

**UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS**

**Giovana Fernandes Nogueira
Lorena Colautto Borges**

**Análise de Business Intelligence para Aplicações de
Robótica**

São Carlos

2019

**Giovana Fernandes Nogueira
Lorena Colautto Borges**

Análise de Business Intelligence para Aplicações de Robótica

Monografia apresentada ao Curso de Engenharia Elétrica com Ênfase em Eletrônica, da Escola de Engenharia de São Carlos da Universidade de São Paulo, como parte dos requisitos para obtenção do título de Engenheiro Eletricista.

Orientador: Prof. Dr. Thiago Boaventura Cunha

**São Carlos
2019**

AUTORIZO A REPRODUÇÃO E DIVULGAÇÃO TOTAL OU PARCIAL DESTE TRABALHO, POR QUALQUER MEIO CONVENCIONAL OU ELETRÔNICO, PARA FINS DE ESTUDO E PESQUISA, DESDE QUE CITADA A FONTE.

Ficha catalográfica elaborada pela Biblioteca Prof. Dr. Sérgio Rodrigues
Fontes da EESC/USP

N778a	<p>Nogueira, Giovana Fernandes</p> <p>Análise de <i>business intelligence</i> para aplicações de robótica / Giovana Fernandes Nogueira, Lorena Colautto Borges; orientador Thiago Boaventura Cunha. -- São Carlos, 2019.</p> <p>Monografia (Graduação em Engenharia EElétrica com Ênfase em Eletrônica) -- Escola de Engenharia de São Carlos da Universidade de São Paulo, 2019.</p> <p>1. Big data. 2. Aplicações de <i>business intelligence</i>. 3. Robótica vestível. 4. Robôs quadrúpedes. 5. <i>Design de data warehouse</i>. 6. ETL. 7. Dashboard. I. Borges, Lorena Colautto. II. Título.</p>
-------	--

FOLHA DE APROVAÇÃO

Nome: Giovana Fernandes Nogueira

Título: "Análise de business intelligence para aplicações de robótica"

Trabalho de Conclusão de Curso defendido e aprovado
em 11 / 06 / 2019,

com NOTA 9,5 (nove, cinco), pela Comissão Julgadora:

Prof. Dr. Thiago Boaventura Cunha - Orientador - SEM/EESC/USP

Prof. Titular Glauco Augusto de Paula Caurin - SAA/EESC/USP

Prof. Associado Rogério Andrade Flauzino - SEL/EESC/USP

Coordenador da CoC-Engenharia Elétrica - EESC/USP:
Prof. Associado Rogério Andrade Flauzino

FOLHA DE APROVAÇÃO

Nome: Lorena Colautto Borges

Título: "Análise de business intelligence para aplicações de robótica"

Trabalho de Conclusão de Curso defendido e aprovado
em 11 / 06 / 2019,

com NOTA 9,5 (nove, cinco), pela Comissão Julgadora:

Prof. Dr. Thiago Boaventura Cunha - Orientador - SEM/EESC/USP

Prof. Titular Glauco Augusto de Paula Caurin - SAA/EESC/USP

Prof. Associado Rogério Andrade Flauzino - SEL/EESC/USP

Coordenador da CoC-Engenharia Elétrica - EESC/USP:
Prof. Associado Rogério Andrade Flauzino

AGRADECIMENTOS

Em nome de Giovana, agradeço em primeiro lugar a minha mãe Ana Maria, meu maior exemplo e incentivadora assídua que sempre me encorajou a seguir meus sonhos e acreditar no meu potencial. Agradeço também ao restante da nossa pequena (grande) família: minha irmã e guerreira Aline, meu irmão e companheiro Alan, e meu irmão e parceiro Jari; sem os *petelecos* de vocês eu não teria chegado até aqui. Amanda, Bia, Sarah e Luigi: vocês foram as melhores adições que poderíamos ter tido, obrigada por agregarem tanto à nossa história!

Agradeço também aos queridos amigos que me suportaram e apoiaram ao longo dessa (não tão trivial) graduação; nossas alegrias e apertos compartilhados ficarão para sempre gravados em mim. À Grifinória, eterna melhor casa da 014. Aos piratas do *Warthog*, que nunca falham uma missão. Aos amigos de tantas gerações da SA-SEL, muito obrigada pela confiança e ensinamentos compartilhados.

Claramente não posso deixar de citar a Casinha de Sapê, o bonde mais convicto do eixo *sanca-sampa*. Especial agradecimento à Dani, minha companheira, amiga, conselheira... tantas coisas - eu adorei dividir essa jornada com você. Agradecimento especial também pra Lorena, amiga de sala, futsal, casa e agora TCC! Obrigada pela parceria em tudo que fizemos juntas. Isadora, Mari, Lara e Ágata, vocês moram no meu coração.

Em fim, agradeço à Universidade de São Paulo, que me forma não só como Engenheira Eletricista mas também como cidadã. Espero com todas as forças que muitos outros brasileiros possam desfrutar do mesmo ensino público de excelência que eu desfrutei. Professores, obrigada pelos ensinamentos ao longo desses anos. Em especial, agradeço ao professor Thiago Boaventura por prontamente nos receber e ajudar a construir o trabalho aqui apresentando.

Em nome de Lorena, agradeço à Universidade de São Paulo e aos bons professores que passaram por meu caminho e que, cada um a sua maneira, me prepararam para as oportunidades que estão por vir. Em especial, agradeço ao professor Thiago pelo suporte e orientação durante o desenvolvimento deste projeto.

Aos meus pais, Nivaldo e Simone, pessoas imprescindíveis em todos os momentos de minha vida, que me dão apoio incondicional, me auxiliando nos momentos mais difíceis e ficando orgulhosos com cada vitória. Obrigada por todos os sacrifícios e por terem me proporcionado todas essas oportunidades. Ao meu irmão, Pedro, que desde sua chegada mudou a maneira como enxergo o mundo. Espero e desejo muito que, apesar dos anos conturbados que estão por vir, você possa ter oportunidades no mínimo equivalentes.

Ao Centro Acadêmico Armando de Salles Oliveira, que me proporcionou tantos amigos, atividades, reflexões e bons momentos. Em especial, agradeço ao time de futsal feminino da Associação Atlética Acadêmica Campus de São Carlos USP, lugar que aprendi muito sobre trabalho em equipe, dedicação e liderança.

À República Sedução, lugar que foi meu lar durante todo esse período, e à todas as pessoas que por lá passaram e moraram, em especial à Ana Caroline, Carolina, Marina, Livia, Marilizi, Pamela, Bianca, Caroline e Milena, amigas que tanto me apoiaram e que sempre terei boas lembranças.

À turma da Eletrônica 014, em especial aos Grifinórios, pessoas que muito contribuíram nessa caminhada e que a deixaram menos tortuosa e com mais sorrisos. Também agradeço especialmente à Giovana, amiga na vida e parceira neste projeto: obrigada pela construção que fizemos!

Às amigas do Sapê, Isadora, Daniele, Alessandra, Lara e Ágata que estão me acompanhando nessa próxima etapa. Agradeço também à Mariana, amiga e companheira: se você houvesse aparecido no meu caminho há alguns anos atrás, o meu mundo seria diferente e talvez um pouco menos colorido.

“[...] collecting data isn’t enough. Data science only matters if data drives action.”
Jeremy Stanley

RESUMO

NOGUEIRA, G.F.; BORGES, L. C. **Análise de Business Intelligence para Aplicações de Robótica**. 2019. 96p. Monografia (Trabalho de Conclusão de Curso) - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2019.

O presente trabalho relata o processo de desenvolvimento de um *dashboard* na plataforma Power BI para análises de experimentos realizados com robôs. Esta é uma aplicação de *business intelligence* em uma área ainda pouco explorada na literatura acadêmica. O principal objetivo é proporcionar, de maneira intuitiva e amigável ao usuário, um ambiente interativo e com visualizações adequadas para a geração de *insights* sobre os experimentos, os diferentes tipos de controladores utilizados nos mesmos e também sobre os robôs. Além disso, o trabalho apresenta em detalhes as etapas necessárias para trazer os dados dos robôs às visualizações: design de um *data warehouse* e processos de ETL (*extract, transform and load*); para isso, *softwares* como MatLab e Power Query foram utilizados. Os experimentos analisados neste estudo de caso são provenientes de testes de diferentes controladores com aplicação para robôs quadrúpedes e exoesqueletos (robôs vestíveis), e uma proposta para padronização de armazenamento e registro deles também é apresentada.

Palavras-chave: *Big data*. Aplicações de *business intelligence*. Robótica vestível. Robôs quadrúpedes. Design de *data warehouse*. ETL. *Dashboard*.

ABSTRACT

NOGUEIRA, G.F.; BORGES, L. C. **Business Intelligence Analysis for Control System Applied to Quadruped Robots**. 2019. 96p. Monografia (Trabalho de Conclusão de Curso) - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2019.

This paper reports a dashboard development process in Power BI platform for experiment analysis performed with robots. This is a business intelligence application in an area that has not yet been well explored in the academic literature. The main goal is to provide an intuitive and user-friendly environment with adequate visualizations for insights generation about the experiments, different types of controllers and also about the robots. In addition, this study presents in detail all necessary steps to bring the data from robots to visualizations: design of a data warehouse and ETL (extract, transform and load) processes; for this, softwares like MatLab and Power Query were used. The experiments analyzed in this case study come from tests of different controllers with application for hydraulic quadruped robots and exoskeletons (wearable robots). At last, a proposal for storage and record is presented.

Keywords: Big data. Business intelligence applications. Wearable robots. Quadruple robots. Design of data warehouse. ETL. Dashboard.

LISTA DE FIGURAS

Figura 2.1 – Tabela fato e tabelas dimensionais em esquema estrela	33
Figura 2.2 – Fluxo dos dados	33
Figura 2.3 – Quadrante mágico de Gartner para plataformas de BI	35
Figura 2.4 – Robô quadrúpede com atuadores hidráulicos - HyQ	36
Figura 2.5 – Composição de uma perna do robô quadrúpede HyQ	37
Figura 2.6 – Diagrama de blocos do sistema de controle do robô HyQ	38
Figura 2.7 – Representação linear da interação robô-humano	39
Figura 2.8 – Plataforma de teste FC2D	40
Figura 3.1 – Exemplo de menu de seleção do Power Query para estabelecer conexão da ferramenta com uma base de dados	46
Figura 3.2 – Substituir valores no Power Query	47
Figura 3.3 – Remover colunas e/ou linhas no Power Query	47
Figura 3.4 – Aplicar filtros de acordo com o nome de uma coluna no Power Query	48
Figura 3.5 – Mudar o tipo de dado armazenado em uma coluna (inteiro, decimal, texto, etc) no Power Query	48
Figura 3.6 – Apendar dados de diferentes tabelas em uma única no Power Query	48
Figura 3.7 – Página inicial do Power BI	50
Figura 4.1 – Tabela "Robot"	53
Figura 4.2 – Tabela "Controller"	54
Figura 4.3 – Tabela "Experiments"	54
Figura 4.4 – <i>Data warehouse</i> do projeto	55
Figura 4.5 – Exemplo de uma tabela despivotada	57
Figura 4.6 – Página "Home" do <i>dashboard</i>	60
Figura 4.7 – Página "FC2D - Kinematics" do <i>dashboard</i>	61
Figura 4.8 – Menu de seleção de experimentos da página "FC2D - Kinematics" e análise de dois experimentos simultaneamente	61
Figura 4.9 – Botões de escolha das variáveis da página "FC2D - Kinematics"	62
Figura 4.10– <i>Slider</i> de tempo da página "FC2D - Kinematics"	62
Figura 4.11–Cartões de descrição do experimento e controlador da página "FC2D - Kinematics"	62
Figura 4.12–Menu padrão para navegação no <i>dashboard</i>	63
Figura 4.13–Página "FC2D - Pressures" do <i>dashboard</i>	63
Figura 4.14–Página "FC2D - References" do <i>dashboard</i>	64
Figura 4.15–Página "FC2D - Kinematics x F" do <i>dashboard</i>	64
Figura 4.16–Página "HYQ Leg - Variables x Time" do <i>dashboard</i>	65

Figura 4.17–Botões de seleção do experimento da página "HYQ Leg - Variables x Time"	65
Figura 4.18–Lista de primeiros atributos das variáveis do experimento "K3000" . . .	66
Figura 4.19–Página "HYQ Leg - Additional Information"do <i>dashboard</i>	66
Figura 4.20–Fluxograma para a inserção de novos experimentos no <i>dashboard</i>	67

LISTA DE TABELAS

Tabela 2.1 – Principais especificações do robô HyQ	38
Tabela 3.1 – Descrição dos experimentos que geraram os <i>logs</i> usados no projeto . . .	43
Tabela 3.2 – Descrição das principais variáveis da perna do robô HyQ	45
Tabela 3.3 – Descrição das variáveis da plataforma FC2D	46
Tabela 4.1 – Descrição das características dos elementos que descrevem um experimento	53

LISTA DE ABREVIATURAS E SIGLAS

BI	- <i>Business Intelligence</i> (Inteligência de Negócio)
HyQ	- <i>Hydraulic Quadruped robot</i> (Robô Quadrúpede Hidráulico)
FC2D	- <i>Force Control 2 Degrees of Freedom</i> (Controle de Força com 2 Graus de Liberdade)
IoT	- <i>Internet of Things</i> (Internet das Coisas)
BA	- <i>Business Analyst</i> (Analista de Negócios)
ETL	- <i>Extract, Transform and Load</i> (Extrair, Transformar e Carregar)
DW	- <i>Data Warehouse</i> (Armazém de Dados)
TI	- Tecnologia da Informação
API	- <i>Application Programming Interface</i> (Interface de Programação de Aplicação)
DOF	- <i>Degree of Freedom</i> (Grau de Liberdade)
HAA	- <i>Hip Abduction/Adduction</i> (Abdução e Adução do Quadril)
HFE	- <i>Hip Flexion Extension</i> (Flexão e Extensão do Quadril)
KFE	- <i>Knee Flexion Extension</i> (Flexão e Extensão do Joelho)
L	- Largura
C	- Comprimento
A	- Altura
ITT	- Instituto Italiano de Tecnologia
ETH Zurich	- Instituto Federal Suíço de Tecnologia de Zurique
ASCII	- <i>American Standard Code for Information Interchange</i> (Código Padrão Americano para Intercâmbio de Informações)

LISTA DE SÍMBOLOS

m	- Metro
kg	- Kilograma
N	- Newton
MPa	- Mega Pascal

SUMÁRIO

1	INTRODUÇÃO	27
1.1	Motivação do projeto	28
1.2	Objetivos	28
2	REVISÃO BIBLIOGRÁFICA	29
2.1	<i>Big data</i>	29
2.2	<i>Business intelligence</i> (BI)	31
2.3	Aplicações robóticas	36
2.3.1	Robô quadrúpede (HyQ - <i>Hydraulic Quadruped</i>)	36
2.3.2	<i>Force Control 2 Degrees of Freedom</i> (FC2D)	38
3	MATERIAIS E MÉTODOS	41
3.1	Design do <i>data warehouse</i>	41
3.2	ETL - <i>Extract, Transform and Load</i>	42
3.3	Criação do <i>dashboard</i>	49
4	DESENVOLVIMENTO E RESULTADOS EXPERIMENTAIS	53
4.1	Design do <i>data warehouse</i>	53
4.2	Desenvolvimento do ETL	55
4.2.1	Transformação dos dados da perna do robô quadrúpede HyQ	56
4.2.2	Transformação dos dados da plataforma FC2D	58
4.3	Criação do <i>dashboard</i>	60
4.3.1	FC2D	60
4.3.2	HyQ - Leg	64
4.4	Inserção de novos experimentos	67
5	CONCLUSÃO	69
	APÊNDICES	71
	APÊNDICE A – CÓDIGO QUE DE DECODIFICAÇÃO DOS AR- QUIVOS <i>LOGS</i> DO PROTÓTIPO DA PERNA DE UM ROBÔ QUADRÚPEDE (HYQ) - FUN- ÇÃO CLMCPLT_CONVERT	73

APÊNDICE B – CÓDIGO QUE SALVA AS VARIÁVEIS LIDAS NO LOG DE UM PROTÓTIPO DA PERNA DE UM ROBÔ QUADRÚPEDE (HYQ) E AS EXPORTA PARA UM ARQUIVO "XLSX" - FUNÇÃO GETTINGVARIABLES	77
APÊNDICE C – CÓDIGO RESULTANTE DAS OPERAÇÕES REALIZADAS NO POWER QUERY PARA O EXPERIMENTO "K3000" - HYQ	79
APÊNDICE D – CÓDIGO DO MATLAB PARA UMA DAS ETAPAS DE TRANSFORMAÇÃO DOS DADOS NA PLATAFORMA FC2D - FUNÇÃO MATLAB_TO_XLS	83
APÊNDICE E – CÓDIGO RESULTANTE DAS OPERAÇÕES REALIZADAS NO POWER QUERY PARA O EXPERIMENTO "SINE5HZ" - HYQ	85
APÊNDICE F – CÓDIGO RESULTANTE DAS OPERAÇÕES REALIZADAS NO POWER QUERY PARA O EXPERIMENTO "EXPSRING_T" - HYQ	89
APÊNDICE G – CODIGO GERADO PELAS TRANSFORMAÇÕES FEITAS NO POWER QUERY PARA OS LOGS DA PLATAFORMA FC2D	93
REFERÊNCIAS	95

1 INTRODUÇÃO

Com o avanço da tecnologia e a consequente onda de digitalização do mundo, uma grande quantidade de dados vem sendo gerada. Esses dados são provenientes de inúmeras fontes, desde sistemas de comunicação e *softwares* até sensores de todos os tipos (BRYANT et al., 2008). Assim, qualquer transação, email, clique, visualização e interação em mídias, *sites* e aplicativos, além de comunicação entre sensores, computadores, aplicações e *logs* de máquinas são geradores de dados (SAGIROGLU; SINANC, 2013).

Então, segundo (AALST, 2014): "[...] dados são coletados sobre qualquer coisa, em qualquer lugar e a qualquer hora [...]". No entanto, apenas a coleta de dados não é suficiente para os *insights* de produto, negócios e melhorias que são gerados a partir deles. Entre outros motivos, é por isso que frequentemente é feita a metáfora que compara dados com petróleo: ambos precisam ser localizados, extraídos, transformados e armazenados; só depois desse processo estarão prontos para uso e poderão gerar o devido valor.

Esse valor gerado é extremamente poderoso e vem transformado principalmente o mundo dos negócios *online*: agora, além de ter conhecimento do que o consumidor compra de fato, é possível entender quais outros produtos ele olha, por onde ele navega, qual *layout* da página tem um efeito mais positivo e como o usuário responde a uma ação realizada (promoções, ações de *marketing*, etc) (MCAFEE et al., 2012).

Nesse contexto, a fim de entender a relação entre esses comportamentos e seus efeitos, surgiu o conceito de *business intelligence* (BI). A definição do escopo dessa área de estudo passa pelo processo de coleta, armazenamento e análise de um grande volume de dados, tendo principalmente como objetivo final a entrega de insumos com alto nível de detalhamento para serem usados como base em tomadas de decisão (NEGASH; GRAY, 2008).

Para que esses *insights* sejam possíveis dentro da área, é importante que os dados a serem analisados tenham grande volume, de forma a ditar comportamentos em larga escala. E é justamente por esse motivo que, apesar do nome, a aplicação não se restringe à área de negócios: campos dos mais diversos que tenham a capacidade de gerar tal volume de dados podem usufruir da implementação de um processo de BI (LOHR, 2012).

É justamente esse propósito que levou à motivação da presente monografia: fazer uma aplicação de *business intelligence* para uma área científica, mais precisamente a da robótica.

1.1 Motivação do projeto

A modelagem de sistemas para controle de robôs tem uma direta relação com dados: é por meio da transmissão dos mesmos que o controlador toma uma decisão do que fazer em seguida. Assim, qualquer experimento ou robô em atividade gera um volume grande de dados que, normalmente, são gravados em arquivos que registram eventos do experimento em questão.

Esses arquivos podem conter informações preciosas sobre o desempenho do robô. No entanto, existe uma dificuldade de obter essas informações e analisar as mesmas por conta de não haver um armazenamento centralizado e/ou padronizado dos dados contidos nesses arquivos. A implementação de um sistema que supre essa necessidade pode trazer análises e *insights* muito mais profundos sobre o desempenho dos experimentos, permitindo inclusive a comparação efetiva entre eles ou até mesmo entre diferentes robôs.

1.2 Objetivos

Esse projeto tem como objetivo trazer uma alternativa de *framework* que permita gerar *insights* profundos sobre o funcionamento e desempenho de robôs a partir dos dados gerados pelos mesmos em seus experimentos. Para isso, será proposta uma solução para possíveis problemas de armazenamento, processamento e visualização dos dados.

Como estudo de caso, a proposta e o projeto terão como base experimentos realizados em duas aplicações de robótica: protótipo da perna de robôs quadrúpedes com atuadores hidráulicos (HyQ) e a plataforma FC2D de testes de controladores aplicados a robôs vestíveis.

2 REVISÃO BIBLIOGRÁFICA

2.1 *Big data*

O termo *big data* passou a ser utilizado com muita frequência, tanto na academia, quanto no mercado de trabalho (MAURO et al., 2015). Por aparecer em contextos diversos, seu significado tornou-se difuso ao longo dos anos. Segundo a abordagem de (MAURO et al., 2015), *big data* deve ser estudado sob a ótica de quatro principais temas: informação, tecnologias, métodos e impacto.

A informação é o combustível do fenômeno *big data* (MAURO et al., 2015). Passando pelos processos de digitalização (onde dados analógicos e contínuos são convertidos para digitais e discretos no tempo) e, posteriormente, dataficação (onde os dados já digitalizados são organizados de forma a transmitirem *insights*), cada vez mais informações são geradas e disponibilizadas (MAURO et al., 2015).

É importante ressaltar que um volume tão grande de informações exige ferramentas apropriadas para tornar possível sua utilização. É nesse aspecto que as tecnologias se tornam essenciais para a manipulação dos dados (MCAFEE et al., 2012), onde memória e capacidade de processamento são os recursos limitantes das ferramentas e, a cada dia, busca-se vencer tais barreiras (MAURO et al., 2015). Com as tecnologias adequadas, o *big data* pode e já está transformando as atividades das empresas, pesquisas científicas, práticas médicas e até mesmo operações de inteligência governamental e defesa dos países (BRYANT et al., 2008).

Por outro lado, um conjunto de dados só passa a ter real valor quando estudado e analisado de maneira adequada. Nesse sentido, os métodos são indispensáveis para o processo de identificação de padrões e comportamentos presentes nos dados (MAURO et al., 2015). (MANYIKA et al., 2011) e (CHEN et al., 2012) propõem uma lista de métodos analíticos para *big data*, e alguns exemplos são: teste A/B (*A/B testing*), classificação (*classification*), análise de agrupamentos (*cluster analysis*), algoritmos genéticos (*genetic algorithms*), aprendizado de máquina (*machine learning*) e redes neurais (*neural networks*). A partir da utilização de tais métodos, que vão muito além de uma simples análise probabilística, é possível respaldar tomadas de decisão críticas com alto nível de confiabilidade (MAURO et al., 2015).

O impacto do fenômeno *big data* está presente tanto na sociedade como nas empresas, afetando diarimamente a organização tradicional desses domínios (MAURO et al., 2015) e traduzindo informação em valor para negócios, ciências, governo e sociedade (BRYANT et al., 2008). É possível que uma mesma técnica de análise seja utilizada em diferentes situações e problemas, o que torna seu impacto cada vez maior. Um exemplo é o método

de análise de correlação, que foi aplicado para registros de pesquisas do Google tornando possível prever epidemias de gripe (GINSBERG et al., 2009), desemprego (ASKITAS; ZIMMERMANN, 2009) e inflação (GUZMAN, 2011).

Uma das grandes contribuições do fenômeno *big data* é a mudança na forma como decisões são tomadas, uma vez que este permite mensurar cada detalhe sobre um determinado assunto e traduzir tal conhecimento em decisões melhores (MCAFEE et al., 2012) - é uma nova abordagem para entender o mundo (LOHR, 2012). Assim, o que antes era muitas vezes guiado por intuição e subjetividade passa a ser respaldado por evidências lógicas que definem comportamentos e identificam padrões com base na causalidade (MCAFEE et al., 2012). Em última instância, o *big data* permite ir ainda mais além, partindo para o reconhecimento de ligações de correlação entre eventos (MAURO et al., 2015).

Dessa forma, o movimento *big data* é muitas vezes referido como uma revolução (LOHR, 2012). Sua diversidade de aplicações promete afetar todas as áreas do conhecimento de tal forma que os dados passaram a ser considerados uma nova classe de ativos econômicos, como moeda ou ouro (LOHR, 2012). Pesquisas do Google, *posts* no Facebook e Twitter permitem medir o comportamento e o sentimento das pessoas em detalhes (LOHR, 2012). Lojas de varejo, a exemplo do Walmart, analisam os dados de vendas, preços e dados econômicos, demográficos e meteorológicos para adaptar as seleções de produtos em lojas específicas e determinar o momento correto para promoções (LOHR, 2012).

Atualmente os avanços na área de eletrônica e aquisição de dados resultam nos mais variados sensores digitais, que também são fontes valiosas de informação e possuem aplicações em equipamentos industriais, automóveis, medidores elétricos e caixas de transporte (LOHR, 2012). Eles podem medir e comunicar a localização, movimento, vibração, temperatura, umidade e até mesmo mudanças químicas no ar (LOHR, 2012). Se tais sensores estiverem de alguma forma relacionados e transmitirem informação para uma inteligência computacional, tem-se o que hoje em dia ficou conhecido como internet das coisas (*internet of things* - IoT) (LOHR, 2012).

De acordo com (AHMED et al., 2017), o número de dispositivos conectados à internet em 2017 já era maior do que a população de humanos na Terra; e a previsão é de que esse número seja atualizado para 50 bilhões em 2020. Entretanto, para que essa enorme quantidade de dados gerados e transmitidos seja útil é indispensável que exista uma plataforma robusta para trabalhar com eles - e é assim que as tecnologias de *big data* oferecem os serviços de armazenamento e processamento necessários para o ambiente da internet das coisas (AHMED et al., 2017).

Vale ressaltar a relação entre o movimento *big data* e seu antecessor *analytics*. Embora ambos trabalhem a premissa de buscar informações inteligentes nos dados e aplicá-las para a resolução de problemas, três grandes características os distinguem: volume,

velocidade e variedade (MCAFEE et al., 2012). Uma grande quantidade de dados, vindos das mais variadas fontes e ferramentas capazes de processar tais informações são recursos chave para que os mais ricos *insights* sejam obtidos (MAURO et al., 2015). Fica claro, porém, que a herança do movimento *analytics* é responsável por trazer valor ao grande volume de dados armazenados e processados pelas ferramentas de *big data* (AHMED et al., 2017).

Assim, entende-se o *big data* como uma das maiores inovações na área de computação dos últimos anos (BRYANT et al., 2008). Decisões respaldadas por dados podem ser aplicadas a todas as áreas da sociedade e são, sem dúvidas, decisões melhores do que as guiadas por intuição (MCAFEE et al., 2012). Embora o processo de geração de valor a partir da informação demande tecnologias e métodos analíticos robustos (MAURO et al., 2015), seu retorno é garantido - uma vez que os dados já são considerados como uma nova classe de ativos econômicos da nossa sociedade (LOHR, 2012).

2.2 *Business intelligence* (BI)

O conceito de *business intelligence* - BI (inteligência de negócios) se popularizou em meados dos anos 90 (CHEN et al., 2012) e pode ser entendido como "[...] as técnicas, tecnologias, sistemas, práticas, metodologias e aplicações que analisam dados críticos de negócios com o objetivo de ajudar uma empresa a melhor entender seu contexto de forma a auxiliar a tomada de decisões oportunas [...]"(CHEN et al., 2012).

Estritamente ligada aos movimentos *analytics* e *big data*, a abordagem de BI apresenta valor na exibição de dados de forma intuitiva e simplificada em relatórios e *dashboards* (também referidos como painéis) (CHEN et al., 2012). Analistas de BI, também conhecidos como *business analysts* - BAs, são responsáveis por apresentar os dados de forma a facilitar a interpretação humana, ajudando os tomadores de decisões na interpretação das informações contidas neles (NEGASH; GRAY, 2008); para isso, o processo de representação dos dados em imagens gráficas (conhecido como processo de visualização) é indispensável (NEGASH; GRAY, 2008).

Dessa forma, explora-se o sistema visual humano para extrair informações, fornecendo uma visão geral de conjuntos de dados complexos (NEGASH; GRAY, 2008). A ideia é identificar estrutura, padrões, tendências, anomalias e relacionamentos entre os dados para auxiliar na definição das áreas de interesse e gerar *insights* (NEGASH; GRAY, 2008). As visualizações usualmente são apresentadas em *dashboards*, contendo grandes quantidades de informações interativas em uma única tela (NEGASH; GRAY, 2008).

Entretanto, na grande maioria dos casos as informações exigem um pré-processamento para que sejam trabalhadas nas ferramentas de visualização (CHEN et al., 2012). Os processos de extração, transformação e carregamento (*extraction, transformation, and*

load - ETL) dos dados têm um importante papel em projetos de BI (JUN et al., 2009), uma vez que garantem a integração, limpeza e estruturação das informações de maneira consolidada (JUN et al., 2009).

Mais especificamente, pode-se dividir o ETL em cinco módulos: extração, validação, limpeza, conversão e carregamento (JUN et al., 2009). Uma vez que os processos forem realizados, os dados estão prontos para serem armazenados em um local definitivo, conhecido como *data warehouse* (DW).

Um DW é definido como uma estrutura consistente de dados que estão orientados em função dos assuntos a que se referem, definindo processos integrados, variados e não voláteis (IMNON, 1996). Eles armazenam as informações que foram extraídas de diversas fontes de dados heterogêneas por meio de ferramentas de ETL, tornando-as disponíveis em forma multidimensional para análises que visam melhorar o conhecimento dos usuários sobre seus negócios (GOLFARELLI; RIZZI, 2018).

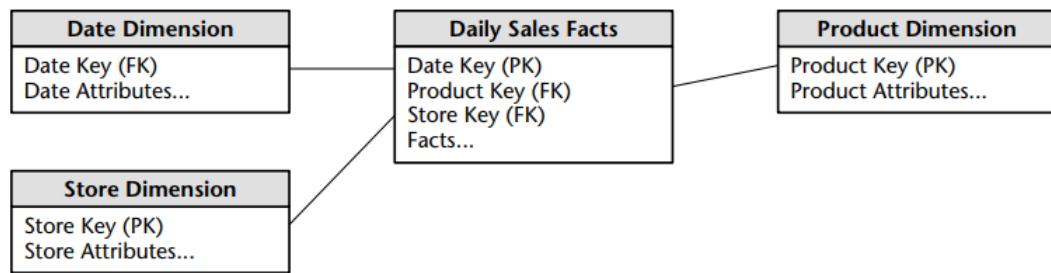
Para garantir que os requisitos do usuário sejam atingidos sem erros, de maneira compreensível e escalável, deve-se manter especial atenção na fase de modelagem conceitual do DW (TRYFONA et al., 1996), pois sua estruturação afetará a forma como consultas futuras à base de dados serão feitas e também definirá se o processo ali representado está compatível com o modelo real (TRYFONA et al., 1996).

Um dos *data models* (modelos de dados) mais utilizados para atender a tais requisitos é o multidimensional (GOLFARELLI; RIZZI, 2018). De acordo com (TRYFONA et al., 1996), para entendê-lo é necessário definir alguns conceitos:

- **Fatos:** são fatos reais do mundo e representam processos que geram dados ao longo do tempo - caracterizam-se por propriedades;
- **Dimensões:** refletem a representação dos fatos ao longo do tempo;
- **Hierarquias:** relação entre conjuntos de dados com dimensões de granularidades (i.e. nível de detalhes) diferentes.

Em implementações relacionais de DWs, o esquema em estrela (*star schema*) e o esquema floco de neve (*snowflake schema*), ambos introduzidos por (KIMBALL; ROSS, 1996), são as soluções mais adotadas para modelagem multidimensional no nível lógico (GOLFARELLI; RIZZI, 2018). Nesses modelos, as tabelas fato estão ligadas às tabelas dimensionais por meio de relacionamentos, como mostra o exemplo da Figura 2.1:

Figura 2.1: Tabela fato e tabelas dimensionais em esquema estrela

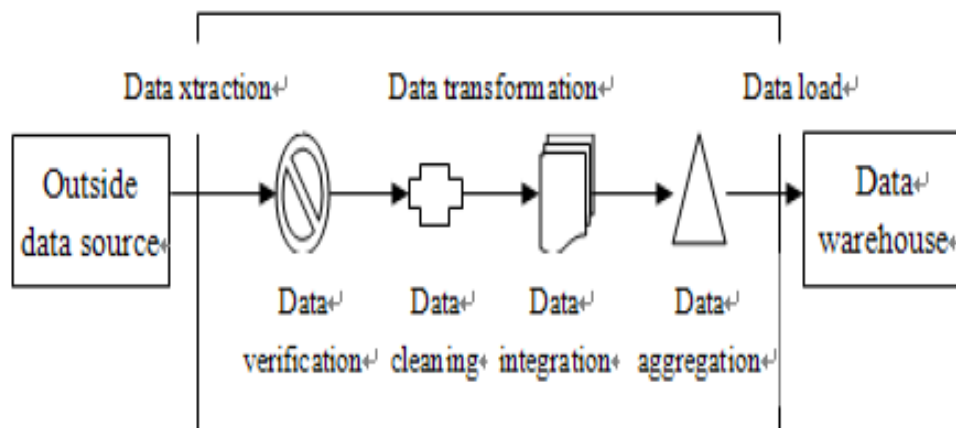


Fonte: (KIMBALL; ROSS, 1996)

O que diferencia os esquemas estrela e floco de neve é o grau de normalização das tabelas dimensionais, ou seja, uma tabela dimensional do esquema estrela passa a ser representada em uma estrutura de árvore com níveis de granularidades diferentes (KIMBALL; ROSS, 1996). A normalização é aplicada para evitar a redundância e aumentar a integridade dos dados, porém sua utilização deve ser ponderada caso a caso, uma vez que bases normalizadas podem acarretar em um menor desempenho de consultas (GOLFARELLI; RIZZI, 2018). Como dito anteriormente, o processo de design de um *data model* demanda bastante atenção, pois cada aplicação terá uma solução que melhor se adapta aos seus requisitos (TRYFONA et al., 1996).

A Figura 2.2 apresenta o fluxo completo dos dados, partindo de uma fonte externa e passando por processos de ETL até chegarem no DW:

Figura 2.2: Fluxo dos dados



Fonte: (JUN et al., 2009)

As plataformas mais modernas de BI são caracterizadas por simplicidade no uso, o que garante a exploração visual dos dados e geração de *insights* sem a necessidade de um

profissional especializado em tecnologia da informação (TI) (HOWSON et al., 2019). Um estudo realizado pelo *Gartner Group* comparou diferentes plataformas modernas de BI (HOWSON et al., 2019) com base nas 15 competências críticas a seguir:

- **Administração, segurança e arquitetura:** Recursos que permitem a segurança da plataforma, administração de usuários e auditoria do acesso e utilização;
- **Nuvem:** Recursos para a construção, implementação e gerenciamento em nuvem com base em dados vindos da nuvem e do servidor local;
- **Conectividade da base de dados:** Recursos que permitem a conexão com dados não estruturados contidos em diferentes tipos de plataformas de armazenamento;
- **Gerenciamento de metadados:** Recursos para análise de informações sobre os próprios dados, bem como métricas e indicadores de performance;
- **Opções de armazenamento e carregamento de dados:** Recursos de plataforma para acessar, integrar, transformar e carregar dados com a capacidade de gerenciar carregamentos e agendar atualizações;
- **Preparação de dados:** Recursos para combinar dados de diferentes fontes e criação de medidas e modelos pelo usuário;
- **Escalabilidade e complexidade de modelos de dados:** Recursos para lidar com um grande volume de dados e manter a performance otimizada para modelos complexos;
- **Análises avançadas para cientistas de dados:** Recursos para análises avançadas disponíveis na plataforma e capacidade de integração e importação de modelos externos;
- **Dashboards analíticos:** Recursos para criação de *dashboards* interativos e conteúdo visual para análises geoespaciais avançadas;
- **Exploração visual interativa:** Recursos para exploração visual dos dados nas mais variadas formas, indo além dos gráficos básicos (pizza, barras e linhas);
- **Descoberta de dados ativa:** Recursos para automaticamente detectar correlações, exceções, agrupamentos e previsões nos dados;
- **Exploração de dispositivos móveis:** Recursos de desenvolvimento de conteúdo para dispositivos móveis;
- **Incorporação de conteúdo analítico:** Recursos para desenvolver *softwares* com APIs (Interfaces de programação de aplicações) e suporte a padrões abertos para criar e modificar conteúdo;

- **Publicação e compartilhamento de conteúdo:** Recursos que permitem aos usuários publicar e compartilhar conteúdo por diferentes meios;
- **Facilidade no uso, apelo visual e integração do fluxo de trabalho:** Se a plataforma é simplesmente administrada e visualmente atraente;

Após uma análise minuciosa dos produtos de cada um dos vendedores a pesquisa os categorizou em relação a abrangência de visão e habilidade de execução, definindo assim os Quatro Quadrantes Mágicos de Gartner ([HOWSON et al., 2019](#)), que podem ser observado na Figura 2.3:

Figura 2.3: Quadrante mágico de Gartner para plataformas de BI



Fonte: ([HOWSON et al., 2019](#))

De acordo com a pesquisa realizada pelo *Gartner Group*, as plataformas Power BI (Microsoft) e Tableau (Amazon) são as mais completas para análises de BI. Tais ferramentas facilitam a geração de *insights* a partir de um grande volume de dados, representando os

mesmos de forma simples e intuitiva em visualizações que facilitam a interpretação humana. Do momento em que são adquiridos, os dados passam por ETLs e são armazenados em data warehouses para então serem requisitados a partir de consultas (*queries*) e utilizados na construção de visualizações e *dashboards*.

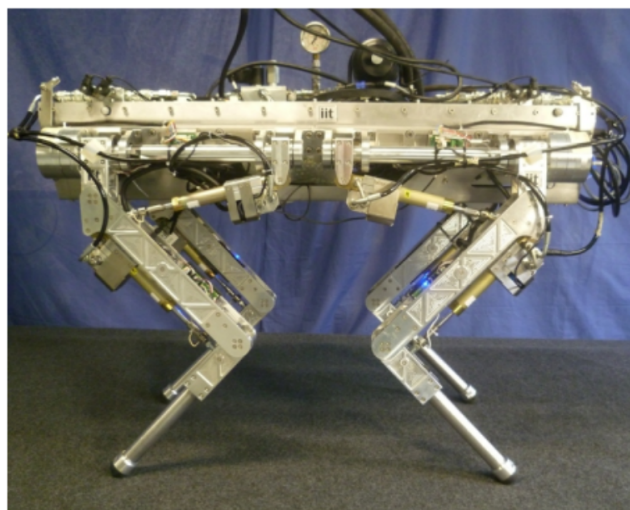
2.3 Aplicações robóticas

2.3.1 Robô quadrúpede (HyQ - *Hydraulic Quadruped*)

Existe uma expectativa de que robôs com pernas tenham a capacidade de superar o desempenho de robôs com rodas em aplicações onde, hoje, eles alcançaram seu limite máximo, principalmente em situações nas quais os robôs são submetidos a andar por terrenos acidentados. Em especial, no caso de estudos com robôs quadrúpedes espera-se que eles possam operar em ambientes desafiadores e de alto risco, como disastres naturais e desabamento de prédios, florestas e montanhas, com a possibilidade de auxiliar humanos em resgates (SEMINI et al., 2016).

Para que isso seja possível, o robô em questão deve ser altamente capaz de resistir a impactos e quedas, ter uma alta performance em largura de banda e de controle, razão elevada entre potência e peso, oferecer sensores de força, controle do torque das juntas, aceleração e orientação do corpo bem como de percepção do terreno ao qual está submetido, entre outras características. O robô quadrúpede HyQ, retratado na Figura 2.4, respeita quase todas dessas necessidades. Ele se caracteriza por possuir atuadores hidráulicos com sistema de controle do torque (SEMINI et al., 2016).

Figura 2.4: Robô quadrúpede com atuadores hidráulicos - HyQ

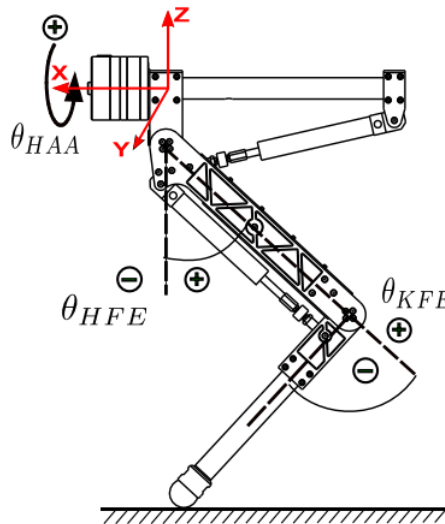


Fonte: (SEMINI et al., 2016)

A [Tabela 2.1](#) lista as principais especificações do robô HyQ enquanto que a [Figura 2.5](#) exemplifica a composição de juntas de uma perna do robô. Ele possui 12 graus de liberdade (DOFs) ativos, sendo que cada perna possui 3 atuadores hidráulicos, listados a seguir ([SEMINI et al., 2016](#)):

- HAA - *Hip Abduction/Adduction*: Responsável pela abdução e adução do quadril;
- HFE - *Hip Flexion Extension*: Responsável pela flexão e extensão do quadril;
- KFE - *Knee Flexion Extension*: Responsável pela flexão e extensão do joelho.

Figura 2.5: Composição de uma perna do robô quadrúpede HyQ



Fonte: ([SEMINI et al., 2016](#))

O sistema de controle desses robôs aceita como entrada medidas de torque para performance de seus movimentos e dinâmica. O sistema de controle possui duas malhas fechadas, mostradas na [Figura 2.6](#). Uma é externa e se refere à posição e a outra é interna e se refere ao torque ([BOAVENTURA et al., 2012](#)).

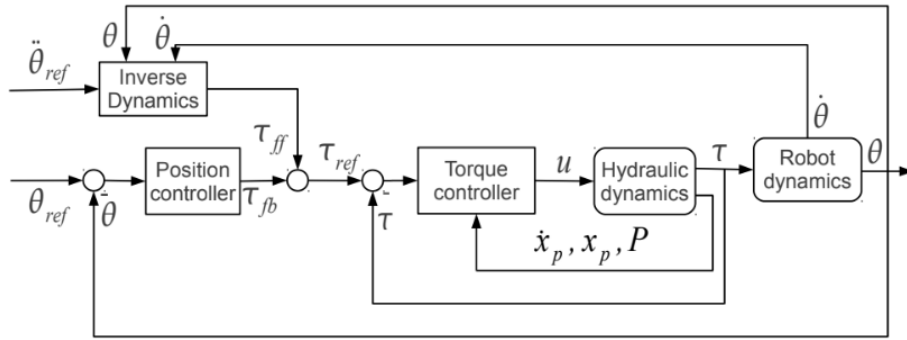
O protótipo da perna de um robô quadrúpede, mostrado na [Figura 2.5](#) também é estudado e usado separadamente em experimentos ([FOCCHI et al., 2010](#)). O protótipo é feito de uma liga entre alumínio e aço inoxidável; possui 3 graus de liberdade, dois deles acionados hidraulicamente e um passivo; como o robô quadrúpede, também tem acionadores hidráulicos com controle baseado em entrada de torque ([FOCCHI et al., 2010](#)).

Tabela 2.1: Principais especificações do robô HyQ

Dimensões	1.0 m x 0.5 m 0.98 m (L x C x A)
Tamanho e peso das conexões	Quadril (HAA-HFE): 0.08 m, 1.9 kg Perna dianteira (HFE-KFE): 0.35 m, 2.6 kg Perna traseira (KFE-pé): 0.35 m, 0.8 kg
Peso	80 kg
DOFs ativos	12
Atuadores HAA	Atuadores hidráulicos rotativos de palhetas duplas
Atuadores HFE/KFE	Cilindros hidráulicos assimétricos com junta articulada
Amplitude de movimento da junta	90° (HAA), 120° (HFE, KFE)
Torque máximo (HAA)	120 N.m (pico de torque a 20 MPa)
Torque máximo (HFE/KFE)	181 N.m (pico de torque a 20MPa)
Sensores de posição	80000CPR
Sensores de torque	Sensor de torque customizado (HAA), <i>loadcell</i> (HFE, KFE)

Fonte: (SEMINI et al., 2016)

Figura 2.6: Diagrama de blocos do sistema de controle do robô HyQ



Fonte: (BOAVENTURA et al., 2012)

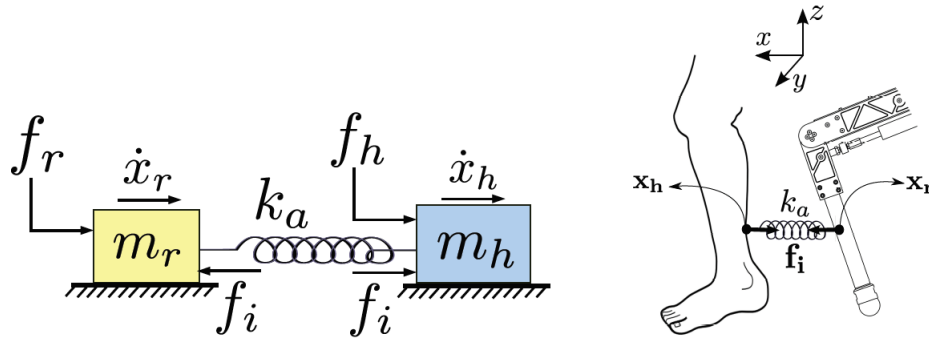
2.3.2 Force Control 2 Degrees of Freedom (FC2D)

Atualmente, uma das grandes contribuições da robótica para a melhoria da qualidade de vida humana está nas aplicações vestíveis - como exoesqueletos e próteses de membros (BOAVENTURA; BUCHLI, 2016). Dentro da grande área da robótica reabilitativa, tais robos são categorizados como assistivos, uma vez que têm como principal função auxiliar os indivíduos no desempenho de suas atividades diárias (RAJASEKARANA et al., 2015). Os dispositivos ficam conectados ao corpo de maneira simbiótica, ou seja, se movendo e trabalhando junto com o humano (BOAVENTURA; BUCHLI, 2016).

Entretanto, tornar um robô vestível imperceptível ao seu usuário é uma tarefa de controle bastante desafiadora (BOAVENTURA; BUCHLI, 2016). Manter o equilíbrio e reagir às forças de interação homem-órtese de acordo com o tipo de movimento (flexão ou extensão) se tornam atividades complexas dada a constante interação entre robô e humano (RAJASEKARANA et al., 2015).

Tal interação robô-humano pode ser representada por um modelo linear simplificado com um grau de liberdade, como mostra a Figura 2.7.

Figura 2.7: Representação linear da interação robô-humano



Fonte: (BOAVENTURA; BUCHLI, 2016)

O modelo considera as ligações entre humanos e robôs como inerciais e de conexão puramente elástica (BOAVENTURA; BUCHLI, 2016), onde a força f_h representa uma força arbitrária aplicada pelo humano ao seu próprio corpo e f_r caracteriza a força exercida pelo robô vestível (BOAVENTURA; BUCHLI, 2016).

Dado um movimento relativo entre robô e humano, a rigidez de fixação, representada por k_a , define a dinâmica da força de interação f_i . De acordo com (BOAVENTURA; BUCHLI, 2016), o sistema acoplado pode ser descrito através das Equações 2.1, 2.2 e 2.3

$$\dot{f}_i = k_a(\dot{x}_r - \dot{x}_h) \quad (2.1)$$

$$f_r - f_i = m_r \ddot{x}_r \quad (2.2)$$

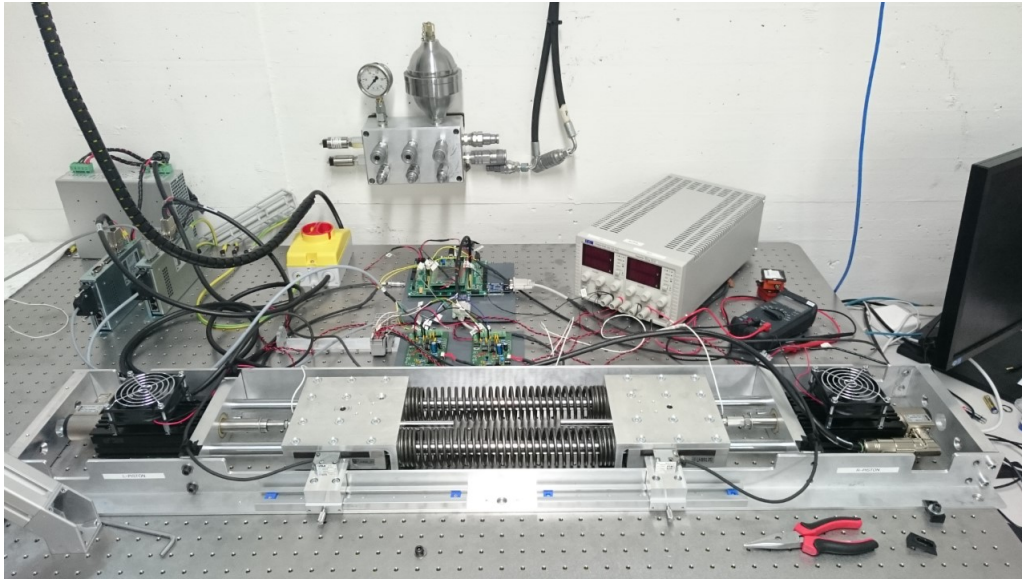
$$f_h + f_i = m_h \ddot{x}_h \quad (2.3)$$

Como a entrada de controle é f_i e a saída desejada é f_r , combinando as equações pode-se reescrever a dinâmica da força de interação como mostrado na Equação 2.4.

$$m_r \ddot{f}_i + k_a f_i = k_a (f_r - m_r \ddot{x}_h) \quad (2.4)$$

A plataforma *Force Control 2 Degrees of Freedom* - FC2D (Controle de Força com 2 Graus de Liberdade) da Figura 2.8 foi desenvolvida para testar controladores de robôs vestíveis, seguindo o modelo linear da Figura 2.7.

Figura 2.8: Plataforma de teste FC2D



Fonte: (BOAVENTURA; BUCHLI, 2016)

Seu objetivo é simular a interação robô-humano a partir de dois atuadores, um pistão hidráulico e um motor elétrico. Cada um dos atuadores movimenta um suporte diferente sobre um mesmo trilho. Os suportes permitem: a simulação de um ambiente sem ação da gravidade, uma vez que possuem apoios laterais; a variação de suas massas pela adição de pesos em suas superfícies e, em especial, a simulação da interação entre os dois suportes a partir da conexão dos mesmos por uma mola.

Dessa forma, o FC2D se caracteriza como uma bancada de teste para controladores que serão aplicados posteriormente em robôs vestíveis.

3 MATERIAIS E MÉTODOS

Esta seção descreve quais materiais foram utilizados durante a execução do projeto, bem como a metodologia adotada para se alcançar o objetivo proposto. O projeto tem como ponto de partida arquivos que contém o histórico de execução (chamados também de *logs* de dados) dos controladores de robôs durante um experimento. Esses *logs* foram manipulados por diferentes *softwares* de forma a gerar dados devidamente construídos que são usados por um *dashboard* que contém diversas visualizações sobre os mesmos. Além disso, um sistema de armazenamento de dados, baseado na definição de *data warehouse* (DW), foi proposto de forma a padronizar a maneira como as informações descritivas dos experimentos é salva.

Por isso, o projeto pode ser dividido em três principais etapas: design do *data warehouse* usado para armazenamento de informações que descrevem o experimento, processos de ETL (*extract, transform and load*) dos dados provenientes de *logs* dos experimentos e, por último, criação de um *dashboard* a partir de consultas aos dados armazenados no *data warehouse*.

3.1 Design do *data warehouse*

Uma questão importante no armazenamento dos arquivos *logs* é a estruturação de um *data warehouse* centralizado que armazene não só o conteúdo dos experimentos (proveniente dos *logs*) mas também informações que possam classificá-los e defini-los. Dessa forma, além de unificar as informações, facilitando buscas e entendimento dos diversos experimentos e robôs, é possível usar esses dados para gerar visualizações sobre todo o conteúdo do DW em questão, além de permitir comparações entre experimentos.

Após estudo e entendimento dos dados que seriam trabalhados durante o projeto, chegou-se à conclusão de que o melhor modelo para o relacionamento entre os dados do DW a ser construído é o esquema em estrela (*star schema*). Essa modelagem possui uma tabela central, chamada tabela fato, que se conecta com tabelas chamadas dimensionais - como mostra a Figura 2.1.

A tabela fato armazena os eventos ocorridos e indicadores (ou métricas) desse evento. Além disso, ela também contém chaves que possibilitam a conexão com as tabelas dimensionais. Essas, por sua vez, possuem características específicas de cada evento que está escrito na tabela fato.

Em um *data warehouse*, uma chave primária é um índice único, que não se repete e que pode identificar um conjunto de valores específicos de uma tabela, ou seja, identifica unicamente aquela linha. Já a chave estrangeira é um tipo de identificação que possibilita

o relacionamento entre duas tabelas: cria-se uma correspondência entre colunas. Em um caso hipotético, para construir a relação entre duas tabelas nomeadas A e B, sendo que cada uma possui sua própria chave primária, é necessário inserir uma nova coluna na tabela A caracterizada como chave estrangeira. Essa coluna deve sempre trazer o valor da chave primária da tabela B ao qual aquele registro específico da tabela A corresponde.

Existem outros tipos de chave além da primária e da estrangeira; uma delas é a chave alternativa, que também identifica unicamente um registro mas que não foi um elemento usado como chave primária.

Com base nisso e utilizando-se do esquema em estrela, tem-se que todas as tabelas do banco possuem suas próprias chaves primárias; no entanto, a tabela fato se relaciona com todas as demais tabelas (e as demais apenas se relacionam com a tabela fato). Portanto, a tabela fato possui chaves estrangeiras que possibilitam essa conexão. O número de chaves estrangeiras que a tabela fato contém corresponde ao número de tabelas dimensionais existentes no DW.

Existem diversas ferramentas que podem ser utilizadas para o armazenamento de um *data warehouse* desse tipo. Em sua maioria, são ferramentas mais robustas que exigem um grande sistema de programação que construa sua estrutura e a mantenha de pé, disponível para uso. Além disso, nenhuma delas possui uma interface para administração do sistema que possibilite uma simplificação de uso e edição do DW.

Por esses motivos, optou-se por manter as tabelas fato e dimensionais no Microsoft Excel. A estrutura citada foi construída como planilhas de um arquivo e foi proposto um procedimento padrão que deverá ser executado pelo usuário ao inserir novos arquivos *logs* no *data warehouse*. A ferramenta escolhida para armazenamento do DW foi o Microsoft Power BI.

3.2 ETL - *Extract, Transform and Load*

O projeto proposto surgiu da necessidade de utilizar de uma maneira mais efetiva os *logs* que são gerados pelo controlador de um robô durante um dado experimento, de forma a possibilitar *insights* sobre o desempenho do mesmo. Assim, mesmo que a definição e execução dos experimentos não sejam parte do escopo do projeto, os arquivos *logs* gerados por eles são a base para o início.

Foi definido que o estudo de caso desse projeto seriam *logs* de dados provenientes de experimentos realizados com uma das pernas de um robô quadrúpede de atuadores hidráulicos (HyQ) e na plataforma FC2D (*Force Control 2 Degrees of Freedom*) para testes de controladores de robôs vestíveis; esses *logs* foram gerados por experimentos realizados no Instituto Italiano de Tecnologia (IIT) e também no Instituto Federal Suíço de Tecnologia de Zurique (ETH Zurich).

Para este projeto, foram utilizados *logs* de dois robôs diferentes (um de cada modelo citado) e 18 experimentos no total. A [Tabela 3.1](#) contém a descrição dos diferentes tipos de experimentos que geraram os *logs* usados ao longo do projeto.

Tabela 3.1: Descrição dos experimentos que geraram os *logs* usados no projeto

Robô	Nome do experimento	Controladores usados
FC2D	impedanceKvc065Kp2000	Controlador de impedância linear e controlador de força não linear
FC2D	sin20mmKvc065Cold	Controlador de impedância linear e controlador de força não linear
FC2D	sin20mmKvc065Warm	Controlador de impedância linear e controlador de força não linear
FC2D	sin50NKvc045LwprV2PEst	Controlador de impedância linear, controlador de força não linear e LWPR
FC2D	sin50nKvc045LwprV2PMeas	Controlador de impedância linear, controlador de força não linear e LWPR
FC2D	sin50NKvc045PEst	Controlador de impedância linear e controlador de força não linear
FC2D	sin50NKvc045PMeas	Controlador de impedância linear e controlador de força não linear
FC2D	sin50NKvc065Cold	Controlador de impedância linear e controlador de força não linear
FC2D	sin50NKvc065Warm	Controlador de impedância linear e controlador de força não linear
FC2D	impedanceKvc070	Controlador de impedância linear e controlador de força não linear
FC2D	impedanceKvc070LwprVB2	Controlador de impedância linear e controlador de força não linear
FC2D	sin20mmKvc070	Controlador de impedância linear e controlador de força não linear
FC2D	sin20mmKvc070LwprVB2	Controlador de impedância linear e controlador de força não linear
FC2D	sin50NKvc070	Controlador de impedância linear e controlador de força não linear
FC2D	sin50NKvc070LwprVB2	Controlador de impedância linear, controlador de força não linear e LWPR
Perna HyQ	expSpring_t	Controlador de impedância não linear no espaço de tarefas e controlador de força não linear
Perna HyQ	sine5Hz	Controlador de impedância linear, controlador de força não linear e controlador feedforward de dinâmica inversa
Perna HyQ	K3000_t	Controlador de impedância linear no espaço de tarefas e controlador de força não linear

Assim sendo, o processo de extração dos dados (E, em ETL) foi realizado na

execução do experimento. A próxima etapa se refere à transformação desses dados, que pode englobar sua verificação, limpeza, integração e agregação.

Para o projeto, foram executados tanto a limpeza quanto a agregação dos dados. Além disso, um processo adicional de conversão de arquivos foi necessário: os *logs* recebidos possuem extensões diferentes de acordo com o robô que executou o experimento; nenhuma delas eram extensões em que os dados seriam facilmente lidos por outras ferramentas.

O *software* utilizado para essa primeira transformação dos dados foi o MatLab. O código desenvolvido lê as variáveis contidas no arquivo em questão e as salva em um arquivo externo com extensão "xlsx". Dessa forma, esse novo arquivo pode ser facilmente lido por diversas ferramentas, como por exemplo o Microsoft Excel.

MatLab é um *software* e uma linguagem de programação, ou seja, possui uma sintaxe de programação própria, de alta performance e voltada principalmente para desenvolvimento de algoritmos que envolvam cálculos numéricos e tenham uma natureza técnica matemática, como manipulação de matrizes e vetores.

O arquivo "xlsx" gerado tem como conteúdo as variáveis obtidas a partir do *log* de dados. O número de colunas corresponde ao número de variáveis que foram lidas e o número de linhas aos diferentes valores das mesmas em um determinado tempo, sendo que a primeira linha contém o nome das variáveis.

A partir desse momento, os dados foram estudados para que houvesse um entendimento mais profundo de suas relações. Observou-se que a chave primária de todos os arquivos é a variável do tempo, que registra o tempo decorrido desde o início do experimento. Cada registro na planilha, ou tabela, armazena o valor das variáveis em um dado tempo, sendo que a segmentação do seu valor entre os registros é dada pela taxa de amostragem em que o experimento foi realizado.

Como os experimentos tem suas próprias características, eles podem possuir variáveis diferentes entre si; experimentos do mesmo tipo e modelo contém sempre as mesmas variáveis. A [Tabela 3.2](#) mostra as principais variáveis observadas nos experimentos da perna do robô quadrúpede enquanto que a [Tabela 3.3](#) contém as variáveis da plataforma FC2D.

Tabela 3.2: Descrição das principais variáveis da perna do robô HyQ

Sigla da Variável	Significado
LF_HFE_hst_xp	Posição do pistão do atuador hidráulico de flexão e extensão do quadril da perna dianteira
LF_HFE_hst_xpd	Velocidade do pistão do atuador hidráulico de flexão e extensão do quadril da perna dianteira
LF_HFE_hst_xpdd	Aceleração do pistão do atuador hidráulico de flexão e extensão do quadril da perna dianteira
LF_HFE_hst_force	Força do pistão do atuador hidráulico de flexão e extensão do quadril da perna dianteira
LF_HFE_hst_uvc	Comando de compensamento da velocidade do pistão do atuador hidráulico de flexão e extensão do quadril da perna dianteira
LF_HFE_hst_pa	Pressão na câmara A do pistão do atuador hidráulico de flexão e extensão do quadril da perna dianteira
LF_HFE_hst_pb	Pressão na câmara B do pistão do atuador hidráulico de flexão e extensão do quadril da perna dianteira
LF_HFE_hst_pa	Pressão de suprimento do pistão do atuador hidráulico de flexão e extensão do quadril da perna dianteira
LF_KFE_hst_ps	Pressão de suprimento do pistão do atuador hidráulico de flexão e extensão do joelho da perna dianteira
LF_KFE_hst_xp	Posição do pistão do atuador hidráulico de flexão e extensão do joelho da perna dianteira
LF_KFE_hst_xpp	Velocidade do pistão do atuador hidráulico de flexão e extensão do joelho da perna dianteira
LF_KFE_hst_xpp	Aceleração do pistão do atuador hidráulico de flexão e extensão do joelho da perna dianteira
LF_KFE_hst_fref	Força de referência do pistão do atuador hidráulico de flexão e extensão do joelho da perna dianteira
LF_KFE_hst_force	Força do pistão do atuador hidráulico de flexão e extensão do joelho da perna dianteira
LF_KFE_hst_uvc	Comando de compensamento da velocidade do pistão do atuador hidráulico de flexão e extensão do joelho da perna dianteira
LF_HFE_th	Posição angular da junta de abdução e adução do quadril da perna dianteira
LF_HFE_des_th	Posição angular desejada da junta de abdução e adução do quadril da perna dianteira
LF_HFE_thd	Velocidade angular da junta de abdução e adução do quadril da perna dianteira
LF_HFE_des_thd	Velocidade angular desejada da junta de abdução e adução do quadril da perna dianteira
LF_HFE_thdd	Aceleração angular da junta de abdução e adução do quadril da perna dianteira

Tabela 3.3: Descrição das variáveis da plataforma FC2D

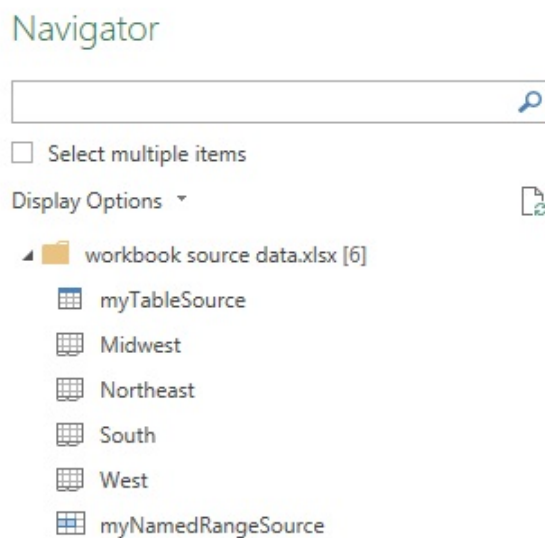
Sigla da Variável	Significado
x	Posição em x
dx	Velocidade em x
d2x	Aceleração em x
F	Força
Pa	Pressão na câmara A do cilindro
Pb	Pressão na câmara B do cilindro
Ps	Pressão de suprimento
Pt	Pressão do tanque
u	Saída do controlador
xRef	Valor de referência para a posição em x
FRef	Valor de referência para força
rx	Posição em x sem filtragem

Ainda dentro do processo de transformação, com os arquivos "xlsx" de cada experimento prontos, outros procedimentos foram aplicados com o objetivo de limpar e agregar os dados. Novamente, o processo para cada tipo de robô foi diferente, respeitando suas configurações e particularidades.

Para isso, foi utilizado o Power Query, uma extensão de *softwares* da Microsoft que se conecta com as planilhas a serem trabalhadas e permite diferentes manipulações das mesmas.

A conexão com uma fonte de dados é bem simples e um exemplo de menu onde pode-se escolher com qual tabela se quer trabalhar é mostrado na [Figura 3.1](#).

Figura 3.1: Exemplo de menu de seleção do Power Query para estabelecer conexão da ferramenta com uma base de dados



Algumas das funções que o Power Query tem disponível são: substituir valores (Figura 3.2), remover colunas e linhas (Figura 3.3), aplicar filtros de acordo com o nome de uma coluna (Figura 3.4), mudar o tipo de dado que uma coluna armazena (Figura 3.5) e apendar dados de diferentes tabelas (duas ou mais) em um único arquivo (Figura 3.6). Por trás de cada uma dessas operações, o Power Query na verdade realiza consultas (*queries*) aos dados que estão sendo trabalhados e opera a manipulação solicitada.

Figura 3.2: Substituir valores no Power Query

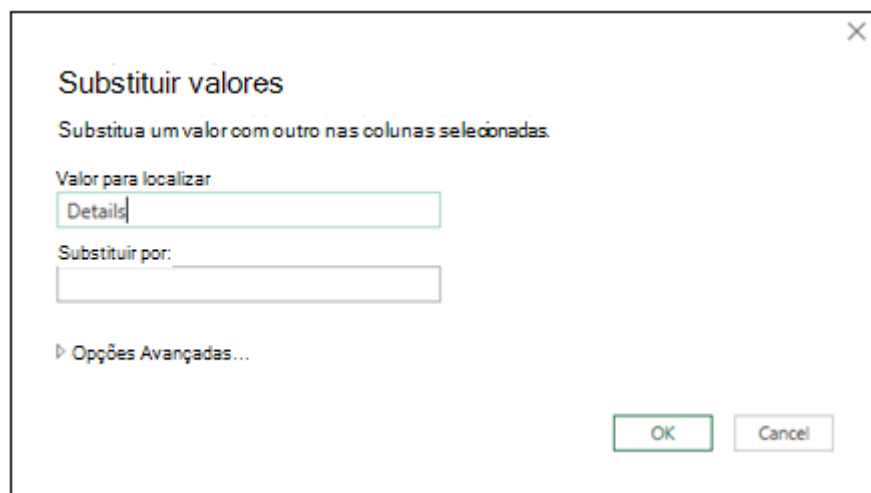


Figura 3.3: Remover colunas e/ou linhas no Power Query

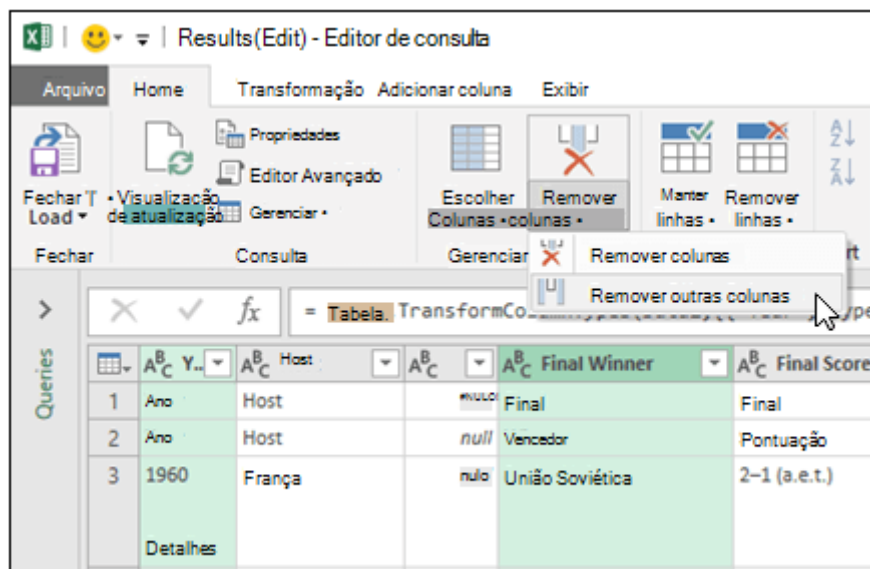


Figura 3.4: Aplicar filtros de acordo com o nome de uma coluna no Power Query

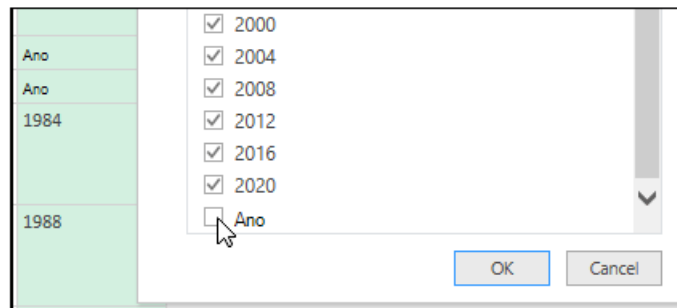


Figura 3.5: Mudar o tipo de dado armazenado em uma coluna (inteiro, decimal, texto, etc) no Power Query

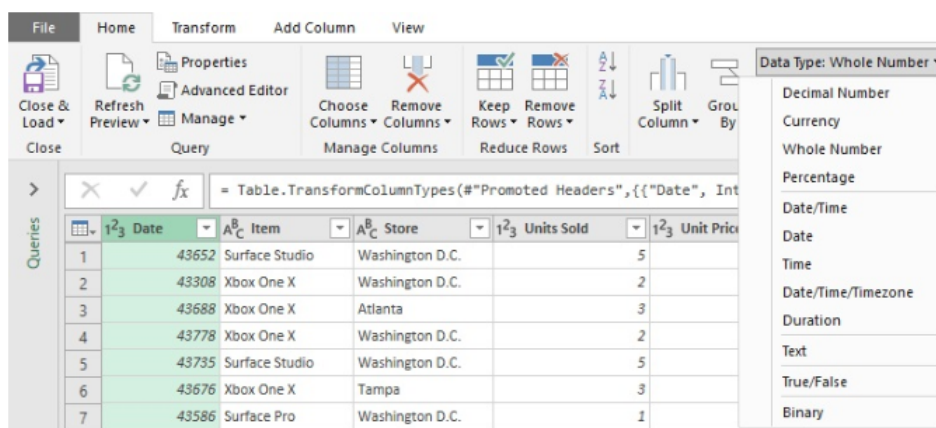
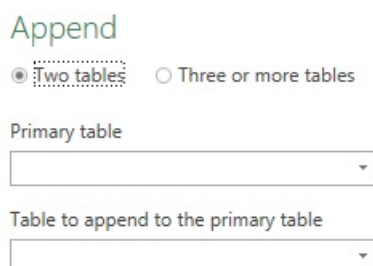


Figura 3.6: Apendar dados de diferentes tabelas em uma única no Power Query



A última etapa do ETL consiste em salvar os dados, agora transformados. Isso é possível pelo próprio Power Query, que carrega os dados como uma nova planilha no arquivo original. Uma vez feito isso, o arquivo deve ser carregado no *data warehouse* que foi criado no *software* Power BI.

3.3 Criação do *dashboard*

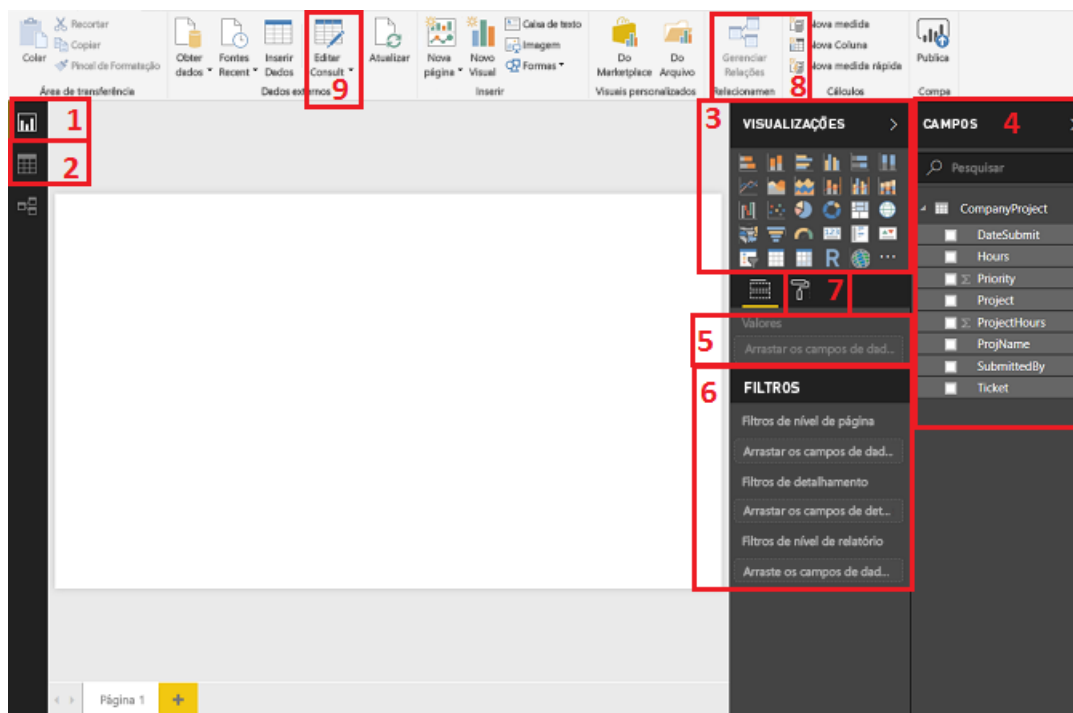
Com toda a estrutura de dados bem definida e entendida, a última etapa é a criação do *dashboard* com as visualizações de interesse. Enxergou-se a necessidade de desenvolver três tipos páginas, cada uma com visualizações e objetivos diferentes:

- Dados gerais: Essa página contém informações gerais sobre o *data warehouse*, com métricas sobre a variedade de experimentos registrados e seus respectivos controladores e robôs;
- Dados comparativos: Essa tipo de página contém visualizações que comparam dados de mesmas variáveis entre experimentos do mesmo tipo de robô. Dessa forma, pode-se relacionar o desempenho de experimentos diferentes para uma mesma variável.
- Dados específicos de experimentos: A ideia é exibir visualizações de variáveis de um único experimento ao longo do tempo.

Entre as ferramentas disponíveis para criação do *dashboard*, ponderou-se principalmente entre os *softwares* Tableau e Power BI. Apesar de o Tableau apresentar vantagens de performance para esse estudo de caso, optou-se pelo uso do Power BI pela possibilidade de estruturação do DW no próprio *software* e de criação de visualizações fixas que podem se atualizar automaticamente com a inserção de novos dados.

A [Figura 3.7](#) mostra uma visão global da página inicial do Power BI e destaca as principais funcionalidades, explicadas a seguir.

Figura 3.7: Página inicial do Power BI



1. Visualização do *Dashboard*: este botão dá acesso ao *dashboard* construído até então, bem como às demais funcionalidades disponíveis;
2. Visualização dos dados: este botão direciona o usuário para uma tela onde ele pode visualizar os dados de todas as planilhas às quais o Power BI está conectado naquele momento;
3. Visualizações: esse quadro mostra os tipos de gráficos que podem ser construídos (colunar, tabela, gráfico de pizza, gráfico de linha, etc). Ao clicar em um deles, ele cria uma visualização vazia na área de trabalho do Power BI;
4. Campos: este quadro mostra as variáveis disponíveis para serem plotadas de acordo com as tabelas que estão conectadas ao *software*;
5. Campos para visualização: essa área exibe os possíveis campos para a visualização que está sendo trabalhada, como eixos e detalhes da visualização;
6. Filtros: Permite que campos de variáveis sejam adicionados para aplicação de filtros de acordo com os valores da variável;
7. Formato: Este botão abre opções para formatar o *design* da visualização que está sendo trabalhada, como tamanho do texto, cores, legenda, etc;
8. Gerenciar relações: abre o *data warehouse* que está sendo trabalhado, possibilitando gerenciar as conexões entre tabelas;

9. Editar consulta: abre o Power Query dentro do Power BI, possibilitando fazer edições no ETL.

4 DESENVOLVIMENTO E RESULTADOS EXPERIMENTAIS

Com base na metodologia exposta e utilizando os materiais citados, esta seção apresenta o desenvolvimento do projeto, bem como os resultados obtidos em cada etapa. Por último, todas as informações são condensadas para formalizar o procedimento de inclusão de um novo experimento ao DW e, conseqüentemente, ao *dashboard*.

4.1 Design do *data warehouse*

Foi observado que existem três elementos principais que, dada suas características, são responsáveis por identificar e classificar um experimento: o robô que o realizou, as configurações do controlador utilizado no robô e dados inerentes ao experimento. A [Tabela 4.1](#) mostra quais dados foram levados em conta para cada um desses elementos.

Tabela 4.1: Descrição das características dos elementos que descrevem um experimento

Elemento	Características observadas
Robô	Nome e descrição
Controlador	Nome e descrição
Experimento	Nome do arquivo <i>log</i> , data, tempo total e taxa de amostragem do experimento

Com base nestas características e seguindo a metodologia do esquema em estrela, foram criadas três tabelas: "Robot" (Figura 4.1), "Controller" (Figura 4.2) e "Experiments" (Figura 4.3). Cada uma delas possui uma chave primária, identificada pela coluna "Id"; essa chave é um número inteiro que é incrementado toda vez que um novo registro for adicionado.

Figura 4.1: Tabela "Robot"

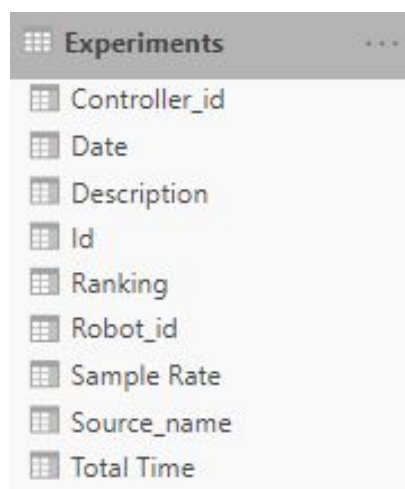
Robot
Description
Id
Name

Figura 4.2: Tabela "Controller"



Controller	
Description	
Id	
Name	

Figura 4.3: Tabela "Experiments"



Experiments	
Controller_id	
Date	
Description	
Id	
Ranking	
Robot_id	
Sample Rate	
Source_name	
Total Time	

A tabela fato desse esquema é a "Experiments". Então, além da chave primária, ela possui as chaves estrangeiras que conectam essa tabela com as tabelas dimensionais. Essas chaves são referenciadas pelas colunas "Robot_id" e "Controller_id", respectivamente.

A chave estrangeira da tabela fato tem correspondência com a chave primária na tabela dimensional a qual ela está se referindo; logo, na tabela "Experiments", a coluna "Robot_id" é equivalente ao "Id" da tabela "Robot" e a coluna "Controller_id" ao "Id" da tabela "Controller".

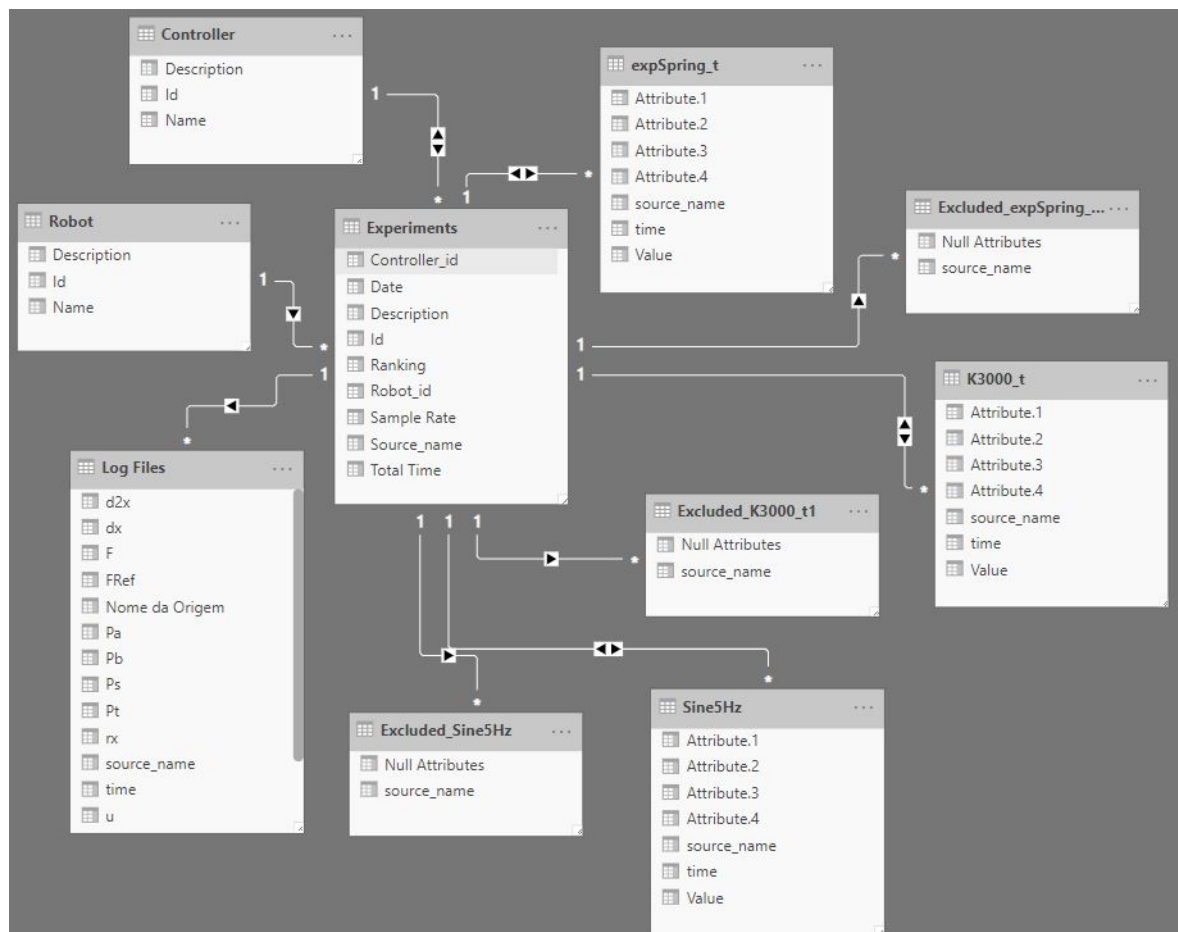
Como a tabela "Experiments" é a tabela fato, além das chaves das outras tabelas ela também possui indicadores do experimento realizado. Esses indicadores são data, descrição, nome do arquivo *log* correspondente, a taxa de amostragem e o tempo total do experimento. As três tabelas foram criadas em um único arquivo do Excel, chamado "Experiment Records", e separadas em abas. Fez-se isso com a intenção de concentrar as informações de identificação dos experimentos.

Apesar da tabela fato se relacionar com as tabelas dimensionais através de chaves estrangeiras, é interessante para o projeto que seja estabelecido uma relação entre a tabela "Experiments" e o *log* correspondente daquele experimento. Por isso, a coluna "source_name" foi adicionada tanto na tabela fato quanto nos arquivos *logs*.

Na tabela "Experiments" essa coluna é uma chave alternativa, pois ela não é definida como a chave primária mas é um valor único para cada registro. Na tabela de *logs* ela é um atributo que tem o mesmo valor para todas as linhas. Dessa forma, pode-se estabelecer esse tipo de relação respeitando-se a ideia de que a tabela fato se conecta com todas as tabelas do esquema e as tabelas dimensionais se relacionam apenas com a tabela fato.

A Figura 4.4 mostra como o DW ficou estruturado, incluindo a tabela fato, as tabelas dimensionais e as tabelas dos *logs*, bem como a relação entre elas. As tabelas que se referem aos experimentos do robô HyQ - Leg são conectadas separadamente, uma vez que apresentam variáveis distintas entre si. Já os experimentos do plataforma FC2D foram agregados em uma só tabela, pois possuem um padrão das variáveis amostradas em diferentes experimentos, o que permite tratá-las com o mesmo processo de ETL.

Figura 4.4: *Data warehouse* do projeto



4.2 Desenvolvimento do ETL

Como mencionado na Seção 3, a extração dos dados é operada no momento em que o experimento é realizado. Portanto, o desenvolvimento desse projeto cobre as etapas de transformação e carregamento dos dados.

A etapa de transformação foi diferente para os *logs* de cada tipo de robô. Ambas são explicadas nas subseções que se seguem. A etapa de carregamento dos dados (L, em ETL) foi a mesma para os dois tipos: após transformados os dados foram salvos e armazenados no *data warehouse* construído.

4.2.1 Transformação dos dados da perna do robô quadrúpede HyQ

Os *logs* de dados gerados durante um experimento da perna do robô quadrúpede (HyQ) são arquivos encriptografados (geralmente em formato binário ou ASCII - Código Padrão Americano para Intercâmbio de Informações) e possuem uma extensão pouco usual.

Logo, o primeiro passo para transformá-los em arquivos do tipo "xlsx" consiste em fazer uma decodificação do arquivo, com uma subsequente leitura dos dados. Esse processo já era performado por usuários que lidavam com esses *logs* através de duas funções no MatLab que, por resultado, salva cada variável lida no ambiente de trabalho do MatLab como uma matriz de mesmo nome da variável, que possui uma coluna e número de linhas igual ao número de linhas registradas no arquivo. Além disso, uma variável do tipo *struct* é criada; ela armazena o nome de todas as variáveis lidas bem como a medida de cada uma delas.

A partir daí novas linhas de código foram adicionadas ao arquivo original; a operação das mesmas consiste em salvar os valores de cada variável como uma matriz de tamanho (X,Y), onde X é o número de variáveis lidas e Y o número de linhas contidas em cada uma. Além disso, foi realizada uma leitura da variável *struct* criada anteriormente no ambiente de trabalho do MatLab, para que os respectivos nomes de cada variável fossem passados também para a nova matriz. Uma linha foi adicionada na mesma e os nomes lidos foram escritos nela.

Em seguida, uma nova coluna foi adicionada a matriz; ela tem como nome "source_name" e foi criada com o objetivo de levar para dentro da matriz o nome do arquivo que originou aqueles dados. Todas as linhas dessa coluna, exceto a primeira, contém o nome do arquivo que foi inicialmente carregado.

Por último, essa matriz foi escrita em uma planilha (*Worksheet*) de um novo arquivo do tipo "xlsx" salvo na mesma pasta em que o código está contido e com o mesmo nome do *log* original. Os arquivos contendo os códigos referentes a essa etapa são mostrados no [Apêndice A](#) e no [Apêndice B](#).

A etapa de transformação feita no Power Query para esses *logs* consiste principalmente em limpá-los e rearranjá-los de forma a garantir uma boa visualização e navegação para o usuário no *dashboard* final.

Em sua maioria, os *logs* gerados por experimentos nesse tipo de robô possuem

um número muito grande de colunas (na ordem da centena). Estudando os dados, foi observado que haviam algumas linhas e colunas que estavam completamente vazias, ou seja, não possuíam nenhum valor diferente de zero.

Dessa forma, a primeira operação realizada foi a de remover as linhas que não carregavam nenhum valor. Em seguida, o mesmo aconteceu para as colunas; no entanto, essa foi uma operação mais difícil porque o Power Query só disponibiliza a função de remover uma coluna especificada. O objetivo era fazer a operação desejada de uma forma automática.

Portanto, foi feita uma transposição dos dados de forma que as linhas se transformaram em colunas e vice-versa. Nessa nova configuração, linhas que continham apenas valores nulos foram removidas e a tabela foi novamente transposta para a sua configuração inicial. Nesse procedimento, algumas manobras precisaram ser realizadas para manter-se a integridade e característica dos dados e da tabela como um todo.

Os nomes das colunas que foram removidas foram salvos em uma segunda tabela, para que essa informação não fosse perdida e possíveis erros de extração do *log* pudessem ser futuramente analisados no *dashboard*.

Por fim, observou-se que as variáveis desses *logs* seguem um padrão de nome que as dividem por um *underscore*, sendo que cada abreviatura traz alguma referência para aquela variável, como mostra a [Tabela 3.2](#).

Por esse motivo, a tabela foi despivotada em relação à variável do tempo, ou seja, todas as colunas foram agrupadas em uma única, tendo como valor o nome da variável à qual aquela linha corresponde. Então, esse processo aumenta o número de linhas da tabela final já que, para cada valor no tempo, será adicionado um número de linhas correspondentes ao número de colunas que havia antes. Um exemplo genérico do resultado final é mostrado na [Figura 4.5](#).

Figura 4.5: Exemplo de uma tabela despivotada

	COLUNA	VALOR
1	INVESTIMENTOS	9613906084.01
2	DESPESAS	8102644.84
3	BANCO	BANCO ALVORADA S/A
4	ANO	2010
5	INVESTIMENTOS	174343.35
6	DESPESAS	7935411.15
7	BANCO	BANCO ALVORADA S/A
8	ANO	2011
9	INVESTIMENTOS	8202652.29
10	DESPESAS	114215.13
11	BANCO	BANCO ARBI S/A

Feito isso, essa nova coluna criada é separada em quatro, cada uma correspondendo

ao conteúdo entre os *underscores*. Cada uma dessas quatro colunas foi definida como um atributo da variável.

Por conta de os *logs* desse tipo de robô nem sempre terem as mesmas variáveis, foi necessário que um processo fosse criado para cada um deles. No entanto, o embasamento das operações realizadas é sempre o mesmo.

Como mencionado na Seção 3, o Power Query executa a cada operação consultas nas tabelas que estão sendo tratadas por meio de linhas de comando. O código gerado pelos processos em cada um dos *logs* da perna do robô quadrúpede estão contidos nos [Apêndice C](#), [Apêndice E](#) e [Apêndice F](#).

4.2.2 Transformação dos dados da plataforma FC2D

Os *logs* provenientes de experimentos realizados na plataforma FC2D, por outro lado, foram salvos como um arquivo de extensão "mat", que se caracteriza por ser também uma extensão compatível com o MatLab.

Portanto, no código desenvolvido, foi feita uma chamada para trazer o que está armazenado nesse tipo de arquivo para o ambiente de trabalho do MatLab. Para esse caso, o arquivo carregou uma variável do tipo *struct*, contendo dentro dela diversas matrizes de tamanho (X,1), sendo X o número de linhas registrado em cada um dos experimentos. Essas matrizes correspondem às variáveis que de fato foram extraídas durante o experimento.

Feito isso, o mesmo procedimento foi aplicado: os dados foram armazenados em uma matriz de múltiplas colunas (correspondente ao número de variáveis lidas), uma nova coluna contendo o nome original do arquivo foi adicionada à essa matriz e, por último, ela foi salva em uma planilha de um novo arquivo "xlsx".

A necessidade de desenvolver um novo código apenas se deu por conta da maneira como foi feita a leitura do arquivo *log* inicial e também porque o tipo de resultado que se deu por essa leitura foi diferente para cada um. O arquivo contendo o código referente a essa etapa é mostrado no [Apêndice D](#).

Com os novos arquivos gerados, deu-se início à construção do processamento dos mesmos pelo *software* Power Query. Observou-se que para esse robô todos os *logs* possuíam as mesmas variáveis, tanto em nome, quanto em número. Por esse motivo, o processo consistiu em apendar todos os elementos de cada uma das planilhas em uma nova e única tabela. Dessa forma, a tabela final possui os dados de todos os *logs* em questão, sendo que o que os diferencia é a coluna "source_name", que contém o nome do arquivo original do *log*. Isso foi feito selecionando-se a pasta que continha todos os arquivos desse tipo como a fonte de dados.

Uma grande vantagem desse procedimento é que qualquer novo experimento incluso na pasta será automaticamente apendado à essa tabela final e consequentemente incluído

no *dashboard* construído, desde que a tabela referente a esse experimento tenha as mesmas características (mesmas variáveis). Caso contrário, o ETL não será compilado e nenhuma visualização no *dashboard* será processada.

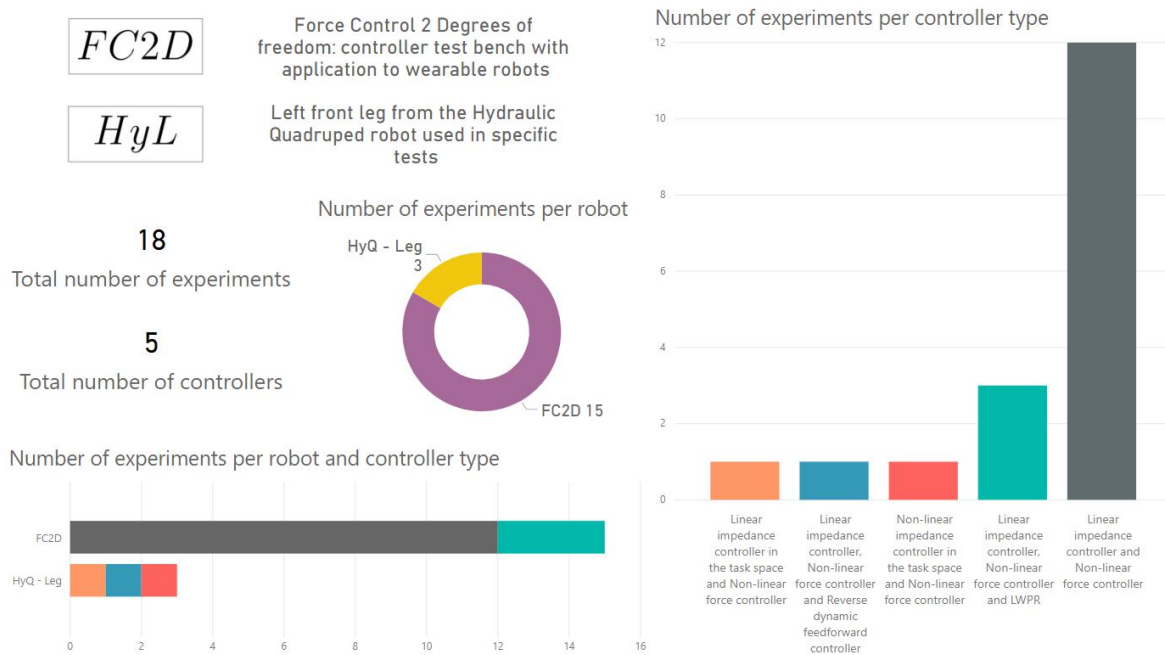
O código completo realizado para este caso está mostrado no [Apêndice G](#).

4.3 Criação do *dashboard*

Conforme dito na Seção 3, o *software* Power BI foi escolhido para desenvolver o produto final do projeto. Um só *dashboard* concentra as informações provenientes dos arquivos de *log* dos robôs HyQ - Leg e FC2D. Seu principal conceito é oferecer visualizações interativas a respeito dos robôs, controladores e experimentos. Para isso, o usuário pode navegar por diferentes páginas, de acordo com as informações de seu interesse.

A primeira página, chamada "Home", traz informações sobre a quantidade de robôs, controladores e experimentos que estão disponíveis na aplicação. Além disso, é nela que o usuário começa a navegar: existem dois botões (um para cada robô) que direcionam para as respectivas páginas de análise. É importante ressaltar que todas as visualizações presentes na página são interativas, permitindo ao usuário destacar dados e ver relações entre os gráficos. A Figura 4.6 apresenta uma visão geral da "Home" do *dashboard*.

Figura 4.6: Página "Home" do *dashboard*



Ao clicar no botão de seleção do robô, o usuário é redirecionado para as análises de sua escolha.

4.3.1 FC2D

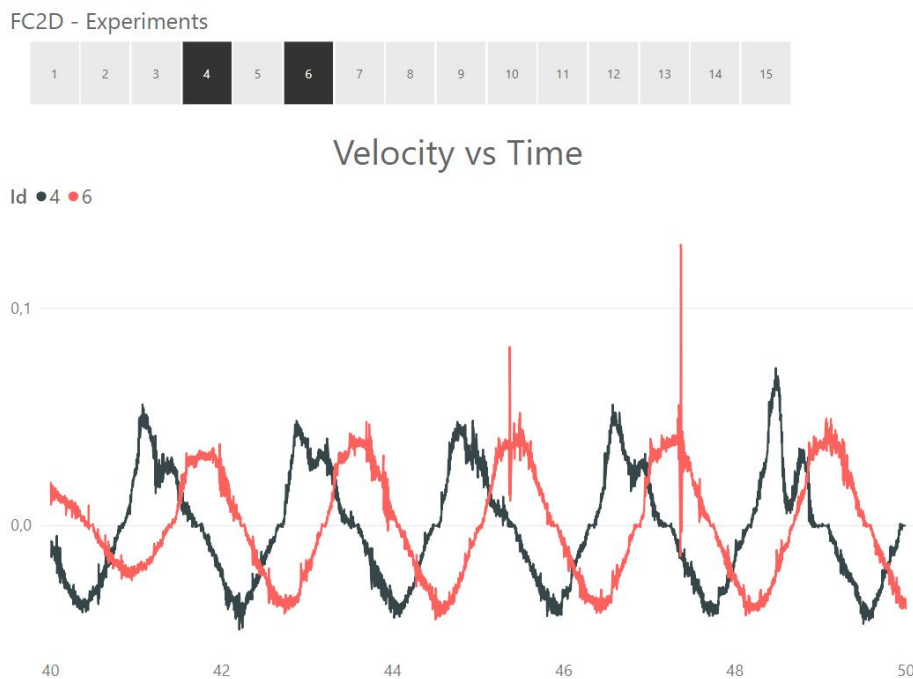
Começando pelo FC2D, a primeira página "FC2D - Kinematics" (Figura 4.7) apresenta os gráficos das grandezas cinemáticas do experimento, a saber: posição, velocidade e aceleração. Além disso, pode-se visualizar os gráficos de *raw position* x_r .

Figura 4.7: Página "FC2D - Kinematics" do *dashboard*



O mesmo padrão de menus é adotado para todas as páginas referentes a este robô. Na parte superior pode-se visualizar todos os experimentos realizados para o FC2D, e o usuário seleciona um ou mais deles para sua análise, como mostra a [Figura 4.8](#).

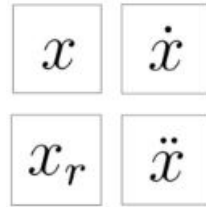
Figura 4.8: Menu de seleção de experimentos da página "FC2D - Kinematics" e análise de dois experimentos simultaneamente



Ao lado do menu de seleção do experimento, quatro botões permitem que o usuário

escolha qual variável cinemática será analisada.

Figura 4.9: Botões de escolha das variáveis da página "FC2D - Kinematics"



No canto superior direito da página há um *slider* para escolha do tempo de amostragem do(s) experimento(s). Essa é uma importante funcionalidade pois permite ver em mais detalhes determinados gráficos ou então estudar de forma ampla os experimentos no decorrer do tempo. Os valores iniciais do tempo podem ser digitados ou então o usuário pode arrastar a barra inferior para determinar qual parte do experimento deseja ver:

Figura 4.10: *Slider* de tempo da página "FC2D - Kinematics"



Por fim, os últimos elementos da página são cartões com as descrições do experimento e do controlador, respectivamente:

Figura 4.11: Cartões de descrição do experimento e controlador da página "FC2D - Kinematics"

Kvc = 065
Kp = 2000
f = 5 kHz
 Experiment Description

**Linear impedance
 controller in joint space
 and Power controller
 based on linearizing
 output feedback**
 Controller Description

Para continuar a navegar pelo *dashboard* o usuário pode selecionar algum dos botões do menu fixo inferior, onde as setas o direcionam para a página anterior/posterior e o símbolo do meio o faz retornar para a página "Home".

Figura 4.12: Menu padrão para navegação no *dashboard*



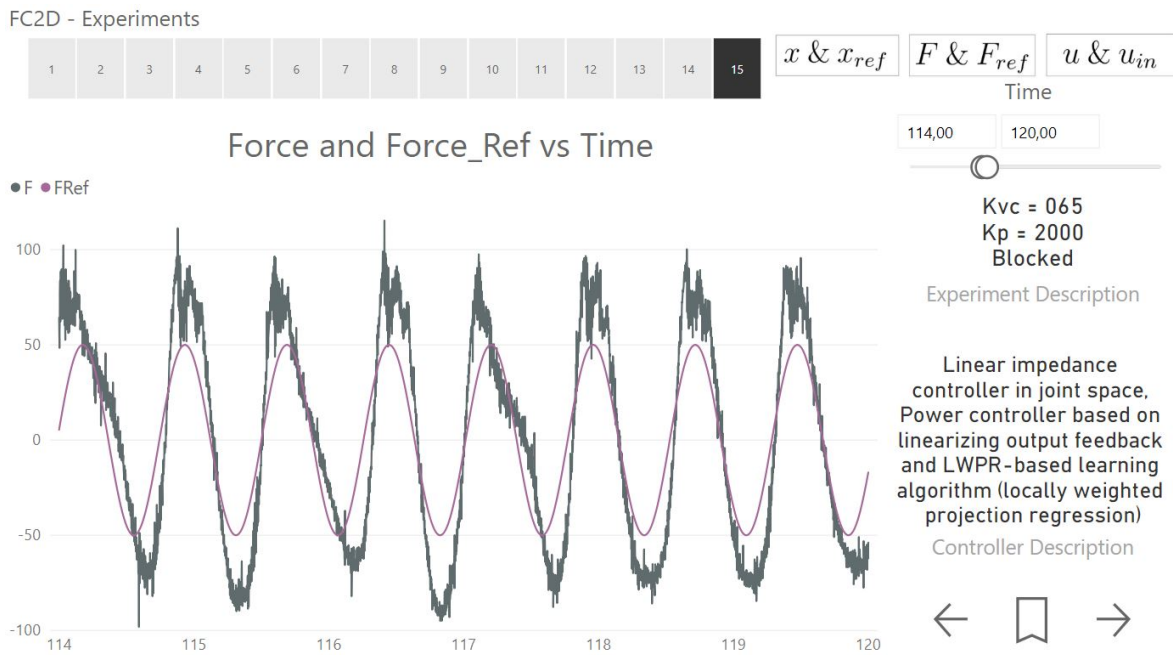
A próxima página, chamada "FC2D - Pressure", traz informações sobre as diferentes pressões de tanque da plataforma. De maneira bastante similar à página anterior, o usuário pode escolher os experimentos de seu interesse, bem como a pressão a ser plotada e o tempo de amostragem. A Figura 4.13 apresenta uma visão geral da página.

Figura 4.13: Página "FC2D - Pressures" do *dashboard*



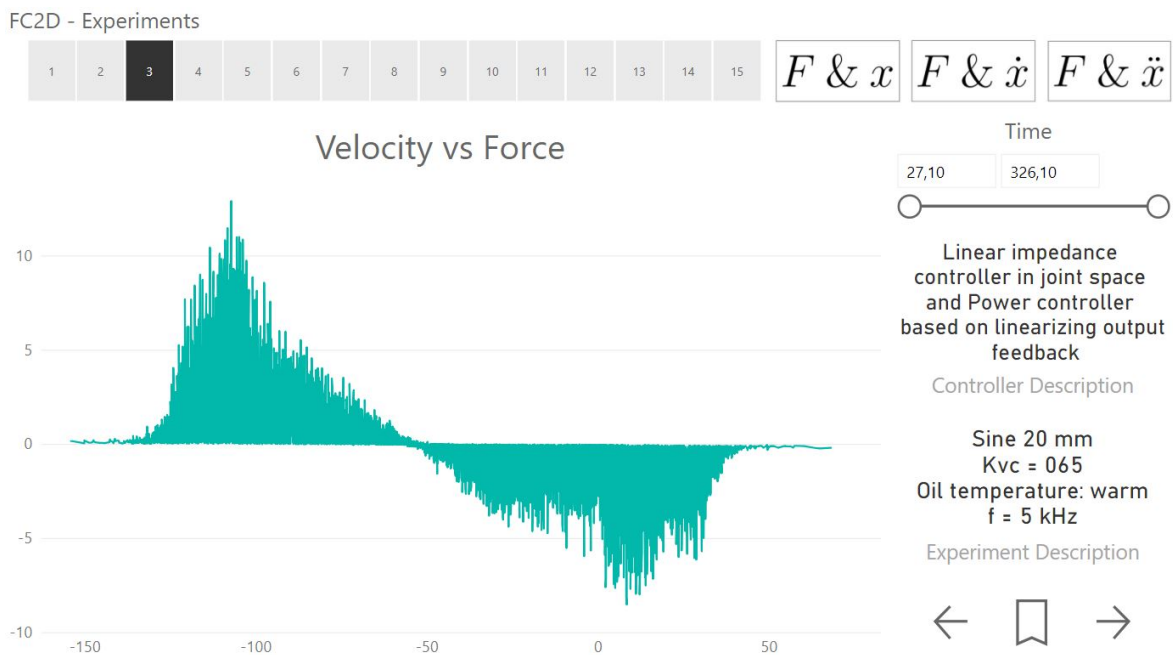
Ao selecionar a seta da direita do menu de navegação do *dashboard* o usuário é redirecionado para a próxima página: "FC2D - References". Nela é possível visualizar as variáveis x , F e u em relação a seus respectivos sinais de referência: x_{ref} , F_{ref} e u_{in} . Vale ressaltar que nesta página não é possível selecionar mais de um experimento para análise, uma vez que o gráfico já possui duas variáveis sendo plotadas.

Figura 4.14: Página "FC2D - References" do *dashboard*



A Figura 4.15 apresenta a última página do *dashboard* referente ao FC2D, onde é possível plotar gráficos das variáveis cinemáticas em função da força.

Figura 4.15: Página "FC2D - Kinematics x F" do *dashboard*

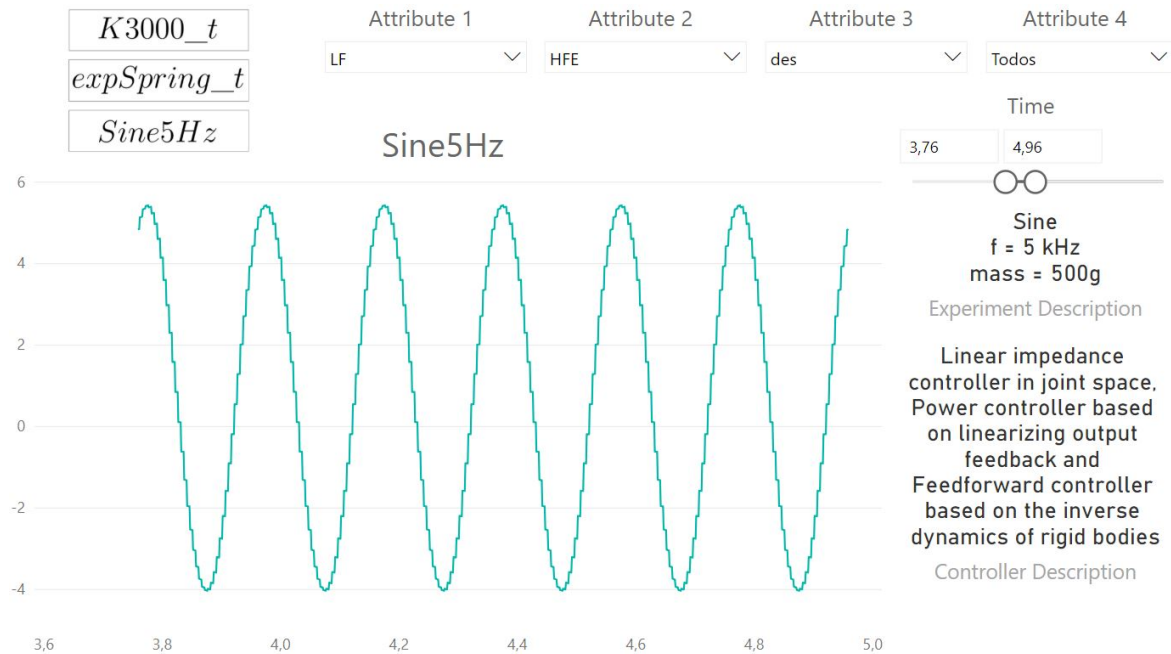


4.3.2 HyQ - Leg

Se o usuário selecionar o botão de seta para a direita do menu de navegação na última página do FC2D, ou então selecionar o botão HyL na página "Home", será

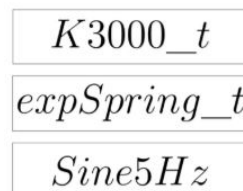
direcionado para "HYQ Leg - Variables x Time" (Figura 4.16).

Figura 4.16: Página "HYQ Leg - Variables x Time" do *dashboard*



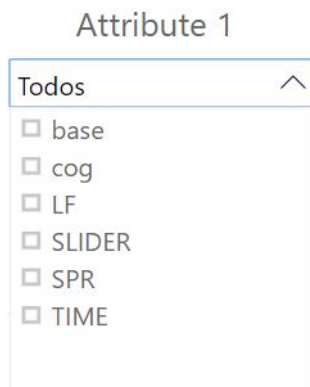
Nela inicialmente seleciona-se o experimento desejado nos botões do canto superior esquerdo, como mostra a Figura 4.17.

Figura 4.17: Botões de seleção do experimento da página "HYQ Leg - Variables x Time"



Os detalhes do experimento e do controlador utilizado, bem como o *slider* de tempo, seguem o mesmo padrão das páginas do FC2D. A principal diferença consiste na forma como as variáveis são selecionadas: aqui, cada um dos 4 atributos que identificam uma variável dos experimentos do HyQ - Leg é selecionado a partir de listas suspensas nos menus superiores. A Figura 4.18 apresenta em mais detalhes as opções de seleção para o primeiro atributo do experimento "K3000_t".

Figura 4.18: Lista de primeiros atributos das variáveis do experimento "K3000"



Por fim, a última página do *dashboard* traz algumas informações adicionais sobre os experimentos do HyQ - Leg: data de realização, tempo total, taxa de amostragem e variáveis nulas (não mostradas na página anterior), como mostra a [Figura 4.19](#).

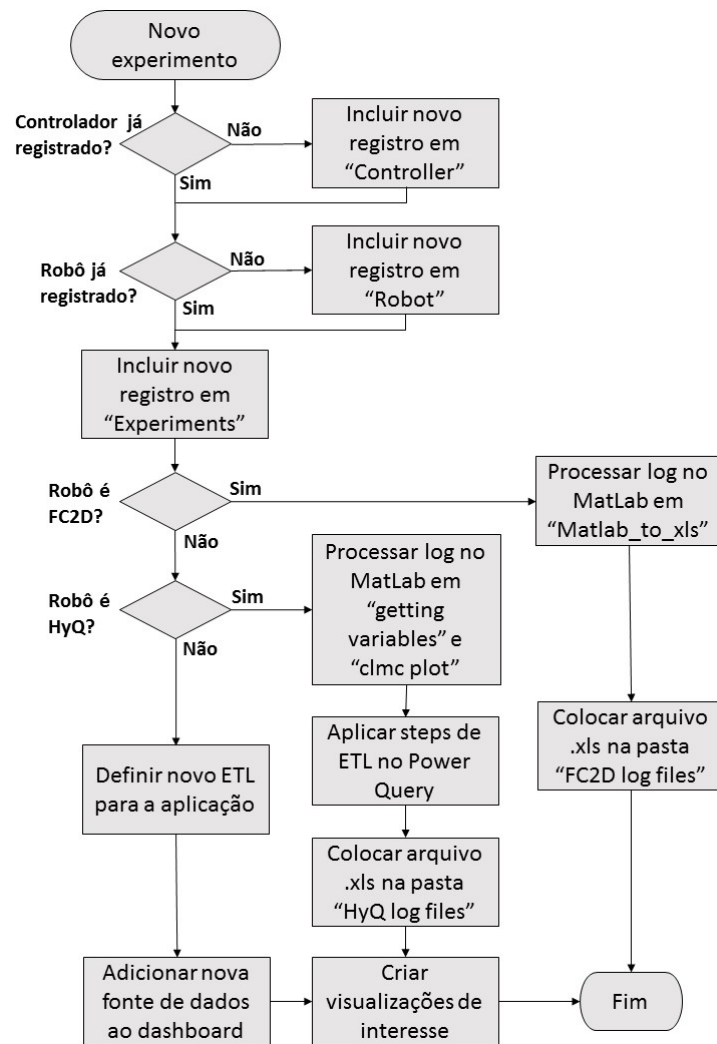
Figura 4.19: Página "HYQ Leg - Additional Information" do *dashboard*

<i>K3000_t</i>	<i>expSpring_t</i>	<i>Sine5Hz</i>	Null Attributes
Experiment Date	Total Time (s)	Sample Rate (Hz)	base_ad
quarta-feira, 26 de abril de 2017	10,00	500,10	base_add
Experiment Description	Controller Description		base_bd
Experiment performed in spring	Impedance controller in task space with exponential stiffness and Power controller based on linearizing output feedback		base_bdd
			base_gd
			base_gdd
			base_q1
			base_q2
			base_q3
			base_qd0
			base_qd1
			base_qd2
			base_qd3
			base_x
			base_xd
			base_xdd

4.4 Inserção de novos experimentos

A Figura 4.20 a seguir mostra o processo de inserção de novos experimentos ao *dashboard*.

Figura 4.20: Fluxograma para a inserção de novos experimentos no *dashboard*



5 CONCLUSÃO

Com base no estudo das características específicas dos dados que foram trabalhados e de suas aplicações, foi feita uma abordagem pensando-se na melhor maneira para processá-los e armazená-los de forma que todas as informações necessárias para a identificação de um experimento pudessem ser salvas e facilmente localizadas. Essa estrutura compõe o *Data Warehouse* do projeto, enquanto que a lógica de processamento dos dados compõe o ETL dos mesmos.

Um dos grandes desafios para essa etapa foi justamente a falta de padronização dos dados extraídos do experimento (*logs*): eles possuíam extensões diferentes, o que causou a necessidade de uma abordagem inicial específica para cada um deles.

Além disso, especificamente para os *logs* provenientes do protótipo da perna do robô quadrúpede HyQ, houve uma dificuldade a mais: cada *log* tinha um número e tipos de variáveis diferentes, o que dificultou a determinação de um ETL unificado para os *logs* desses robôs. O grande número de variáveis também apresentou uma dificuldade em termos de performance dos dados no Power Query.

O produto final do projeto é o painel (ou *dashboard*) de visualizações. Foi construído um *dashboard* que possibilita um alto nível de interação com o usuário: ele possui um menu de navegação entre as páginas, permitindo que, dentro delas, o usuário possa escolher de forma bastante simples e rápida as variáveis que se deseja observar. Além disso, por meio de um outro menu de seleção é possível plotar dentro de uma mesma visualização as variáveis de dois ou mais experimentos, dando a opção de fazer um comparativo entre o desempenho dos mesmos.

Essa solução final mostrou ter uma navegação muito amigável ao usuário; somando-se a isso os benefícios mencionados anteriormente, o *dashboard* construído propôs uma solução de visualização melhor que a usada atualmente, que consiste em criar as visualizações das variáveis no MatLab por linhas de código ou comando.

Como sugestões de trabalhos futuros, cabe algumas melhorias no processamento dos dados. Um exemplo seria criar uma aplicação robusta que pudesse fazer as manipulações dos dados e inserção das características do mesmo nas tabelas do *Data Warehouse* de forma automática. Para isso, seria interessante que houvesse também um servidor *web* que fizesse a administração da aplicação, trazendo mais uma facilidade para o usuário final, agora no que tange a gestão dos dados contidos no *Data Warehouse*.

Apêndices

APÊNDICE A – CÓDIGO QUE DE CODIFICAÇÃO DOS ARQUIVOS LOGS DO PROTÓTIPO DA PERNA DE UM ROBÔ QUADRÚPEDE (HYQ) - FUNÇÃO CLMCLOT_CONVERT

```

1      function [D,vars,freq] = clmcplot_convert(fname)
2      % [D,vars,freq] = clmcplot_convert(fname)
3      %
4      % This function converts an CLMCLOT binary file into a Matlab matrix
5      % and
6      % a struct array of variable names and variable units. If fname is
7      % given, the file is processed immediately. If no filename is given,
8      % a dialog box will ask to located the file.
9      %
10     % fname (i): input file name (optional)
11     % D      (o): data matrix
12     % vars   (o): struct array containing variable names and units
13     % freq   (o): sampling frequency
14     %
15     % Stefan Schaal, March 2006
16
17     % read in the file name
18     if ~exist('fname') | isempty(fname),
19         [fname, pathname] = uigetfile('*','Select Data File');
20         if (fname == 0),
21             return;
22         end;
23     % concatenate pathname and filename and open file
24     fname_store = fname;
25     fname=strcat(pathname, fname);
26     end;
27
28     fid=fopen(fname, 'r', 'ieee-be');
29     if fid == -1,
30         return;
31     end;
32
33     % check for BDILOT file format
34     string = fscanf(fid, '%s', 1);
35     frewind(fid);

```

```

37 if strcmp(string , '$BEGIN_HEADER'), % this is a BDILOT file
38
39     vars = [];
40
41     while (1),
42
43         string=fscanf(fid , '%s' ,1);
44
45         if (strcmp(string , '$DT' ) ,
46             dt=fscanf(fid , '%f' ,1);
47             freq = 1/dt;
48         end
49
50         if (strcmp(string , '$NVAR' ) ,
51             cols=fscanf(fid , '%d' ,1);
52         end
53
54         if (strcmp(string , '$VAR' ) ,
55             vars(end+1).name = fscanf(fid , '%s' ,1);
56             vars(end).unit   = '-';
57         end
58
59         if (strcmp(string , '$OUT' ) ,
60             rows=fscanf(fid , '%d' ,1);
61         end
62
63         if (strcmp(string , '$END_HEADER' ) ,
64             break;
65         end
66     end
67
68     % read the data
69     fscanf(fid , '%c' ,1);
70
71 else % this is a CLMCLOT file
72
73     specs=fscanf(fid , '%d %d %d %f' ,4); % [dummy, cols ,rows ,freq]
74     cols = specs(2);
75     rows = specs(3);
76     freq = specs(4);
77     vars = [];

```

```
79 % read all variable names
    for i=1:cols ,
81     vars(i).name=fscanf(fid , '%s' ,1);
        vars(i).unit=fscanf(fid , '%s' ,1);
83     end;
        fscanf(fid , '%c' ,3); % there are three characters which must be
            skipped
85
    end
87
% read the data
89 D = fread(fid , [cols ,rows] , 'float32 ');
    D=D';
91 fclose(fid);
```


APÊNDICE B – CÓDIGO QUE SALVA AS VARIÁVEIS LIDAS NO LOG DE UM PROTÓTIPO DA PERNA DE UM ROBÔ QUADRÚPEDE (HYQ) E AS EXPORTA PARA UM ARQUIVO "XLSX" - FUNÇÃO GETTINGVARIABLES

```

1 clear all

3 % load data file

5 filepath = '/Users/Lorena C. Borges/Documents/Graduação/TCC2/log/
    Log files/K3000_t';
    split_path = strsplit(filepath, '/');
7 filename_extension = char(split_path(length(split_path)));
    split_file = strsplit(filename_extension, '.');
9 filename = split_file(1);

11 [D,vars,freq] = clmcplot_convert(filepath);

13 % create the named variables by looping through the data and using
    eval
    % to assign the data to the labels
15 for i=1:length(vars);
        str = [vars(i).name, 'D(:,', num2str(i), ');'];
17        eval(str);
    end

19

21 % code developed for project
    for i=1:length(vars);
23        var_name = vars(i).name;
        teste = num2cell(evalin('base', vars(i).name));
25        parcial_result(:,i) = [var_name; teste];
    end

27
    source_name = cell(length(parcial_result),1);

29
    for i=1:length(parcial_result);
31        if i == 1
            source_name{i,1} = 'source_name';
33        else
            source_name{i,1} = char(filename);

```

```
35     end
end
37
result = [source_name parcial_result];
39
workbook = char(strcat(filename, '.xlsx'));
41
xlswrite(strcat('data/', workbook), result);
```


APÊNDICE C – CÓDIGO RESULTANTE DAS OPERAÇÕES REALIZADAS NO POWER QUERY PARA O EXPERIMENTO "K3000" - HYQ

```

1 let
    Source = Excel.CurrentWorkbook() {[Name="Table1 "]}[Content],
3     #"Changed Type" = Table.TransformColumnTypes(Source,{{"Column1",
    type text}, {"Column2", type any}, {"Column3", type any}, {"
    Column4", type any}, {"Column5", type any}, {"Column6", type any},
    {"Column7", type any}, {"Column8", type any}, {"Column9", type
    any}, {"Column10", type any}, {"Column11", type any}, {"Column12",
    type any}, {"Column13", type any}, {"Column14", type any}, {"
    Column15", type any}, {"Column16", type any}, {"Column17", type
    any}, {"Column18", type any}, {"Column19", type any}, {"Column20",
    type any}, ..., {"Column215", type any}}),
    #"Removed Columns" = Table.RemoveColumns(#"Changed Type",{"
    Column1"}),
5     #"Replaced Value" = Table.ReplaceValue(#"Removed Columns",0,null,
    Replacer.ReplaceValue,{"Column2", "Column3", "Column4", "Column5",
    "Column6", "Column7", "Column8", "Column9", "Column10", "Column11
    ", "Column12", "Column13", "Column14", "Column15", "Column16", "
    Column17", "Column18", "Column19", "Column20", "Column21", "
    Column22", ..., "Column215"}),
    #"Removed Blank Rows" = Table.SelectRows(#"Replaced Value", each
    not List.IsEmpty(List.RemoveMatchingItems(Record.FieldValues(_),
    {"", null}))),
7     #"Transposed Table" = Table.Transpose(#"Removed Blank Rows"),
    #"Removed Columns1" = Table.RemoveColumns(#"Transposed Table",{"
    Column1"}),
9     #"Removed Blank Rows1" = Table.SelectRows(#"Removed Columns1",
    each not List.IsEmpty(List.RemoveMatchingItems(Record.FieldValues(
    _), {"", null}))),
    #"Merged Queries" = Table.NestedJoin(#"Removed Blank Rows1",{"
    Column2", "Column3", "Column4", "Column5", "Column6", "Column7", "
    Column8", "Column9", "Column10", "Column11", "Column12", "Column13
    ", "Column14", "Column15", "Column16", "Column17", "Column18", "
    Column19", "Column20", "Column21", "Column22", "Column23", "
    Column24", "Column25", ..., "Column2501"},
11    #"K3000_t (2)",{"Column2", "Column3", "Column4", "Column5", "
    Column6", "Column7", "Column8", "Column9", "Column10", "Column11",
    "Column12", "Column13", "Column14", "Column15", "Column16", "

```

```

Column17", "Column18", "Column19", "Column20", "Column21", "
Column22", "Column23", "Column24", ..., "Column2501"}, "K3000_t (2)
", JoinKind.LeftOuter),
#"Expanded K3000_t (2)" = Table.ExpandTableColumn("#Merged
Queries", "K3000_t (2)", {"Column1"}, {"Column1"}),
13  #"Removed Duplicates" = Table.Distinct("#Expanded K3000_t (2)",
{"Column1"}),
#"Reordered Columns" = Table.ReorderColumns("#Removed Duplicates
", {"Column1", "Column2", "Column3", "Column4", "Column5", "Column6
", "Column7", "Column8", "Column9", "Column10", "Column11", "
Column12", "Column13", "Column14", "Column15", "Column16", "
Column17", "Column18", "Column19", "Column20", "Column21", "
Column22", "Column23", ..., "Column2501"}),
15  #"Transposed Table1" = Table.Transpose("#Reordered Columns"),
#"Replaced Value1" = Table.ReplaceValue("#Transposed Table1", null
, 0, Replacer.ReplaceValue, {"Column1", "Column2", "Column3", "
Column4", "Column5", "Column6", "Column7", "Column8", "Column9", "
Column10", "Column11", "Column12", "Column13", "Column14", "
Column15", "Column16", "Column17", "Column18", "Column19", "
Column20", "Column21", "Column22", "Column23", "Column24", "
Column25", "Column26", "Column27", "Column28", "Column29", "
Column30", "Column31", "Column32", "Column33", "Column34", "
Column35", "Column36", "Column37", "Column38", "Column39", "
Column40", "Column41", "Column42", "Column43", "Column44", "
Column45", "Column46", "Column47", "Column48", "Column49", "
Column50", "Column51", "Column52", "Column53", "Column54", "
Column55", "Column56", "Column57", "Column58", "Column59", "
Column60", "Column61", "Column62", "Column63", "Column64", "
Column65", "Column66", "Column67", "Column68", "Column69", "
Column70", "Column71", "Column72", "Column73", "Column74", "
Column75", "Column76", "Column77", "Column78"}),
17  #"Merged Queries1" = Table.NestedJoin("#Replaced Value1", {"
Column1"}, Name, {"Column2"}, "Name", JoinKind.LeftOuter),
#"Expanded Name" = Table.ExpandTableColumn("#Merged Queries1", "
Name", {"Column1"}, {"Column1.1"}),
19  #"Reordered Columns1" = Table.ReorderColumns("#Expanded Name", {"
Column1.1", "Column1", "Column2", "Column3", "Column4", "Column5",
"Column6", "Column7", "Column8", "Column9", "Column10", "Column11
", "Column12", "Column13", "Column14", "Column15", "Column16", "
Column17", "Column18", "Column19", "Column20", "Column21", "
Column22", "Column23", "Column24", "Column25", "Column26", "
Column27", "Column28", "Column29", "Column30", "Column31", "

```

21

```

Column32 ", "Column33 ", "Column34 ", "Column35 ", "Column36 ", "
Column37 ", "Column38 ", "Column39 ", "Column40 ", "Column41 ", "
Column42 ", "Column43 ", "Column44 ", "Column45 ", "Column46 ", "
Column47 ", "Column48 ", "Column49 ", "Column50 ", "Column51 ", "
Column52 ", "Column53 ", "Column54 ", "Column55 ", "Column56 ", "
Column57 ", "Column58 ", "Column59 ", "Column60 ", "Column61 ", "
Column62 ", "Column63 ", "Column64 ", "Column65 ", "Column66 ", "
Column67 ", "Column68 ", "Column69 ", "Column70 ", "Column71 ", "
Column72 ", "Column73 ", "Column74 ", "Column75 ", "Column76 ", "
Column77 ", "Column78 "}) ,
#"Promoted Headers" = Table.PromoteHeaders("#Reordered Columns1 ",
[ PromoteAllScalars=true ]) ,
#"Changed Type1" = Table.TransformColumnTypes("#Promoted Headers
",{{"source_name", type text}, {"time", type number}, {"
SLIDER_Z_th", type number}, {"SLIDER_Z_thd", type number}, {"
SLIDER_Z_thdd", type number}, {"SLIDER_Z_u", type number}, {"
SLIDER_Z_ufb", type number}, {"LF_HAA_u", type number}, {"
LF_HAA_uff", type number}, {"LF_HFE_th", type number}, {"
LF_HFE_des_th", type number}, {"LF_HFE_thd", type number}, {"
LF_HFE_des_thd", type number}, {"LF_HFE_thdd", type number}, {"
LF_HFE_u", type number}, {"LF_HFE_ufb", type number}, {"
LF_HFE_load", type number}, {"LF_HFE_uff", type number}, {"
LF_KFE_th", type number}, {"LF_KFE_des_th", type number}, {"
LF_KFE_thd", type number}, {"LF_KFE_des_thd", type number}, {"
LF_KFE_thdd", type number}, {"LF_KFE_u", type number}, {"
LF_KFE_ufb", type number}, {"LF_KFE_load", type number}, {"
LF_KFE_uff", type number}, {"LF_LEG_x", type number}, {"LF_LEG_z",
type number}, {"LF_LEG_xd", type number}, {"LF_LEG_zd", type
number}, {"LF_LEG_xdd", type number}, {"LF_LEG_zdd", type number},
{"LF_LEG_bd", type number}, {"LF_LEG_bdd", type number}, {"
LF_LEG_des_x", type number}, {"LF_LEG_des_z", type number}, {"
LF_LEG_des_xd", type number}, {"LF_LEG_des_zd", type number}, {"
LF_LEG_des_bd", type number}, {"LF_LEG_q0", type number}, {"
LF_LEG_q1", type number}, {"LF_LEG_q2", type number}, {"LF_LEG_q3
", type number}, {"LF_LEG_q0d", type number}, {"LF_LEG_q1d", type
number}, {"LF_LEG_q2d", type number}, {"LF_LEG_q3d", type number},
{"LF_LEG_q0dd", type number}, {"LF_LEG_q1dd", type number}, {"
LF_LEG_q2dd", type number}, {"LF_LEG_q3dd", type number}, {"
LF_LEG_des_q0", type number}, {"LF_LEG_des_q1", type number}, {"
LF_LEG_des_q2", type number}, {"LF_LEG_des_q3", type number}, {"
LF_LEG_des_q0d", type number}, {"LF_LEG_des_q1d", type number}, {"
LF_LEG_des_q2d", type number}, {"LF_LEG_des_q3d", type number}, {"

```

```

TIME_MOTOR", type number}, {"cog_x", type number}, {"cog_y", type
number}, {"cog_z", type number}, {"cog_des_x", type number}, {"
cog_des_y", type number}, {"cog_des_z", type number}, {"base_q0",
Int64.Type}, {"SPR_l", type number}, {"SPR_v", type number}, {"
SPR_F", type number}, {"SPR_Fs", type number}, {"SPR_K", Int64.
Type}, {"SPR_L0", type number}, {"SPR_u_X", type number}, {"
SPR_u_Z", type number}, {"SPR_J_LF_KFE_X", type number}, {"
SPR_J_LF_KFE_Z", type number}, {"SPR_J_LF_KFE_B", Int64.Type})),
  #"Unpivoted Columns" = Table.UnpivotOtherColumns(#"Changed Type1
", {"source_name", "time"}, "Attribute", "Value"),
23   #"Split Column by Delimiter" = Table.SplitColumn(#"Unpivoted
Columns", "Attribute", Splitter.SplitTextByDelimiter("_",
QuoteStyle.Csv), {"Attribute.1", "Attribute.2", "Attribute.3", "
Attribute.4"})
in
25   #"Split Column by Delimiter"
```

APÊNDICE D – CÓDIGO DO MATLAB PARA UMA DAS ETAPAS DE TRANSFORMAÇÃO DOS DADOS NA PLATAFORMA FC2D - FUNÇÃO MATLAB_TO_XLS

```

1  filepath = '/Documents/Experiments/logs/log_example.mat';
   split_path = strsplit(filepath, '/');
3  filename_extension = char(split_path(length(split_path)));
   split_file = strsplit(filename_extension, '.');
5  filename = split_file(1);

7  if ~exist(filepath, 'file')
       message = sprintf('%s does not exist', filepath);
9     uiwait(warndlg(message));
   else
11     vars = structfun(@(x) x, load(filepath));
   end
13
   field_names = fieldnames(vars);
15  value = struct2cell(vars);
   teste = value{1,1};
17
   for i=1:length(field_names);
19       var_name = cellstr(field_names{i});
       field_var = vars.(field_names{i});
21       if isa(field_var, 'double')
           field_value = num2cell(field_var);
23           parcial_result(:,i) = [var_name; field_value];
       end
25   end

27  source_name = cell(length(parcial_result),1);

29  for i=1:length(parcial_result);
       if i == 1
31           source_name{i,1} = 'source_name';
       else
33           source_name{i,1} = char(filename);
       end
35   end

```

```
37 result = [source_name partial_result];  
    workbook = char(strcat(filename, '.xlsx'));  
39 xlswrite(strcat('data/', workbook), result);
```

APÊNDICE E – CÓDIGO RESULTANTE DAS OPERAÇÕES REALIZADAS NO POWER QUERY PARA O EXPERIMENTO "SINE5HZ" - HYQ

```

1 let
    Source = Excel.CurrentWorkbook() {[Name="Table1 "]}[Content],
3     #"Changed Type" = Table.TransformColumnTypes(Source,{{"Column1",
    type text}, {"Column2", type any}, {"Column3", type any}, {"
    Column4", type any}, {"Column5", type any}, {"Column6", type any},
    {"Column7", type any}, {"Column8", type any}, {"Column9", type
    any}, {"Column10", type any}, {"Column11", type any}, {"Column12",
    type any}, {"Column13", type any}, {"Column14", type any}, {"
    Column15", type any}, {"Column16", type any}, {"Column17", type
    any}, {"Column18", type any}, {"Column19", type any}, {"Column20",
    type any}, ..., {"Column125", type any}}),
    #"Removed Columns" = Table.RemoveColumns(#"Changed Type",{"
    Column1"}),
5     #"Replaced Value" = Table.ReplaceValue(#"Removed Columns",0,null,
    Replacer.ReplaceValue,{"Column2", "Column3", "Column4", "Column5",
    "Column6", "Column7", "Column8", "Column9", "Column10", "Column11
    ", "Column12", "Column13", "Column14", "Column15", "Column16", "
    Column17", "Column18", "Column19", "Column20", "Column21", "
    Column22", "Column23", "Column24", "Column25", "Column26", ..., "
    Column125"}),
    #"Removed Blank Rows" = Table.SelectRows(#"Replaced Value", each
    not List.IsEmpty(List.RemoveMatchingItems(Record.FieldValues(_),
    {"", null}))),
7     #"Transposed Table" = Table.Transpose(#"Removed Blank Rows"),
    #"Removed Columns1" = Table.RemoveColumns(#"Transposed Table",{"
    Column1"}),
9     #"Removed Blank Rows1" = Table.SelectRows(#"Removed Columns1",
    each not List.IsEmpty(List.RemoveMatchingItems(Record.FieldValues(
    _), {"", null}))),
    #"Merged Queries" = Table.NestedJoin(#"Removed Blank Rows1",{"
    Column2", "Column3", "Column4", "Column5", "Column6", "Column7", "
    Column8", "Column9", "Column10", "Column11", "Column12", "Column13
    ", "Column14", "Column15", "Column16", "Column17", "Column18", "
    Column19", "Column20", "Column21", "Column22", "Column23", "
    Column24", "Column25", ..., "Column3000"},
11    #"sine5Hz (2)",{"Column2", "Column3", "Column4", "Column5", "
    Column6", "Column7", "Column8", "Column9", "Column10", "Column11",

```

```

    "Column12", "Column13", "Column14", "Column15", "Column16", "
Column17", "Column18", "Column19", "Column20", "Column21", "
Column22", "Column23", "Column24", ..., "Column3000"},"sine5Hz (2)
",JoinKind.LeftOuter),
    #"Expanded sine5Hz (2)" = Table.ExpandTableColumn(#"Merged
Queries", "sine5Hz (2)", {"Column1"}, {"Column1"}),
13    #"Removed Duplicates" = Table.Distinct(#"Expanded sine5Hz (2)",
{"Column1"}),
    #"Reordered Columns" = Table.ReorderColumns(#"Removed Duplicates
",{"Column1", "Column2", "Column3", "Column4", "Column5", "Column6
", "Column7", "Column8", "Column9", "Column10", "Column11", "
Column12", "Column13", "Column14", "Column15", "Column16", "
Column17", "Column18", "Column19", "Column20", "Column21", "
Column22", "Column23", ..., "Column10001"}),
15    #"Transposed Table1" = Table.Transpose(#"Reordered Columns"),
    #"Replaced Value1" = Table.ReplaceValue(#"Transposed Table1",null
,0,Replacer.ReplaceValue,{"Column1"}),
17    #"Merged Queries1" = Table.NestedJoin(#"Replaced Value1",{"
Column1"},Name,{"Column2"},"Name",JoinKind.LeftOuter),
    #"Expanded Name" = Table.ExpandTableColumn(#"Merged Queries1", "
Name", {"Column1"}, {"Column1.1"}),
19    #"Reordered Columns1" = Table.ReorderColumns(#"Expanded Name",{"
Column1.1", "Column1", "Column2", "Column3", "Column4", "Column5",
"Column6", "Column7", "Column8", "Column9", "Column10", "Column11
", "Column12", "Column13", "Column14", "Column15", "Column16", "
Column17", "Column18", "Column19", "Column20", "Column21", "
Column22", "Column23", "Column24", "Column25", "Column26", "
Column27", ..., "Column66"}),
    #"Promoted Headers" = Table.PromoteHeaders(#"Reordered Columns1",
[PromoteAllScalars=true]),
21    #"Changed Type1" = Table.TransformColumnTypes(#"Promoted Headers
",{{"source_name", type text}, {"time", type number}, {"
LF_HFE_hst_xp", type number}, {"LF_HFE_hst_xpd", type number}, {"
LF_HFE_hst_xpdd", type number}, {"LF_HFE_hst_fref", type number},
{"LF_HFE_hst_force", type number}, {"LF_HFE_dsp_hst_forc", type
number}, {"LF_HFE_hst_uv", type number}, {"LF_HFE_hst_uvc", type
number}, {"LF_HFE_hst_flow", type number}, {"LF_HFE_hst_pa", type
number}, {"LF_HFE_hst_pb", type number}, {"LF_HFE_hst_ps", Int64.
Type}, {"LF_KFE_hst_ps", Int64.Type}, {"LF_HFE_dsp_hst_xp", type
number}, {"LF_HFE_dsp_hst_uv", type number}, {"LF_HFE_dsp_hst_uvc
", type number}, {"LF_HFE_dsp_hst_pa", type number}, {"
LF_HFE_dsp_hst_pb", type number}, {"LF_HFE_dsp_hst_ps", type

```



```

number}, {"LF_KFE_dsp_hst_ps", type number}, {"LF_KFE_hst_xp",
type number}, {"LF_KFE_hst_xpd", type number}, {"LF_KFE_hst_xpdd",
type number}, {"LF_KFE_hst_fref", type number}, {"
LF_KFE_hst_force", type number}, {"LF_KFE_dsp_hst_forc", type
number}, {"LF_KFE_hst_uv", type number}, {"LF_KFE_hst_uvc", type
number}, {"LF_KFE_hst_flow", type number}, {"LF_KFE_hst_pa", type
number}, {"LF_KFE_hst_pb", type number}, {"LF_KFE_dsp_hst_xp",
type number}, {"LF_KFE_dsp_hst_uv", type number}, {"
LF_KFE_dsp_hst_uvc", type number}, {"LF_KFE_dsp_hst_pa", type
number}, {"LF_KFE_dsp_hst_pb", type number}, {"LF_HFE_rth", type
number}, {"LF_HFE_rthd", type number}, {"LF_HFE_rload", type
number}, {"LF_KFE_rth", type number}, {"LF_KFE_rthd", type number
}, {"LF_KFE_rload", type number}, {"SLIDER_Z_u", type number}, {"
SLIDER_Z_uff", type number}, {"LF_HAA_u", type number}, {"
LF_HAA_uff", type number}, {"LF_HFE_th", type number}, {"
LF_HFE_thd", type number}, {"LF_HFE_thdd", type number}, {"
LF_HFE_u", type number}, {"LF_HFE_load", type number}, {"
LF_HFE_des_th", type number}, {"LF_HFE_des_thd", type number}, {"
LF_HFE_uff", type number}, {"LF_KFE_th", type number}, {"
LF_KFE_thd", type number}, {"LF_KFE_thdd", type number}, {"
LF_KFE_u", type number}, {"LF_KFE_load", type number}, {"
LF_KFE_des_th", type number}, {"LF_KFE_des_thd", type number}, {"
LF_KFE_uff", type number}, {"MSNoReceive", Int64.Type}, {"
real_time", Int64.Type}, {"real_time_dt", Int64.Type})),
#"Unpivoted Columns" = Table.UnpivotOtherColumns("#Changed Type1
", {"source_name", "time"}, "Attribute", "Value"),
23 #"Split Column by Delimiter" = Table.SplitColumn("#Unpivoted
Columns", "Attribute", Splitter.SplitTextByDelimiter("_",
QuoteStyle.Csv), {"Attribute.1", "Attribute.2", "Attribute.3", "
Attribute.4"}),
#"Changed Type2" = Table.TransformColumnTypes("#Split Column by
Delimiter",{{"Attribute.1", type text}, {"Attribute.2", type text
}, {"Attribute.3", type text}, {"Attribute.4", type text}})
25 in
#"Changed Type2"

```


APÊNDICE F – CÓDIGO RESULTANTE DAS OPERAÇÕES REALIZADAS NO POWER QUERY PARA O EXPERIMENTO "EXPSRING_T" - HYQ

```

let
2   Source = Excel.CurrentWorkbook() {[Name="Table1 "]}[Content],
   #"Changed Type" = Table.TransformColumnTypes(Source,{{"Column1",
type text}, {"Column2", type any}, {"Column3", type any}, {"
Column4", type any}, {"Column5", type any}, {"Column6", type any},
   {"Column7", type any}, {"Column8", type any}, {"Column9", type
any}, {"Column10", type any}, {"Column11", type any}, {"Column12",
   type any}, {"Column13", type any}, {"Column14", type any}, {"
Column15", type any}, {"Column16", type any}, {"Column17", type
any}, {"Column18", type any}, {"Column19", type any}, {"Column20",
   type any}, ..., {"Column244", type any}}),
4   #"Removed Columns" = Table.RemoveColumns(#"Changed Type",{"
Column1"}),
   #"Replaced Value" = Table.ReplaceValue(#"Removed Columns",0,null,
Replacer.ReplaceValue,{"Column2", "Column3", "Column4", "Column5",
   "Column6", "Column7", "Column8", "Column9", "Column10", "Column11
", "Column12", "Column13", "Column14", "Column15", "Column16", "
Column17", "Column18", "Column19", "Column20", "Column21", "
Column22", "Column23", "Column24", "Column25", "Column26", ..., "
Column244"}),
6   #"Removed Blank Rows" = Table.SelectRows(#"Replaced Value", each
not List.IsEmpty(List.RemoveMatchingItems(Record.FieldValues(_),
{"", null}))),
   #"Transposed Table" = Table.Transpose(#"Removed Blank Rows"),
8   #"Removed Columns1" = Table.RemoveColumns(#"Transposed Table",{"
Column1"}),
   #"Removed Blank Rows1" = Table.SelectRows(#"Removed Columns1",
each not List.IsEmpty(List.RemoveMatchingItems(Record.FieldValues(
_), {"", null}))),
10  #"Merged Queries" = Table.NestedJoin(#"Removed Blank Rows1",{"
Column2", "Column3", "Column4", "Column5", "Column6", "Column7", "
Column8", "Column9", "Column10", "Column11", "Column12", "Column13
", "Column14", "Column15", "Column16", "Column17", "Column18", "
Column19", "Column20", "Column21", "Column22", "Column23", "
Column24", "Column25", ..., "Column3000"},
   #"expSpring_t (2)",{"Column2", "Column3", "Column4", "Column5", "
Column6", "Column7", "Column8", "Column9", "Column10", "Column11",

```

```

    "Column12", "Column13", "Column14", "Column15", "Column16", "
Column17", "Column18", "Column19", "Column20", "Column21", "
Column22", "Column23", ..., "Column3000"}, expSpring_t (2)",
JoinKind.LeftOuter),
12    #"Expanded expSpring_t (2)" = Table.ExpandTableColumn(#"Merged
Queries", "expSpring_t (2)", {"Column1"}, {"Column1"}),
    #"Removed Duplicates" = Table.Distinct(#"Expanded expSpring_t (2)
", {"Column1"}),
14    #"Reordered Columns" = Table.ReorderColumns(#"Removed Duplicates
", {"Column1", "Column2", "Column3", "Column4", "Column5", "Column6
", "Column7", "Column8", "Column9", "Column10", "Column11", "
Column12", "Column13", "Column14", "Column15", "Column16", "
Column17", "Column18", "Column19", "Column20", "Column21", "
Column22", "Column23", "Column24", "Column25", "Column26", ..., "
Column5001"}),
    #"Transposed Table1" = Table.Transpose(#"Reordered Columns"),
16    #"Replaced Value1" = Table.ReplaceValue(#"Transposed Table1", null
, 0, Replacer.ReplaceValue, {"Column1", "Column2", "Column3", "
Column4", "Column5", "Column6", "Column7", "Column8", "Column9", "
Column10", "Column11", "Column12", "Column13", "Column14", "
Column15", "Column16", "Column17", "Column18", "Column19", "
Column20", "Column21", "Column22", "Column23", "Column24", "
Column25", ..., "Column105"}),
    #"Merged Queries1" = Table.NestedJoin(#"Replaced Value1", {"
Column1"}, Name, {"Column2"}, "Name", JoinKind.LeftOuter),
18    #"Expanded Name" = Table.ExpandTableColumn(#"Merged Queries1", "
Name", {"Column1"}, {"Column1.1"}),
    #"Reordered Columns1" = Table.ReorderColumns(#"Expanded Name", {"
Column1.1", "Column1", "Column2", "Column3", "Column4", "Column5",
"Column6", "Column7", "Column8", "Column9", "Column10", "Column11
", "Column12", "Column13", "Column14", "Column15", "Column16", "
Column17", "Column18", "Column19", "Column20", "Column21", "
Column22", "Column23", "Column24", "Column25", "Column26", "
Column27", ..., "Column105"}),
20    #"Promoted Headers" = Table.PromoteHeaders(#"Reordered Columns1",
[PromoteAllScalars=true]),
    #"Changed Type1" = Table.TransformColumnTypes(#"Promoted Headers
", {{ "source_name", type text }, { "time", type number }, { "
SLIDER_Z_th", type number }, { "SLIDER_Z_thd", type number }, { "
SLIDER_Z_thdd", type number }, { "SLIDER_Z_u", type number }, { "
SLIDER_Z_ufb", type number }, { "LF_HAA_u", type number }, { "
LF_HAA_uff", type number }, { "LF_HFE_th", type number }, { "

```

LF_HFE_des_th", type number}, {"LF_HFE_thd", type number}, {"
 LF_HFE_des_thd", type number}, {"LF_HFE_thdd", type number}, {"
 LF_HFE_u", type number}, {"LF_HFE_ufb", type number}, {"
 LF_HFE_load", type number}, {"LF_HFE_uff", type number}, {"
 LF_KFE_th", type number}, {"LF_KFE_des_th", type number}, {"
 LF_KFE_thd", type number}, {"LF_KFE_des_thd", type number}, {"
 LF_KFE_thdd", type number}, {"LF_KFE_u", type number}, {"
 LF_KFE_ufb", type number}, {"LF_KFE_load", type number}, {"
 LF_KFE_uff", type number}, {"LF_LEG_x", type number}, {"LF_LEG_z",
 type number}, {"LF_LEG_xd", type number}, {"LF_LEG_zd", type
 number}, {"LF_LEG_xdd", type number}, {"LF_LEG_zdd", type
 number}, {"LF_LEG_bd", type number}, {"LF_LEG_bdd", type
 number}, {"LF_LEG_des_x", type number}, {"LF_LEG_des_z", type
 number}, {"LF_LEG_des_xd", type number}, {"LF_LEG_des_zd", type
 number}, {"LF_LEG_des_bd", type number}, {"LF_LEG_q0", type
 number}, {"LF_LEG_q1", type number}, {"LF_LEG_q2", type
 number}, {"LF_LEG_q3", type number}, {"LF_LEG_q0d", type
 number}, {"LF_LEG_q1d", type number}, {"LF_LEG_q2d", type
 number}, {"LF_LEG_q3d", type number}, {"LF_LEG_q0dd", type
 number}, {"LF_LEG_q1dd", type number}, {"LF_LEG_q2dd", type
 number}, {"LF_LEG_q3dd", type number}, {"LF_LEG_des_q0", type
 number}, {"LF_LEG_des_q1", type number}, {"LF_LEG_des_q2", type
 number}, {"LF_LEG_des_q3", type number}, {"LF_LEG_des_q0d", type
 number}, {"LF_LEG_des_q1d", type number}, {"LF_LEG_des_q2d", type
 number}, {"LF_LEG_des_q3d", type number}, {"FP_X12", type
 number}, {"FP_X34", type number}, {"FP_Y14", type
 number}, {"FP_Y23", type number}, {"FP_Z1", type
 number}, {"FP_Z2", type number}, {"FP_Z3", type
 number}, {"FP_Z4", type number}, {"FP_Fx", type
 number}, {"FP_Fy", type number}, {"FP_Fz", type
 number}, {"TIME_MOTOR", type number}, {"cog_x", type
 number}, {"cog_y", type number}, {"cog_z", type
 number}, {"cog_des_x", type number}, {"cog_des_y", type
 number}, {"cog_des_z", type number}, {"base_q0", Int64.Type}, {"SPR_l", type
 number}, {"SPR_v", type number}, {"SPR_F", type
 number}, {"SPR_Fs", type number}, {"SPR_Fd", type
 number}, {"SPR_K", Int64.Type}, {"SPR_D", Int64.Type}, {"
 SPR_L0", type number}, {"SPR_u_X", type number}, {"SPR_u_Z", type
 number}, {"SPR_J_LF_KFE_X", type number}, {"SPR_J_LF_KFE_Z", type
 number}, {"SPR_J_LF_KFE_B", Int64.Type}, {"E_SPR_l", type
 number}, {"E_SPR_v", type number}, {"E_SPR_F", type
 number}, {"E_SPR_Fs", type number}, {"E_SPR_Fd", type
 number}, {"E_SPR_A", Int64.Type}, {"E_SPR_B", Int64.Type}, {"E_SPR_D", Int64.Type}, {"E_SPR_L0",
 type number}, {"E_SPR_u_X", type number}, {"E_SPR_u_Z", type
 number}, {"E_SPR_J_LF_KFE_X", type number}, {"E_SPR_J_LF_KFE_Z",

```
type number}, {"E_SPR_J_LF_KFE_B", Int64.Type})),  
22   #"Unpivoted Columns" = Table.UnpivotOtherColumns(#"Changed Type1  
", {"source_name", "time"}, "Attribute", "Value"),  
   #"Split Column by Delimiter" = Table.SplitColumn(#"Unpivoted  
Columns", "Attribute", Splitter.SplitTextByDelimiter("_",  
QuoteStyle.Csv), {"Attribute.1", "Attribute.2", "Attribute.3", "  
Attribute.4"}),  
24   #"Changed Type2" = Table.TransformColumnTypes(#"Split Column by  
Delimiter",{{"Attribute.1", type text}, {"Attribute.2", type text  
}, {"Attribute.3", type text}, {"Attribute.4", type text}})  
in  
26   #"Changed Type2"
```

APÊNDICE G – CODIGO GERADO PELAS TRANSFORMAÇÕES FEITAS NO POWER QUERY PARA OS LOGS DA PLATAFORMA FC2D

```

let
2   Fonte = Folder.Files("C:\Users\Lorena Borges\Documents\tcc\FC2D\
    Log Files"),
    #"Arquivos Ocultos Filtrados1" = Table.SelectRows(Fonte, each [
        Attributes]?[Hidden]? <> true),
4   #"Invocar Função Personalizada1" = Table.AddColumn(#"Arquivos
    Ocultos Filtrados1", "Transformar Arquivo de Log Files", each #"
    Transformar Arquivo de Log Files"([Content])),
    #"Colunas Renomeadas1" = Table.RenameColumns(#"Invocar Função
    Personalizada1", {"Name", "Nome da Origem"}),
6   #"Outras Colunas Removidas1" = Table.SelectColumns(#"Colunas
    Renomeadas1", {"Nome da Origem", "Transformar Arquivo de Log Files
    "}),
    #"Erros Removidos1" = Table.RemoveRowsWithErrors(#"Outras Colunas
    Removidas1", {"Transformar Arquivo de Log Files"}),
8   #"Coluna de Tabela Expandida1" = Table.ExpandTableColumn(#"Erros
    Removidos1", "Transformar Arquivo de Log Files", Table.ColumnNames
    (#"Transformar Arquivo de Log Files"("#Arquivo de Amostra"))),
    #"Tipo Alterado" = Table.TransformColumnTypes(#"Coluna de Tabela
    Expandida1",{{"Nome da Origem", type text}, {"source_name", type
    text}, {"time", type number}, {"x", type number}, {"d2x", Int64.
    Type}, {"F", type number}, {"Pa", type number}, {"Pb", type number
    }, {"Ps", Int64.Type}, {"Pt", type number}, {"u_in", type number},
    {"xRef", type number}, {"FRef", type number}, {"rx", type number
    }, {"dx", type number}, {"u", type number}}})
10 in
    #"Tipo Alterado"

```


REFERÊNCIAS

- AALST, W. M. Van der. Data scientist: The engineer of the future. In: **Enterprise interoperability VI**. [S.l.]: Springer, 2014. p. 13–26.
- AHMED, E. et al. The role of big data analytics in internet of things. **Computer Networks**, Dec 2017.
- ASKITAS, N.; ZIMMERMANN, K. F. Google econometrics and unemployment forecasting. Jun 2009.
- BOAVENTURA, T.; BUCHLI, J. Acceleration-based transparency control framework for wearable robots. 2016.
- BOAVENTURA, T. et al. Dynamic torque control of a hydraulic quadruped robot. In: IEEE. **2012 IEEE international conference on robotics and automation**. [S.l.], 2012. p. 1889–1894.
- BRYANT, R.; KATZ, R. H.; LAZOWSKA, E. D. **Big-data computing: creating revolutionary breakthroughs in commerce, science and society**. [S.l.]: December, 2008.
- CHEN, H.; CHIANG, R. H. L.; STOREY, V. C. Business intelligence and analytics: From big data to big impact. In: MIS QUARTERLY. [S.l.], 2012.
- FOCCHI, M. et al. Control of a hydraulically-actuated quadruped robot leg. In: IEEE. **2010 IEEE International Conference on Robotics and Automation**. [S.l.], 2010. p. 4182–4188.
- GINSBERG, J. et al. Detecting influenza epidemics using search engine query data. **Nature**, v. 457, p. 1012, 2009.
- GOLFARELLI, M.; RIZZI, S. From star schemas to big data: 20+ years of data warehouse research. 2018.
- GUZMAN, G. Internet search behavior as an economic forecasting tool: The case of inflation expectations. **Economic Alchemy LLC**, p. 38, Nov 2011.
- HOWSON, C.; RICHARDSON, J.; SALLAM, R.; KRONZ, A. Magic quadrant for analytics and business intelligence platforms. Feb 2019.
- IMNON, W. H. Building the data warehouse. 1996.
- JUN, T.; KAI, C.; YU, F.; GANG, T. The research application of etl tool in business intelligence project. 2009.
- KIMBALL, R.; ROSS, M. **The Data Warehouse Toolkit**. 2^a. ed. [S.l.]: Wiley Computer Publishing, 1996.
- LOHR, S. The age of big data. **New York Times**, v. 11, n. 2012, 2012.
- MANYIKA, J. et al. Big data: The next frontier for innovation, competition, and productivity. In: MCKINSEY GLOBAL INSTITUTE. [S.l.], 2011.

MAURO, A. D.; GRECO, M.; GRIMALDI, M. What is big data? a consensual definition and a review of key research topics. In: AIP. **AIP conference proceedings**. [S.l.], 2015. v. 1644, n. 1, p. 97–104.

MCAFEE, A.; BRYNJOLFSSON, E.; DAVENPORT, T. H.; PATIL, D.; BARTON, D. Big data: the management revolution. **Harvard business review**, v. 90, n. 10, p. 60–68, 2012.

NEGASH, S.; GRAY, P. Handbook on decision support systems 2. **New York: SpringerLink**, 2008.

RAJASEKARANA, V.; ARANDA, J.; CASALS, A.; PONS, J. L. An adaptive control strategy for postural stability using a wearable robot. **Elsevier**, 2015.

SAGIROGLU, S.; SINANC, D. Big data: A review. In: IEEE. **2013 International Conference on Collaboration Technologies and Systems (CTS)**. [S.l.], 2013. p. 42–47.

SEMINI, C. et al. Design of the hydraulically actuated, torque-controlled quadruped robot hyq2max. **IEEE/ASME Transactions on Mechatronics**, IEEE, v. 22, n. 2, p. 635–646, 2016.

TRYFONA, N.; BUSBORG, F.; CHRISTIANSEN, J. G. B. Starer: A conceptual model for data warehouse design. 1996.