo em vista as diferenças dos mecanismos de movimentação, vale ressaltar que o ″coreXY" possui diferenças estruturais em relação ao ″cartesiano", entretanto tais diferenças não possuem influência significativa no desenvolvimento do sistema. Portanto, é possível utilizar todo o sistema implementado na Hevo em qualquer outra impressora 3D FDM sendo necessário apenas se atentar a forma com que os sensores utilizados serão acoplados, pois podem existir diferenças estruturais que tornam necessário remodelar as peças utilizadas no acoplamento dos sensores.

Na parte de software da impressora foi utilizado o firmware do Klipper. O firmware serve para transformar os comandos enviados pelo software em comandos para o hardware, possibilitando desta forma movimentar os motores da impressora, liberar energia para o bloco aquecedor e a mesa aquecida, entre outras ações que a impressora executa. Os comandos que iremos utilizar são básicos suportados pela maioria dos firmwares disponíveis.

3.4.1.2 Raspberry Pi

Foi utilizado um Raspberry Pi 3 Model B que é um computador de bolso desenvolvido pela Raspberry Pi Foundation, permitindo seu uso em várias aplicações (RASPBERRY FOUNDATION, s.d.). O raspberry utilizado conta com 4 portas USB, conexão blutooth e conexão WiFi.

Este componente permite a utilização do Octoprint, pois cobre os requisitos mínimo, sendo um dos dispositivos recomendados no próprio site do Octoprint, sendo que seu download pode ser feito a partir do link `https://octoprint.org/download/`.

Outro ponto positivo do uso do Raspberry é a disponibilidade de portas USB, possibilitando o uso de componentes que necessitem de conexão USB para funcionar.

Sua escolha foi feita por ser uma plataforma aberta que poderá ser utilizada em trabalhos futuros para implementar um sistema completo de Digital Twin.

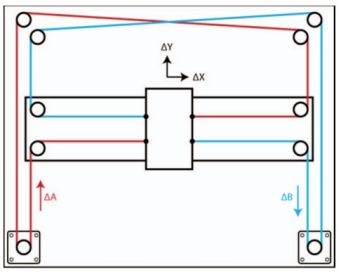


Figura 13 – Ilustração do mecanismo de movimentação CoreXY.

Fonte: (ZHU; LEE, J.-S., 2019)

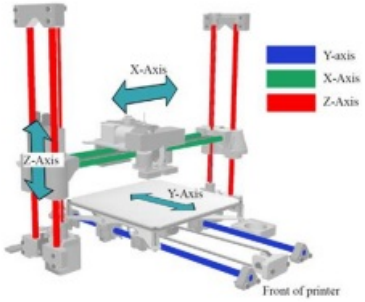


Figura 14 – Ilustração do mecanismo de movimentação Cartesiano.

Fonte: (CARNEIRO; SOUZA TAVARES, 2021)

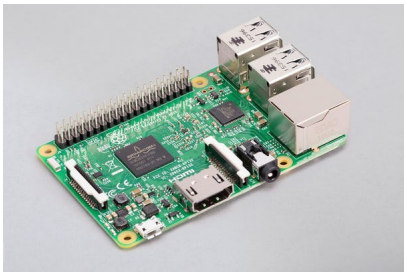


Figura 15 – Imagem representativa do Raspberry Pi 3 Model B.

Fonte: (WAGNER PEDRO, 2022)

Para utilizá-lo foi necessário seguir os seguintes passos:

1°. Passo: foi baixado no site `https://www.raspberrypi.com/software/` o instalador para o Windows do sistema operacional que foi instalado no raspberry;

2°. Passo: ao executar o instalador foi escolhida primeiramente a opção 'CHOOSE STORAGE' e selecionado o cartão SD de 16Gb, logo após foram escolhidas as opções 'CHOOSE OS', 'Other specific-purpose OS', '3D printing' e 'OctoPi' sequencialmente. Desta forma o Octoprint é instalado automaticamente no cartão SD que é inserido no raspberry;

3°. Passo: instalado o programa PuTTY que possibilita acessar o terminal do raspberry remotamente e o programa Advanced IP Scanner para poder descobrir o ip que o raspberry está utilizando, desde que estejam na mesma rede de internet. A Figura 16 ilustra a interface inicial do PuTTY, no qual é necessário entrar com o endereço de IP descoberto. A Figura 17 ilustra a interface do Advanced IP Scanner;

4°. Passo: foi seguido o tutorial para instalar o firmware da impressora 3D, as instruções podem ser encontradas no site `https://www.klipper3d.org/Installation.html`.

Desta forma teremos o raspberry devidamente configurado com o framework que foi utilizado, sendo ele o Octoprint.

3.4.1.3 Acelerômetro

Foi utilizado um acelerômetro de 3 eixos chamado ADXL345. Este componente possui dimensões pequenas e uma baixa potência de utilização, entregando uma alta resolução de medição. Os dados de saída digital são formatados como complemento de dois de 16 bits e podem ser acessados por meio de uma interface digital SPI ou I2C. A Tabela 2 possui as especificações do dispositivo.

Este sensor possui a finalidade de obter a vibração no extrusor da impressora 3D, e, a partir dos dados coletados, avaliar a correlação entre a tensão da correia utilizada para

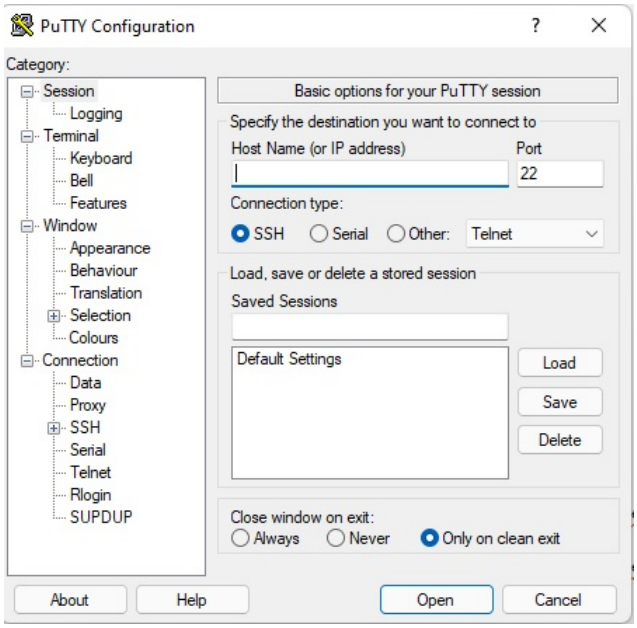


Figura 16 – Interface do programa PuTTY.

Fonte: Produzido pelo autor

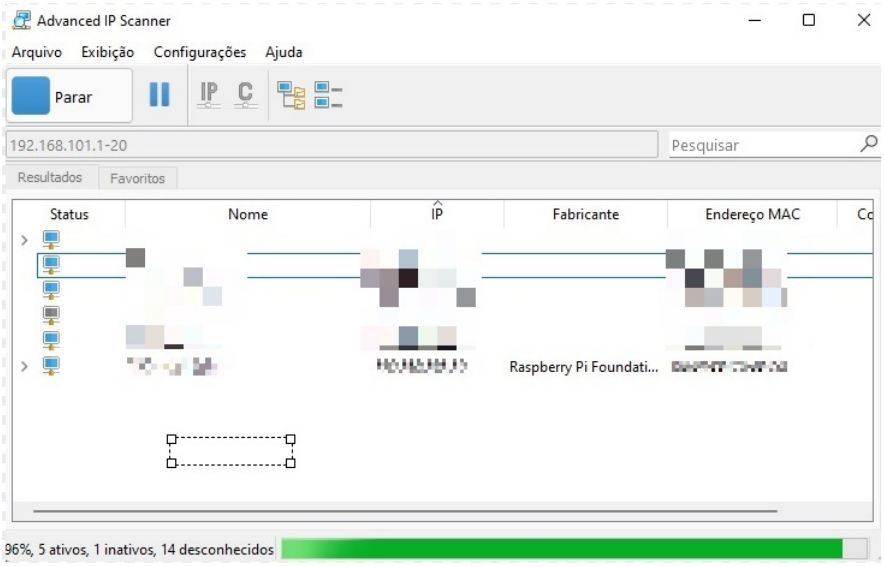


Figura 17 – Interface do programa Advanced IP Scanner.

Fonte: Produzido pelo autor

movimentar o extrusor e a vibração do extrusor, para que desta forma seja possível identificar quando ajustes na tensão vierem a ser necessários.

A partir desses dados, melhorias na estrutura poderão ser propostas e ajuste na tensão da correia poderão ser feitas de forma a melhorar a qualidade das peças produzidas na impressora, pois as vibrações que ocorrem durante o processo de impressão possuem grande influência na qualidade das peças impressas, seja mecanicamente quanto superficialmente

Tabela 2 – Especificações do ADXL345

ADXL345	
Interface	I2C/SPI
Quantidade de eixos	3
Sensibilidade	±2 g/±4 g/±8 g/±16 g
Consumo	25 a 130 μA em 2,5V
Tensão de alimentação	1.8V a 3.6V
Dimensão	20 x 15 mm

Fonte: (ANALOG DEVICES, s.d.)

(MENDERES KAM; SARUHAN, 2018).

A programação do acelerômetro foi feita utilizando python, em que durante sua execução realiza uma chamada de sistema que executa um código em C responsável por coletar os dados do acelerômetro para cada eixo (X, Y e Z) e armazena-los em um arquivo com extensão ´´.csv". A coleta dos dados é realizada em C para evitar que a variação do tempo entre cada coleta seja alta a ponto de invalidar os dados coletados. Após coletar os dados o programa em python continua sua execução, onde o arquivo gerado para armazenas os dados é aberto e o valor da componente DC de cada eixo é subtraída dos dados do seu respectivo eixo, sendo a componente DC a média aritmética dos valores de cada eixo, permitindo desta forma aplicar uma ´´Fast Fourier transform" nos valores de cada eixo criando em seguida uma imagem do gráfico gerado para possa ser analisado.

Os códigos em python e em C podem ser encontrados na seção A.2 e na seção A.3, respectivamente.

A partir da análise dos gráficos é possível validar a tese de que existe uma correlação entre a tensão da correia que movimenta o extrusor com a vibração do mesmo. Portanto, dois cenários foram analisados:

a) Tensão correta na correia: ajustando a correia da impressora de forma correta para obter os dados.

b) Tensão menor na correia: reduzindo a tensão na correia de forma que ainda funcione a impressora para obter os dados da impressora em funcionamento e,

Figura 18 – Imagem representativa do ADXL345.

Fonte: (ELECTROPEAK, 2022)

ao invés de verificar problema superficiais que apareceram na peça, é observado os dados obtidos.

A escolha de obter os dados com a correia com uma tensão menor e não maior foi feita porque, com o uso da impressora, não é possível ter um aumento na tensão sem que algum aconteça uma intervenção externa.

3.4.1.4 Sensor de filamento inteligente

Para realizar o sensoriamento da extrusão foi utilizado o sensor de filamento inteligente da Bigtreetech. Diferentemente de um sensor que não é dito ´´inteligente", no qual detecta apenas existência de filamento, possui também um encoder rotacional que detecta a existência do fluxo de filamento passando pelo sensor (BIGTREETECH, s.d.), possibilitando assim determinar quando não há a presença de filamento ou quando o extrusor está entupido.

Sua conexão pode ser feita diretamente na impressora 3D ou no Raspberry pi.

Foi optado a utilização diretamente no raspberry, pois não depende da placa de controle, que a impressora 3D utiliza, ter suporte para sensor, assim possibilitando a utilização do sensor em qualquer impressora 3D.

A configuração do sensor é feita diretamente no Octoprint através de um plugin chamado ´´Smart Filament Sensor". Podemos visualizar na Figura 21 a interface do plugin que foi feita a configuração, na qual as seguintes configurações podem ser feitas:

- Board Pin Mode: escolhe entre as duas opções de mapeamento dos pinos do raspberry, na qual a opção ´´Board Mode"utiliza a pinagem mais comum, onde em um dos lados ficam os pinos impares e no outro lado ficam os pinos pares. Já a opção ´´BCM Mode" utiliza o mapeamento dos pinos chamados GPIO do raspberry, que altera a posição de alguns pinos, não havendo mais a separação entre pares e impares. A Figura 20 demonstra como é mapeado tanto no ´´Board Mode" quanto no ´´BCM Mode";

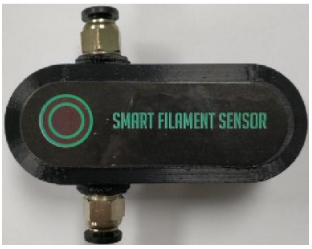


Figura 19 – Imagem representativa do sensor de filamento inteligente da Bigtreetech.

Fonte: Produzido pelo autor

- Detection Method: nesta configuração são fornecidas duas opções: Timeout detection e Distance detection. No Timeout detection a detecção da existência de fluxo de filamento é feita caso após um valor T de tempo passar e não ocorrer ser detectado fluxo de filamento, enquanto que no Distance detection utiliza a distância percorrida pelo extrusor da impressora 3D para realizar a detecção. Os próprios desenvolvedores do plugin recomendam utilizar a opção Distance detection, por ser mais preciso;
- Pause command: nesta opção é feita a escolha do comando que é executado quando é detectado problemas de extrusão. Porém foi apenas utilizado a opção ˝M601 - Pause print", pois a decisão do que vai ser feito quando o erro for detectado foi a partir da programação em python desenvolvida que se encontra na seção A.1;
- Motion Sensor GPIO Pin: no campo disponível é necessário colocar o valor do pino que foi utilizado, se atentando ao Board Pin Mode escolhido;
- Enable Sensor: ativa o sensor para que a detecção seja feita, portanto esta opção deve ficar ativa;
- Detection time: campo que recebe o valor de tempo em segundos que foi utilizado para a detecção. Porém, não foi utilizado, pois esta opção é utilizada apenas quando o Detection Method escolhido for o Timeout detection.
- Detection distance: neste campo é adicionado o valor da distância em milímetros percorrida pela impressora pra realização da detecção de erros de extrusão. Quanto menor o valor, mais sensível será a detecção do sensor, portanto deve-se tomar cuidado com a ocorrência de ˝false flag" caso um valor muito baixo seja escolhido.

A conexão com o raspberry deve ser feita conforme a Figura 22, em que os pinos GND, VCC e SIGNAL do sensor são ligados nos pinos 6, 1 e 11, respectivamente, sendo os pinos mapeados no mapeamento Board Mode.

Desta forma o sensor de filamento inteligente está devidamente configurado para ser utilizado no desenvolvimento do sistema.

3.4.1.5 Câmera

A câmera utilizada é uma das que são recomendadas para o uso com o Raspberry Pi, possuindo 5MP de resolução de vídeo, entrada USB para realizar a conexão e dimensões pequenas de 40x30 mm. Seu ajuste de foco é feito de forma manual girando o mecanismo acoplado a lente.

Ao conectar a câmera em qualquer uma das portas USB do Raspberry, ela já é detectada pelo Octoprint, não necessitando realizar nenhuma configuração adicional.

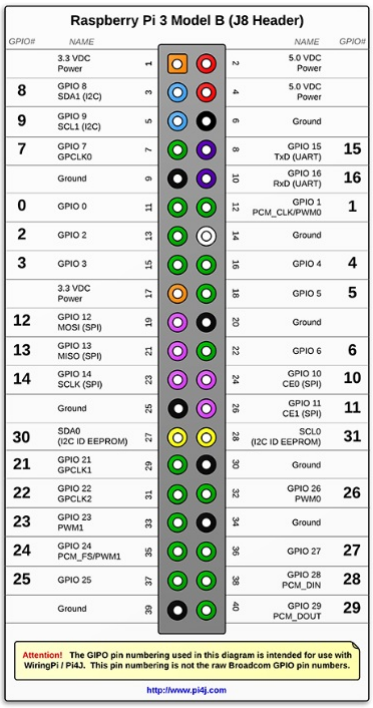


Figura 20 – Ilustração do mapeamento dos pinos do Raspberry Pi 3 Model B.

Fonte: (THE PI4J PROJECT, 2019)

3.4.1.6 BLTouch

A utilização de um sensor de nivelamento da mesa possibilita que uma das etapas de grande importância para que uma impressão de uma peça em 3D aconteça, essa etapa é a primeira camada de impressão. Quando ocorre algum problema na primeira camada as chances de a impressão apresentar algum problema é muito elevada. Sua precisão fica em torno de 0.005mm.

Portanto, possuir um sensor que possibilite acompanhar a distância da mesa com o bico de extrusão em vários pontos diferentes reduz a ocorrência de falhas, possibilita que uma manutenção seja feita quando a distância se torna maior do que deveria e a atuação no sistema.

O BLTouch é um sensor mecânico que faz o uso de um sensor de efeito Hall para detectar a movimentação do pino quando toca a mesa. A partir dessa detecção é marca a posição do eixo Y determinando assim o posicionamento do extrusor. Sua pinagem consiste em dois conectores responsáveis por detectar a mudança de corrente no sensor de efeito Hall, sendo um o GND e o outro o SIGNAL fechando assim o circuito quando há a detecção, e três conectores responsáveis pela movimentação do pino através de um servomotor, sendo o GND, VCC e SIGNAL.

A Figura 25 demonstra a conexão que foi feita, levando em consideração a placa de controle utilizada na impressora 3D na qual o sistema está sendo desenvolvido sendo a SKR

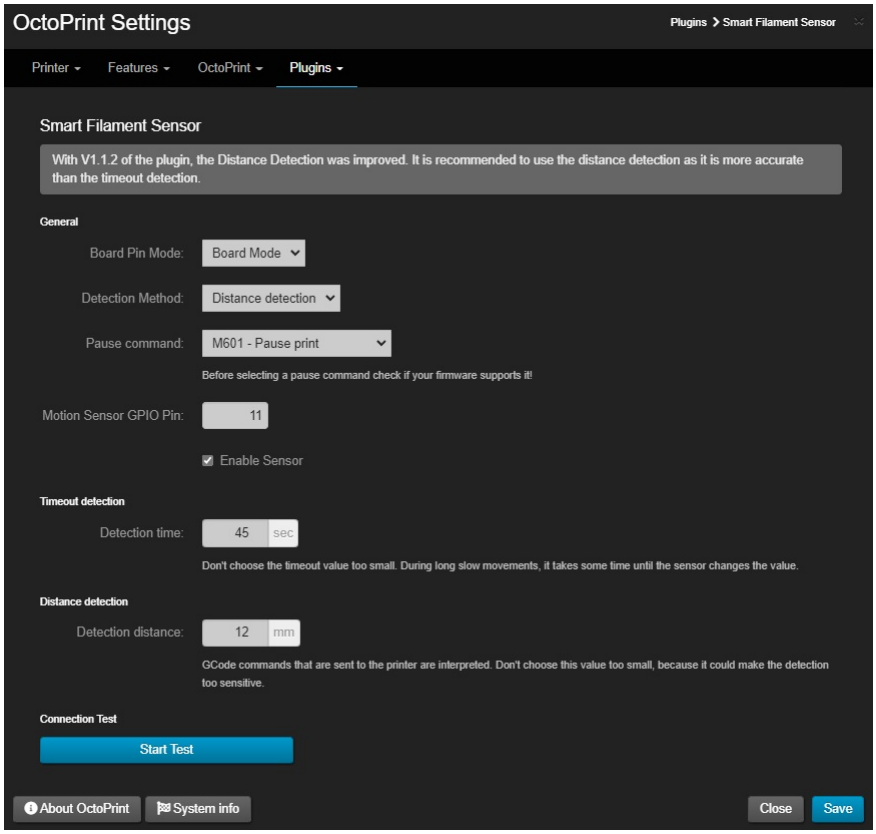


Figura 21 – Demonstração da configuração utilizada para o sensor de filamento inteligente no Octoprint.

Fonte: Produzido pelo autor

1.4.

Com o componente devidamente conectado, é necessário configurá-lo no Octoprint, essa configuração pode ser feita seguindo as instruções encontradas no site https://www.klipper3d.org/BLTouch.html.

Portanto, agora quando a impressão é iniciada o comando de nivelamento é executado e esses dados podem ser obtidos e utilizados. Porém, a aquisição desses dados é feita por uma API que conecta o programa em python, que foi implementado, com o Octoprint permitindo assim obter os valores obtidos pelo sensor e armazená-los, na qual foi escolhido utilizar o Google Sheet para realizar o armazenamento.

Desta forma, durante a inicialização de uma impressão é feita a comparação com os valores de nivelamento armazenados e os valores adquiridos do Octoprint a cada 10 segundos, até que ocorra uma mudança nos dados adquiridos ou então depois de 1 minuto, que é o tempo máximo para realização do nivelamento. Caso não ocorra mudanças significa que a mesa não sofreu mudanças no seu nivelamento. Os valores adquiridos são utilizados para atualizar o campo ″BED_MESH_OUTPUT” no Google Sheet, como será visto na 3.4.1.7.

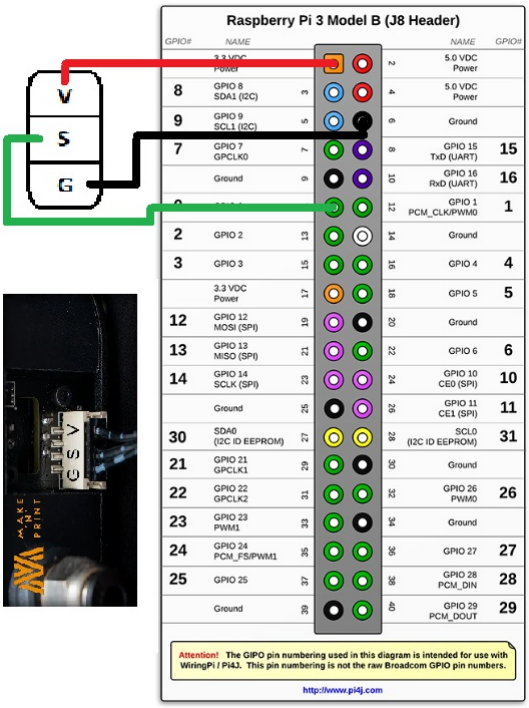


Figura 22 – Conexão do sensor de filamento inteligente com o Octoprint.

Fonte: adaptado de (THE PI4J PROJECT, 2019)

Figura 23 – Imagem representativa da câmera utilizada no projeto.

Fonte: (AUTOCOREROBOTICA, 2022)

3.4.1.7 Google Sheet

Para realizar o armazenamento de alguns dados da impressora foi utilizado o Google Sheet. Para realizar o acesso a planilha criada, alguns passos foram necessários de serem seguidos e estão disponíveis no site `https://developers.google.com/sheets/api/quickstart/python`. Seguindo as instruções do site, é possível obter acesso a planilha por meio de uma API, possibilitando que mudanças sejam feitas, como por exemplo armazenas os valores do nivelamento da mesa, ou então informações podem ser obtidas, como por exemplo o estado em que a impressora se encontra.

A Figura 26 demonstra como a planilha está organizada, na qual são armazenados os dados a seguir:

- Índice da impressora: utilizado para identificar a impressora com um número inteiro

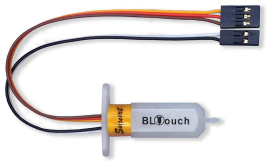


Figura 24 – Imagem representativa do BLTouch, sensor de nivelamento, utilizado no projeto.

Fonte: (3DWORK, 2020)

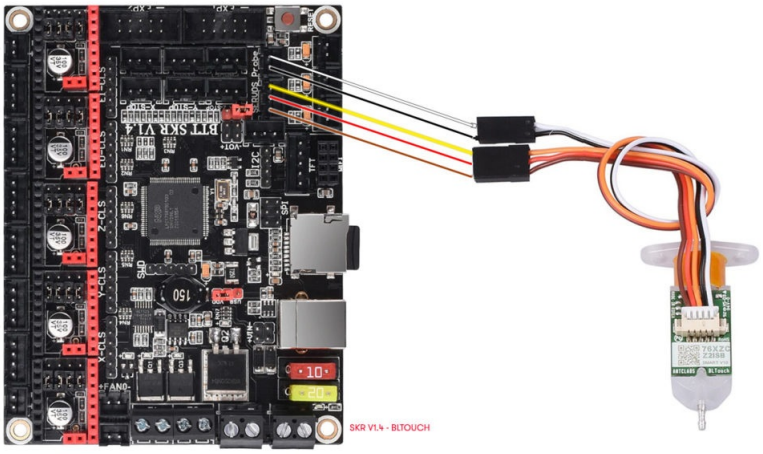


Figura 25 – Ilustração da conexão do BLTouch com a placa de controle SKR 1.4..

Fonte: (TEAMGLOOMY, 2022)

atribuído em ordem crescente e único, sendo que cada impressora terá seu próprio número.

- Marca: registra a marca da impressora utilizada.
- Modelo: registra o modelo da impressora utilizada.
- Status: define a impressora como
  - DESCONECTADA: estado em que a impressora não está conectada ao Octoprint, estando assim inoperante.
  - DISPONÍVEL: estado em que a impressora está disponível para uso.
  - IMPRIMINDO: estado em que a impressora está em operação.
  - EM MANUTENÇÃO: estado em que a impressora está passando por uma manutenção. Deve ser definido manualmente no estado atual do projeto.
  - AGUARDANDO MANUTENÇÃO: estado em que a impressora está aguardando que uma manutenção seja realizada nela.

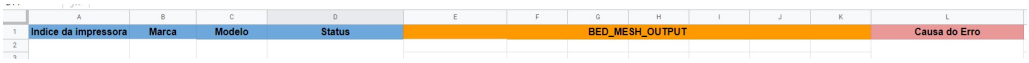


Figura 26 – Figura demonstrando a planilha do Google que foi utilizada no projeto.

Fonte: Produzido pelo autor

- BED_MESH_OUTPUT: registra a última aquisição de valores do nivelamento da mesa de impressão, além de indicar de verde os pontos da mesa em que o valor diminuiu em relação ao valor anterior e de vermelho os pontos em que os valores aumentaram;
- Causa do Erro: registra a causa de erro caso ocorra ou ajustes que devem ser feitos na impressora para evitar que venha ocorrer uma diminuição na qualidade das peças ou erros de impressão, podendo assim direcionar a manutenção que será feita.

3.4.1.8 Pushbullet

A utilização do Pushbullet se fez necessária, pois o Octoprint possui limitações, como por exemplo, não ser possível receber algumas informações úteis sobre o estado da impressora, como o início de uma impressão. Portanto, para contornar algumas das limitações existentes, foi necessário um plugin disponível no próprio Octoprint que permite que algumas das informações necessárias no desenvolvimento do sistema sejam enviadas como notificação para o Pushbullet.

O plugin do Octoprint que foi utilizado é o Octoslack, ele permite que seja feita a conexão do Octoprint com algumas plataformas, como o Discord, o Microsoft Teams, o Pushbullet, entre outras.

Para utilizar o Pushbullet é necessário configurá-lo, na qual primeiramente foi necessário escolher a plataforma utilizada para conexão com o Octoprint no espaço nomeado como ″Connection type”. Após a escolha da plataforma, é pedido, no espaço nomeado como ″Pushbullet configuration”, o token adquirido no site da plataforma, que no caso foi utilizado o Pushbullet, sendo o campo destinado ao token chamado de ″Access Token”. A forma como é possível adquirir o token do Pushbullet é descrito no site `https://docs.pushbullet.com/`. Além do token é necessário também informar o canal que irá receber as notificações no campo chamado de ″Channel(s)”, no qual deve ser preenchido com a palavra ″$myself$”.

Após realizar a configuração, no espaço ″Snapshots” deve ser marcada a opção ″None” e no espaço ″OctoPrint Events” deve ser marcada somente a opção ″Enable notification” para cada um dos seguintes eventos preenchendo também o campo ″Message”: (″OctoPrint Events”: ″Message”)

a) — Print started —: Started

b) — Print failed —: Failed

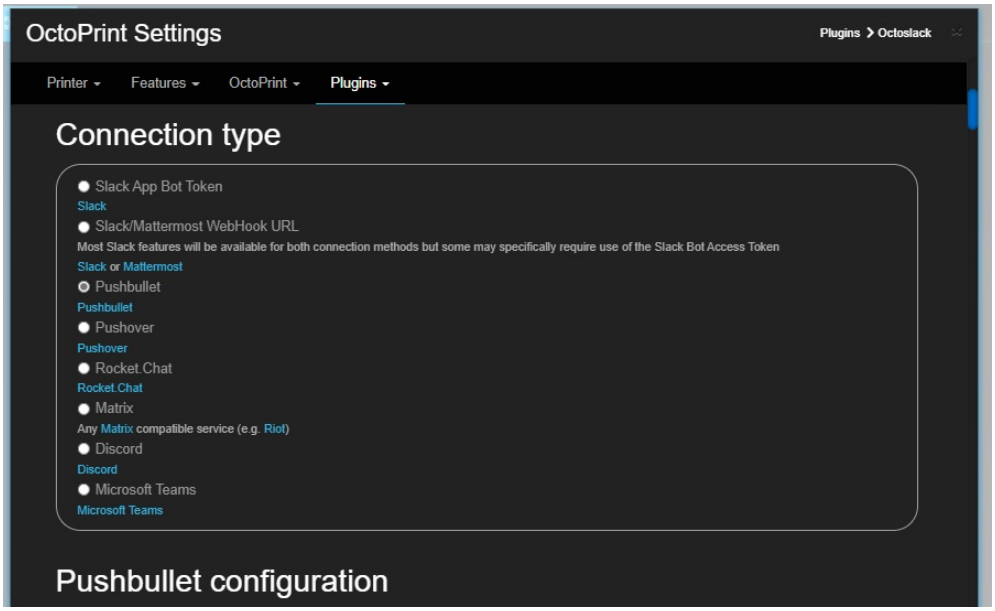


Figura 27 – Figura demonstrando a interface do Octoslack e suas formas de conexão.

Fonte: Produzido pelo autor

c) — Print cancelled —: Cancelled

d) — Print paused —: Paused

e) — Print resumed —: Resumed

f) — Print finished —: Finished

Portanto, o Pushbullet é a ferramenta que irá possibilita adquirir as notificações de início, término, falha, pausa, cancelamento ou retomada de uma impressão, pois receber algumas dessas notificações direto do Octoprint não é possível. Essa verificação é feita continuamente e, quando uma notificação é recebida, e para evitar que as notificações sejam acumuladas, é realizada a limpeza das notificações já recebidas.

3.4.1.9 The Spaghetti Detective

The Spaghetti Detective é uma empresa que fornece um serviço voltado para impressoras 3D que tem a finalidade de tornar mais fácil o gerenciamento das impressoras, realizando o monitoramento e controle remoto da impressora.]

Uma das funções que o software disponibiliza é a utilização de uma inteligência artificial que possibilita a detecção de problemas na peça enquanto está sendo impressa. Quando algum problema é detectado, uma notificação é enviada para o operador notificando o problema detectado, sendo que está notificação pode ser enviada através de algumas opções fornecidas.

Para utilizar o serviço fornecidos dois planos para serem contratados, o plano profissional e o plano grátis (a comparação entre os planos é feita na Tabela 3).

Tabela 3 – Comparação dos planos disponíveis pelo The Spaghetti Detective.

	Plano Profissional	**Plano Grátis**
Webcam streaming	25fps de streaming continuo	0.1fps de streaming apenas quando a impressora 3D está imprimindo
Horas de dectecção grátis	50 horas por mês	10 horas por mês
Tunelamento (acesso remoto)	Ilimitado	50MB por mês

Fonte: Produzido pelo autor

O Octoprint desenvolveu um plugin que possibilita utilizar o TSD chamado de ˝Access Anywhere - The Spaghetti Detective”. Após instalado, é necessário configurá-lo, na qual uma janela se abrirá. Nesta janela é necessário seguir as instruções para configurar corretamente o TSD para ser utilizado no Octoprint. Após a configuração no Octoprint, é necessário realizar a configuração adicional no modelo digital da impressora criada no site do TSD (`https://app.thespaghettidetective.com/printers/`), portanto, após acessar o site e entrar nas configurações da impressora, é necessário escolher a opção ˝Just notify me” em ˝When a potential failure is detected:", colocar a opção ˝How sensitive do you want the Detective to be on this printer?” no máximo, e o restante das opções devem ser desmarcada. Desta forma, quando um problema for detectado na impressão, apenas uma notificação será enviada.

Também é necessário, ainda no site que foi criada a impressora, acessar o campo ˝Preferences” e no espaço ˝Notificarion” escolher o Pushbullet para ser utilizado no envio da notificações de erro detectado pela inteligência artificial, selecionando apenas a notificação de falhas. Semelhante a configuração do Pushbullet no Octoprint, é necessário adicionar o token gerado para assim ser possível receber as notificações. A Figura 28 demonstra como foi configurada as notificações.

Para o projeto foi utilizado o plano grátis e o envio das notificações foram feitas a partir do Pushbullet, semelhante ao Octoprint, facilitando assim o desenvolvimento.

3.4.2 Framework

Para realização do projeto se faz necessário o uso de um framework para centralizar toda informação, gerenciar o sistema e conectar todo o sistema (WAEYENBERGH; PINTELON, 2002).

O funcionamento do sistema como um todo, em conjunto com a forma que eles se conectam e se organizam são essenciais para que um mecanismo de DT seja efetivo. Portanto, foi necessário para permitir que o sistema fosse desenvolvido utilizar um framework já existente, apesar das limitações impostas.

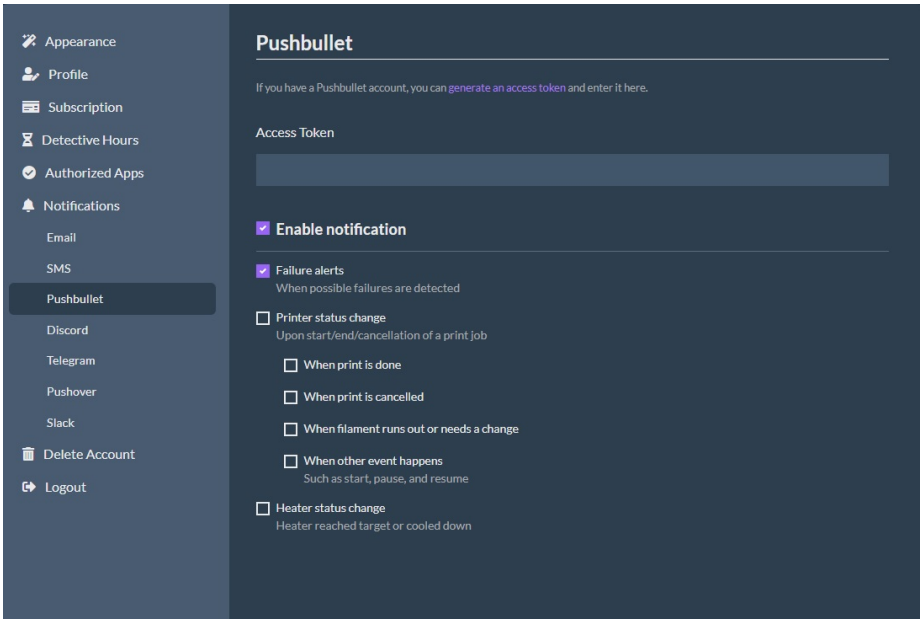


Figura 28 – Demonstração das configurções de notificações utilizadas no TSD.

Fonte: Produzido pelo autor

O framework escolhido para a realização foi o Octoprint, que possibilita realizar o controle da impressora 3D, enviando comandos e requisições de dados, possibilitando atuar no sistema, assim como disponibilizar plugins que auxiliam no gerenciamento das operações de impressão e na aplicação de novas funções, como controlar remotamente a impressora 3D.

A Figura 29 demonstra como o framework é utilizado no sistema de manutenção preditiva, na qual a impressora 3D recebe comandos do raspberry, como por exemplo a inicialização ou cancelamento de uma operação de impressão, movimentação dos motores, entre outros comandos disponíveis. O envio dos comandos é feito via porta serial a partir do Octoprint, na qual a impressora 3D envia de volta dados sobre a operação de impressão, como temperatura do bico e da mesa aquecida.

Durante seu funcionamento as notificações sobre o início, cancelamento, pausa, retomada da impressão ou falha de uma operação de impressão são enviadas pelo Octoprint para o Pushbullet. Então as notificações contidas no Pushbullet são adquiridas por meio de um API para que desta forma sejam repassado o estado da impressora para a planilha do Google Sheet, atualizando-a. Caso a notificação enviada pelo Octoprint seja a detecção de falha, o operador será notificado e a impressora pausada possibilitando analisar se a impressão pode ser retomada. Caso o problema detectado impossibilite a retomada da impressão, a mesma será cancelada e o estado da impressora será definida como "AGUARDANDO MANUTENÇÃO".

Um dos problemas que podem ser detectados pode ser originado do sensor de fila-

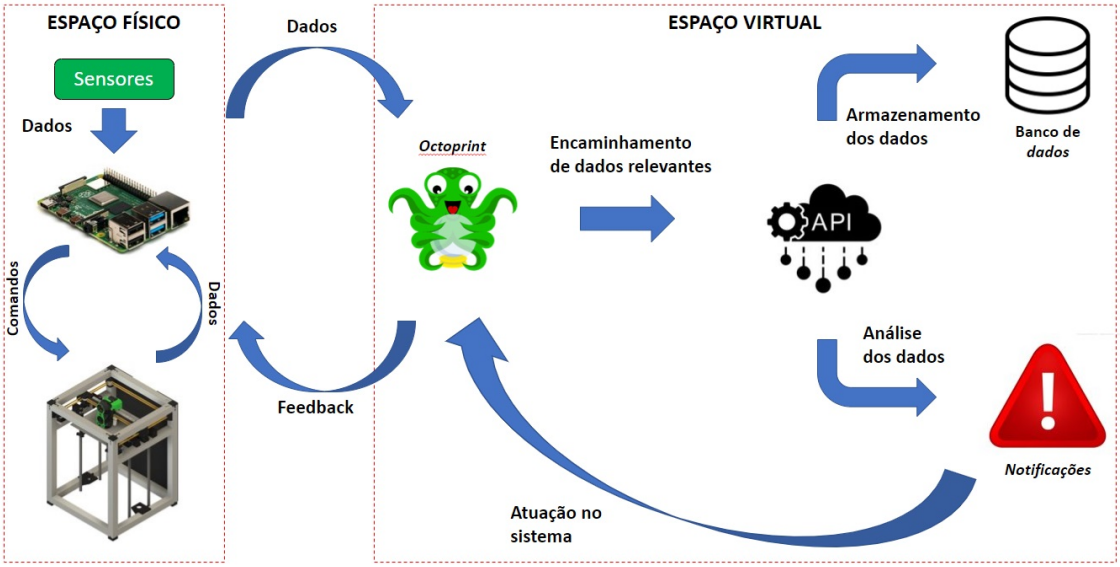


Figura 29 – Representação da utilização do Octoprint, framework escolhido para o projeto.

Fonte: Produzido pelo autor

mento inteligentes, que, durante a impressão de peças está detectando caso alguma falha de extrusão ocorra através da detecção de fluxo de filamento.

Já em relação a câmera, a conexão foi feita via porta USB e durante seu funcionamento os frames são repassados para o Octoprint que utiliza um plugin chamado ´´The Spaghetti Detective" para realiza o processamento em nuvem dos frames utilizando uma inteligência artificial que é capaz de determinar quando problemas de impressão estão ocorrendo, cobrindo uma das necessidades do cliente que é identificar quando ocorre a perda de passo. Quando um problema na impressão é detectado, um aviso é repassado para o Octoprint que, por meio da API do Pushbullet, esta notificação é lida pelo código em python que processa a informação e possibilita tomar a decisão de cancelar a impressão ou avisar o operador para que ele tome a decisão, na qual a ação tomada dependendo do erro detectado e sua proporção.

Enquanto que o sensor de filamento e a câmera são conectados diretamente no Raspberry, o BLTouch é conectado direto na placa de controle da impressora 3D, que pode possuir entrada específica para ser inserido ou então ser inserido no lugar do sensor de fim de curso do eixo Z.

Os dados obtidos pela placa de controle são repassados para o Raspberry via conexão serial. Os dados repassados para o Raspberry são enviados via API para o sistema que irá analisa-los e, caso o valor obtido em cada ponto da impressora for maior que um limiar definido, o sistema cancela a impressão e muda o estado da impressora para ´´AGUARDANDO MANUTENÇÃO" para que a ajustes no nivelamento da mesa sejam feitos.

Os valores do mapeamento da mesa são armazenados no Google Sheet para que

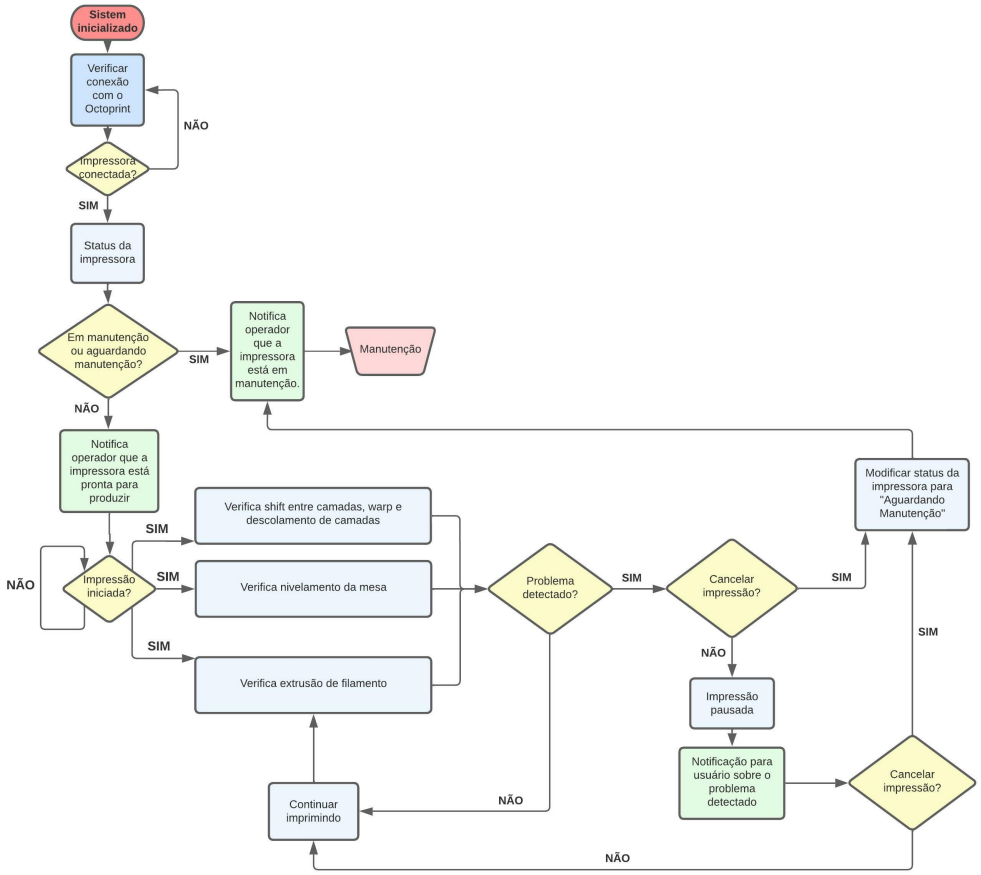


Figura 30 – Fluxograma do funcionamento do sistema.

Fonte: Produzido pelo autor

uma comparação seja feita entre os valores lidos e os valores anteriores, fornecendo uma análise sobre pontos em que a leitura aumentaram ou diminuíram, permitindo visualizar a propagação do erro da mesa da impressora.

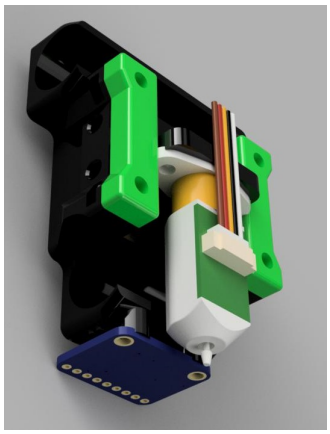
3.4.3 Acoplamento dos componentes

Para acoplar alguns dos componentes foi necessário modelar peças e fabricá-las utilizando impressão 3D. Para a modelagem dessas peças foi utilizado o software Fusion360 e para fabricação foi utilizada a própria impressora que está sendo utilizada neste projeto.

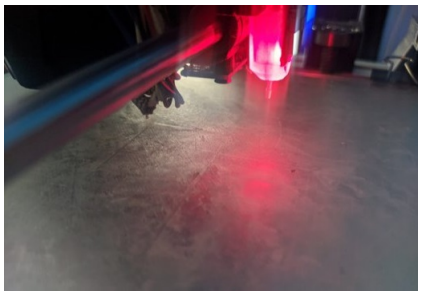
3.4.4 ADXL345 e BLTouch

Ambos, ADXL345 e BLTouch, necessitam ser acoplados ao extrusor da impressora. Em relação ao ADXL345 é necessário colocar o mais perto do extrusor, pois o extrusor é a parte responsável pela deposição do material, onde a vibração tem maior impacto sobre a qualidade da peça impressa. Já em relação ao BLTouch o motivo é que o ponto que se deseja medir a distância é a ponta do bico de extrusão que se encontra no extrusor. A Figura 31

(a) podemos visualizar a modelagem feita para realizar o acoplamento do ADXL345 e do BLTouch, enquanto que na Figura 31 (b) podemos visualizar a peça fabricada e já em uso.



(a) Peça modelada para acoplar o ADXL345 e o BLTouch ao sistema.



(b) Peça fabricada com impressão 3D para acoplar o ADXL345 e o BLTouch ao sistema.

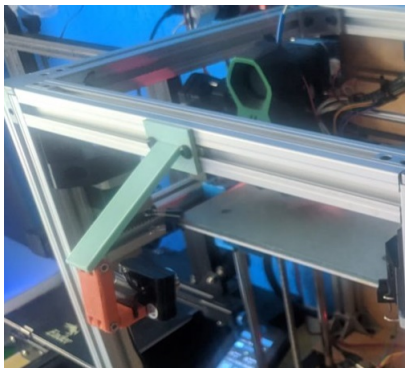
Figura 31 – Acoplamento do ADXL345 e do BLTouch

Fonte: Produzido pelo autor

3.4.5 Câmera

Para realizar o acoplamento da câmera foi modelada uma peça articulável composta por três partes, que se movem permitindo ajustar a posição da câmera facilmente. Como na estrutura da impressora utilizada para o projeto o extrusor fica em uma posição Z fixa, a câmera foi acoplada de modo que o bico do extrusor fosse focado. A Figura 32 (a) podemos visualizar a modelagem feita para realizar o acoplamento da câmera, enquanto que na Figura 32 (b) podemos visualizar a peça fabricada e já em uso.

(a) Peça modelada para acoplar a câmera ao sistema.



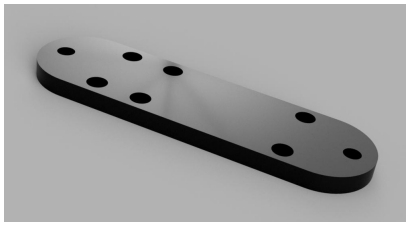
(b) Peça fabricada com impressão 3D para acoplar a câmera ao sistema.

Figura 32 – Acoplamento da câmera

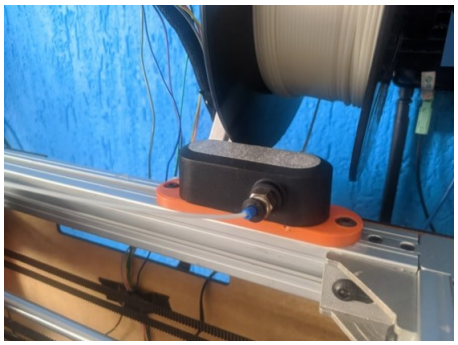
Fonte: Produzido pelo autor

3.4.6 Sensor de filamento inteligente

Temos também que realizar o acoplamento do sensor de filamento inteligente. Para realizar esse acoplamento foi necessário retirar a tampa traseira que vem de fábrica e substituir por uma tampa que permitisse o seu acoplamento, porém não alterando o funcionamento do sensor. A Figura 33 (a) podemos visualizar a modelagem feita para realizar o acoplamento do sensor de filamento, enquanto que na Figura 33 (b) podemos visualizar a peça fabricada e já em uso.



(a) Peça modelada para acoplar do sensor de filamento inteligente ao sistema.



(b) Peça fabricada com impressão 3D para acoplar o sensor de filamento inteligente ao sistema.

Figura 33 – Acoplamento do sensor de filamento inteligente

Fonte: Produzido pelo autor

4 Resultados

Em concordância com o Diagrama em V, antes de serem acopladas as partes do projeto, é necessário realizar testes para verificar o funcionamento de cada componente separadamente e retomar às necessidades dos stakeholder verificando se o que está sendo implementado está de acordo com o desejado. Portanto, será tratado os resultados de cada componente individualmente e, por fim, tratar o sistema de manutenção preditivo implementado.

4.1 Testes e verificação

4.1.1 Sensor de filamento

Iniciando pelo sensor de filamento que foi devidamente configurado na interface do plugin no Octoprint da seguinte forma:

- Foi escolhida a opção Boad Mode no campo Board Pin Mode, porém é de livre escolha, lembrando que o valor pino de sinal que será utilizado deve ser modificado dependendo da escolha feita nesta opção.
- Foi utilizado a distância percorrida pela impressora para realizar a detecção do fluxo de filamento, como recomendado pelos desenvolvedores.
- No campo "Pause command"foi escolhida a opção M601 que apenas pausa a impressora, pois a decisão do que vai ser feito quando o erro for detectado será feito da programação em python, como será visto posteriormente.
- No campo "Motion Sensor GPIO Pin"foi escolhido o pino 11 do raspberry, pois foi escolhida a opção Board Pin Mode anteriormente. Caso fosse escolhido a opção BCM Mode, o valor do pino seria o 0.
- Foi marcado o campo "Enable Sensor"para que o sensor fosse ativado.

O espaço "Timeout detection"não será modificado, pois não será utilizado o tempo para realizar a detecção de erro.

Alguns testes foram realizados para verificar o funcionamento correto do sensor. O primeiro teste teve o objetivo de verificar se o sensor estava detectando a movimentação do filamento. Para realização do teste foi utilizado o próprio plugin que configura o sensor, pois possui um espaço dedicado para verificar seu funcionamento. Podemos visualizar na Figura 34 a interface utilizada.

Figura 34 – Demonstração do teste do funcionamento do sensor de filamento inteligente no plugin do Octoprint.

Fonte: Produzido pelo autor

Como resultado do teste, foi possível concluir que o sensor detecta corretamente a existência de fluxo de filamento, porém é possível notar que não é feito o cálculo de quanto filamento foi extrudado. Isso ocorre, pois, para realizar esse cálculo e comparar com o comando da quantidade de filamento que a impressora irá extrudar, enquanto acontece a impressão não é possível, isso acontece por causa dos limites relacionados ao firmware da impressora e a forma com que o arquivo é lido que impossibilita realizar alguma sincronização entre a placa de controle da impressora e um sistema que simule a extrusão que está ocorrendo.

O próximo teste foi feito com a impressora imprimindo uma peça de teste para verificar se o sensor está funcionando normalmente durante a operação. Foi simulado o entupimento do bico da impressora e a ocorrência de nó no filamento, ambos ocasionam a interrupção da passagem de filamento pelo sensor, desta forma não sendo detectado movimento e acusando problemas durante a impressão. Foram utilizados os valores de 5mm, 10mm e 15mm de distância no campo "Detection distance"no espaço "Distance detection".

Os testes foram bem sucedidos para cada um dos valores de distância utilizados, na qual a impressora pausa a operação e avisa ao operador que ocorreram problemas durante a operação, para que desta forma seja tomada uma decisão do que será feito. Quando ocorre essa detecção, é possível retomar a impressão após o problema ser solucionado, desde que não mova o extrusor acidentalmente, perdendo assim a referência de onde a impressora estava, ou que exista sub extrusão na peça comprometendo-a, impossibilitando assim a retomada da impressão.

Vale ressaltar que a diferença dos resultados apresentados para cada um dos valores de distância utilizados foi a sensibilidade para realizar a detecção, quanto menor o valor, mais rápido o sensor detecta que está ocorrendo problemas de extrusão. Porém valores muitos baixos podem ocasionar "false flag", ou seja, detecta que o filamento não está se movendo mesmo que seja durante uma parte da impressão que não deve ocorrer extrusão de filamento, como por exemplo movimentação do extrusor de um ponto final de extrusão de uma camada da peça que está sendo fabricada até o ponto inicial da extrusão da sua próxima camada. Logo, para evitar que ocorra "false flag", será utilizado como padrão o valor de 10mm que

apresentou bons resultados.

Retornando as necessidades dos Stakeholders, foi verificada que a questão de realizar o sensoriamento da extrusão possibilitando reconhecer o erro e as suas possíveis causas são requisitos alcançados. Realizar este sensoriamento é algo importante, pois geralmente quando ocorre problemas de extrusão, a peça que estava sendo produzida geralmente é descartada, ocasionando um desperdício de tempo e de material.

4.1.2 BLTouch

O BLTouch foi utilizado para acompanhar a degradação do nivelamento da mesa e possibilitar identificar a necessidade de manutenção preditiva, na qual será feito o ajuste do nivelamento antes que erros de impressão possam vir a ocorrer causados pela falta de nivelamento da mesa aquecida.

Ao iniciar uma impressão é feito o nivelamento da mesa aquecida. Durante a realização do nivelamento da mesa é requisitado diretamente ao Octoprint os valores dos pontos obtidos em uma matriz de 7x7, que correspondem a distância em milímetros da ponta do pino do BLTouch à mesa, esses valores são comparados com os valores que foram adquiridos anteriormente, e foram salvos na planilha do Google Sheet. Logo após a comparação, os novos valores são salvos na planilha substituindo os anteriores, sendo que os pontos que tiveram seu valor aumentado são armazenados e a sua célula é marcada de vermelho, já no caso de o valor ter diminuído a célula será marcada de verde.

A identificação da necessidade de manutenção é feita a partir da comparação dos valores obtidos com um limiar de 0.35, que foi obtido a partir de tentativa e erro de forma que a primeira camada de impressão ficasse distante o suficiente da mesa para ocorrer problemas na primeira camada. Portanto, caso o módulo de algum dos valores lidos pelo BLTouch ultrapasse esse limiar, a impressão é cancelada e a notificação sobre a necessidade de manutenção na impressora é mostrada para o operador. Também é feita a modificação no estado da impressora na planilha do Google Sheet, que será alterada para "AGUARDANDO MANUTENÇÃO".

Foram feitos os testes para obter os valores dos pontos da mesa, para isso o BLTouch foi acoplado a impressora e configurado devidamente. Após a configuração, a impressora foi colocada em funcionamento para assim obter os valores dos pontos da mesa. Os valores do nivelamento podem ser obtidos do Octoprint por meio da API, porém não é possível obter apenas os valores do nivelamento, recebendo várias informações de configuração da impressora, de plugins, de conexão, entre outras informações. Logo é necessário filtrar as informações obtidas para obter apenas a informação que será utilizada, que no caso são os pontos lidos pelo BLTouch.

A obtenção dos valores foi bem sucedido, e foi possível comparar cada valor com o

BED_MESH_OUTPUT						
0.01	0.00	0.00	0.00	0.02	0.04	0.07
0.04	0.03	0.02	0.03	0.04	0.05	0.07
0.05	0.04	0.04	0.04	0.04	0.05	0.07
0.05	0.04	0.04	0.03	0.04	0.04	0.05
0.03	0.02	0.02	0.02	0.02	0.02	0.03
0.00	-0.01	-0.01	-0.02	-0.01	-0.01	0.00
-0.05	-0.06	-0.06	-0.06	-0.06	-0.05	-0.04

BED_MESH_OUTPUT						
0.02	-0.01	0.00	0.00	0.02	0.04	0.07
0.04	0.03	0.02	0.03	0.04	0.05	0.07
0.05	0.05	0.03	0.04	0.04	0.05	0.07
0.05	0.04	0.04	0.03	0.04	0.05	0.05
0.03	0.02	0.02	0.02	0.02	0.02	0.03
-0.01	-0.01	-0.01	-0.02	0.00	-0.01	0.00
-0.05	-0.05	-0.06	-0.06	-0.06	-0.05	-0.04

Figura 35 – Comparação entre dois nivelamentos realizados em impressões executadas sequência.

Fonte: Produzido pelo autor

valor do limiar de 0.35 milímetros.

Foi realizado o teste no qual a impressora iniciou um impressão e os pontos foram obtidos e armazenados. Logo após o término da impressão, ou impressão foi iniciada para ver a mudança que ocorreu nos valores. Podemos visualizar na Figura 35 que muitos pontos continuaram com os mesmo valores, porém em alguns dos pontos houveram mudanças pequenas em seus valores, principalmente os situados no canto esquerdo da mesa. Isso pode ter acontecido no momento que a impressão foi retirada da mesa.

Então um novo teste foi realizado, no qual a mesa foi desnivelada manualmente para ser possível visualizar o comportamento quando o valor do limiar é ultrapassado. Neste caso, a impressão foi iniciada e, logo após realizar o nivelamento, foi cancelada, sendo enviado ao operador um aviso sobre a ocorrência do erro, assim como esperado.

Voltando as necessidades dos Stakeholders, foi verificada que a questão de realizar o sensoriamento do nivelamento da mesa, assim como reportar o erro ocorrido e sua causa. Não é possível dizer qual é a causa do erro sem um estudo prévio, realização de sensoriamento capaz de gerar dados que possibilite isso, pois as causas podem várias origens diferentes.

4.1.3 Pushbullet

A utilização do Pushbullet foi feita a partir de uma API que permite receber notificações diretas do Octoprint, para que desta seja possível realizar atuações no sistema.

Foram feitos testes como iniciar uma impressão, pausar, retomar, cancelar, além de simular falhas de impressão. Todos os testes foram bem sucedidos, na qual o Octoprint enviou para o Pushbullet a ocorrência de cada uma das ações realizadas no teste, permitindo assim que decisões sobre a operação de impressão sejam feitas. Apesar dos testes tiverem sido bem sucedidos, é necessário se atentar a frequência de aquisição que poderá ocasionar um timeout de alguns minutos, portanto a utilização de um delay de menos de 1 segundo é necessário para impedir o timeout.

A Figura 36 mostra o teste no qual é recebido a notificação do início de uma impressão e do cancelamento do mesmo, para verificar que as notificações estão sendo enviadas de forma correta.

Apesar de não ser uma necessidade do stakeholder, foi necessário utilizar o Pushbullet

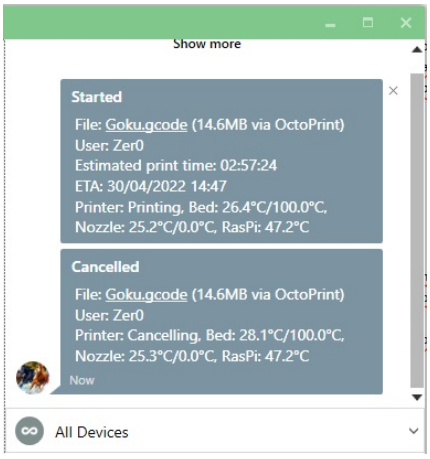


Figura 36 – Demonstração da notificação de início da impressão e do seu cancelamento.

Fonte: Produzido pelo autor

por causa das limitações apresentadas pelo Octoprint, sendo uma das forma possíveis de amenizar essas limitações.

4.1.4 The Spaghetti Detective

Passando para o plugin The Spaghetti Detective que possibilita o uso de uma inteligência artificial para detectar falhas na peça que está sendo impressa.

Para fazer o seu uso foi primeiro feitas as configurações necessárias no site que hospeda o serviço fornecido. Com a impressora configurada devidamente, foi possível continuas com os testes.

Para realizar o teste verificando o funcionamento do correto do plugin foi iniciada uma impressão de uma peça de calibração. Durante a impressão o extrusor foi movido manualmente alguma vezes a fim de que o plugin detectasse o erro ocasionado na impressão. Para fins de comparação, a sensibilidade foi testada na metade do valor máximo e no valor máximo.

No teste realizado a inteligência artificial conseguiu detectar que algo estava estranho na impressão, porém não conseguiu identificar qual o problema que estava ocorrendo, notificando o operador que um erro foi detectado para que, a partir da informação recebida, uma decisão seja tomada. A Figura 37 mostra o plugin detectando a ocorrência da falha, porém, mesmo sendo uma falha evidente, ele não consegue determinar se é uma falha que prejudicará a impressão.

Observando a Figura 38, podemos verificar que o plugin não possui um funcionamento esperado. A causa disso pode ser o plano utilizado no projeto, que utiliza um fps de 0.1s e a inteligência artificial disponibilizada não é a otimizada, afetando a detecção do erro.

Em concordância com as necessidades obtidas a partir dos Stakeholders, o plugin

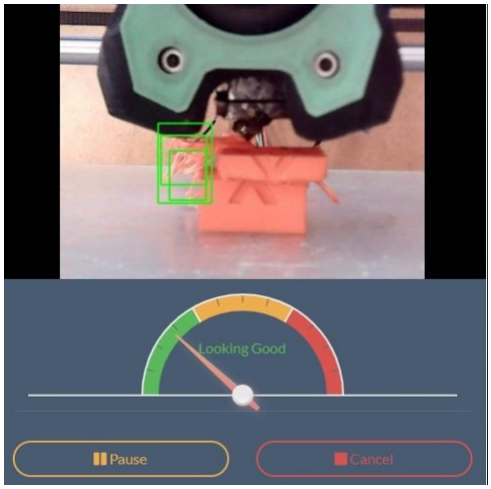


Figura 37 – Monitoramento do detector de falhas do The Spaghetti Detective analisando a impressão da peça teste sendo feita.

Fonte: Produzido pelo autor

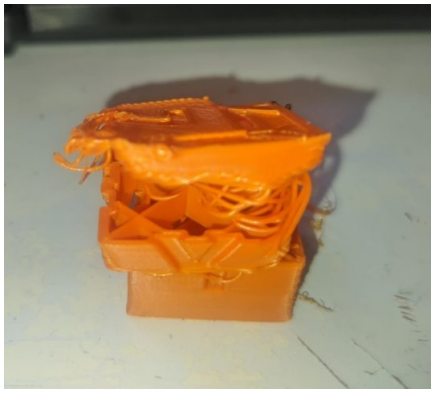


Figura 38 – Peça impressa para o teste do The Spaghetti Detective.

Fonte: Produzido pelo autor

pode ser utilizado para detectar perda de passo, porém não é possível dizer qual a natureza do erro que pode ser causado por dois problemas distintos: correia com baixa tensão ou travamento do motor.

4.1.5 Google Sheet

Utilizando uma API foi implementado um código em python que atualiza as células da planilha do Google e obtém as informações que foram armazenadas nela, como por exemplo os valores do nivelamento e o estado das impressoras. Os pontos obtidos que possuem um valor maior que os pontos anteriormente obtidos são marcados de vermelho na planilha, enquanto aqueles que possuem um valor menor são marcados de verde, possibilitando assim ter um acompanhamento de como o nivelamento está se comportando.

Sua utilização foi necessária para realizar os armazenamentos das informações que

Índice da impressora	Marca	Modelo	Status	BED_MESH_OUTPUT							Causa do Erro
1.00	DIY	Zer0	DISPONÍVEL	0.02	-0.01	0.00	0.00	0.02	0.04	0.07	
				0.04	0.03	0.02	0.03	0.04	0.05	0.07	
				0.05	0.05	0.03	0.04	0.04	0.05	0.07	
				0.05	0.04	0.04	0.03	0.04	0.05	0.05	
				0.03	0.02	0.02	0.02	0.02	0.02	0.03	
				-0.01	-0.01	-0.01	-0.02	0.00	-0.01	0.00	
				-0.05	-0.05	-0.06	-0.06	-0.06	-0.05	-0.04	

Figura 39 – Informações referentes a impressora armazenados no Google Sheet.

Fonte: Produzido pelo autor

são importantes possibilitando o desenvolvimento do projeto.

A Figura 39 mostra a planilha do Google Sheet atualizada com as informações da impressora, contendo seu estado e também os pontos do nivelamento armazenados.

4.1.6 Acelerômetro

Foi optado a utilização da conexão SPI do Raspberry Pi para obtenção dos dados. A escolha da interface se deve a impossibilidade de se utilizar a conexão I2C, pois ocorrem conflitos quando a impressora 3D está imprimindo e o acelerômetro está coletando dados, na qual a aquisição de dados é interrompida independente da taxa dessa aquisição.

A utilização do acelerômetro tem como objetivo identificar variações na leitura quando a tensão da correia utilizada para movimentar o eixo X e o eixo Y está ajustada da forma correta e quando está necessitando de um ajuste.

Para realizar a leitura foi implementado um código em C para obtenção dos dados do acelerômetro para que a leitura feita não tenha variações de tempo muito altas nas leituras, que invalidariam as leituras coletadas.

Os testes iniciais eram apenas a execução do código em python que faz uma chamada de sistema para executar o código me C e logo após gera os gráficos dos dados obtidos com a aplicação de uma FFT. Como não era movido o extrusor, logo não ouve valor de leitura captado a não ser a componente DC.

Após verificar que o acelerômetro estava funcionando corretamente, assim como o código utilizado, foram feitos outros testes, porém agora com a impressora imprimindo uma peça de calibração qualquer. Este teste tinha a finalidade de gerar o gráfico para ser possível verificar a variação do tempo dos dados coletados, para que eles não fossem invalidados.

No início estava sendo utilizado a conexão I2C do raspberry para utilizar o acelerômetro, porém sempre que o código era executado, ocorria um erro que impossibilitava rodar o código por mais de 1 segundo. Após serem feitos testes para solucionar o problema, foi descoberto que quando este problema ocorria a impressora estava funcionando. Então, o problema era gerado porque quando a impressora está em operação, a conexão I2C é utilizada, portanto qualquer outra aplicação utilizando esta conexão é encerrada. Portanto foi feita a mudança da conexão para utilizar a SPI.

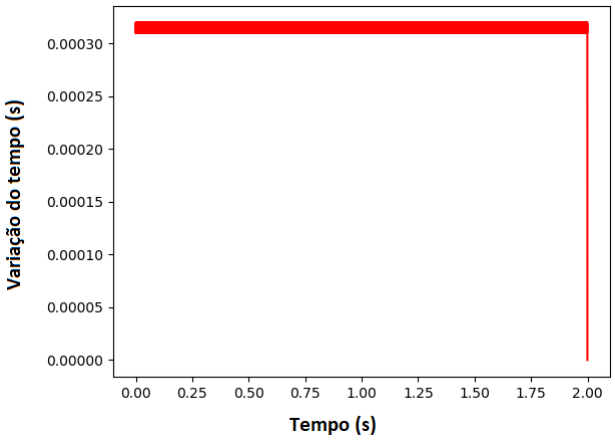


Figura 40 – Gráfico da variação do tempo entre as amostras coletadas.

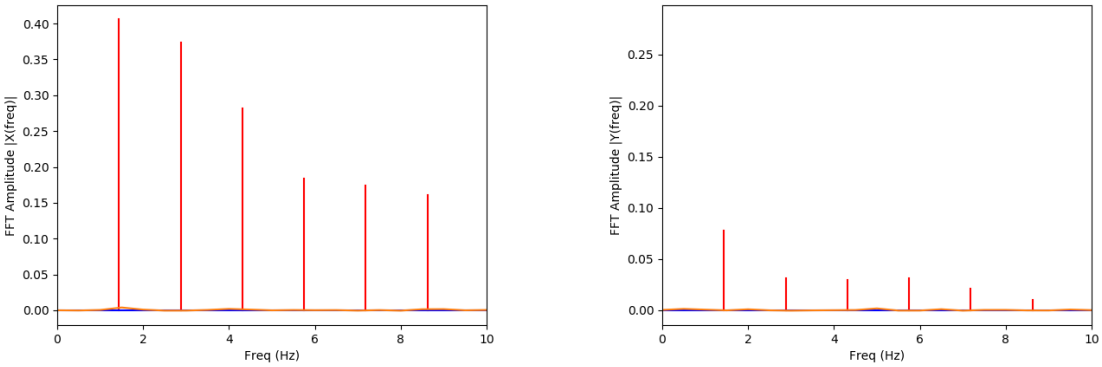
Fonte: Produzido pelo autor

Os testes foram reiniciados utilizando agora a nova conexão e os dados foram obtidos com êxito.

Em posse dos dados foi realizado o cálculo da diferença entre o tempo de aquisição entre os dados. Desta forma foi gerado o gráfico da variação do tempo que pode ser visto na Figura 40. Podemos verificar que a variação do erro foi menor que 0.35 milissegundos, valor este que valida os dados coletados.

Após verificar a variação do tempo, foram feitos dois testes para coletar os dados durante 2 segundos com uma frequência de 3200 Hz, correspondente a 6400 amostras, sendo que o valor da componente DC foi calculada a partir da média aritmética e subtraída dos valores das amostras coletadas. O primeiro teste foi feito com a correia com baixa tensão, enquanto que o segundo foi feito com a correia devidamente tensionada. Em ambos foi gerado o gráfico a partir da aplicação da FFT nas amostras obtidas.

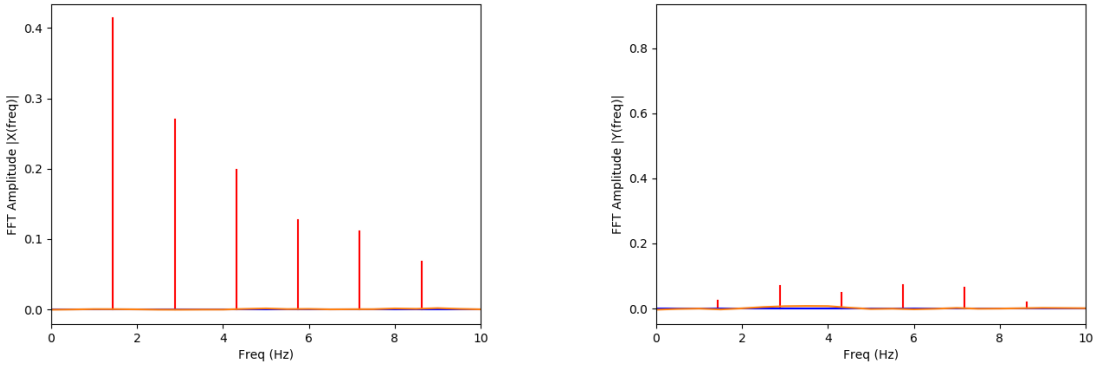
Na Figura Figura 41 estão os gráficos gerados pelos dados obtidos do eixo X e Y com a correia da impressora devidamente tensionada, enquanto que na Figura Figura 42 estão os gráficos gerados pelos dados obtidos do eixo X e Y com a correia da impressora com baixa tensão. Podemos então verificar a amplitude e a frequência dos espectros gerados.



(a) Gráfico do eixo X. (b) Gráfico do eixo Y.

Figura 41 – Gráficos gerados com o tensionamento ideal das correias.

Fonte: Produzido pelo autor



(a) Gráfico do eixo X. (b) Gráfico do eixo Y.

Figura 42 – Gráficos gerados com o tensionamento das correias abaixo do ideal.

Fonte: Produzido pelo autor

Visualizando os gráficos gerados podemos facilmente verificar que quando a correia está devidamente ajustada, as amplitudes são maiores do que quando a correia está folgada. Isso ocorre, pois, com a correia ajustada, a vibração da estrutura e dos motores influenciam mais na vibração do extrusor, enquanto que com a correia folgada essas vibrações são mais absorvidas.

Portanto, criar um sistema em tempo real para obter os dados do acelerômetro e utilizar algum tipo de ferramenta para processá-los para saber quando que a correia está devidamente ajustada ou não é viável utilizando um acelerômetro. A análise dos dados poderia ser feita a partir do valor de densidade espectral de potência.

4.2 Integração dos códigos

A integração do código foi feita em python. Ao iniciar o sistema é feita a verificação se a impressora está conectada a partir de uma requisição enviada para o Octoprint que retorna o estado da impressora. Dependendo do retorno obtido, as seguintes situações podem ocorrer:

a) situação - Closed: O Octoprint retorna a mensagem "Closed"que significa que a impressora não está conectada. Desta forma alguns casos podem ocorrer:

   1°. caso: Se na planilha do Google o estado da impressora está como "DISPONÍVEL"ou "IMPRIMINDO", então será atualizada para "DESCONECTADA".

   2°. caso: Se na planilha do Google o estado da impressora está como EM MANUTENÇÃO ou "ESPERANDO MANUTENÇÃO", então será mantido o estado e será enviado para o usuário uma notificação avisando "Aguardando a realização da manutenção!".

   3°. caso : para qualquer outro estado o operador receberá como retorno a mensagem ´´Impressora está desconectada! Verificação será feita novamente dentro de 1 minuto", então será atualizada para "DESCONECTADA". O valor de 1 minuto para não ocorrer timeout no acesso a planilha.

b) situação - Operational: O Octoprint retorna a mensagem "Operational"que significa que a impressora está conectada. Desta forma alguns casos podem ocorrer:

   1°. caso: Se na planilha do Google o estado da impressora está como "DESCONECTADA", então será atualizada para "DISPONÍVEL".

   2°. caso: Se na planilha do Google o estado da impressora está como EM MANUTENÇÃO ou "ESPERANDO MANUTENÇÃO", então será mantido o estado e será enviado para o usuário uma notificação avisando que "Aguardando a realização da manutenção!".

   3°. caso : Se na planilha do Google o estado da impressora está como "IMPRIMINDO"a flag booleana que informa se a impressora está imprimindo ou não, então será atualizada para "DISPONÍVEL".

   4°. caso : Para qualquer outro estado, será atualizado para "DISPONÍVEL".

Caso o estado seja "Disponível"será verificado a cada 1 segundo se alguma impressão foi inicializada. Essa verificação é feita verificando se existe a notificação no Pushbullet, na qual, caso não possua a notificação de início de impressão o programa roda em loop verificando se a impressora continua conectada e se foi recebida a notificação de início da impressão.

Quando a notificação da impressão é recebida, são iniciados duas threads para realizar o sensoriamento em paralelo, logo é iniciada a thread responsável pela tarefa de realizar o nivelamento da mesa como visto anteriormente, e em paralelo é iniciada a tarefa que recebe as notificações do Octoprint e do The Spaghetti Detective. A thread responsável pelo nivelamento é finaliza após a impressora realizar o comando responsável por pegar os pontos da mesa, enquanto que a thread que recebe as notificações continua sendo executada até o término da impressão, ou caso a impressão seja cancelada ou apareça algum erro que necessite cancelar a impressão.

Enquanto a impressão está acontecendo o sensor de filamento também está funcionando paralelamente através do Octoprint que envia a notificação de erro caso o sensor venha a detectar. Semelhantemente o The Spaghetti Detective, que utilizando a câmera devidamente posicionalmente para a impressão, avalia os frames e caso detecte um erro na peça sendo impressa, envia uma notificação. Desta forma, caso receba a notificação do erro, em ambos os casos o operador deve checar a impressão para definir se pode retomá-la ou deve cancelá-la.

Se for possível retomar uma impressão, no caso de problemas de extrusão, o operador deve antes realizar a manutenção tomando os devidos cuidados para que seja possível retomar a impressão.

Desta forma, os componentes irão funcionar em conjunto, menos o acelerômetro que necessita de um estudo mais aprofundado para poder ser utilizado. Portanto, o sensoriamento da extrusão de filamento, do nivelamento da mesa e da perda de passo foram possíveis de serem implementadas possibilitando realizar o diagnóstico da impressora e realizar a manutenção preditiva, além de possibilitar atuar no sistema de forma a cancelar a operação de impressão e realizar o envio de notificações para o usuário.

Na seção A.1 podemos verificar a programação que foi utilizada.

5 Conclusão

Neste trabalho foram abordados todos os conceitos necessário para a implementação do sistema de manutenção preditiva voltada a impressão 3D utilizando o conceito de DT.

O foco principal do trabalho foi apresentar o mecanismo de DT, que possibilita a mudança para a quarta revolução industrial, chamada de Industria 4.0, que possui o objetivo de convergir o modelo físico com o modelo virtual possibilitando a integração e a extensão dos processos de fabricação em todo o ciclo de vida do produto.

Existem outros mecnismos além do DT, porém seu diferencial em relação a outros mecanismos com o mesmo objetivo como sendo a possibilidade de realizar simulações no modelo virtual para entender o comportamento do modelo fisíco com as mudanças simuladas de forma a otimizar o sistema de maneira eficiente, além de permitir a atuação no sistema com base em dados coletados dos sensores para que ele performe melhor, tornam ele um mecanismo com egrande potencial para a industria.

Para realizar a implementação do DT foi escolhido desenvolver um sistema de manutenção preditiva em impressoras 3D. Essa escolha foi feita com base na necessidade do Laboratório Aberto de Brasília de otimizar o gerenciamento das impressoras, otimizar o processo de produção, aumentar a confiabilidade do serviço de impressão prestado, aumentar a repitibilidade das impressoras, aumentar a qualidade de impressão, etc. Desta forma, o DT foi a forma escolhida para realizar essas mudanças, pois, além de permitar que essas mudanças sejam implementadas, também é uma tecnológia que tem ganhado cada vez mais evidência.

O desenvolvimento do sistema de manutenção preditiva foi feito com base no Diagrama em V que é uma ferramenta que auxilia no desenvolvimento de sistemas com alta complexidade, no qual direciona o desenvolvimento para as necessidades dos stakeholders de tal forma que cada etapa desenvolvida deve ser confrontada com as necessidades obtidas para verificar se está de acordo com o desejado. Portanto, foi feito todo o mapeamento dos stakeholders, o levantamento de suas necessidades, a definição dos requisitos do sistema, a escolha dos componentes, o desenvolvimento de cada componente e por fim o acoplamneto dos componentes obtendo assim o sistema de manutenção preditiva. Cada etapa de desenvolvimento realizado era confrontado com as necessidades obtidas dos stakeholders.

Durante o desenvolvimento foi possível verificar algumas propostas de trabalhos futuros que podem ser desenvolvidos para que seja possível desenvolver o mecanismo de DT, não apenas para manutenção preditiva, mas também para simulação, gerenciamento, entre outras aplicações.

As propostas de trabalhos futuros são:

1. Desenvolvimento de um framework: O framework utilizado neste trabalho foi o Octoprint, porém durante o desenvolvimento algumas dificuldades foram surgindo por causa das limitações que ele possui. Algumas dessas limitações foram possíveis de contornar, porém o funcionamento necessitou ser simplificado, não podendo atuar no sistema da forma que era desejado, ou seja, não apenar pausando ou cancelando a impressão, mas também realizando algumas operações quando o problema surgia para a própria impressora tentar resolver algum problema que venha surgir. Por exemplo podemos citar a possibilidade de quando o bico da impressora entupir, a própria impressora aumentar sua temperatura, retrair e extrudar novamente o filamento, sendo uma operação que em alguns caso resolve o problema de extrusão. Portanto, desenvolver um framework que não ofereça limitações, que possibilite gerenciar todos os sensores que venham a ser implementados e possa centralizar todas as funções que o DT possuirá seria um dos trabalhos de grande relevância. Uma alternativa é utilizar o Microsoft Azure que possui uma ferramenta que possibilita a criação de um framework para ser utilizado justamente no desenvolvimento de um DT.

2. Desenvolvimento do firmware: o firmware utilizado também possui algumas limitações. Logo, é possível adaptar ou contruir totalmente um firmware que forneça as funções necessárias para a implementação do DT. Uma alternativa viável é adaptar o firmware Marlin, pois ele é open-source, possuindo uma vasta comunidade e atualizações constantes, além de que algumas fabricantes de impressoras 3D baseadas na tecnologia de FDM fazem o uso do Marlin ou até mesmo modificam-o.

3. Desenvolver um sistema de detecção de perda de passos: desenvolver um sistema que detecte perda de passo durante a impressão seria interessante, pois evitaria que, quando ocorresse perda de passo durante a impressão seria possível cancelá-la ou então atuar no sistema de forma que o motor retornasse para a possição correta e a impressão continuasse corretamente. A utilização desse sistema evitaria que o material utilizado na impressão na qual o erro ocorreu fosse descartado e o tempo de operação perdido, diminuindo assim os gasto de operação.

4. Desenvolver um sistema de detecção da tensão das correias: as correias que movimentam a impressora possuem um grande influência na qualidade das peças impressas. Portanto, mensurar está variável possibilitaria um controle maior sobre a qualidade das peças impressas, além de identificar quando é necessário ajustá-la, realizando assim uma manutenção preditiva. Durante o desenvolvimento deste trabalho foi visto que existe uma correlação entre a vibração, obtida por meio de um acelerômetro no extrusor da impressora, e a tensão na correia estão correlacionados, logo, é possível desenvolver um sistema que determine, a partir dessa correlação, a tensão da correia e

o ajuste que deve ser feito para otimizar a qualidade de impressão. Uma alternativa válida para processar os dados obtidos pelo acelerômetro e obter assim informações relevantes sobre o sistema é calculando a densidade espectral de potência, porém é necessário realizar estudos mais aprofundados sobre o assunto.

5. Desenvolver o sistema de detecção de problemas estruturais: problemas extruturais também influênciam a qualidade das peças impressas, pois podem aumentar a vibração do sistema. A utilização de um sistema que possibilite identificar problemas estruturais permitiria otimizar a qualidade das peças impressas, além de identificar a necessidade de realizar uma manutenção preditiva. A utilização de acelerômetros pode ser uma alternativa, porém necessita de estudos mais aprofundados sobre o assunto.

6. Desenvolver uma inteligência artificial (IA): desenvolver uma inteligência artificial, não apenas para detectar erros de impressão, mas também que o próprio sistema atue na operação e otimize o sistema. Um exemplo de atuação que a IA poderia realizar é a otimização do PID do extrusor durante a impressão, pois quando o PID não está devidamente ajustado, problemas de impressão, tanto problemas na qualidade superficiais quanto problemas nas propriedades mecânica, podem vir a ocorrer.

7. Desenvolvimento de um banco de dados: uma dos sistemas de grande importância para a Indústria 4.0, assim como para a implementação de um DT, é a aquisição, análise e processamento de dados. Portanto, desenvolver um banco de dadosque possibilite o armazenamento de um número muito grande de dados seria excenssial.

8. Desenvolver um sistema de processamento em nuvem: desenvolver um sistema para que os dados obtidos sejam processados na nuvem é outro ponto de grande importância, sendo uma das tecnologias que habilitam a Industia 4.0.

Referências

3DWORK. **Install and configure BLTouch / 3DTouch in Marlin 2.0.x (MKS Gen, SKR, Anet A8, RAMPS)**. 2020. [Online; acesso em 05 de maio de 2022]. Disponível em: <https://3dwork.io/en/configure-bltouch-in-marlin/>. Citado na p. 50.

ALEX GOROD, B. S.; BOARDMAN, J. System-of-Systems Engineering Management: A Review of Modern History and a Path Forward. **IEEE**, 2008. Citado na p. 28.

ANALOG DEVICES. **ADXL345**. Disponível em: https://pdf1.alldatasheet.com/datasheet-pdf/view/254714/AD/ADXL345.html – acesso em 21 abril 2022. Citado na p. 44.

ANDERSON, C. **Makers: the new Industrial revolution**. Crown Business, 2012. Citado na p. 26.

ANDREW K.S. JARDINE, D. L.; BANJEVIC, D. A review on machinery diagnostics and prognostics implementing condition-based maintenance. **Elsevier**, 2005. Citado na p. 27.

AUTOCOREROBOTICA. **Câmera Usb 5Mp para Raspberry Pi**. 2022. [Online; acesso em 05 de maio de 2022]. Disponível em: <https://www.autocorerobotica.com.br/camera-usb-5mp-para-raspberry-pi>. Citado na p. 49.

B.S. DHILLON, P. **Engineering maintenance: A modern approach**. CRC Express, 2002. Citado na p. 26.

BENJAMIN S BLANCHARD, D. C. V.; PETERSON, E. L. **Maintainability: A Key to Effective Serviceability and Maintenance Management**. 1995. Citado na p. 26.

BIGTREETECH. **File:** Figura ilustrativa do sensor de filamento inteligente da Bigtreetech. Upload de Bigtreetech. Disponível em: https://github.com/bigtreetech/smart-filament-detection-module/blob/master/manual – acesso em 12 abr. 2022. Citado na p. 45.

CARNEIRO, L. R. R.; SOUZA TAVARES, J. J.-P. Z. de. Design and implementation of 3D printer for Mechanical Engineering. **International Journal for Innovation Education and Research**, v. 9, n. 3, 2021. Citado na p. 41.

CLAUS EMMELMANN JANNIS KRANZ, D. H.; WYCISK, E. Additive manufacturing of metals. **Elsevier**, 2016. Citado na p. 26.

CREALITY. **File:** ender5.jpg. Upload de Creality. Disponível em: https://www.creality.com/ – acesso em 11 abr. 2022. Citado na p. 40.

ELECTROPEAK. **ADXL345**. 2022. [Online; acesso em 05 de maio de 2022]. Disponível em: <https://electropeak.com/accelerometer-sensor-adxl345-digital>. Citado na p. 44.

EUGENIO BRUSA, A. C.; FERRETTO, D. **Systems Engineering and Its Application to Industrial Product Development**. Springer, 2017. v. 134. Citado nas pp. 29, 30.

FEI TAO, F. S.; LIU, A. Digital twin-driven product design framework. **International Journal of Production Research**, v. 57, n. 12, 2019. Citado na p. 16.

FENGMING CUI, L. M.; LI, L. Development of smart nursing homes using systems engineering methodologies in industry 4.0. **Enterprise Information Systems**, 2018. Citado nas pp. 29–31.

FORSBERG, K.; MOOZ, H. The Relationship of System Engineering to the Project Cycle. **Center for Systems Management**, 1996. Citado nas pp. 29, 30.

FOUNDATION, N. S. **Workshop on ˝Cyber-Physical Systems"**. Texas, US, 2016. Citado na p. 20.

GLAESSGEN, E.; STARGEL, D. The Digital Twin Paradigm for Future NASA and U.S. Air Force Vehicles. **American Institute of Aeronautics and Astronautics**, 2013. Citado na p. 23.

GRIEVES, M. **Digital twin: manufacturing excellence through virtual factory replication**. Mar. 2015. Citado na p. 23.

HASKINS, C. **SYSTEMS ENGINEERING HANDBOOK**. INCOSE, 2006. Citado nas pp. 28, 29.

JEFF HORN, L.; SMITH, M. **Reconceptualizing the Industrial Revolution**. MIT Press, 2010. Citado na p. 18.

KAI DING, F. C.; ZHANG, F. Defining a Digital Twin-based CyberPhysical Production System for autonomous manufacturing in smart shop floors. **International Journal of Production Research**, 2019. Citado na p. 23.

KUSIAK, A. Smart manufacturing. **International Journal of Production Research**, p. 1–11, jul. 2017. Citado na p. 19.

LEE, E. A.; SESHIA, S. A. **Introduction to Embedded Systems, A Cyber-Physical Systems Approach**. 2. ed.: MIT Press, 2017. Citado na p. 20.

LI DA XU, E. L. X.; LI, L. Industry 4.0: state of the art and future trends. **International Journal of Production Research**, v. 56, n. 8, 2018. Citado nas pp. 18, 19, 21–23.

LI DA XU, W. H.; LI, S. Internet of Things in Industries: A Survey. **IEEE**, 2013. Citado nas pp. 21, 22.

MENDERES KAM, A. İ.; SARUHAN, H. INVESTIGATION THE EFFECTS OF 3D PRINTER SYSTEM VIBRATIONS ON MECHANICAL PROPERTIES OF THE PRINTED PRODUCTS. **Sigma Journal of Engineering and Natural Sciences**, 2018. Citado na p. 44.

PRUSA 3D. **How to assemble the Prusa Face Shield - RC2/RC3**. 2020. [Online; acesso em 05 de maio de 2022]. Disponível em: <https://help.prusa3d.com/guide/how-to-assemble-the-prusa-face-shield-rc2-rc3_125495>. Citado na p. 15.

PRUSA RESEARCH. **File:** Prusa.jpg. Upload de Prusa Research. Disponível em: https://www.prusa3d.com/ – acesso em 11 abr. 2022. Citado na p. 40.

RAGUNATHAN RAJKUMAR INSUP LEE, L. S.; STANKOVIC, J. Cyber-physical systems: The next computing revolution. **Design Automation Conference**, 2010. Citado na p. 20.

RASPBERRY FOUNDATION. **File:** ender5.jpg. Upload de Raspberry Foundation. Disponível em: https://www.raspberrypi.org/ – acesso em 11 abr. 2022. Citado na p. 41.

ROSMAINI AHMAD, S. K. An overview of time-based and condition-based maintenance in Industrial application. **Elsevier**, 2012. Citado nas pp. 26–28.

RUPARELIA, N. B. Software Development Lifecycle Models. **ACM SIGSOFT Software Engineering Notes**, v. 35, n. 3, p. 8–13, 2010. Citado nas pp. 29, 30, 32.

S.O. DUFFUAA M. BEN-DAYA, K. A.-S.; ANDIJANI, A. A generic conceptual simulation model for maintenance systems. **Journal of Quality in Maintenance Engineering**, v. 7, p. 207–219, 2001. Citado na p. 26.

TAN, L.; WANG, N. Future Internet: The Internet of Things. **IEEE**, 2010. Citado na p. 20.

TAO, F.; CHENG, J. Digital twin-driven product design, manufacturing and service with big data. **CrossMark**, p. 1–14, mar. 2017. Citado na p. 24.

TAO, F.; ZHANG, M. Digital Twin Shop-Floor: A New Shop-Floor Paradigm Towards Smart Manufacturing. **IEEE**, 2017. Citado nas pp. 22, 24.

TEAMGLOOMY. **Connecting a BLTouch to an SKR v1.4**. 2022. [Online; acesso em 05 de maio de 2022]. Disponível em: <https://teamgloomy.github.io/skr_1.4_bltouch.html>. Citado na p. 50.

THE PI4J PROJECT. **Pin Numbering - Raspberry Pi 2 Model B**. 2019. [Online; acesso em 05 de maio de 2022]. Disponível em: <https://pi4j.com/1.2/pins/model-2b-rev1.html>. Citado nas pp. 47, 49.

WAEYENBERGH, G.; PINTELON, L. A framework for maintenance concept development. **Elsevier**, 2002. Citado na p. 53.

WAGNER PEDRO. **Raspberry Pi comemora 10 anos; confira todos os modelos lançados até hoje**. 2022. [Online; acesso em 05 de maio de 2022]. Disponível em: <https://tecnoblog.net/especiais/raspberry-pi-comemora-10-anos-confira-todos-os-modelos-lancados-ate-hoje/>. Citado na p. 42.

WEI GAO, Y. Z.; ZAVATTIERI, P. D. The Status, Challenges, and Future of Additive Manufacturing in Engineering. **Computer-Aided Design**, 2015. Citado nas pp. 25, 26.

WEICHAO LUO, T. H.; WEI, Y. Digital twin for CNC machine tool: modeling and using strategy. **CrossMark**, v. 10, p. 1–12, jul. 2018. Citado na p. 25.

WIKTOR, P. Coupled Cartesian Manipulators. **Elsevier**, 2020. Citado na p. 40.

WU, S. Preventive Maintenance Models: A Review. **Series in Reliability Engineering**, 2011. Citado na p. 26.

XIAOLIN JIA1 QUANYUAN FENG, T. F.; LEI, Q. RFID Technology and Its Applications in Internet of Things. **IEEE**, 2012. Citado nas pp. 20, 21.

YONGXIN LIAO, F. D.; RAMOS, L. F. P. Past, present and future of Industry 4.0 - a systematic literature review and research agenda proposal. **International Journal of Production Research**, 2017. Citado na p. 18.

ZHU, G.-H.; LEE, J.-S. Development of Imitated-Handwriting Systems Using PLC-Controlled CoreXY Mechanisms. **IEEE**, 2019. Citado nas pp. 40, 41.

Apêndices

APÊNDICE A – Códigos de programação

A.1 Código para processar os dados e atuar no sistema de manutenção preditiva

Código A.1 – Código de Python

```
from octorest import OctoRest
import gspread
from oauth2client.service_account import ServiceAccountCredentials
import time
from pushbullet import Pushbullet
import numpy as np
from threading import Thread

imprimindo = False

pb = Pushbullet('token')

   # token obtido no Pushbullet

client_octo = OctoRest(url="host-url", apikey="apikey")
                                                    # acessa o
   Octoprint por meio da API

scope = ['https://spreadsheets.google.com/feeds']
cred =
   ServiceAccountCredentials.from_json_keyfile_name('credentials.json',
   scope)                          # acessa o Google Sheet por
   meio das credenciais
client_sheet = gspread.authorize(cred)

wks = client_sheet.open_by_key('apikey')

   # API do Google Sheet

sheet = wks.get_worksheet(0)

   # acessa a primeira aba da planilha

def get_info():

   # obtém o estado da impressora acessando o Octoprint
```

```
24      message = client_octo.connection_info()

           # envia comando para o Pushbullet
25      status = (message.get("current")).get("state")
                                                                  #
           obtém apenas a informação que deseja
26      return status
27
28  def status_da_impressora():

       # atualiza a planilha do Google Sheet com o estado da impressora
29      status_imp = get_info()
30      status_bd = sheet.cell(2, 4).value
31
32      if(status_imp == 'Closed'):

           # caso a impressora esteja desconectada
33          if(status_bd == "DISPONÍVEL" or status_bd == "IMPRIMINDO"):
34              sheet.update_cell(2, 4, "DESCONECTADA")
35              print("Impressora está desconectada! Verificação será
                   feita novamente dentro de 1 minuto.")
36          elif(status_bd == "EM MANUTEN  O" or status_bd ==
               "ESPERANDO MANUTEN  O"):
37              print("Aguardando a realização da manutenção!")
38          else:
39              print("Impressora está desconectada! Verificação será
                   feita novamente dentro de 1 minuto.")
40              sheet.update_cell(2, 4, "DESCONECTADA")
41      elif(status_imp == 'Operational'):

           # caso a impressora esteja conectada
42          if(status_bd == "DESCONECTADA"):
43              sheet.update_cell(2, 4, "DISPONÍVEL")
44              print("Impressora está pronta para produzir!")
45          elif(status_bd == "EM MANUTEN  O" or status_bd ==
               "ESPERANDO MANUTEN  O"):
46              print("Aguardando a realização da manutenção!")
47          elif(status_bd == "IMPRIMINDO" and imprimindo == False):
48              sheet.update_cell(2, 4, "DISPONÍVEL")
49              print("Impressora está pronta para produzir!")
50          else:
51              sheet.update_cell(2, 4, "DISPONÍVEL")
52
53      return status_bd
54
55  def bed_mesh():

       # obt m o valor dos pontos do nivelamento da mesa aquecida
56      cell = ""
57      bed_mesh_old = sheet.get('E2:K8')

           # acessa os valores guardados no Google Sheet
```

```
58
59      message = client_octo.settings()

           # envia para o Octoprint a requisição das configurações
60
61      bed_mesh_new =
           ((message.get("plugins")).get('bedlevelvisualizer')).get('stored_mesh
                                 # obt m das configurações apenas os
           valores do nivelamento
62
63      for i in range(6):

           # verifica se o valor obtido é igual ao anterior, caso seja
           significa que ainda não foi atualizado
64          bed_mesh_new =
               ((message.get("plugins")).get('bedlevelvisualizer')).get('stored_
65          if bed_mesh_new[0][0] != bed_mesh_old[0][0]:
66              break
67
68          time.sleep(10)
69
70      sheet.update('E2:K8', bed_mesh_new)
71
72      bed_mesh_new = list(np.float_(bed_mesh_new))
73
74      maximum = max([max(r) for r in bed_mesh_new])
75      minimum = min([min(r) for r in bed_mesh_new])
76
77      for i in range(7):

           # verifica a progreção do erro do nivelamento para
           atualizar o Google Sheet
78          for x in range(7):
79              if i == 0:
80                  cell = "E"+str(2+x)
81              elif i == 1:
82                  cell = "F"+str(2+x)
83              elif i == 2:
84                  cell = "G"+str(2+x)
85              elif i == 3:
86                  cell = "H"+str(2+x)
87              elif i == 4:
88                  cell = "I"+str(2+x)
89              elif i == 5:
90                  cell = "J"+str(2+x)
91              elif i == 6:
92                  cell = "K"+str(2+x)
93
94              if bed_mesh_new[x][i] > float(bed_mesh_old[x][i]):
95                  sheet.format(cell, {"backgroundColor": {"red":
                       1.0, "green": 0.0, "blue": 0.0}})
96              elif bed_mesh_new[x][i] < bed_mesh_old[x][i]:
```

```
97                  sheet.format(cell, {"backgroundColor": {"red":
                        0.0, "green": 1.0, "blue": 0.0}})
98              else:
99                  sheet.format(cell, {"backgroundColor": {"red":
                        1.0, "green": 1.0, "blue": 1.0}})
100
101     print("BED_MESH_OUTPUT atualizado!")
102
103     if (abs(maximum) or abs(minimum)) >= 0.35:

            # verifica se algum dos valores utrapassou o limiar de
            0.35mm
104          sheet.update_cell(2, 4, "ESPERANDO MANUTEN  O")
105          sheet.update_cell(2, 12, "Mesa descalibrada")
106          client_octo.cancel()
107          print("A mesa aquecida necessita ser recalibrada")
108
109 def notification():

    # obt m as notificaç~eos do Pushbullet
110     time.sleep(0.3)
111     decisao = ""
112
113     while True:
114         try:
115             push = pb.get_pushes()
116             action  = (push[0]).get('title')

                # obt m apenas a última notificação e apenas seu
                título
117
118             pb.delete_pushes()

                # deleta todas as notificações
119
120             if action == 'Cancelled':

                # atualiza o Google Sheet dependendo da notificação
                obtida
121                  sheet.update_cell(2, 4, "DISPONÍVEL")
122                  print("IMPRESS O CANCELADA!")
123                  imprimindo = False
124                  break
125
126             elif action == 'Finished':
127                 sheet.update_cell(2, 4, "DISPONÍVEL")
128                 print("IMPRESS O FINALIZADA!")
129                 imprimindo = False
130                 break
131
132             elif action == 'Failed':
133                 sheet.update_cell(2, 4, "ESPERANDO MANUTEN  O")
```

```
134                 print("IMPRESS O FALHOU!")
135                 imprimindo = False
136                 break
137 
138             elif action == 'The Spaghetti Detective - Failure
                    alert!':
                    # pergunta ao operador qual decisão tomar, ou seja,
139 
140                 decisao = input('Um erro foi detectado durante a
                        impressão. Deseja continuar imprimindo? Digite
                        Y ou N.')
141                 while (decisao != 'Y') or (decisao != 'N'):
142                     decisao = input('Um erro foi detectado durante
                            a impressão. Deseja continuar imprimindo?
                            Digite Y ou N.')
143                 if decisao == "N":
144                     client_octo.cancel()
145 
146                 imprimindo = False
147                 break
148 
149             elif action == 'Filament runout!':
150                 sheet.update_cell(2, 12, "Problema de extrusão")
151                 sheet.update_cell(2, 4, "ESPERANDO MANUTEN  O")
152                 print("A impressora apresentou problemas de
                        extrusão. Portanto a impressão foi cancelada.")
153                 client_octo.cancel()
                        # cancela a impressão
154                 imprimindo = False
155                 break
156 
157         except:
158             decisao = ""
159 
160         time.sleep(1)
161 
162 def main():
163     while True:
            # executa o código em loop
164         time.sleep(1)
165         status = status_da_impressora()
166         while(status != 'DISPONÍVEL'):
```

```
167             time.sleep(60)
168             status = status_da_impressora()
169
170         print("Impressora está pronta para produzir!")
171         try:
172             push = pb.get_pushes()
173             action  = (push[0]).get('title')
174             pb.delete_pushes()
175
176         except:
177             action = ""
178
179         if action == "Started":

            # caso a impressão inicie, são iniciadas duas threads
180             imprimindo = True
181             sheet.update_cell(2, 4, "IMPRIMINDO")

                # atualiza o Google Sheet com o estado Imprimindo
182             t1 = Thread(target=bed_mesh, daemon=True)

                # thread para obter o valores do nivelamento
183             t2 = Thread(target=notification, daemon=True)
                                                                    #
                thread para obter as notificações
184
185             t1.start()
186             t2.start()
187
188 if __name__ == "__main__":
189     main()
```

A.2 Código para criar os gráficos do acelerômetro

Código A.2 – Código de Python

```
1  import os
2  import numpy as np
3  from matplotlib import mlab
4  import matplotlib.pyplot as plt
5  from scipy.fftpack import fft, ifft
6
7  sample_rate_Hz = 3200
8  length_s = 2
9  os.system(f'sudo adxl345spi -t {length_s} -f {sample_rate_Hz} -s
       out.csv')                                  # executa o código em C
       para obter os dados do aceler metro
10 acc_data = np.genfromtxt('out.csv', delimiter=',', names=True)
                                                  # abre o arquivo
       criado pelo código em C que armazena os dados obtidos
```

```
11 acc_x, freq_x, _ = mlab.specgram(acc_data['x'], Fs=sample_rate_Hz,
       NFFT=sample_rate_Hz * length_s)      # obt m apenas os valores
       do eixo X
12 acc_y, freq_y, _ = mlab.specgram(acc_data['y'], Fs=sample_rate_Hz,
       NFFT=sample_rate_Hz * length_s)      # obt m apenas os valores
       do eixo Y
13 acc_z, freq_z, _ = mlab.specgram(acc_data['z'], Fs=sample_rate_Hz,
       NFFT=sample_rate_Hz * length_s)      # obt m apenas os valores
       do eixo Z
14
15 a = np.mean(acc_x)

       # obtém a componente DC do eixo X
16 acc_x = [x-a for x in acc_x]

       # retira a componente DC dos dados do eixo X
17 a = np.mean(acc_y)

       # obtém a componente DC do eixo Y
18 acc_y = [x-a for x in acc_y]

       # retira a componente DC dos dados do eixo Y
19
20 acc_x = np.concatenate( acc_x, axis=0 )
21 acc_y = np.concatenate( acc_y, axis=0 )
22
23 X = fft(acc_x)

       # aplica a FFT aos dados do eixo X
24 Y = fft(acc_y)

       # aplica a FFT aos dados do eixo Y
25
26 sr = 4600
27
28 N = len(X)

       # variáveis para criar os gráficos do eixo X e do eixo Y
29 n = np.arange(N)
30 T = N/sr
31 freq = n/T
32
33 plt.stem(freq, np.abs(X), 'r', \
34          markerfmt=" ", basefmt="-b")
35 plt.xlim(0, 10)
36 plt.xlabel('Freq (Hz)')
37 plt.ylabel('FFT Amplitude |X(freq)|')
38 plt.plot(freq_x, np.real(ifft(X)),  label = "X")
39
40 plt.savefig('FFT_X.png')
41
42 plt.stem(freq, np.abs(Y), 'r', \
```

```
43          markerfmt=" ", basefmt="-b")
44 plt.xlim(0, 10)
45 plt.xlabel('Freq (Hz)')
46 plt.ylabel('FFT Amplitude |Y(freq)|')
47 plt.plot(freq_y, np.real(ifft(Y)),  label = "Y")
48
49 plt.savefig('FFT_Y.png')
```

A.3 Código para ler os dados do acelerômetro

Código A.3 – Código de Python

```
1  #include <stdio.h>
2  #include <pigpio.h>
3  #include <time.h>
4  #include <math.h>
5  #include <string.h>
6  #include <stdlib.h>
7
8  #define DATA_FORMAT   0x31  // formato do endereço do registrador
9  #define DATA_FORMAT_B 0x0B  // formato dos dados em bites: +/- 16g
       range, 13-bit resolução
10 #define READ_BIT      0x80
11 #define MULTI_BIT     0x40
12 #define BW_RATE       0x2C
13 #define POWER_CTL     0x2D
14 #define DATAX0        0x32
15
16 const int timeDefault = 5;  // tempo de aquisição dos dados
17 const int freqDefault = 5;  // taxa de amostragem padrão em Hz
18 const int speedSPI = 2000000;  // velocidade de comunicação da
       porta SPI em bps
19 const int freqMaxSPI = 100000;  // máxima taxa de amostragem
       utilizando o SPI em Hz
20 const double accConversion = 2 * 16.0 / 8192.0;  // +/- 16g range,
       13-bit resolução
21 const double tStatusReport = 1;  // timeout
22
23 int readBytes(int handle, char *data, int count) {
24     data[0] |= READ_BIT;
25     if (count > 1) data[0] |= MULTI_BIT;
26     return spiXfer(handle, data, data, count);
27 }
28
29 int writeBytes(int handle, char *data, int count) {
30     if (count > 1) data[0] |= MULTI_BIT;
31     return spiWrite(handle, data, count);
32 }
33
34 int main(int argc, char *argv[]) {
```

```
35      int i;
36      int bSave = 0;
37      char vSave[256] = "";
38      double vTime = timeDefault;
39      double vFreq = freqDefault;
40      for (i = 1; i < argc; i++) {  // pula o argv[0] que
           corresponde ao nome do programa
41          if ((strcmp(argv[i], "-s") == 0) || (strcmp(argv[i],
               "--save") == 0)) {
42              bSave = 1;
43              if (i + 1 <= argc - 1) {  // verifica o número de
                   argumentos
44                  i++;
45                  strcpy(vSave, argv[i]);
46              }
47              else {
48                  return 1;
49              }
50          }
51          else if ((strcmp(argv[i], "-t") == 0) || (strcmp(argv[i],
               "--time") == 0)) {
52              if (i + 1 <= argc - 1) {
53                  i++;
54                  vTime = atoi(argv[i]);
55              }
56              else {
57                  return 1;
58              }
59          }
60          else if ((strcmp(argv[i], "-f") == 0) || (strcmp(argv[i],
               "--freq") == 0)) {
61              if (i + 1 <= argc - 1) {
62                  i++;
63                  vFreq = atoi(argv[i]);
64                  if ((vFreq < 1) || (vFreq > 3200)) {
65                      printf("Wrong sampling rate specified!\n\n");
66                      return 1;
67                  }
68              }
69              else {
70                  return 1;
71              }
72          }
73          else {
74              return 1;
75          }
76      }
77
78      // ler os dados do sensor
79
80      // configura a comunicação SPI do sensor
81      int samples = vFreq * vTime;
```

```
82      int samplesMaxSPI = freqMaxSPI * vTime;
83      int success = 1;
84      int h, bytes;
85      char data[7];
86      int16_t x, y, z;
87      double tStart, tDuration, t;
88      if (gpioInitialise() < 0) {
89          printf("Failed to initialize GPIO!");
90          return 1;
91      }
92      h = spiOpen(0, speedSPI, 3);
93      data[0] = BW_RATE;
94      data[1] = 0x0F;
95      writeBytes(h, data, 2);
96      data[0] = DATA_FORMAT;
97      data[1] = DATA_FORMAT_B;
98      writeBytes(h, data, 2);
99      data[0] = POWER_CTL;
100     data[1] = 0x08;
101     writeBytes(h, data, 2);
102
103     double delay = 1.0 / vFreq;  // delay entre as leituras
104
105     // cria vetores para cada eixo que será utilizado no arquivo
           de saída
106     double *at, *ax, *ay, *az;
107     at = malloc(samples * sizeof(double));
108     ax = malloc(samples * sizeof(double));
109     ay = malloc(samples * sizeof(double));
110     az = malloc(samples * sizeof(double));
111
112     // cria vetores para cada eixoqeu será utilizado para
           armazenar os dados obtidos
113     double *rt, *rx, *ry, *rz;
114     rt = malloc(samplesMaxSPI * sizeof(double));
115     rx = malloc(samplesMaxSPI * sizeof(double));
116     ry = malloc(samplesMaxSPI * sizeof(double));
117     rz = malloc(samplesMaxSPI * sizeof(double));
118
119     int statusReportedTimes = 0;
120     double tCurrent, tClosest, tError, tLeft;
121     int j, jClosest;
122
123     tStart = time_time();
124     int samplesRead;
125     for (i = 0; i < samplesMaxSPI; i++) {
126         data[0] = DATAX0;
127         bytes = readBytes(h, data, 7);
128         if (bytes == 7) {
129             x = (data[2]<<8)|data[1];
130             y = (data[4]<<8)|data[3];
131             z = (data[6]<<8)|data[5];
```

```
132             t = time_time();
133             rx[i] = x * accConversion;
134             ry[i] = y * accConversion;
135             rz[i] = z * accConversion;
136             rt[i] = t - tStart;
137         }
138         else {
139             gpioTerminate();
140             printf("Error occurred!");
141             return 1;
142         }
143         tDuration = t - tStart;
144         if (tDuration > tStatusReport *
               ((float)statusReportedTimes + 1.0)) {
145             statusReportedTimes++;
146             tLeft = vTime - tDuration;
147             if (tLeft < 0) {
148                 tLeft = 0.0;
149             }
150             printf("%.2f seconds left...\n", tLeft);
151         }
152         if (tDuration > vTime) {
153             samplesRead = i;
154             break;
155         }
156     }
157     gpioTerminate();
158     printf("Writing to the output file...\n");
159     for (i = 0; i < samples; i++) {
160         if (i == 0) {
161             tCurrent = 0.0;
162             jClosest = 0;
163             tClosest = rt[jClosest];
164         }
165         else {
166             tCurrent = (float)i * delay;
167             tError = fabs(tClosest - tCurrent);
168             for (j = jClosest; j < samplesRead; j++) {
169                 if (fabs(rt[j] - tCurrent) <= tError) {
170                     jClosest = j;
171                     tClosest = rt[jClosest];
172                     tError = fabs(tClosest - tCurrent);
173                 }
174                 else {
175                 break;
176                 }
177             }
178         }
179         ax[i] = rx[jClosest];
180         ay[i] = ry[jClosest];
181         az[i] = rz[jClosest];
182         at[i] = tCurrent;
```

```
183     }
184     FILE *f;
185     f = fopen(vSave, "w");
186     fprintf(f, "time, x, y, z\n");
187     for (i = 0; i < samples; i++) {
188         fprintf(f, "%.5f, %.5f, %.5f, %.5f \n", at[i], ax[i],
            ay[i], az[i]);
189     }
190     fclose(f);
191 
192     printf("Done\n");
193     return 0;
194 }
```