UNIVERSIDADE FEDERAL DE ITAJUBA
PROGRAMA PÓS-GRADUAÇÃO
EM ENGENHARIA ELÉTRICA

José Carlos Grilo Rodrigues

Implementação de um Simulador para as mais Comumente encontradas Cargas de Motores Industriais, baseado em Inversores PWM e Estimador de Torque

Tese submetida ao Programa de Pós-Graduação em Engenharia Elétrica como parte dos requisitos para obtenção do título de Doutor em Ciências em Engenharia Elétrica.

Área de concentração:
Sistemas Elétricos de Potência

Orientador: prof. Dr. Manuel Luís Barreira Martinez
Orientador: prof. Dr. Ângelo José Junqueira Rezek

Junho - 2013
Itajubá

Aos meus pais (in memória) ***Eloisa*** *e* ***Renato****, razão pela qual cheguei até aqui, ao meu irmão* ***Antônio Mário*** *e aos meus irmãos (in memória)* ***Carlos*** *e* ***Paulo*** *eternos torcedores pelo meu sucesso, a minha esposa* ***Sandra****, cúmplice de todos os momentos e aos meus filhos,* ***José Renato*** *e* ***João Rafael****, frutos de um amor sem fim.*

É graça divina começar bem. Graça maior persistir na caminhada certa. Mas a graça das graças é não desistir nunca.

(Dom Hélder Câmara)

AGRADECIMENTOS

A DEUS, pelo crescimento pessoal e profissional ao longo desta caminhada.

Aos meus orientadores, Prof. Dr. Ângelo José Junqueira Rezek e Prof. Dr. Manuel Luís Barreira Martinez, pelos incentivos, paciência e por acreditarem na minha capacidade.

Aos meus grandes amigos em especial aos do LEPCH, Délvio, Lucio, Luciano, Luiz Otavio, Luiz Sérgio, Suely e Claudia, eternos torcedores pelo meu crescimento.

A todos os alunos que de maneira dedicada fizeram parte dos nossos trabalhos, ao longo destes anos.

Ao colega Nery de Oliveira Junior pela amizade e pelas sugestões ao longo do trabalho, ressaltando-se que foi dele a ideia desta implementação.

RESUMO

Este trabalho apresenta a implementação de um simulador genérico de cargas, utilizando inversores PWM. Cargas as mais diversas podem ser simuladas permitindo-se, entre outros, por exemplo, a avaliação de desempenho de motores acionadores. O sistema implementado apresentou resultados satisfatórios, mesmo para baixas velocidades. Utilizou-se no arranjo do sistema, dois inversores PWM um deles para o motor e o outro para o gerador, um microcomputador, uma placa de aquisição de dados PCL 711 da ADVANTECH, instalada em slot livre do microcomputador, tendo como entradas analógicas, conversores A/D, os sinais analógicos de velocidade proveniente do taco gerador e do transdutor de torque do inversor PWM do motor (saída analógica 1). A saída analógica conversor D/A, da placa é responsável para definição da frequência do inversor PWM 2 do gerador de indução que funciona como carga do sistema. Um sistema de aquisição de dados LABVIEW®, foi também utilizado para filtragem dos sinais de velocidade e Torque para melhor visualização e interpretação dos resultados obtidos.

ABSTRACT

This work presents the implementation of a generic load torque Simulator, using PWM inverters. The most diverse loads can be simulated for evaluation, for example the electric motor drives performance. The implemented system has presented satisfactory results, even for low speeds. The system hardware was assembled, two PWM inverters one for the motor and one for the generator, a computer, a data acquisition card PCL 711 of ADVANTECH, installed in free slot of the microcomputer, with analog inputs, A/D converters, analog signals from the speed (analog taco generation) and Torque transducer from PWM inverter motor (analog output 1 isq). The analog output D/A converter, of the board is responsible for the frequency setup of PWM inverter 2 of induction generator, which works as system motor load. A LABVIEW® data acquisition system platform was also used to the filtering of the signals of speed and torque assuring filtering for better visualization and interpretation of the results.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SÍMBOLOS E NOTAÇÕES

s: escorregamento

n_1: velocidade do campo girante no estator [rpm]

n_2: velocidade do eixo do rotor [rpm]

n : rotação mecânica do rotor [rpm]

f : frequência de alimentação da rede [Hz]

p : numero de polos

T : torque disponível na ponta do eixo [N.m]

ϕ_m: fluxo de magnetização [Wb]

I : corrente rotórica [A]; (depende da carga)

U : tensão estatórica

K1 e K2 : constantes que dependem das caracteristicas construtivas da máquina.

P : potência mecânica no eixo [W]

f_M: frequência da tensão aplicada nos terminais da máquina motor [Hz]

f_G: frequência da tensão aplicada nos terminais da máquina geradora [Hz]

VOut: tensão de saída do microcomputador em PU

sn: escorregamento nominal do motor;

fc: fator de carga;

nrum: rotação média do motor em PU.

tr: torque de referência

PREÂMBULO

Esta tese que verifica a possibilidade de se utilizar inversores PWM associados à circuitos convenientemente projetados para projetar e construir um simulador de cargas capaz de fornecer os mais comuns tipos de cargas encontrados na indústria. Deste modo, se encontra dividida em quatro capítulos. O primeiro descreve as motivações que levaram a esta proposta, bem como lança as hipóteses a serem demonstrada, a arquitetura básica da proposta e as contribuições geradas. O segundo fornece detalhes do desenvolvimento proposto e o terceiro os resultados obtidos e as subsequentes discussões, ou seja o núcleo desta proposta. Finalmente, o quarto e último capítulo desta tese apresenta as considerações, novamente as contribuições e sugestões finais para possíveis continuações dentro desta linha de pesquisa.

CAPITULO 1 – INTRODUÇÃO

1.1 - MOTIVAÇÃO

Invariavelmente a comunidade científica é incentivada a buscar soluções aceitáveis com argumentos técnicos e financeiros, para a obtenção de sistemas robustos com reduzida necessidade de manutenção e de alto desempenho, congregando os interesses da indústria em relação a produtividade, qualidade e eficiência energética.

Devido à sua relativa simplicidade de construção, manutenção, e sua capacidade de operar com uma grande diversidade de cargas em condições adversas, o Motor de Indução Trifásico (MIT) com rotor em gaiola de esquilo está amplamente difundido em vários ramos da indústria, acionando sistemas que podem requerer o controle da velocidade de rotação, posição e Torque. Entretanto, o forte acoplamento, as características não-lineares e as estruturas multivariáveis, limitaram por um grande período de tempo, a aplicação deste tipo de motores em certos tipos de acionamentos. A crescente utilização das máquinas de indução em acionamento de alto desempenho em detrimento das máquinas de corrente contínua foi observada após o surgimento de diferentes estratégias de controle. No entanto, somente com o avanço da tecnologia dos semicondutores é que puderam ser projetados conversores estáticos de frequência, possibilitando a composição de sistemas de acionamento viáveis com motores de indução.

Já faz alguns anos que o advento dos inversores de frequência, utilizando tecnologia digital com microprocessadores, tem tornado vantajosa a relação custo benefício para o uso de acionamentos em corrente alternada. Este fato tem levado as indústrias a trocarem os tradicionais acionamentos com motores de corrente contínua por acionamentos com motores de indução, alimentados através de inversores de frequencia. Como é bem conhecido, motores de corrente contínua têm um elevado custo de manutenção. Não obstante, principalmente quando de motores de menor potência, apresenta a vantagem de manter o torque de modo mais eficiente que motores de corrente alternada. Deste modo, para produzir um mesmo Torque, o motor de corrente alternada deve ser mais robusto, ou seja, apresentar maior potência. Ainda assim mesmo com o sobredimensionamento do motor, o custo de um motor de indução trifásico ainda é muito menor do que um motor de corrente contínua. Os inversores de frequência atuais permitem uma vasta gama de parâmetros para o controle de velocidade. Assim, a precisão e a leveza no ajuste dos acionamentos em corrente alternada

não deixam nada a desejar se comparado a um motor de corrente contínua. Convém ainda ressaltar que há outros diversos recursos que o inversor de frequência pode proporcionar.

Atualmente, o estágio de desenvolvimento da tecnologia de acionamento dos motores de indução está consolidado, tornando-se um desafio atual a proposição de soluções que possam reduzir o custo final do conjunto motor-conversor.

Desde o início do século XX até sua metade, a válvula termoiônica reinou absoluta, quando na metade do século, em 1948, a gigante em telecomunicações Bell Telephone, desenvolveu um dispositivo que em comparação à válvula termoiônica era simplesmente minúsculo. Era o primeiro transistor. Aí dava-se início à era do semicondutor.

Com o transistor e o desenvolvimento das técnicas de miniaturização, ficou cada vez mais acelerada a confecção e projeto de componentes e equipamentos eletrônicos.

Isto culminou com a construção do primeiro circuito integrado no final da década de sessenta, quando apareceu o primeiro amplificador operacional integrado. Este nada mais era que a montagem miniaturizada de transistores, capacitores, resistores e diodos semicondutores, todos feitos numa só base, inicialmente em germânio.

Logo após, no início da década de setenta, os componentes passaram a ser fabricados em silício, elemento de mais fácil manipulação e menos sensível aos efeitos de avalanche térmica.

Foram sendo desenvolvidas assim exponencialmente novas tecnologias para a fabricação seriada em alta velocidade. Estas utilizavam componentes de larga escala de integração, (LSI), sendo que logo após, nos anos oitenta, foi desenvolvida a extra larga escala de integração, (ELSI). Esta tecnologia deu origem aos microprocessadores de alta velocidade e desempenho.

Nos dias de hoje, depois do trabalho de milhares, senão milhões de colaboradores anônimos, a Eletrônica está finalmente entrando na era da nanotecnologia.

Demonstrações matemáticas e práticas comprovam que um “Motor de Indução Trifásica” - MIT, trabalhando de forma sobredimensionada apresenta redução do seu fator de potência e diminuição de sua eficiência (Fitzgerald *et. al.*, 1975; Kosov, 1972). Por outro lado, estando o MIT trabalhando de forma subdimensionada este apresenta sobreaquecimento e por conseguinte, uma drástica redução de sua vida útil (Kosov, 1972).

Segundo (Goedtel,2006), uma pesquisa feita pela “Companhia Energética de Minas Gerais” - CEMIG contemplando 3425 motores de indução trifásicos, em diversos setores industriais, mostrou que 28,7% destes motores trabalhavam de forma sobredimensionada enquanto 5,9% estavam operando de forma subdimensionada. Um outro

estudo, realizado pela “Companhia Energética do Estado do Paraná” - COPEL com 6108 motores de indução trifásicos, mostrou que 37,75% estavam trabalhando de forma sobredimensionada (Marach, 2001).

A estimativa de torque de carga de motores de indução tem três objetivos principais (Goedtel, 2006). O primeiro, mais importante, é prover informações a respeito da carga contribuindo para o correto dimensionamento do motor. O segundo objetivo é prover dados relativos ao comportamento da carga no eixo de forma a determinar a eficiência e desempenho da conversão de energia. Em terceiro, tem-se que ter em mente que a estimação do torque aplicado nos eixos dos motores de indução é de fundamental importância para o desenvolvimento de técnicas eficientes de controle nos regimes transitório e permanente.

Dentro deste contexto, é também necessário atentar que o correto dimensionamento de um sistema de velocidade variável depende do conhecimento do comportamento da carga, ou seja, do torque na ponta de eixo do motor. Assim sendo, as cargas podem ser classificadas em três tipos: torque variável, torque constante e potência constante [19].

Cargas com torque variável: Bombas centrífugas, Exaustores centrífugos ventiladores, compressores centrífugos. A variação da velocidade por meio de acionamento eletrônico permite grandes economias de energia com esse tipo de carga, uma vez que a potência mecânica disponibilizada no eixo do motor não é constante, mas varia convenientemente de acordo com a exigência da carga. Nestes casos, variações quadráticas do torque com a velocidade implicam em variações cúbicas da potência com a velocidade, bem como variações lineares do torque com a velocidade implicam em variações quadráticas da potência com a velocidade.

Cargas com torque constante: Compressores alternados, compressores helicoidais, elevadores de caneca, esteiras transportadoras, bombas de deslocamento positivo, extrusoras, trituradores. Nestes casos, a potência varia linearmente com a velocidade.

Cargas com potência constante: Ferramentas de usinagem, bobinadeiras. Nestes casos, observa-se uma variação inversa do torque com a velocidade. O torque, com o aumento da velocidade cai proporcionalmente com a frequência sendo a potência constante.

Visto a ampla aplicabilidade em diversos estudos que buscam inovações no controle da máquina, eficiência energética, avaliação do torque de carga e predição de falhas no motor e no mecanismo acoplado, torna-se importante estudar a possibilidade de se implantar um simulador universal de cargas, baseado em inversores PWM e estimador de

torque que proporcione simplicidade na concepção e considerável robustez quanto às variações das cargas, sendo este o moto que orienta este trabalho.

1.2 **HIPÓTESE**

Esta tese tem por hipótese demonstrar que, através do acoplamento mecânico direto de duas máquinas de indução trifásicas, acionadas por dois inversores PWM, é possível implantar um sistema de imposição de cargas empregando-se, complementarmente, um estimador de torque. Para tanto as máquinas operam, uma como motor e a outra como gerador. O sistema assim construído passa então a incorporar todas as facilidades para simular as mais importantes cargas industriais o que, por sua vez, permite, dentro de limites, avaliar o desempenho de motores trifásicos e sua especificação.

1.3 **ARQUITETURA BÁSICA**

A máquina de indução trifásica pode funcionar tanto como motor quanto como gerador. Na operação como motor o fluxo magnético resultante gira a uma velocidade superior a do rotor, "arrastando-o", e na operação como gerador é o rotor que gira a uma velocidade superior à do fluxo magnético criado pelas correntes que circulam pelo estator. Portanto, ao acoplar os eixos de duas máquinas de indução com o objetivo de que uma opere como carga para a outra, deve-se aplicar tensões nos terminais da máquina que opera como motor com frequência tal que a velocidade de seu eixo seja superior à rotação do fluxo magnético resultante criado pela tensão aplicada aos terminais da máquina que opera como gerador. Ou seja, a frequência da tensão aplicada ao motor deve seguir a Equação 1.1.

$$f_M \times (1 - s) > f_G \tag{1.1}$$

Onde:

f_M : frequência da tensão aplicada nos terminais da máquina motor [Hz];

f_G : frequência da tensão aplicada nos terminais da máquina gerador [Hz];

s: escorregamento do motor.

No simulador de cargas proposto é isto que ocorre. Os eixos dos motores são acplopados através de acoplamento flexível, o Inversor 1 aciona a máquina motor e o Inversor 2 aciona a máquina gerador, como mostra a Figura 1.1

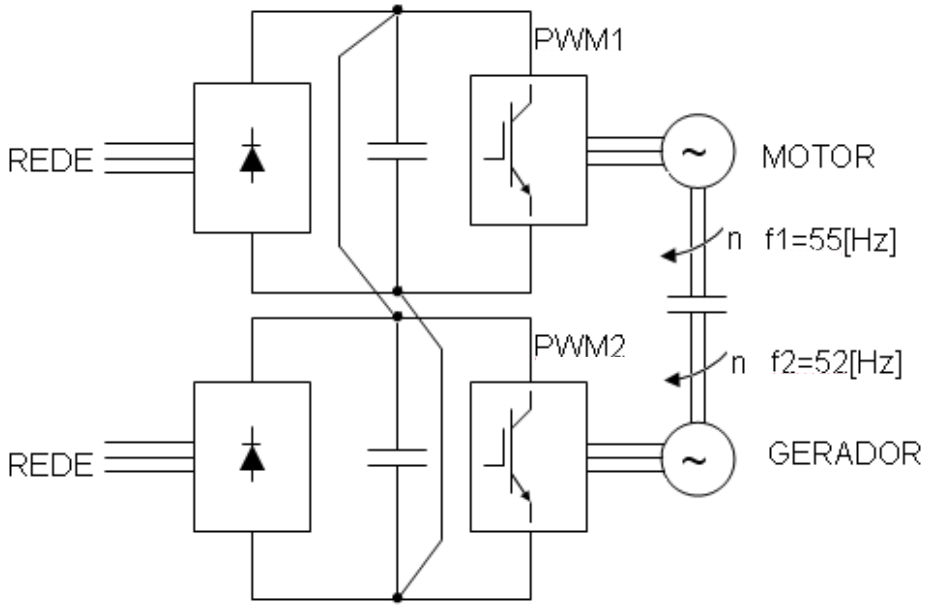


Figura 1.1 – Configuração da arquitetura

A lógica do sistema permite, através de duas chaves, uma para a máquina motora e outra para a geradora se utilizar de controle manual ou automático. No comando manual, pode-se visualizar os valores das frequências de alimentação das máquinas através da Interface homem-máquina dos inversores, este modo de operação foi usado apenas para realização de ensaios de avaliação de comportamente e parte de validação. No comando automático, o algoritmo computacional recebe os sinais de velocidade e de torque através de placa de aquisição de dados instalada em slot livre do PC, a qual possui entradas analógicas (conversores A/D) e uma saída analógica (conversor D/A), realiza os cálculos e libera um sinal que, após passar pelo conversor D/A, aciona o Inversor 2. Os bancos de capacitores dos inversores são ligados em paralelo, de modo que a energia gerada na máquina gerador seja aproveitada pelo Inversor 1 no acionamento do motor, como mostra a Figura 1.1.

O algoritmo computacional desenvolvido permite fazer a escolha de simulação de três tipos de cargas: carga tipo torque constante, tipo torque linear e tipo torque quadrático.

1.4 CONTRIBUIÇÕES

Demonstrar que é possível, com o auxílio de uma lógica de controle adequada, a construção de bancadas para a realização de estudos em motores de indução trifásicos, simulando as várias condições de cargas através do uso apropriado de qualquer inversor com controle vetorial. Isto resulta em redução de custos de engenharia e desenvolvimento, bem como permite o desenvolvimento de várias análises que podem ter por objetivo melhorar a utilização da energia na indústria. Deste modo, torna-se possível ensaiar motores com as mais variadas cargas, como por exemplo, do tipo torque constante (Correias transportadoras), do tipo torque variável com o quadrado da rotação (Bombas, ventiladores e compressores), do tipo torque proporcional a rotação (Calandras) ou ainda do tipo torque proporcional ao inverso da rotação (Bobinadeiras). Estas cargas representam a grande maioria das cargas industriais que passam a ser simuladas de forma eficiente sem a necessidade de arranjos físicos complementares.

Considera-se uma importante contribuição complementar demonstrar que é possível utilizar um transdutor de torque estimado pelo próprio inversor PWM do motor, o que facilita a implementação da técnica e reduz custos.

Finalmente, em termos físicos e também base importante para novos trabalhos nas áreas de controle e acionamentos na Universidade Federal de Itajubá, construir no "Laboratório Eletromecânico para Pequenas Centrais Hidroelétricas" - LEPCH uma bancada contendo como elementos principais 02 inversores PWM SIMOVERT® Siemens acionando 02 motores de indução com rotor em gaiola. Com o arranjo é possível implementar um simulador genérico de cargas, sendo um dos inversores PWM utilizado para alimentar uma das máquinas como motor e o outro para alimentar a segunda máquina de indução, funcionando como gerador, desempenhando a função de carga.

CAPITULO 2 – DESENVOLVIMENTO

2.1 - CONCEITOS E CONSIDERAÇÕES

2.1.1 – Motor de Indução

Em uma máquina de indução, correntes alternadas são aplicadas diretamente nos enrolamentos do estator, já as correntes nos enrolamentos do rotor surgem por indução eletromagnética, ou seja, por efeito transformador. Portanto, a máquina de indução apresenta-se como um transformador generalizado, transferindo potência elétrica com variação de tensão, corrente e frequência.

Neste tipo de máquina o fluxo magnético resultante do estator não está em sincronismo com o giro do rotor, há, portanto, um escorregamento, que proporciona a indução de correntes no rotor, dando origem ao torque eletromecânico. Pode-se ver então que as máquinas de indução operam com velocidades ligeiramente diferentes da velocidade mecânica síncrona. O escorregamento pode ser calculado através da Equação 2.1.

$$s = \frac{n_1 - n_2}{n_1} \qquad 2.1$$

Onde:

s : escorregamento;

n_1 : velocidade do campo girante no estator [rpm];

n_2 : velocidade do eixo do rotor [rpm];

A máquina de indução pode funcionar tanto como motor quanto como gerador. Na operação como motor o fluxo magnético resultante do estator gira a uma velocidade superior à velocidade do eixo do rotor, “arrastando” este, disponibilizando potência mecânica no eixo. Já na operação como gerador o eixo do rotor deve ser posto para girar com velocidade superior a do fluxo magnético resultante do estator, disponibilizando assim potência elétrica para a rede.

A relação entre a velocidade de rotação do rotor, o número de polos, o escorregamento e a frequência de alimentação da rede pode ser expressa através da Equação 2.2.

$$n_2 = \frac{120 \times f(1-s)}{p} \qquad 2.2$$

Onde:

n_2 : velocidade do eixo do rotor [rpm];

f : frequência de alimentação da rede [Hz];

s : escorregamento;

p : número de polos.

Analisando a equação anterior verifica-se que existem três possibilidades de alteração da velocidade de operação da máquina de indução: alteração no número de polos, alteração no escorregamento e mudança na frequência de alimentação. Para conseguir alteração no número de polos seriam necessárias mudanças estruturais na máquina; mudanças no escorregamento são possíveis através de variação da resistência do rotor, porém dentro de uma faixa muito restrita, não possibilitando todas as velocidades necessárias, mudanças na frequência são possíveis com a utilização de inversores de frequência. Portanto, o meio largamente utilizado para controle de velocidade das máquinas de indução são os inversores de frequência.

O torque desenvolvido no eixo de um motor de indução depende diretamente do fluxo magnetizante, da corrente e das características construtivas da máquina, de acordo com as Equações 2.3 e 2.4 – esta última é válida quando se despreza a queda de tensão nas impedâncias dos enrolamentos estatóricos.

$$T = K_1 \times \phi_m \times I \times \cos\varphi_2 \qquad 2.3$$

$$\phi_m = K_2 \times \frac{U}{f} \qquad 2.4$$

Onde:

T : torque disponível na ponta do eixo [N.m];

ϕ_m : fluxo de magnetização [Wb];

I : corrente rotórica [A]; (depende da carga)

U : tensão estatórica [V];

$\cos\varphi_2$: Fator de potência do rotor

K1 e K2 : constantes que dependem das caracteristicas construtivas da máquina.

Pode-se também relacionar o torque desenvolvido no eixo da máquina com a potência, como mostra a Equação 2.5.

$$T = \frac{60}{2\pi} \times \frac{P}{n_2} \qquad 2.5$$

Onde:

P : potência mecânica no eixo [W];

n_2 : velocidade do eixo do rotor [rpm];

2.1.2 – Inversores de frequência PWM

A utilização de inversores estáticos de frequência atualmente compreende o método mais eficiente para controlar a velocidade de motores de indução. Os inversores transformam a tensão da rede, de amplitude e freqüência constantes, em uma tensão de amplitude e freqüência variáveis. Eles operam como uma interface entre a fonte de energia (rede) e o motor de indução.

Os inversores funcionam basicamente em três estágios:

Ponte de diodos: onde ocorre a retificação do sinal alternado obtido da rede de alimentação;

Barramento CC ou Filtro: é feita a regulação da tensão por meio de banco de capacitores;

Transistores IGBT: o sinal contínuo proveniente do enlace CC é convertido em sinal alternado, com tensão e frequência variáveis.

Podem-se ver na Figura 2.1 os três estágios de funcionamento dos inversores PWM.

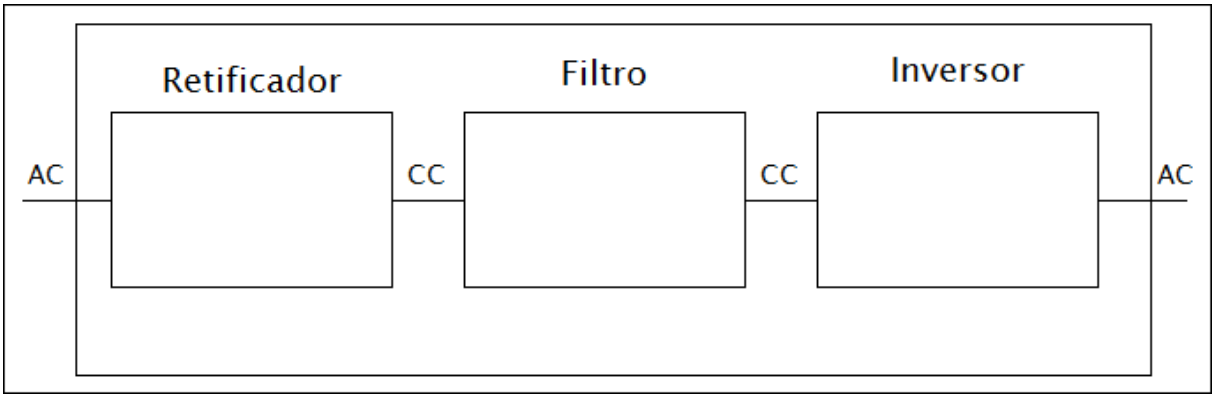


Figura 2.1 – Estágios de funcionamento dos inversores PWM

Existem dois modos de controle de velocidade nos inversores eletrônicos:

O controle escalar baseia-se no conceito original do inversor de frequência: impõe no motor uma determinada tensão/frequência, visando manter a relação U/f constante, ou seja, o motor trabalha com fluxo aproximadamente constante. É aplicado quando não há necessidade de respostas rápidas (1000 ms) a comandos de torque e velocidade sendo particularmente interessante quando há conexão de múltiplos motores a um único inversor. O controle é realizado em malha aberta e a precisão da velocidade é função do escorregamento do motor, que varia em função da carga, já que a frequência no estator é imposta. Para melhorar o desempenho do motor nas baixas velocidades, alguns inversores possuem funções especiais como a compensação de escorregamento (que atenua a variação da velocidade em função da carga) e o *boost* de tensão (aumento da relação U/f para compensar o efeito da queda de tensão na resistência estatórica), de maneira que a capacidade de torque do motor seja mantida. O controle escalar é o mais utilizado devido à sua simplicidade e devido ao fato de que a grande maioria das aplicações não requer alta precisão e/ou rapidez no controle da velocidade.

O controle vetorial possibilita atingir um elevado grau de precisão e rapidez (25 a 50 ms) no controle do torque e da velocidade do motor. O controle decompõe a corrente do motor em dois vetores: um que produz o fluxo magnetizante e outro que produz torque, regulando separadamente o torque e o fluxo. O controle vetorial pode ser realizado em malha aberta (“sensorless”) ou em malha fechada (com realimentação).

Com sensor de velocidade – requer a instalação de um sensor de velocidade (por exemplo, um encoder incremental) no motor. Este tipo de controle permite a maior precisão possível no controle da velocidade e do torque, inclusive em rotação zero.

Sensorless – mais simples que o controle com sensor, porém, apresenta limitações de torque principalmente em baixíssimas rotações. Em velocidades maiores é praticamente tão bom quanto o controle vetorial com realimentação.

As principais diferenças entre os dois tipos de controle são que o controle escalar só considera as amplitudes das grandezas elétricas instantâneas (fluxos, correntes e tensões), referindo-as ao estator, e seu equacionamento baseia-se no circuito equivalente do motor, ou seja, são equações de regime permanente. Já o controle vetorial admite a representação das grandezas elétricas instantâneas por vetores, baseando-se nas equações espaciais dinâmicas da máquina, com as grandezas referidas ao fluxo enlaçado pelo rotor, ou seja, o motor de indução é visto pelo controle vetorial como um motor de corrente contínua, havendo regulação independente para torque e fluxo.

2.1.3 – Conversores A/D D/A

Quando há necessidade de processamento de dados oriundos de sistemas analógicos por circuitos digitais (por exemplo, um microcomputador) são os conversores A/D e D/A que tornam viável a comunicação entre o ambiente analógico e o ambiente digital. A Figura 2.2 ilustra uma situação destas.

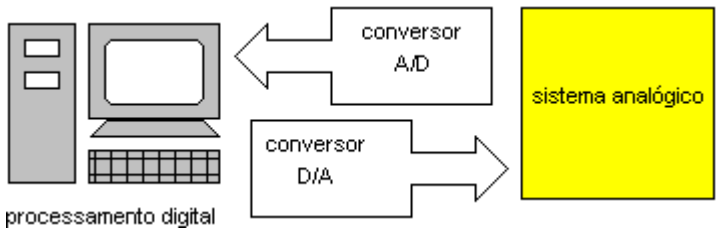


Figura 2.2 – Conversão dos sinais analógicos e digitais

A Figura 2.3 mostra um conversor A/D tipo rampa. Neste tipo de conversor é gerado um sinal de *clear* para "resetar" o contador, enquanto a tensão Ue (tensão analógica de entrada) for maior que a tensão de referência Ur – que é gerada por um conversor D/A ligado a saída do contador – o contador é incrementado.

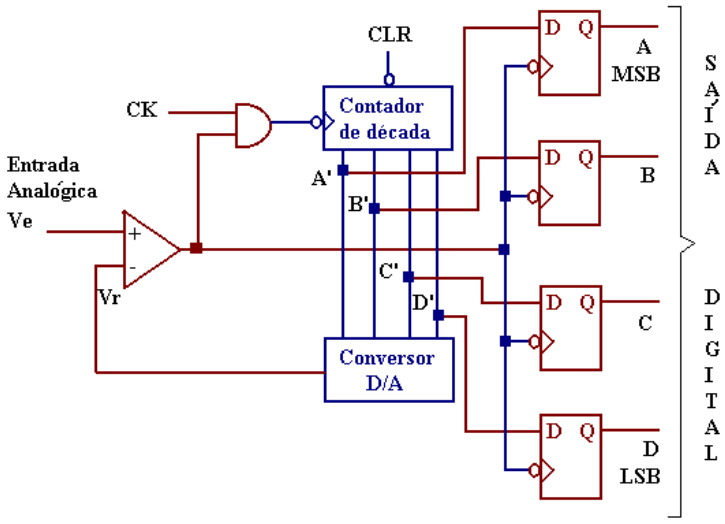


Figura 2.3 – Conversor A/D tipo rampa

O conversor D/A de resistores com pesos ponderados é o mais simples dos conversores D/A. Construído a partir de um circuito básico de resistores em paralelo controlado por corrente, onde a corrente é somada num ponto em comum, passando por um resistor de carga, criando assim uma saída analógica. Os valores dos resistores são distribuídos ponderadamente, de forma a obter pesos de acordo com a numeração binária, como mostra a Figura 2.4

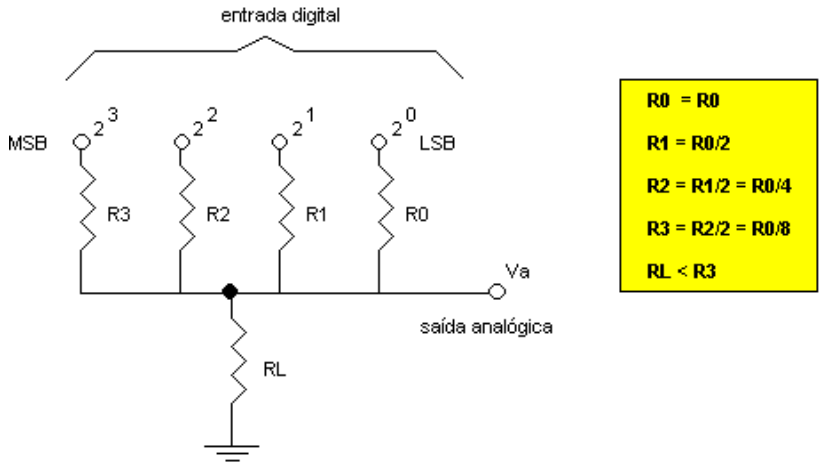


Figura 2.4 – Conversor D/A de resistores

2.2 - MATERIAL

Na Tabela 2.1 tem-se a lista de equipamentos utilizados para a viabilização do simulador genérico de cargas proposto. Alguns destes equipamentos já estavam disponínveis no Laboratório Hidromecânico para Pequenas Centrais Hidroelétricas e no Laboratório de Acionamentos Elétricos da Unifei, alguns outros como, por exemplo, placas com circuito impresso – foram desemvolvidos de acordo com as necessidades.

Tabela 2.1: Lista de equipamentos, algoritmo e circuitos.

Item	Especificação	Quantidade
1	Inversor de frequência PWM Fabricante: Simens Simovert Master Drives 380/460 [V], 25,5 [A]	2
2	Motor de indução trifásico 60 [Hz] Fabricante: WEG e EBERLE 10 [cv] 1780 [rpm] - 1765 [rpm]	2
3	Microcomputador	1
4	Notebook	1
5	NI USB-6212 – National Instruments	1
6	Algoritmo LABVIEW	1
7	Algoritmo em linguagem c	1
8	Placas com circuito impresso	3
9	Torquímetro	1
10	Fonte CC 15 [V]	2
11	Placa com conversor A/D e D/A	1

12	Chave liga/desliga com tensão de saída variável de 0 – 10[V]	1
13	Chave seletora manual/desliga/automático com tensão de saída variável de 0-10 [V]	1

2.3 CIRCUITOS INTRMEDIÁRIOS

Foram desenvolvidos cinco circuitos intermediários: duas chaves com tensão variável de saída, 0 – 10 [V], necessárias para a alimentação do controle analógico dos inversores, e três placas com circuito impresso para a interligação entre a saída do conversor D/A e o inversor que aciona a máquina em operação como gerador. A Figura 2.5 mostra a arquitetura experimental com todos os equipamentos representados. A seguir, será feito o detalhamento dos circuitos intermediários.

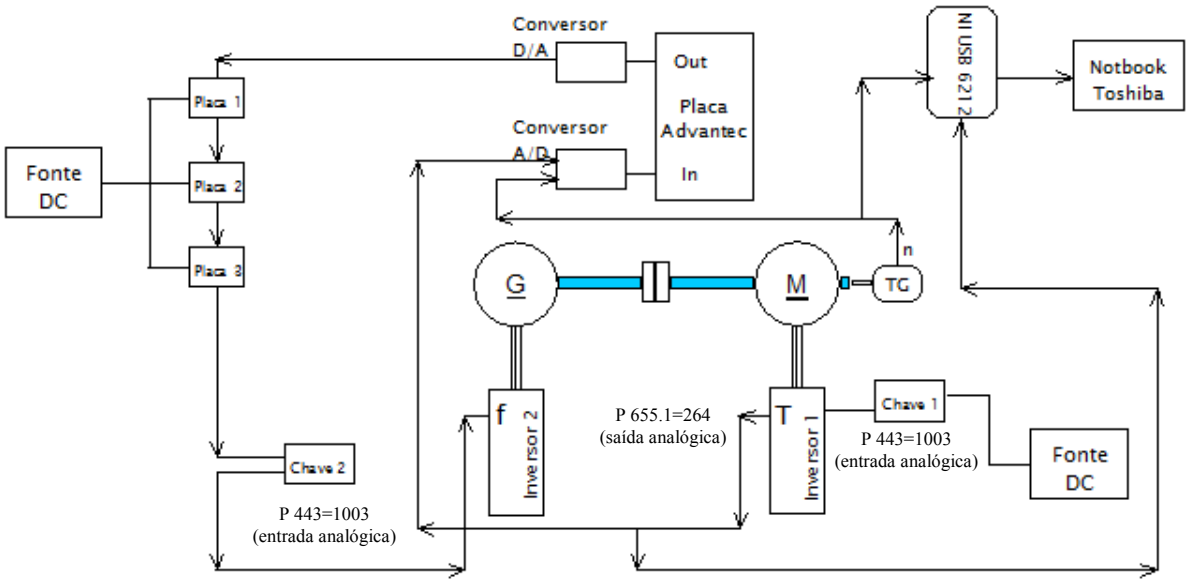


Figura 2.5 – Arquitetura operacional

Placa 1: consiste de dois amplificadores operacionais interligados, ambos com a saída invertida com relação a entrada. Como pode ser observado na Figura 2.6, o primeiro amplificador aplica um ganho unitário ao sinal de entrada. Já o segundo amplificador permite ganho variavél, com ajuste através do potênciometro de 15 [kΩ].

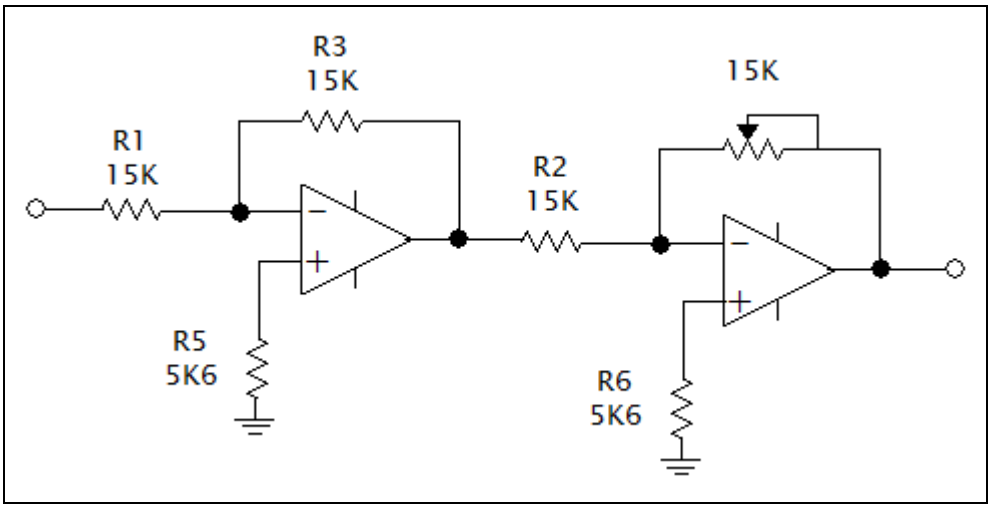


Figura 2.6 – Placa 1

Placa 2: como pode ser observado a Figura 2.7, é um aplificador *buffer*. Este tipo de amplificador apresenta ganho unitário e permite um ótimo acoplamento de impedâncias, principalmente quando há necessidade de conectar um estágio com alta impedânia de saída a uma carga de baixa impedância.

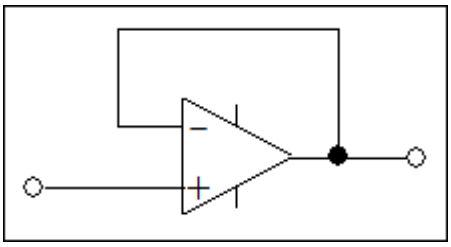


Figura 2.7 – Placa 2

Placa 3: esta última placa consiste de um aplificador operacional com saída conectada na base de um transistor NPN, como pode ser observado na Figura 2.8. A figura 2.9 mostra o circuito final, com as três placas interligadas.

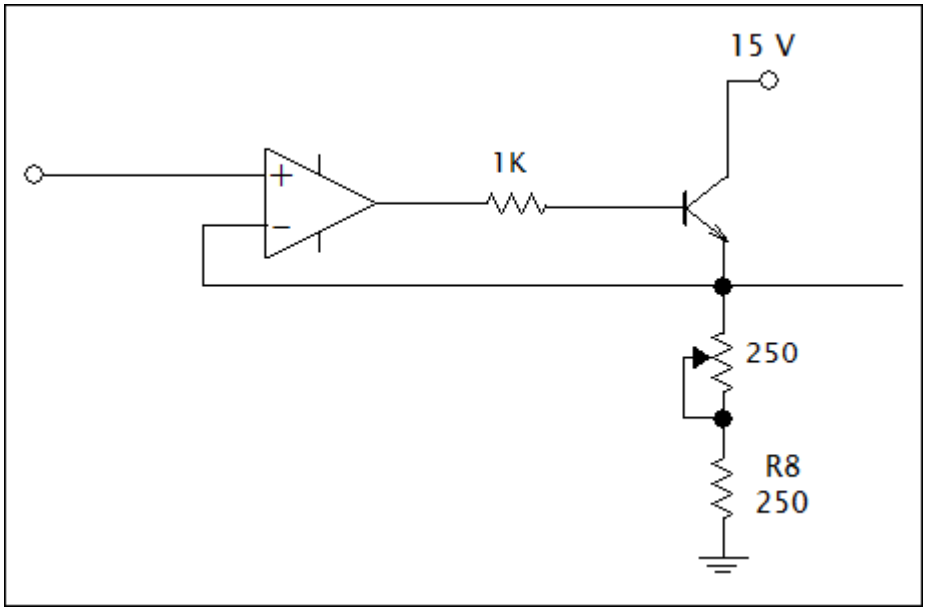


Figura 2.8 – Placa 3

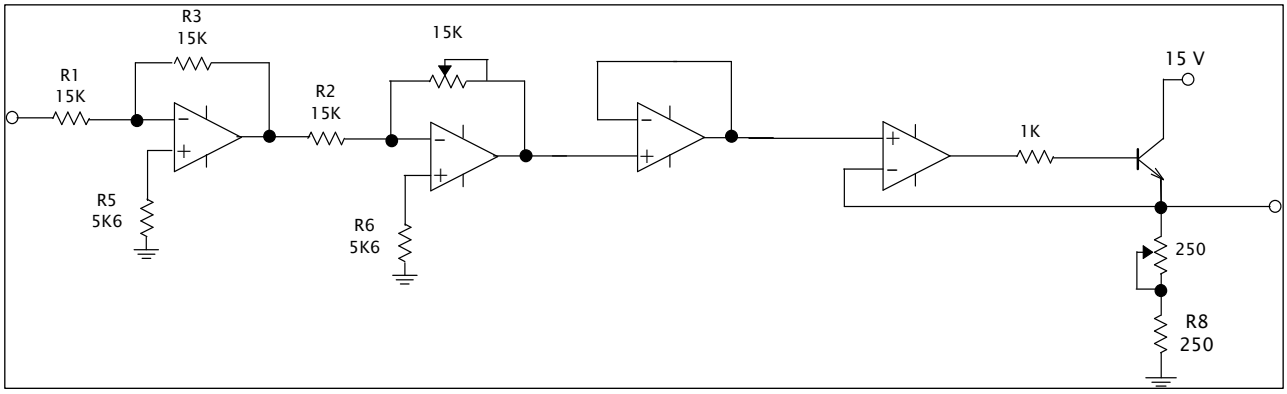


Figura 2.9 – Placa 4

Foi necessária a ligação destas placas devido à perda de tensão que ocorria quando era feita a ligação direta, causado pela baixa impedância da entrada analógica do inversor de frequência. O circuito da figura 2.9 tem, portanto, como finalidade entregar na saída o mesmo valor de tensão recebido na entrada, sendo possível realizar o ajuste no potênciometro da Placa 1.

Chave 1: esta chave permite o controle manual de velocidade do motor. Como pode ser observado na Figura 2.10, é aplicada uma tensão DC de 15 [V] em seus terminais de entrada, por divisor de tensão a saída fica limitada em 10 [V], podendo variar deste valor de tensão até 0 [V], ajustando-se o potênciometro. Sua saída é conectada à entrada analógica do Inversor 1.

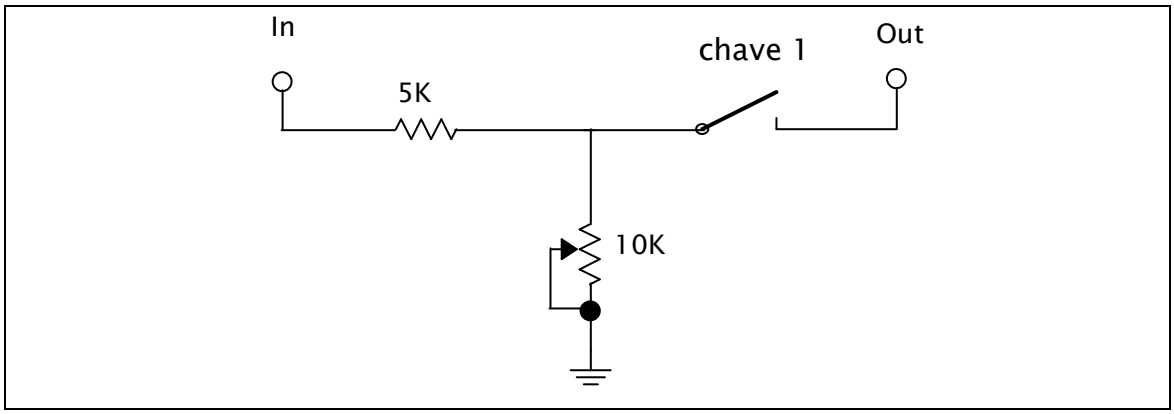


Figura 2.10 – chave 1

Chave 2: permite escolher entre o controle manual ou automático de rotação do gerador. Na operação manual, funciona exatamente como a Chave 1, recebendo tensão de 15 [V] da Fonte DC e entregando tensão variável de 0 a 10 [V]. Na operação como automático o sinal vindo da Placa 3 é entregue em sua saída. Portanto, opera como chave seletora (manual/desligado/automático), como pode ser observada na Figura 2.11.

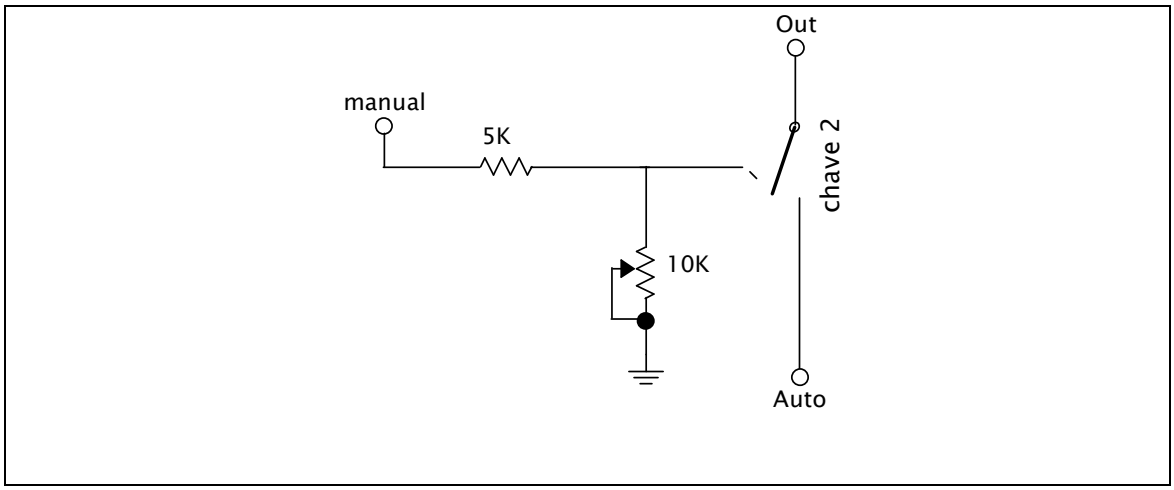


Figura 2.11 – Chave 2

2.4 – ALGORITMO EM LINGUAGEM C ++

Foi desenvolvido um algoritmo computacional em linguagem C++, que recebe os sinais de velocidade e de torque através de placa de aquisição de dados dedicada PCL 711 da Advantech, instalada em slot livre do PC, a qual possui entradas analógicas (conversores A/D) e uma saída analógica (conversor D/A), no algoritmo realizam-se os cálculos e gera um terceiro sinal que, após passar pelo conversor D/A, aciona o Inversor 2.

A placa utilizada é uma placa PCL-711B PC-Multilab Card da Advantech Co, com conversão A/D, e D/A, de entradas e saídas digitais. A conversão A/D, tem resolução de 12 bits, com 8 canais de entrada, programáveis para faixas de entrada de ±5 [V], ±2,5 [V], ±1,25 [V], ±0,625 [V], ±0,3125 [V], com tempo de conversão de até 25[μs]. A conversão D/A, tem a mesma resolução (12 bits), mas apenas com um canal de saída, com tempo de acomodação de 30 [μs], e faixas de saída de 0 a +5 [V] ou 0 a +10 [V].

O algoritmo computacional implementado em C++ permite fazer a escolha de simulução de três tipos de cargas: carga tipo torque constante, tipo torque linear e tipo torque quadrático. Estas cargas são escolhidas no algoritmo em "Menu". Este algoritmo encontra-se no Apêndice A.

Torque Constante (Menu 1): Neste caso, o algoritmo gera, para qualquer rotação do motor, um valor de tensão capaz de manter a diferença de frequência sempre a mesma. A diferença de frequência (f_M - f_G) depende do torque, que é escolhido via algoritmo, pela variável "*fc*"(fator de carga), *"fc"=1* significa torque nominal, *"fc"=0* significa torque nulo. A tensão de saída do microcomputador, em PU, é calculada através da Equação 2.6.

$$UOut = (1 - sn \times fc) \times nrum \qquad 2.6$$

Onde:

UOut: tensão de saída da placa Advantec, instalada no microcomputador, em PU;

sn: escorregamento nominal do motor;

fc: fator de carga;

nrum: rotação média do motor em PU.

Desta forma, fica garantido que a tensão aplicada na entrada analógica do Inversor 2 seja sempre inferior à tensão aplicada na entrada analógica do Inversor 1. A diferença de tensões é dada pelo produto do fator de carga pelo escorregamento, ou seja, quanto maior o fator de carga, maior é a diferença, garantindo um torque maior. Como pode ser observado na Equação 2.6, quando há variação da rotação as duas tensões variam com a mesma proporção, dado que a tensão na entrada do Inversor 1 é proporcional a rotação.

Torque linear (Menu: 2): Nesta carga, a diferença entre as frequências não se mantem constante. Portanto, a diferença entre as tensões aplicadas nas entradas analógicas dos dois inversores é variável. Esta variação é proporcinal a rotação. Aqui, dois tipos de cargas podem ser simuladas, cargas que têm torque linear crescente com a velocidade ou torque linear decrescente com a velocidade. A tensão de saída da placa instalada no microcomputador é calculada de acordo com a Equação 2.7.

$$UOut = (1 - sn \times fc \times nrum) \times nrum \qquad 2.7$$

Assim, fica garantido que a diferença entre a tensão de entrada de controle analógio do Inversor 1 e o Inversor 2 seja proporcional a rotação do eixo do motor.

Torque quadrático (Menu 3): Neste tipo de operação, a diferença entre as frequências deve variar com o quadrado da velocidade. Portanto, a diferença entre a tensão na entrada do controle analógico do Inversor 1 e do Inversor 2 também deve variar proporcionalmente ao quadrado da velocidade, como mostra a Equação 2.8.

$$UOut = (1 - sn \times fc \times nrum^2) \times nrum \qquad 2.8$$

Uma outra parte do algoritmo implementado, denominada de “Ajuste fino”, que pode ser acionada ou não – via teclado – realiza uma correção no sinal de saída da tensão (VOut), caso seu valor esteja proporcionando um torque com erro superior a um valor permitido, escolhido de acordo com os critérios de cada simulação, em relação a um torque de referência calculado. As Equações 2.9, 2.10 e 2.11 mostram como é calculado os torques de referência para cada tipo de carga.

$$tr = fc \qquad 2.9$$

$$tr = fc \times nrum \qquad 2.10$$

$$tr = fc \times nrum^2 \qquad 2.11$$

Onde

tr: Torque de referência

Assim, a partir do torque de referência calculado para cada tipo de carga, o algoritmo realiza um ajuste, de modo que a medida de torque vinda do inversor seja tão proxima do valor de referência quanto desejado.

A seguir na figura 2.12 apresenta-se o fluxograma do algoritmo

LEGENDA

Velocidade(pu) – nru
Velocidade média (pu) – nrum
Tensão de saída – Vout (0 a 10 V)
Referencia de torque (pu) – tr
Torque real (pu) – treal
Torque médio (pu) – mtrealu
Ganho – kpt
Ajuste fino – 0 – Não
1 – Sim
Ajuste fino variável – ajf
Erro torque (pu) – terr
NA – numero de amostras para cálculo da média de torque e velocidade
Fator de carga – fc
Constante de torque - kt
Máximo torque – 1 pu – 22 NM
Escorregamento nominal – sn
Tolerância – erro de torque – mnerr (pu)

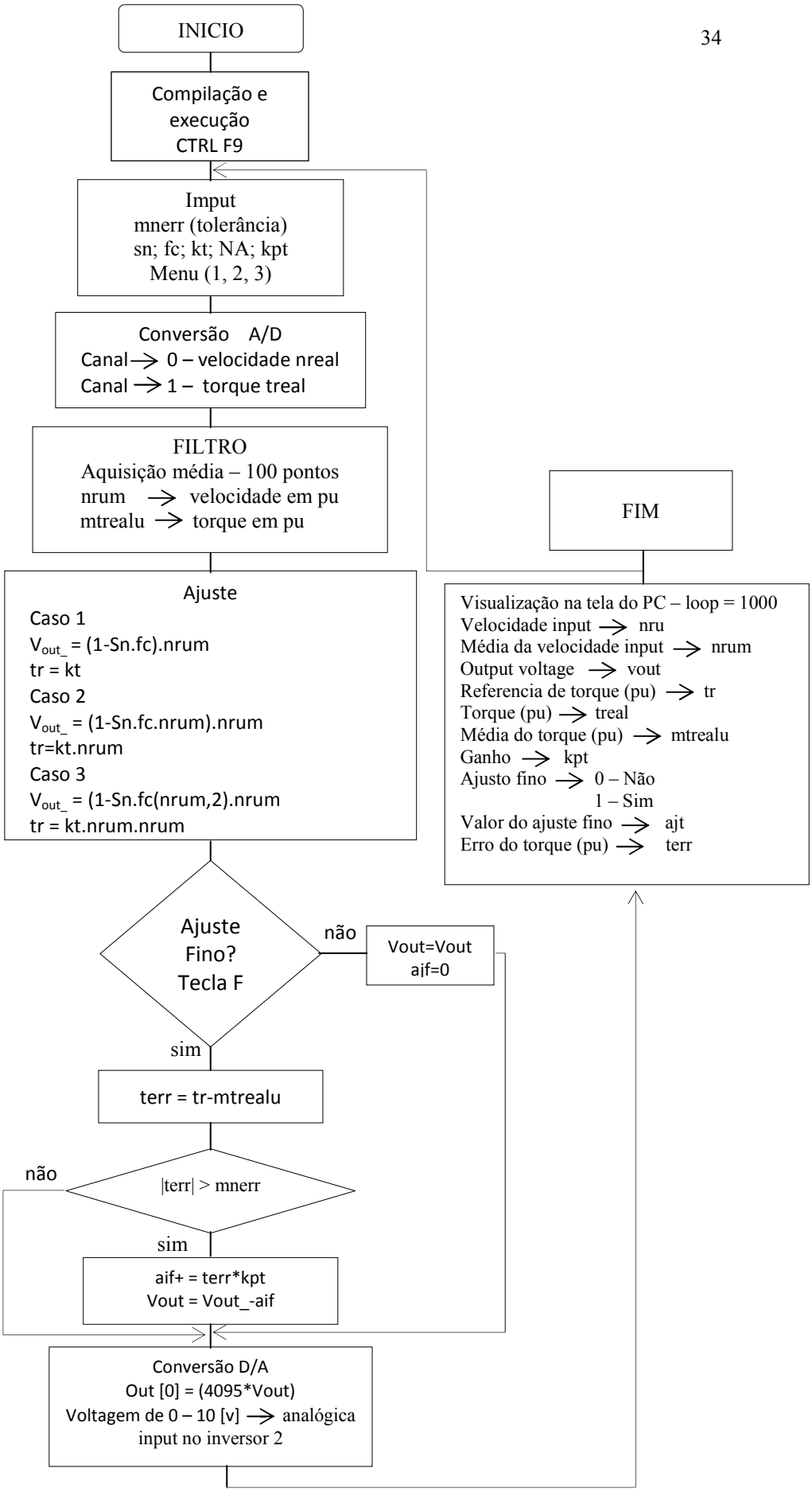


Figura 2.12 – Fluxograma do algoritmo em linguagem C++

2.5 - ALGORITMO LABVIEW

O LABVIEW é uma linguagem de programação gráfica baseado em fluxo de dados em vez de linhas de comandos. Ele foi criado para ser utilizado principalmente para aquisição e processamento de sinais. Por isso e por ser de rápida implementação, este algoritmo foi utilizado para fazer as aquisições e exibições dos resultados da bancada montada no LEPCH.

Para se ler os sinais elétricos foi necessário a utilização de um conversor A/D. Como conversor A/D utilizou-se o NI USB-6212 que é uma DAQ (data aquisition), que possui vários conversores internos. Após a captura do sinal ele foi processado para se retirar a informação realmente útil, ou seja, a média da tensão dos canais de entrada, um valor proporcional a rotação do eixo do motor e outro proporcional ao torque.

No painel frontal (Figura 2.13), encontra-se três gráficos e alguns botões de controle. Os gráficos nomeados de “sem filtro” e “filtrado” exibem o valor medido do torque e rotação ao longo do tempo, porém um deles mostra o valor lido sem processamento algum. Já o gráfico à direita mostra o valor do torque em relação à rotação, como descrito em cada um de seus eixos. Abaixo dos gráficos os indicadores “rotação” e “torque” indicam o valor médio de suas respectivas medidas. Por fim, os dois campos que podem ser preenchidos servem para indicar o local e o nome do arquivo onde os dados serão salvos.

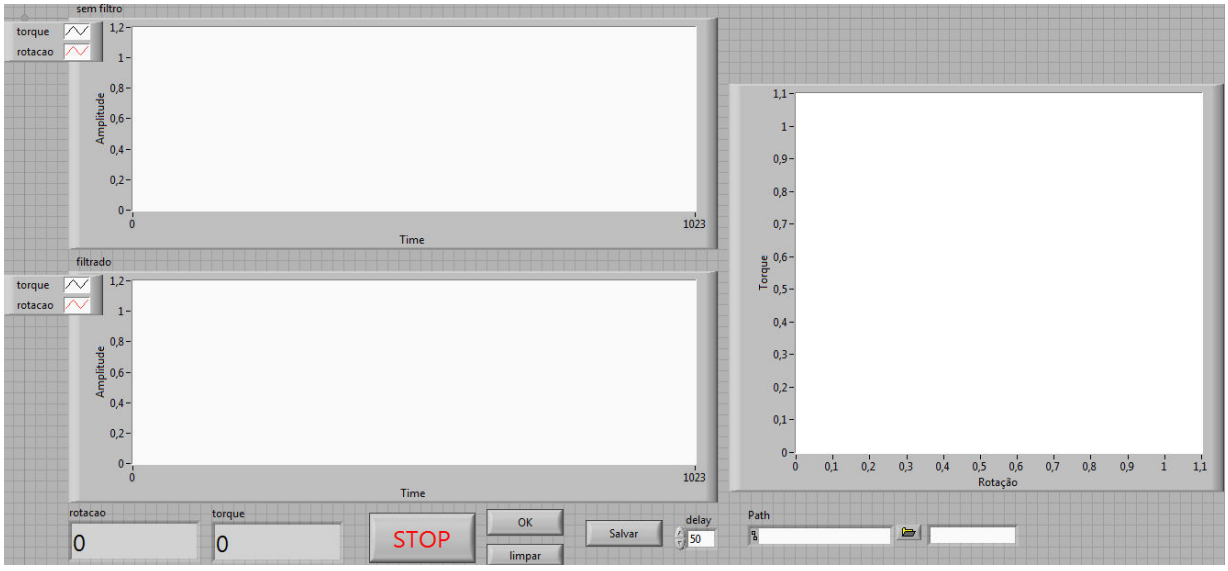


Figura 2.13 – Painel frontal do LABVIEW

A primeira parte do algoritimo cria os canais de comunicação entre o LABVIEW e a DAQ. Em seguida o sinal é capturado e exibido nos gráficos, um deles é apenas exibido e o outro é tratado. O tratamento utilizado é a média móvel, que cria uma janela de 100 pontos e vai "correndo" enquanto calcula a média desses pontos, obtendo-se um sinal com quase nenhum ruído. Já a outra parte do algoritmo é utilizada para se fazer os testes com o motor já estabilizado em vez de estudá-lo na partida. O algoritmo plota um ponto cada vez que o usuário clica em OK. Este ponto é incluso no gráfico da direita e os dados podem também serem salvos em um arquivo txt em forma de tabela quando se clica em salvar e os campos estiverem preenchidos com valores válidos.

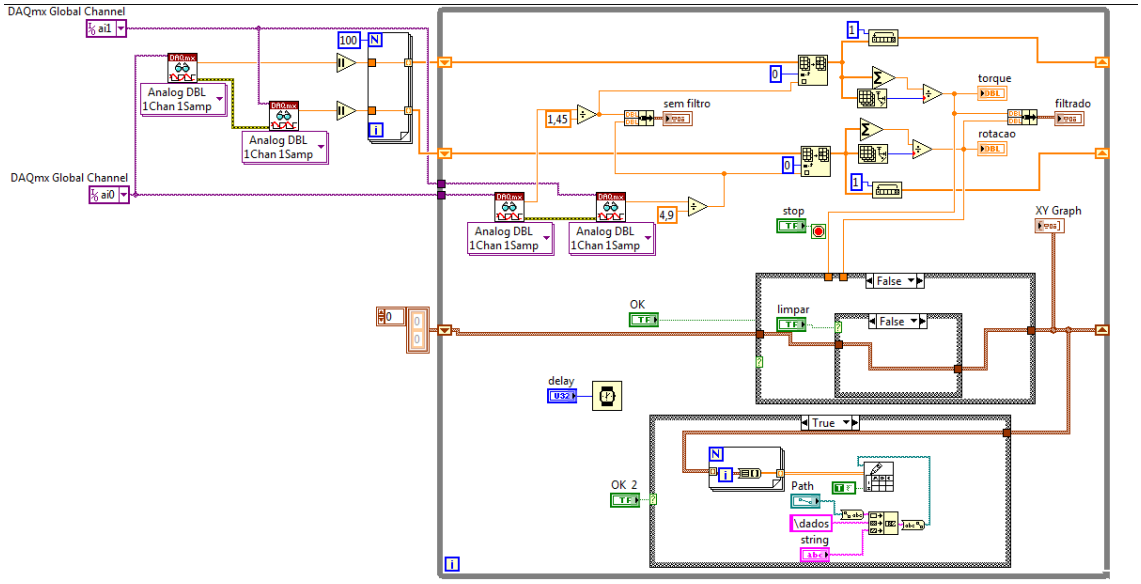


Figura 2.14 – Diagrama de bloco do LABVIEW

2.6 – MONTAGEM DA ARQUITETURA BÁSICA

A bancada foi montada com dois motres de Indução trifásicos (Figura 2.15), potência de 10 [cv], acoplados no mesmo eixo através de correias e posteriormente substituadas por acoplamentos flexível. Ambos os motores são alimentados por Inversor de frequência PWM (Figura 2.16) do fabricante Siemens.

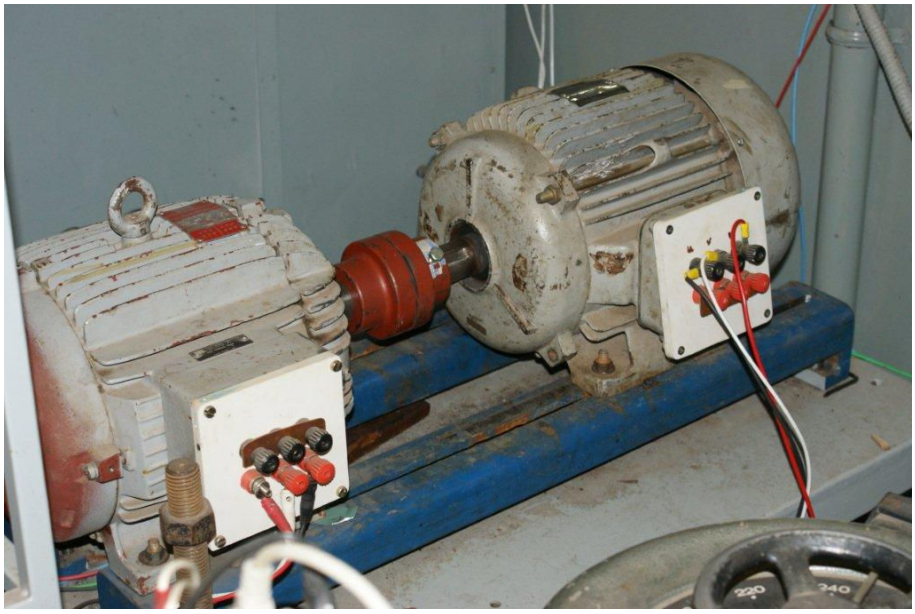


Figura 2.15 – Acoplamento dos motores

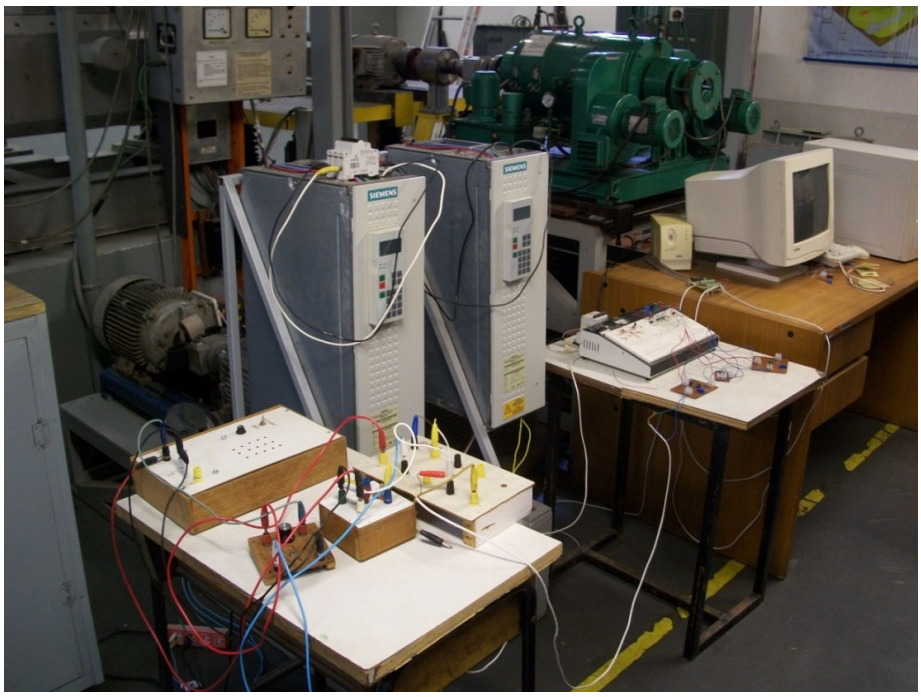


Figura 2.16 – Inversores PWM Siemens

CAPITULO 3 – RESULTADOS E DISCUSSÃO

3.1 – RESULTADOS COM COLETA DE DADOS

Conforme mostram as Tabelas 3.1 e 3.2, fez-se a coleta de dados com a opção "ajuste fino" ativado e dasativado, para efeito de comparação. Segue abaixo as tabelas com os valores coletados para os três tipos de cargas já definidas anteriormente para serem simuladas, torque constante, torque linear e torque quadrático.

Tabela 3.1 – Torques obtidos para a "Condição de Ajuste Fino Ativado"

Torque Constante		**Torque Linear**		**Torque Quadrático**	
Rotação [pu]	**Torque [pu]**	**Rotação [pu]**	**Torque [pu]**	**Rotação [pu]**	**Torque [pu]**
0,061	0,591	0,044	0,008	0,170	0,015
0,151	0,602	0,087	0,060	0,252	0,039
0,240	0,602	0,173	0,104	0,338	0,081
0,328	0,609	0,251	0,137	0,424	0,111
0,410	0,607	0,335	0,203	0,507	0,168
0,499	0,607	0,420	0,241	0,589	0,202
0,583	0,592	0,505	0,317	0,673	0,300
0,666	0,608	0,505	0,317	0,760	0,337
0,752	0,615	0,583	0,345	0,841	0,434
0,838	0,582	0,674	0,401	0,927	0,518
0,923	0,603	0,755	0,457	1,012	0,617
1,011	0,604	0,840	0,517		

Tabela 3.2 – Torques obtidos para a “Condição de Ajuste Fino Desativado”

Torque Constante		**Torque Linear**		**Torque Quadrático**	
Rotação [pu]	**Torque [pu]**	**Rotação [pu]**	**Torque [pu]**	**Rotação [pu]**	**Torque [pu]**
0,164	0,304	0,176	0,108	0,175	0,024
0,243	0,374	0,248	0,146	0,254	0,080
0,330	0,437	0,335	0,146	0,338	0,045
0,412	0,438	0,420	0,137	0,420	0,022
0,494	0,477	0,506	0,203	0,505	0,084
0,579	0,484	0,588	0,228	0,588	0,126
0,662	0,481	0,671	0,296	0,677	0,112
0,752	0,432	0,754	0,325	0,765	0,139
0,839	0,446	0,835	0,376	0,841	0,161
0,925	0,368	0,926	0,471	0,924	0,214
1,002	0,482	0,176	0,108	1,007	0,496
0,164	0,304	0,248	0,146		

3.2 – ANÁLISE DE DADOS

A seguir será feita a análise dos dados coletados, para cada tipo de carga selecionada no “Menu”, a partir das equações desenvolvidas e implementadas no algoritmo em C++ que se encontra no Apêndice A.

3.2.1 – Torque constante – ajuste fino ativado

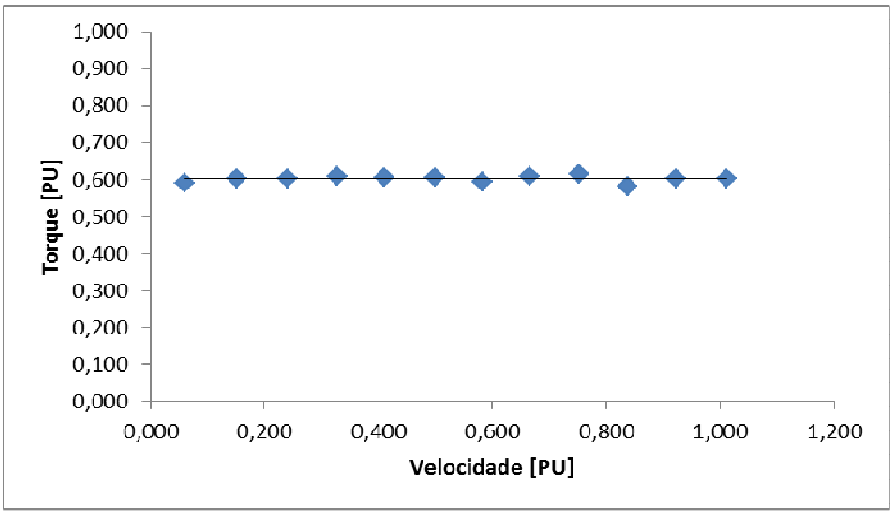


Figura 3.1 – Torque x velocidade – Ajuste fino *ON*

Tabela 3.3 - Equação Y = A + B.x (Figura 3.1)

Parametros	Valores
A	0,6011
R^2	0,9763

3.2.2 – Torque constante – ajuste fino desativado

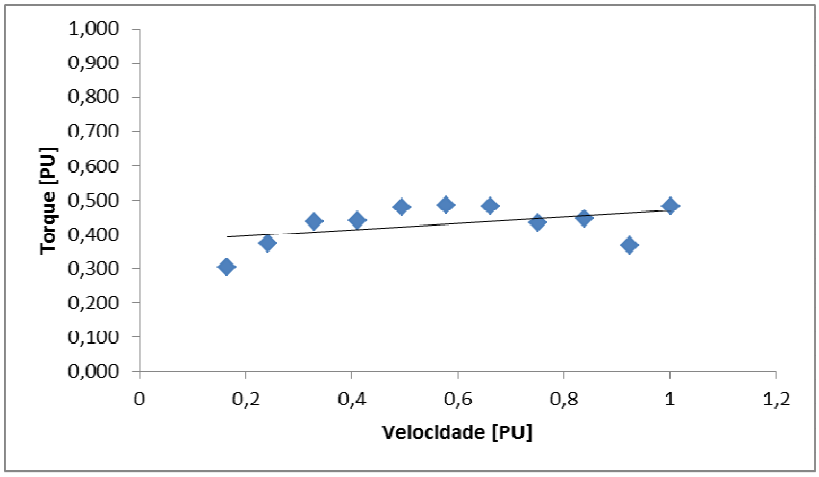


Figura 3.2 – Torque x velocidade – Ajuste fino *OFF*

Tabela 3.4 - Equação Y = A + B.x (Figura 3.2)

Parametros	Valores
A	0,3750
R^2	0,2035

Pode-se observar que com a função ajuste fino ativada o toque permanece praticamente constante, pode-se considerar como uma reta paralela ao eixo das abscissas, evidênciado a condição de torque constante.

Com a função ajuste fino desativada, como pode ser observado na Figura 3.2, o parãmetro B da equação da reta apresenta valor sifnificante, tornando o torque maior com o aumento da velocidade, não evidenciando uma situação de torque constante.

3.2.3 – Torque linear – ajuste fino ativado

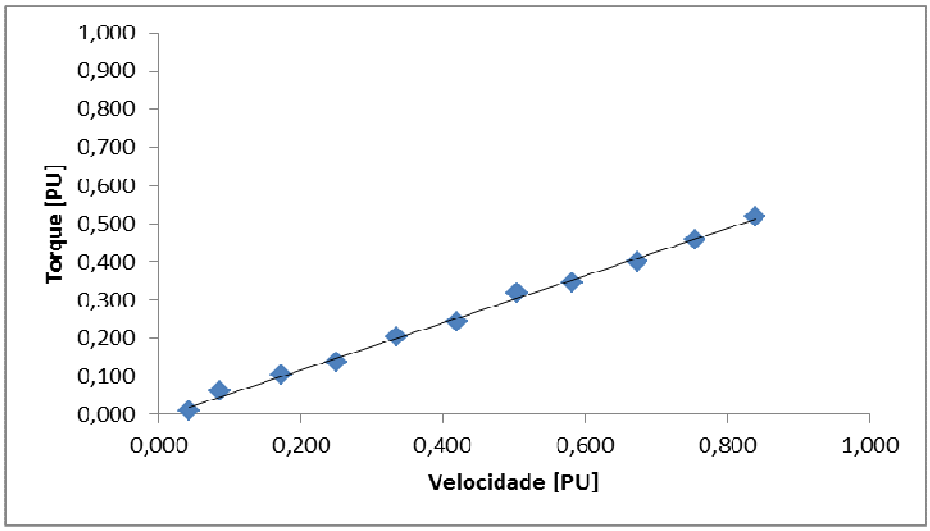


Figura 3.3 – Torque x velocidade – Ajuste fino *ON*

Tabela 3.5 - Equação Y = A + B.x (Figura 3.3)

Parametros	Valores
B	0,6187
R^2	0,9963

3.2.4 – Torque linear – ajuste fino desativado

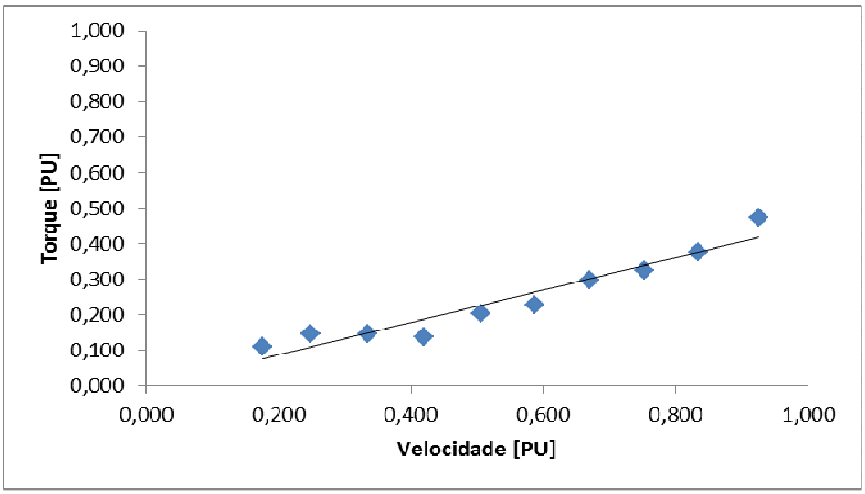


Figura 3.4 – Torque x velocidade – Ajuste fino *OFF*

Tabela 3.6 - Equação Y = A + B.x (Figura 3.4)

Parametros	Valores
B	0,4554
R^2	0,9241

Na opção ajuste fino pode-se observar que o parâmetro B tem valor muito próximo ao fator de carga escolhido (fc=0,6), evidênciando a caracteristica linear do torque.

Na operação em baixa rotação pode-se verificar que ocorrem maiores variações dos pontos com relação a reta. Estas variações são reduzidas quando a rotação se aproxima da nominal.

Já na opção ajuste fino desativado o comportamento do torque também é tipicamente linear, com variações bruscas, principalmente quando a máquina opera em baixa rotação.

3.2.5 – Torque quadrático – ajuste fino ativado

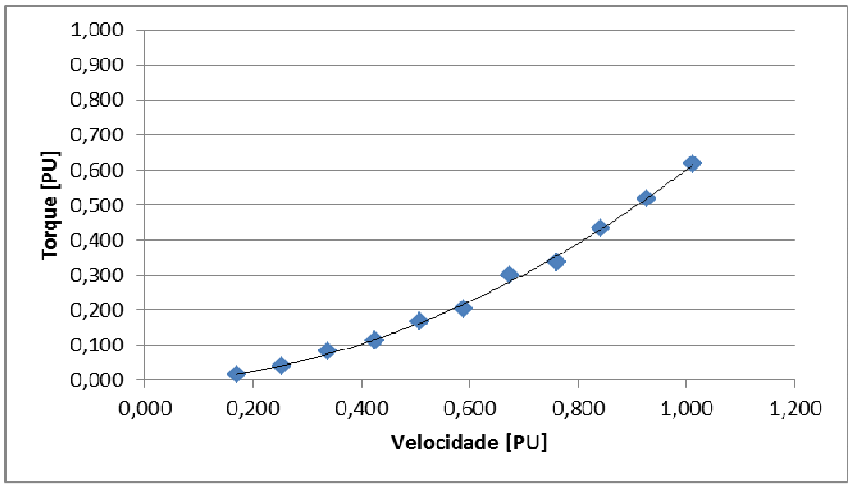


Figura 3.5 – Torque x velocidade – Ajuste fino *ON*

Tabela 3.7 - Equação $Y = A + B1.X + B2.X^2$ (Figura 3.5)

Parametros	Valores
B2	0,5541
R^2	0,9975

3.2.6 – Torque quadrático – ajuste fino desativado

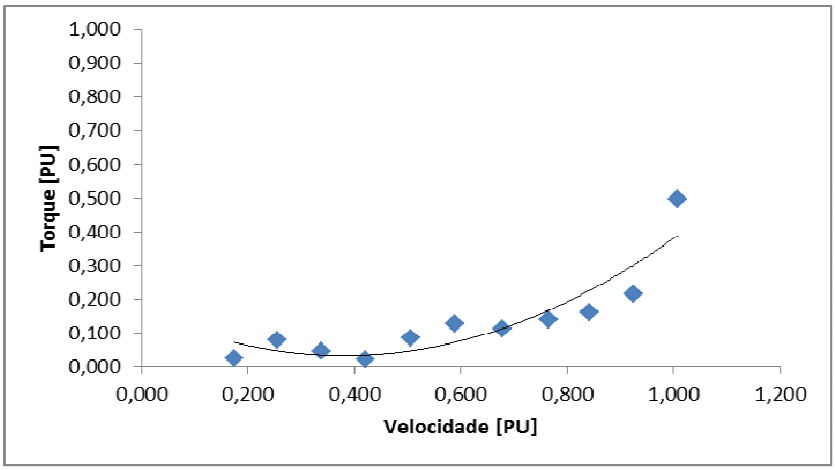


Figura 3.6 – Torque x velocidade – Ajuste fino *OFF*

Tabela 3.8 - Equação $Y = A + B1.X + B2.X^2$ (Figura 3.6)

Parametros	Valores
B2	0,9073
R^2	0,8202

Para a opção ajuste fino ativado o parâmetro A da equação do segundo grau é praticamente nulo, resultando em torque zero para o rotor parado. Para rotação de 1pu tem-se torque bem próximo ao fator de carga escolhido (fc=0,6). Como no caso de torque constante, os pontos na zona de baixa rotação apresentam maior distância em relação a curva ajustada, convergindo para a curva quando a rotação se aproxima da nominal.

Para a opção ajuste fino desativado houve muita oscilação dos pontos, com erros significativos para todos os parâmetros da curva. Os pontos divergem muito da característica ajustada, principalmente na zona de baixa rotação. Não representando bem o perfil de torque quadrático desejado.

Como pode-se obvervar nas análises anteriores, os resultados obtidos com a opção ajuste fino ativada foram mais satisfatórios em relação aos resultados esperados, portanto a seguir será determinada a curva Potência mec. x Velocidade apenas para os dados obtidos com ajuste fino.

3.2.7 Torque constante – Potência x Rotação

Tabela 3.9 – Torque constante

Rotação [pu]	Potência [pu]
0,061	0,036
0,151	0,091
0,240	0,144
0,328	0,199
0,410	0,249
0,499	0,303
0,583	0,345
0,666	0,405
0,752	0,462
0,838	0,488
0,923	0,557
1,011	0,611

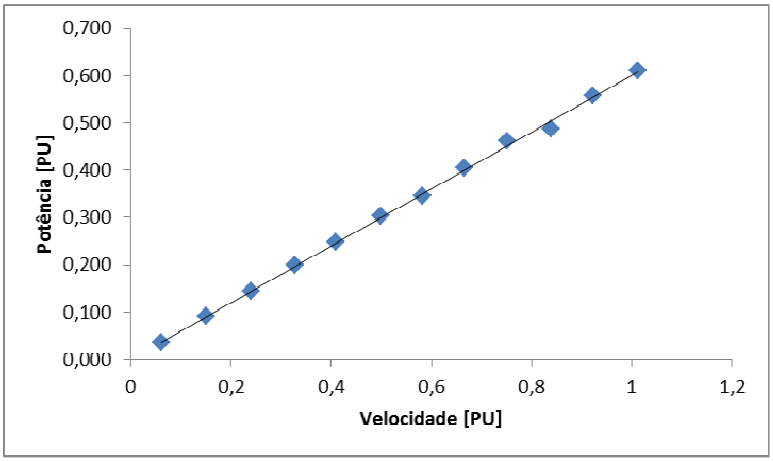


Figura 3.7 – Potência x rotação – Ajuste fino *ON*

Tabela 3.10 - Equação Y = A + B.x (Figura 3.7)

Parametros	Valores
B	0,6008
R^2	0,9989

O parâmetro B representa o próprio fator de carga, resultando na potência em p.u igual ao fator de carga para rotação nominal.

3.2.8 Torque linear – Potência x Rotação

Tabela 3.11 – Torque linear

Rotação [pu]	Potência [pu]
0,087	0,005
0,173	0,018
0,251	0,034
0,335	0,068
0,420	0,101
0,505	0,160
0,583	0,201
0,674	0,270
0,755	0,345
0,840	0,434
0,925	0,524
1,011	0,629

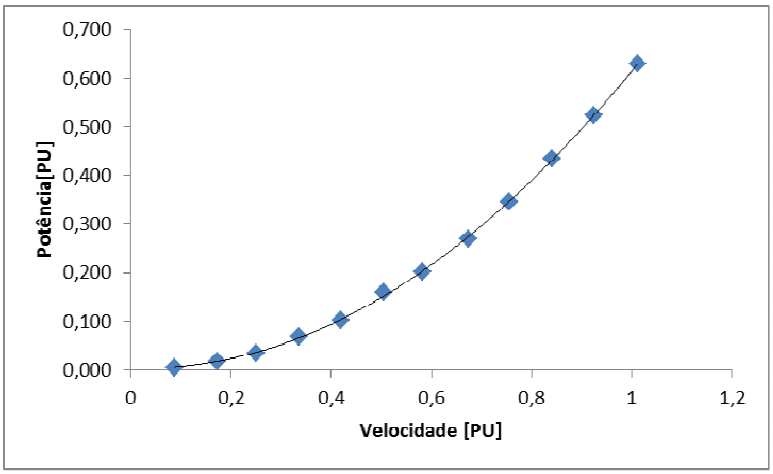


Figura 3.8 – Potência x velocidade – Ajuste fino *ON*

Tabela 3.12 - Equação $Y = A + B1.X + B2.X^2$ (Figura 3.8)

Parametros	Valores
B2	0,6413
R^2	0,9997

O parâmetro B2 é próximo ao valor do fator de carga escolhido e apresenta valor mais significante perante os demais parâmetros, evidênciando o perfil quadrático da curva

3.2.9 Torque quadrático – Potência x Rotação

Tabela 3.13 – Torque quadrático

Rotação [pu]	Potência [pu]
0,170	0,002
0,252	0,009
0,338	0,027
0,424	0,047
0,507	0,085
0,589	0,119
0,673	0,201
0,760	0,256
0,841	0,365
0,927	0,480
1,012	0,624

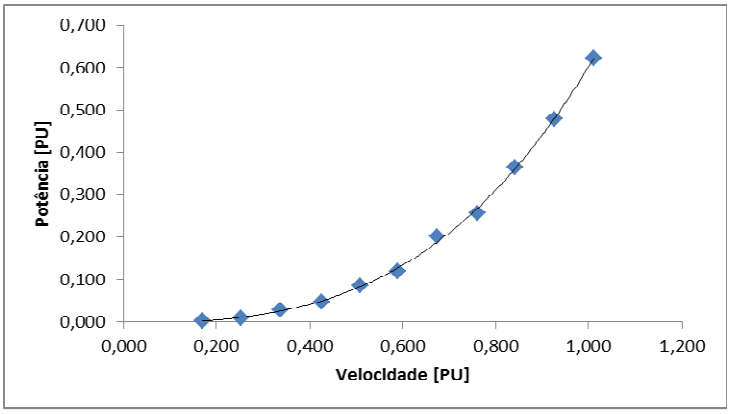


Figura 3.9 – Potência x velocidade – Ajuste fino *ON*

Tabela 3.14 - Equação $Y = A + B1.X + B2.X^2 + B3.X^3$ (Figura 3.9)

Parametros	Valores
B3	0,6085
R^2	0,9990

Neste ultimo caso, pode-se notar que o parâmetro relevante na equação do terceiro grau ajustada é B3, com os demais parâmetros apresentando valores pouco significantes em relação a B3. Assim, fica evidente a característica de variação da potência com o cubo da rotação.

3.3 – RESULTADOS OBTIDOS ATRAVÉS DA PLATAFORMA DE AQUISIÇÃO LABVIEW

3.3.1 – Aquisição dos pontos em regime estático

3.3.1.1 – Aquisição dos pontos utilizando o ajuste grosso

As figuras 3.10, 3.11 e 3.12 ilustram os resultados obtidos utilizando-se apenas o ajuste grosso para cargas do tipo torque constante, torque linear e torque quadrático, para fator de carga fc ou constante de torque kt igual a 0.6. Dados: torque 1[pu]= 22 [N.m] ; rotação 1[pu]= 1800 [rpm].

3.3.1.1.1 – Torque constante

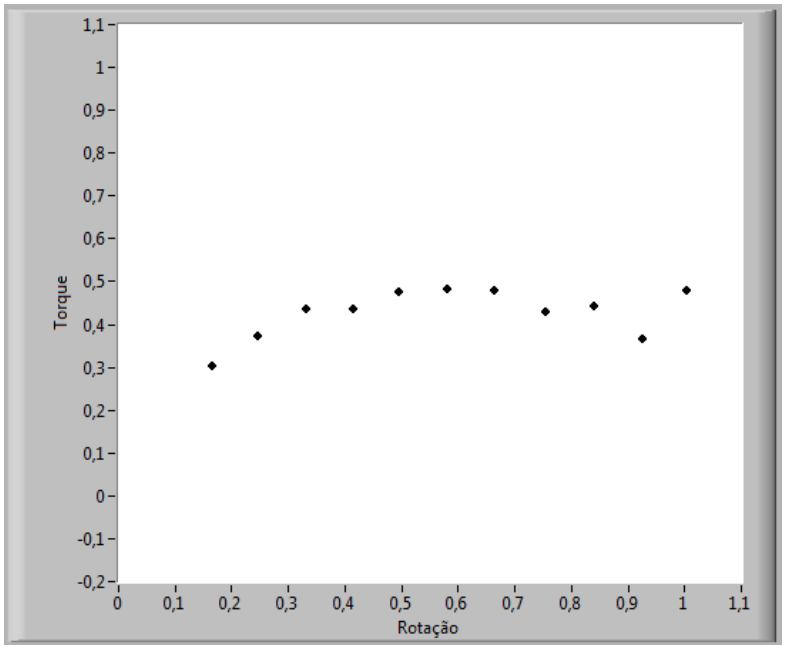


Figura 3.10 Torque versus rotação para torque constante utilizando-se apenas ajuste grosso. Dados: torque 1 [pu]= 22 [N.m] ; rotação 1 [pu]= 1800 [rpm].

3.3.1.1.2 – Torque linear

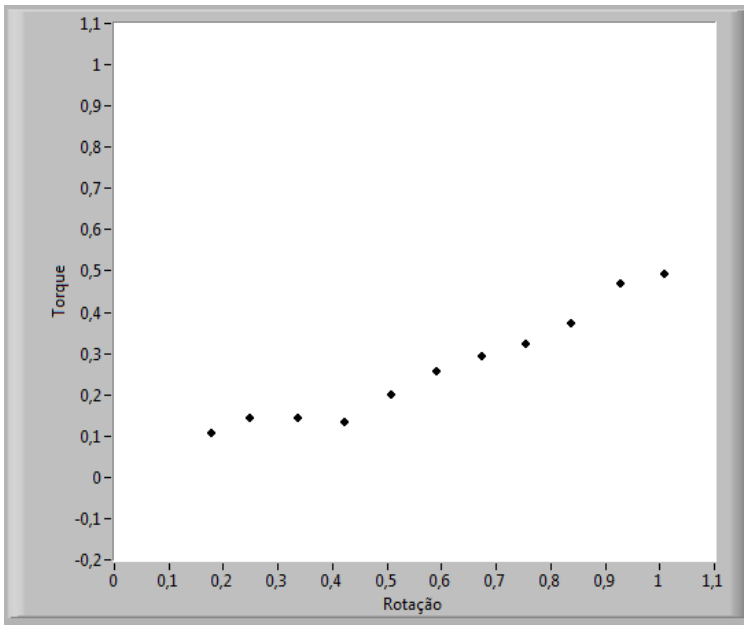


Figura 3.11 Torque versus rotação para torque linear com a rotação utilizando-se apenas ajuste grosso. Dados: torque 1 [pu]= 22 [N.m] ; rotação 1 [pu]= 1800 [rpm].

3.3.1.1.3 – Torque quadrático

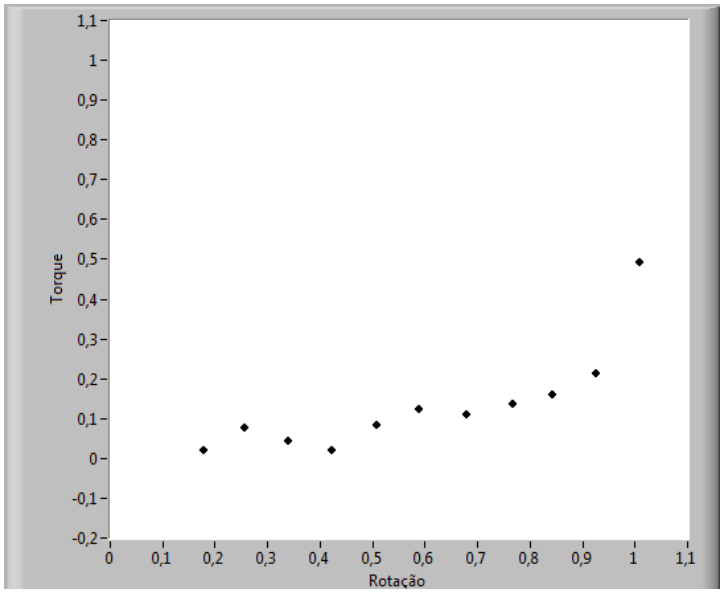


Figura 3.12: Torque versus rotação para torque quadrático com a rotação utilizando-se apenas ajuste grosso. Dados: torque 1 [pu]= 22 [N.m] ; rotação 1 [pu]= 1800 [rpm].

3.3.1.2 – Aquisição dos pontos utilizando o ajuste fino

As figuras 3.13, 3.14 e 3.15 ilustram os resultados obtidos utilizando-se também o ajuste fino para cargas do tipo torque constante, torque linear e torque quadrático, para fator de carga fc ou constante de torque kt igual a 0.6. Dados: torque 1 [pu]= 22 [N.m] ; rotação 1 [pu]= 1800 [rpm].

3.3.1.2.1 – Torque constante

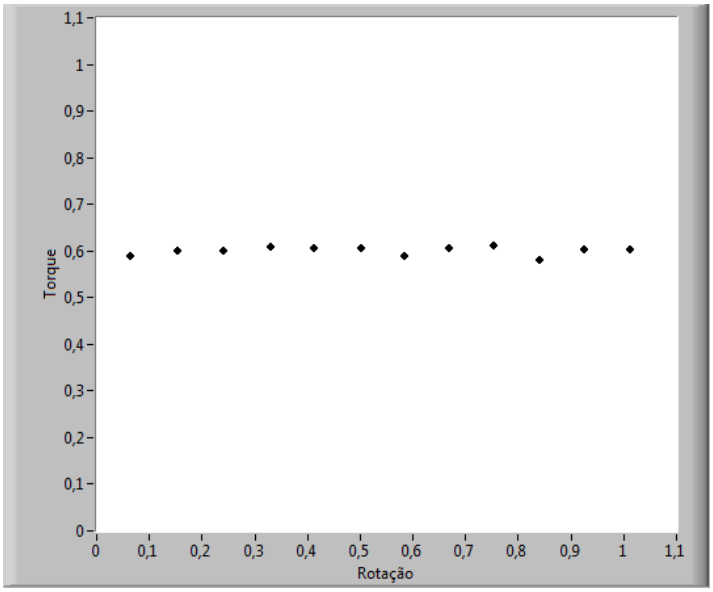


Figura 3.13: Torque versus rotação para torque constante com a rotação utilizando-se ajuste fino. Dados: torque 1 [pu]= 22 [N.m] ; rotação 1 [pu]= 1800 [rpm]

3.3.1.2.2 – Torque linear

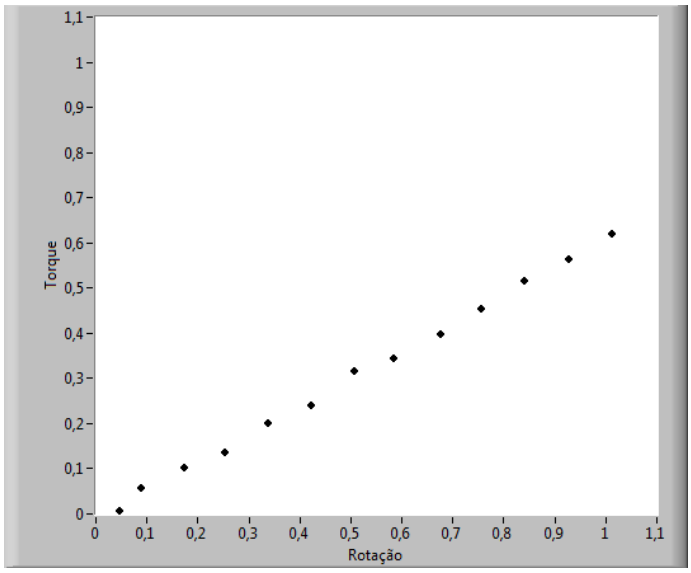


Figura 3.14: Torque versus rotação para torque linear com a rotação utilizando-se ajuste fino. Dados: torque 1 [pu]= 22 [N.m] ; rotação 1[pu]= 1800 [rpm].

3.3.1.2.3 – Torque quadrático

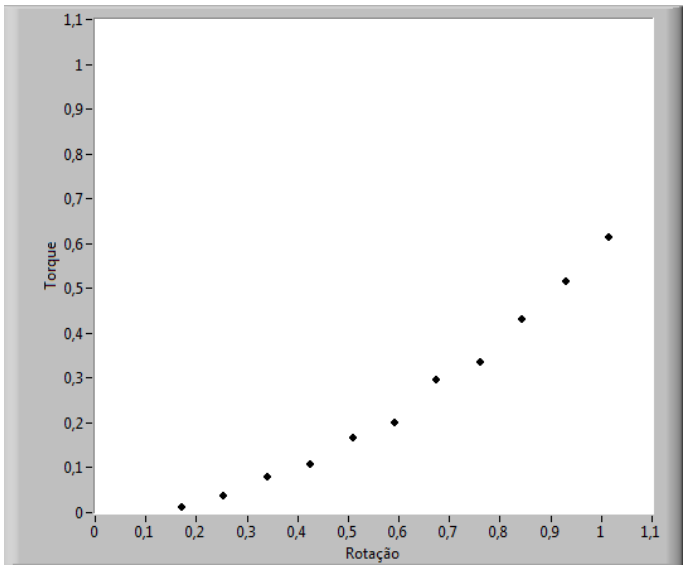


Figura 3.15: Torque versus rotação para torque quadrático com a rotação utilizando-se ajuste fino. Dados: torque 1 [pu]= 22 [N.m] ; rotação 1 [pu]= 1800 [rpm].

Pode-se observar que os resultados utilizando-se o ajuste fino foram bem melhores.

3.3.2 – Aquisição dos pontos em regime dinâmico

Nas figuras 3.16, 3.17 e 3.18 ilustram os resultados obtidos para cargas de torque constante, linear e quadrático com a velocidade, utilizando-se ajuste fino e regime dinâmico para variação da velocidade de forma linear de 300 [rpm] até 1800 [rpm] para fator de carga fc ou constante de torque kt igual a 0.6. Dados torque 1(pu)= 22[N.m] ; rotação 1[pu]= 1800 [rpm].

3.3.2.1 – Torque constante – fc = 0,6

.

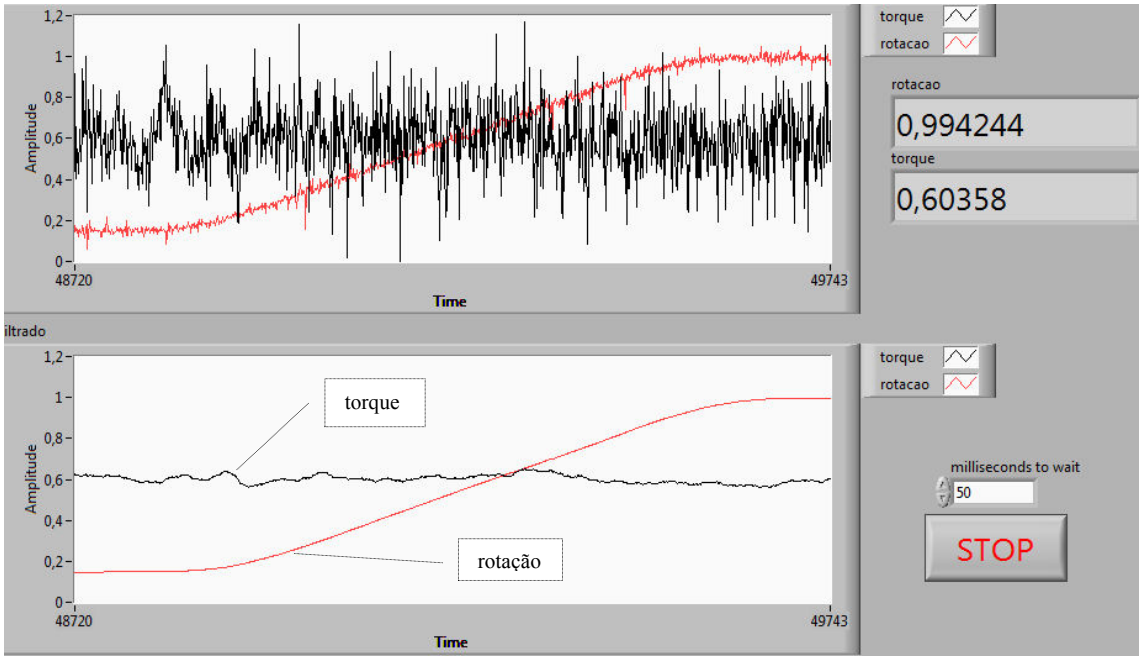


Figura 3.16: Resultados obtidos para cargas de torque constante com a velocidade, utilizando-se ajuste fino e regime dinâmico para variação da velocidade de forma linear de 300 [rpm] até 1800 [rpm]. Curva preta: torque, curva vermelha: velocidade (torque máximo 0.6 pu).

3.3.2.2 – Torque linear - fc = 0,6

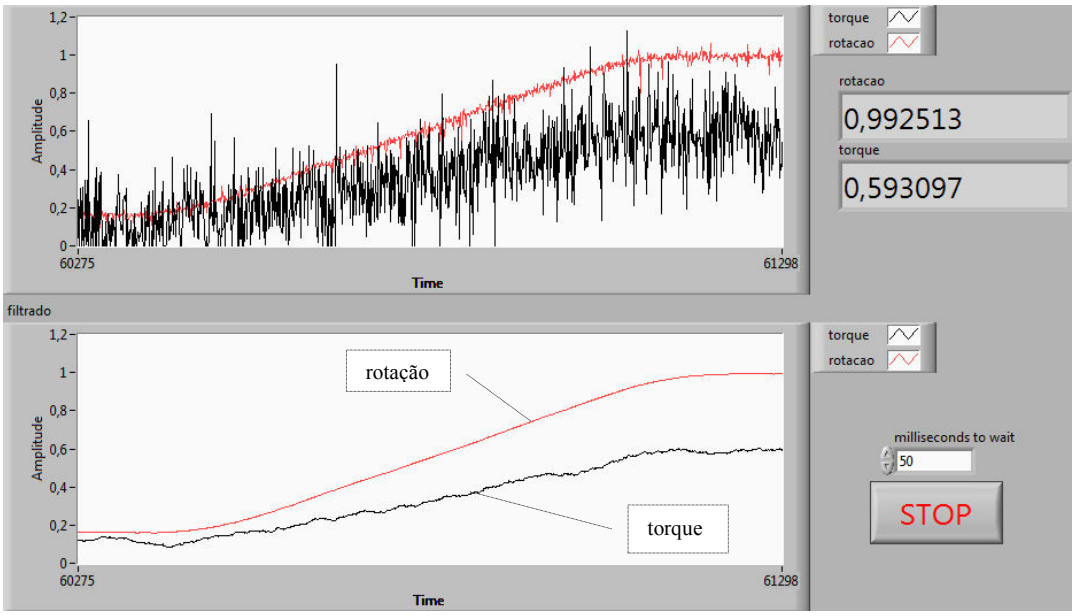


Figura 3.17: Resultados obtidos para cargas de torque linear com a velocidade, utilizando-se ajuste fino e regime dinâmico para variação da velocidade de forma linear de 300 [rpm] até 1800 [rpm]. Curva preta : torque, curva vermelha : velocidade (torque máximo 0.6 pu).

3.3.2.3 – Torque quadrático - fc = 0,6

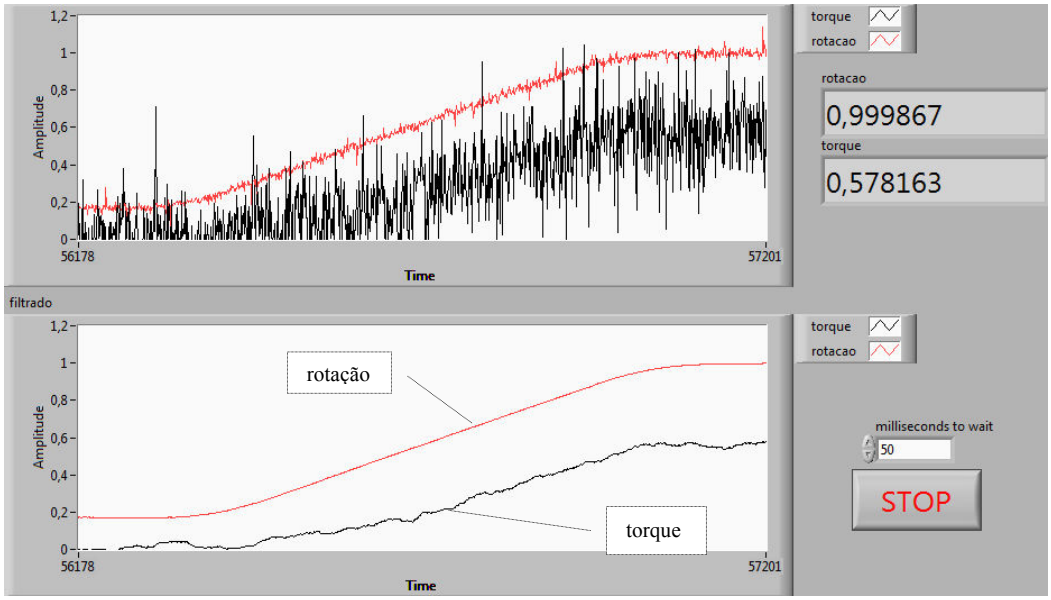


Figura 3.18 Resultados obtidos para cargas de torque quadrático com a velocidade, utilizando-se ajuste fino e regime dinâmico para variação da velocidade de forma quadrática de 300 [rpm] até 1800 [rpm]. Curva preta: torque, curva vermelha: velocidade (torque máximo 0.6 pu).

Nas figuras 3.19, 3.20 e 3.21 as curvas superiores ilustram a velocidade do taco gerador e torque do transdutor de torque do inversor PWM 1 sinais não filtrados. As curvas inferiores apresentam os mesmos gráficos para sinais filtrados. O torque máximo foi tomado igual a 0.9 pu.

3.3.2.4 – Torque constante – fc = 0,9

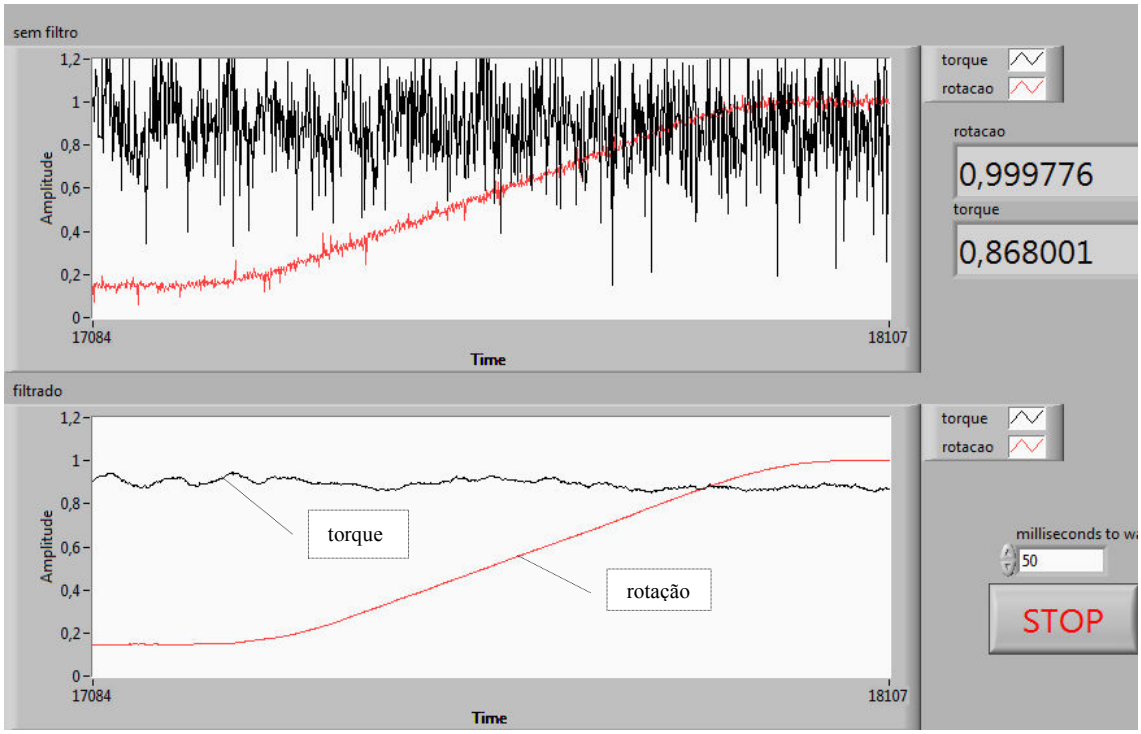


Figura 3.19 Resultados obtidos para cargas de torque constante com a velocidade, utilizando-se ajuste fino e regime dinâmico para variação da velocidade de forma linear de 300 [rpm] até 1800 [rpm]. Curva preta : torque, curva vermelha : velocidade (torque máximo 0.9 pu).

3.3.2.5 – Torque linear – fc = 0,9

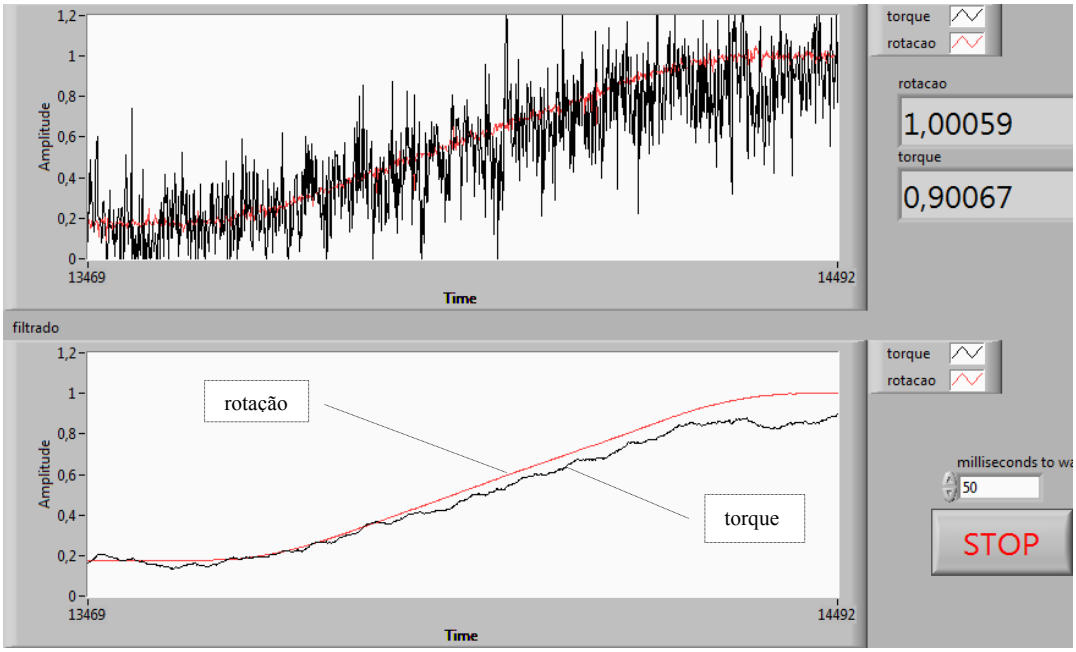


Figura 3.20: Resultados obtidos para cargas de torque linear com a velocidade, utilizando-se ajuste fino e regime dinâmico para variação da velocidade de forma linear de 300 [rpm] até 1800 [rpm]. Curva preta : torque, curva vermelha : velocidade (torque máximo 0.9 pu).

3.3.2.6 – Torque quadrático – fc = 0,9

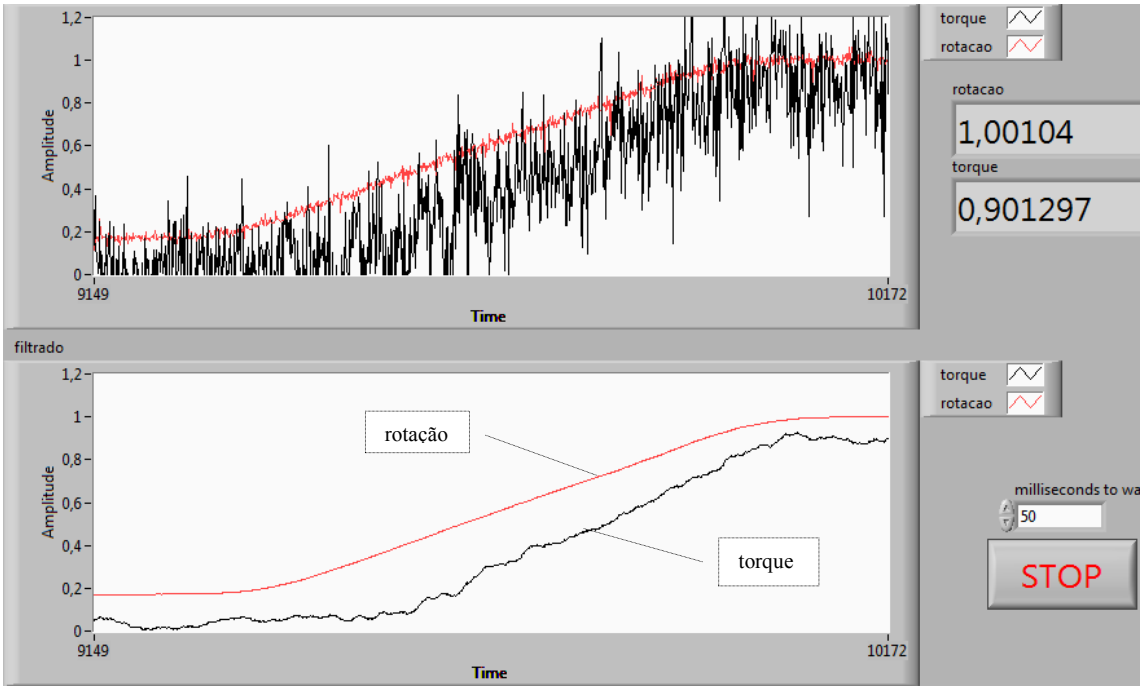


Figura 3.21: Resultados obtidos para cargas de torque quadrático com a velocidade, utilizando-se ajuste fino e regime dinâmico para variação da velocidade de forma linear de 300 [rpm] até 1800 [rpm]. Curva preta : torque, curva vermelha : velocidade (torque máximo 0.9 pu).

3.3.2.7 – Variação de torque constante para torque quadrático – fc = 0,6

A figura 3.22 ilustra as curvas de velocidade e torque em regime dinâmico para alteração do menu de torque constante para torque quadrático e posteriormente de quadrático para torque constante.

Dados: velocidade igual a 0.5 [pu] , torque constante igual a 0.6 [pu].

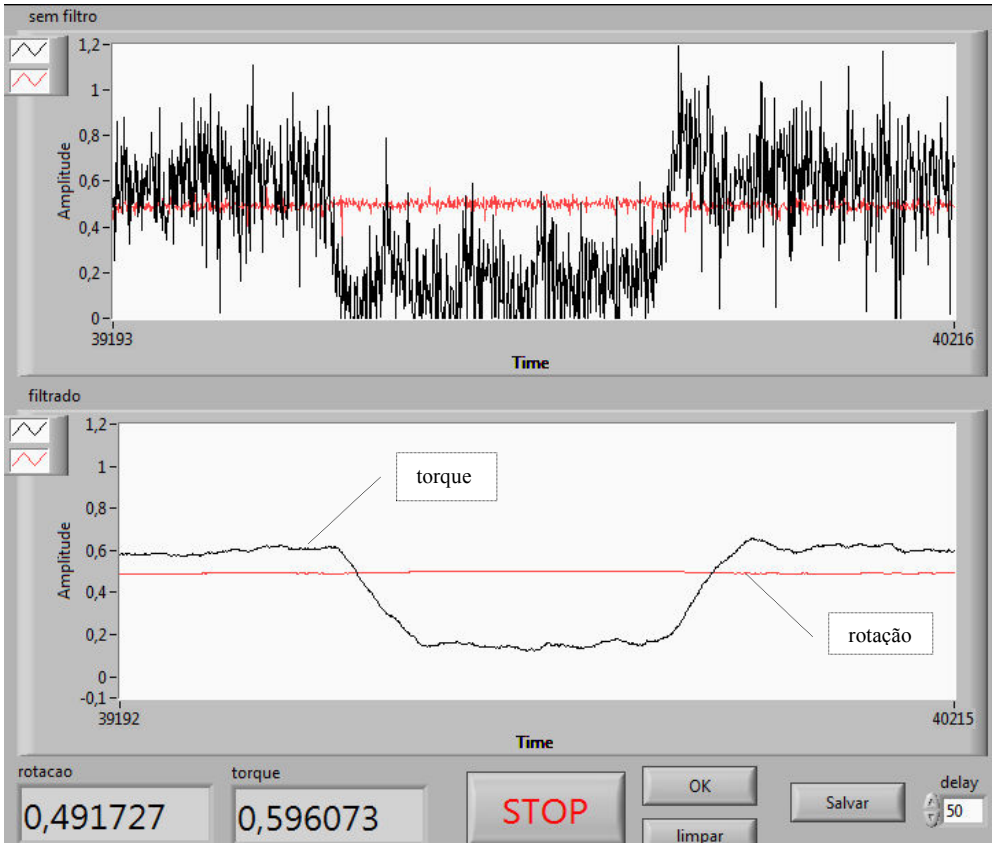


Figura 3.22: Curvas de velocidade e torque em regime dinâmico para alteração do menu de torque constante para torque quadrático e posteriormente de quadrático para torque constante. Sinais não filtrados, curvas superiores, sinais filtrados curvas inferiores.

CAPITULO 4 CONCLUSÃO

4.1 CONSIDERAÇÕES FINAIS

O sistema implantado de um simulador genérico de cargas apresentou resultados satisfatórios. Utilizou-se como transdutor de torque a estimação da corrente isq do inversor do motor, pois esta parcela é proporcional ao torque do motor.

O uso deste sistema torna possível ensaiar motores com as mais variadas cargas, como por exemplo, do tipo torque constante (correias transportadoras), do tipo torque variável com o quadrado da rotação (bombas, ventiladores e compressores), do tipo torque proporcional a rotação (calandras) ou ainda do tipo torque proporcional ao inverso da rotação (bobinadeiras), Assim, este recurso mostra-se uma contribuição importante e útil para a análise e o ensino de acionamentos elétricos.

O ajuste fino apresentou resultados mais precisos, tanto na análise estática de variação de torque com a velocidade, torque constante, torque linear e torque quadrático, conforme pode ser comprovado comparando-se os resultados das figuras 3.13, 3.14 e 3.15 (ajuste fino utilizado) com aqueles obtidos nas figuras 3.10, 3.11 e 3.12 (apenas ajuste grosso utilizado). Também em regime dinâmico de torque, para variação de velocidade ajustada linearmente na faixa de 300 a 1800 [rpm], os resultados foram satisfatórios, conforme pode-se constatar nas figuras 3.16, 3.17, 3.18, 3.19, 3.20, 3.21.

O sistema LABVIEW® de aquisição de sinais foi muito útil para filtragem dos sinais provenientes dos transdutores de velocidade e de torque, para a melhor avaliação dos resultados e do desempenho da bancada de ensaios desenvolvida.

4.2 CONTRIBUIÇÕES

Como proposto, ficou demonstrado que é possível, com o auxílio de uma lógica de controle adequada, a construção de bancadas para a realização de estudos em motores de indução trifásicos, simulando as várias condições de cargas através do uso apropriado de qualquer inversor com controle vetorial.

Isto resulta em redução de custos de engenharia e desenvolvimento, bem como permite o desenvolvimento de várias análises que podem ter por objetivo melhorar a utilização da energia na indústria. As cargas simuladas representam a grande maioria das

cargas industriais que passam a ser analisadas de forma eficiente sem a necessidade de arranjos físicos complementares.

Considera-se uma importante contribuição complementar demonstrar que é possível utilizar um transdutor de torque estimado pelo próprio inversor PWM do motor, o que facilita a implementação da técnica e reduz custos.

4.3 SUGESTÕES PARA FUTUROS TRABALHOS

Como continuação deste trabalho é possível considerar a utilização de um transdutor de torque instalado no eixo do motor para a comparação dos resultados obtidos com os da estimação de torque feita pelo inversor de frequência. Também cargas do tipo torque inversamente proporcional à rotação, caso, por exemplo, de bobinadeiras podem e devem ser consideradas em um próximo trabalho.

Por certo, também é possível considerar a utilização da Plataforma LABVIEW® e o Hardware NI USB 6212 em substituição à Placa de Aquisição PCL 711.

Alguns inversores específicos, como os utilizados - Siemens Máster - Drives possuem um recurso adicional de parametrização adequada para obtenção de torques desejados, como os do tipo que foram obtidos no trabalho (torque constante, torque linear e torque quadrático). Sugere-se então a utilização desta e de outras ferramentas similares em um próximo trabalho.

Ressalta-se, entretanto, que a proposta discutida nesta tese é aplicável para quaisquer tipos de inversores, incluindo os que não possuem recursos adicionais, pois é possível, como demonstrado elaborar-se um algoritmo dedicado para desempenhar funções básicas de controle.

REFERÊNCIAS

[1] Akpolat,Z., Hakan; Asher, Greg, Clare, M., Jon C. "Experimental Dynamometer Emulation of Non-linear Mechanical Load", **IEEE;** pp. 532-539, 1998.

[2] Rodríguez, José*;* Kennel, Ralph, M*;* Espinoza, José, R, "High-Performance Control Strategies for Electrical Drives: An Experimental Assessment", IEEE Transactions on Industrial Electronics, VOL. 59, N0. 2, pp. 1208-1216, February 2012.

[3] Vodyakho, Oleg*;* Steurer ,Mischa; Edrington,Chris, S; Fleming, Fletcher*, "*An Induction Machine Emulator for High-Power Applications Utilizing Advanced Simulation Tools With Graphical User Interfaces" IEEE Transactions on Energy Conversion, VOL. 27, NO. 1, pp. 160-172, March 2012.

[4] Akpolat, Z., Hakan; Asher, Greg, M.; Clare, Jon, C., "Experimental Dynamometer Emulation of Nonlinear Mechanical Loads", IEEE Transactions on Industrial Applications, VOL. 35, NO. 6, pp. 1367-1373, November/December 1999.

[5] Hassania, Baibanou; Sicard, Pierre; Ba-razzouk, Abdellfattah, "Solutions to typical motor Load emulation Control Problems, Electrimacs 2002, August 18-21.

[6] Davari, S., Alireza*;* Khaburi, Davood, Arab; Wang, Fengxiang; Kennel, Ralph, M.*, "*Using Full Order and Reduced Order Observers for Robust Sensorless Predictive Torque Control of Induction Motors", IEEE Transactions on Power Electronics, VOL. 27, NO. 7, pp. 3424-3432, July 2012.

[7] Bose, B. K. " Modern power electronics and variable frequency AC motor drives", IEEE press, 1997.

[8] PappasC.H. ; Murray W. H. " Turbo C++ Total and complete " (In portuguese), Macron Books do Brasil, 1991.

[9] ADVANTECH Co., Ltda. "PCL-711B - PC-MultiLab User's Manual"; Taiwan: Advantech Co., Ltd., aug., 1993.

[10] Simovert – Master Drives, Siemens : "Operation Manual".

[11] Fitzgerald A .E. ; Kingsley Jr. C.; Kusko A. "Electrical Machines" (In portuguese), Ed. Mc Graw Hill do Brasil, 1977.

[12] Kosow, I.L. "Electrical Machines and Transformers" (In portuguese) – Ed. Globo S.A., Brasil, 1987.

[13] V. Fernão Pires; J. F. Martins; Tito G. Amaral "Web Based Teaching of Electrical Drives Using a Mechanical Load Simulator", IEEE pp. 3545-3550, 2008.

[14] Michael Angelo A. Pedrasa ; Vincent Louie S. Delfin “Low Cost Mechanical Load Emulator” , IEEE, 2006.

[15] Rezek, A.J.J.“ Electrical Machines Basic Fundamentals: Theory and tests” (In Portuguese), Acta and Synergia, Editora, Brasil, 2011.

[16] SIMOES, M. G. ; Farret, F.A**.** . Alternative Energy Systems: Design and Analysis with Induction Generators. 2. ed. Boca Raton-Florida-USA: Taylor & Francis, CRC Press, 2008.

[17] Nery Trade and Representations Engineering ltd; www.nery.com.br.

[18] Goedte, Alessandro; Serni, Paulo, J., A.; Silva, Ivan N. da **“**Uma Abordagem neural para Estimativa de Torque em Motores de Indução”; 364 Revista Controle & Automação/Vol.17 no.3/Julho, Agosto e Setembro 2006.

[19] Guia Tecnico: www.weg.net “Motores de indução alimentados por inversores de frequência PWM”

APÊNDICE A

Algoritmo desenvolvido EM C++

Programa computacional implementado em C++

```
/*********************************************************************
*
   * Programa    : Simulador Generico de Cargas
   *Descrição   : Simulador Generico de Cargas utilizando o cartão PCL-711B
   *  Versão     : 4.0
   *  Data        : 18/03/11
   *  Ult.Mod.   : 18/03/10

*********************************************************************
*/
/* Inclusão de Diretivas */

#include <stdio.h>
#include <conio.h>          /* Aceita diretivas, incluindo códigos */
#include <stdlib.h>               /* de fontes de outros */
#include <math.h>
#include <dos.h>               /* programas ou diretórios. */
#define  DATABUFNUM 40
//#include <timer.h>

  /* Declaração de Variáveis Globais */

extern "C" pcl711(int, unsigned int *);   /* Inclui função "pcl711" definida
      inteiro, sem sinal      em um módulo separado utilizando
                              linguagem "C" */
unsigned int param[100];                  /* Definição de um vetor de dados -
                              array - que formam a tabela de
                              parâmetros inteiros e s/ sinal */
unsigned int datain[DATABUFNUM], dataout[DATABUFNUM];     /* Buffer de 10
dados inteiros +
                              para conversão */
unsigned int far *datin, *datout;      /* Endereço do buffer de dados acima
                              - pointer - tipo inteiro e longo:
                              2 palavras c/ range de 1Mbyte */
int tecla,i;                              /* Variáveis de leitura do teclado
                              e numero de canais */
/* Variáveis pontos flutuantes do controle */
float  nReal=0.0, tReal=0.0, VOut=0.0, nReal_ =0.0;
float  DataBuf[DATABUFNUM];
```

```
/* Declaração de variáveis void - significa que não retorna um valor */
void conv_ad(void);
void conv_da(void);

/* Conversão AD - Tabela de parâmetros */

void conv_ad(void)
{
      unsigned int i;
/* Pointer - Espaço de memória - Variável que contém um endereço que,
normalmente , é endereço de outra variável."                    */
      datin = datin;                /* Atribui ao pointer datin o valor
equivalente … variável datain   */
      param[0]  = 0;                /* Número do cartão              */
      param[1]  = 0x220;              /* Endereço de Base I/O          */
/* Frequência de amostragem = Frequência de base do cartão/(C1 * C2) */
                    /* 2M / (10 * 10) = 20 KHz */
      param[5]  = 10;                /* Divisor constante pacer C1 */
      param[6]  = 10;                /* Divisor constante pacer C2 */
      param[7]  = 0;                /* Modo Trigger, 0 : pacer trigger
Permite funções D/I     */
      /* Offset do Buffer , o endereço de memória (buffer) onde os dados
      serão guardados. Segmento , o comprimento do buffer de dados */
      param[10] = FP_OFF(datin);       /* Offset do Buffer A do A/D       */
      param[11] = FP_SEG(datin);       /* Segmento do Buffer A do A/D     */
      param[12] = 0;                /* Endereço do Buffer B (não usado)*/
      param[13] = 0;                /* Segmento- Não usado, setar em 0 */
      /* A conversão A/D envolve dois canais de entrada, canal 1 - corrente,
    e canal 0 - velocidade, com valores em pu ajustados em +/- 5 V */
      param[14] = 2;                /* Número de conversões A/D        */
      param[15] = 0;                /* Canal de inicio da conversão A/D*/
      param[16] = 1;                /* Canal de parada da conversão A/D*/
      param[17] = 0;                /* Ganho dos canais, 0 : +/- 5V    */

/* Indicação de falha na conversão A/D */

      pcl711(3, param);       /* Func. 3 : Inicialização do Hardware      */
      if (param[45] != 0) {   /* Se parâmetro 45 diferente de 0, fazer:   */
      clrscr();         /* Limpar a tela                        */
      printf("\n FALHA NA INICIALIZA?ÇO DO DRIVER !"); /* Imprimir   */

      getch();          /* Mostrar a tela de saída                */
      exit(1);          /* Fecha o loop e sai com status 1 - Erro   */
      }

      pcl711(4, param);       /* Func 4 : Inicialização do conversor A/D*/
      if (param[45] != 0) {
      clrscr();
      printf("\n FALHA NA INICIALIZA?ÇO DO A/D !");
```

```
      getch();
      exit(1);
       }

       pcl711(5, param);       /* Func 5 : Verificação número conversões A/D*/
       if (param[45] != 0) {
      clrscr();
      printf("\n FALHA NO ALGORITMO DE TRANSFERÒNCIA DE DADOS A/D
!");
      getch();
      exit(1);
       }

                        /* Conversões A/D */


       for (i = 0; i < param[14]; i++)   /* Dados amostrados - canais 0 e 1 */
    {
      DataBuf[i] = datain[i]  & 0xFFF;
      /* Coleta de dados para o buffer no endereço 0xFFF
        (os três primeiros dígitos hexadecimais podem ser zerados pois o
        restante , suficiente p/ suportar 4096 dígitos binários ) */
      DataBuf[i] =((5.0 - (-5)) * DataBuf[i] / 4096) + (-5);
            /* Conversão para que o sinal de tensão seja disponível para
                aplicação nas equações recursivas de controle
        (5 - (-5)) : Faixa de entrada A/D (-5V  to  5V)
        4096       : Faixa da escala do A/D - 12 bit
        DataBuf    : Dado de entrada do A/D
        (-5)       : Inicio da escala do A/D "-5" V
      */
     }
        /* Leitura da tensão de realimentação para a malha de velocidade
             e de corrente, sob condições de velocidade e carga nominal /*
        /* Conversão do sinal de velocidade - correção para pu */
  nReal=(DataBuf[0]);

/* Conversão do sinal de torque - correção para pu */
  tReal=(DataBuf[1]);
}

/*  Conversão D/A - Tabela de Parâmetros */

void conv_da()
{
      datout = dataout;           /* Atribui ao pointer datout o valor
                                        equivalente … variável dataout       */
      param[0]=0;               /* Número do cartão                    */
      param[1]=0x220;           /* Endereço de base I/O                */
     /* Offset do Buffer , o endereço de memória (buffer) onde os dados
        serão guardados. Segmento , o comprimento do buffer de dados */
```

```
    param[20] = FP_OFF(datout); /* Offset do buffer A dados  saída D/A  */
    param[21] = FP_SEG(datout); /* Segmento do buffer A dados saída D/A */
    param[22] = 0;          /* Endereço do Buffer B saída(não usado) */
    param[23] = 0;          /* Segmento saída- Não usado, setar em 0 */
    param[24] = 1;          /* Número de conversões D/A            */
    param[25] = 0;          /* Canal de inicio da conversão D/A    */
    param[26] = 0;          /* Canal de parada da conversão D/A    */

/* Indicação de falha na conversão D/A */

    pcl711(3, param);      /* Func 3 : Inicialização do hardware      */
    if (param[45] != 0)  {  /* Se parâmetro 45 diferente de 0, fazer:   */
          clrscr();        /* Limpar a tela                          */
          printf("\n FALHA NA INICIALIZA?ÇO DO DRIVER !"); /* Imprimir   */
          getch();         /* Mostrar a tela de saída                */
          exit(1);         /* Fecha o loop e sai com status 1 - Erro   */
    }
    pcl711(12, param);     /* Func 12: Inicialização do conversor D/A  */
    if (param[45] != 0) {
          clrscr();
          printf("\n FALHA NA INICIALIZA?ÇO DO D/A !");
          getch();
          exit(1);
    }
    pcl711(13, param);     /* Func 13: Verificação número conversões D/A*/
    if (param[45] != 0) {
            clrscr();
            printf("\n FALHA NO ALGORITMO DE TRANSFERÒNCIA DE
DADOS D/A !");
            getch();
            exit(1);
    }

/* Conversão para que o sinal de tensão de saída das equações
recursivas de controle em pu ocupe um espaço de endereço do
buffer de dados de saída.                              */
    dataout[0]=(4095*VOut);

}

/* Programa principal */

void main(void)   /* Garante que as variáveis globais não retornam valores */
{             /* Declaração de variáveis */
  int   k1=0, k2=0;
  float sn = 0.0277, fc = 0.6, nru, nrum = 0.0,Ths = 0.0e-2, VOut_=0.0, snru=0.0;
  float strealu=0, mtrealu=0;
  //-------------------------------------------------------------
  int   NA = 100; // <---- Alterar aqui o número de amostras para
            //       o cálculo da média!
```

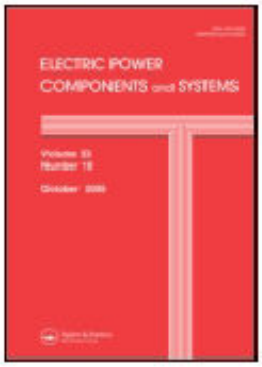
```
//-------------------------------------------------------------
int   NS = 1000; // <---- Alterar aqui o intervalo de atualização
            //       da tela!
//-------------------------------------------------------------
float tnru = 0.1, kt = 0.60, tr = 0, terr=0, mnerr =1e-3, kpt=3e-5; // <----
//-------------------------------------------------------------
float ajf = 0; // <----
//-------------------------------------------------------------
unsigned char Lajf = 0; // <----
int   imenu = 3;
asm cli;
clrscr();
VOut = 0.0;
conv_ad();       /* Efetua a subrotina de conversão A/D */
conv_da(); //Ajusta a saida para 0V.
nReal_ = nReal;
clrscr();
gotoxy(1,18);
cprintf("fc  (+) = <A> / fc  (-) = <Z> / fc  <- default = <Q>\n\r");
cprintf("kt  (+) = <S> / kt  (-) = <X> / kt  <- default = <W>\n\r");
cprintf("kpt (+) = <D> / kpt (-) = <C> / kpt <- default = <E>\n\r");
cprintf("Ajuste fino (on/off) = <F>\n\r");
cprintf("imenu <1>, <2>, <3>");
do {
conv_ad();       /* Efetua a subrotina de conversão A/D */
if (kbhit()) tecla=getch();
else tecla = 0;

if (fabs(nReal-nReal_)>=Ths)
  nReal_ = nReal;

nru      = nReal_ / 4.90;
strealu += tReal;
snru    += nru;

if (++k1>=NA)
{
   nrum  = snru /(float) NA;
   if (nrum<0) nrum = 0.0;
   snru    = 0.0;
   mtrealu = (strealu / (float) NA) / 1.45;
   strealu = 0;
   k1      = 0;
}
switch(imenu)
{
   case 1: if (nrum<tnru) VOut_ = nrum;
               else           VOut_ = (1-sn*fc)*nrum;
               tr = kt;
               break;
```

```
    case 2: VOut_ = (1-sn*fc*nrum)*nrum;
            tr = kt * nrum;
            break;

    case 3: VOut_ = (1-sn*fc*pow(nrum,2))*nrum;
            tr = kt * (nrum*nrum);
            break;
  }

//---------------------------------------------------------------------
  /* Ajuste fino */
  if (Lajf)
  {
    terr = tr - mtrealu;
    if (fabs(terr) > mnerr)
    {
      ajf += terr * kpt;
    }
  }
  else
  {
    VOut = VOut_;
    ajf = 0;
  }
  VOut = VOut_ - ajf;
  if (VOut>1) VOut=1;
  if (VOut<0) VOut=0;
  conv_da();    /* Efetua a subrotina de conversão D/A */

//---------------------------------------------------------------------

  if (++k2>=NS)
  {
    k2 = 0;
    gotoxy(1,1);
    cprintf("imenu=%d,   \n\rfc=%2.3f,   \n\rsn=%2.3f      \n\r",  imenu, fc, sn);
    cprintf("Tensao entrada     = %2.3f     \n\r", nru);
    cprintf("Tensao entrada med = %2.3f     \n\r", nrum);
    cprintf("Tensao Saida       = %2.3f     \n\r", VOut  * 10.0);
    cprintf("Torque ref.        = %2.3f     \n\r", tr);
    cprintf("Torque (V).        = %2.3f     \n\r", tReal);
    cprintf("Torque medio pu    = %2.3f     \n\r", mtrealu);
    cprintf("Kpt                = %2.5f     \n\r", kpt);
    cprintf("Ajuste fino        = %d        \n\r", Lajf);
    cprintf("ajf        = %2.5f             \n\r", ajf);
    cprintf("Torque de erro         = %2.5f  \n\r", terr);
    //cprintf("Tecla = %d  \n\r", tecla);
  }
```

```
  switch (tecla)
  {
    case 65:  case 97:  fc += 1e-3; tecla = 0; break;
    case 90:  case 122: fc -= 1e-3; tecla = 0; break;
    case 81:  case 113: fc = 1;     tecla = 0; break;

    case 83:  case 115: kt += 1e-3; tecla = 0; break;
    case 88:  case 120: kt -= 1e-3; tecla = 0; break;
    case 87:  case 119: kt = 0;     tecla = 0; break;

    case 68:  case 100: kpt += 1e-5; tecla = 0; break;
    case 67:  case 99 : kpt -= 1e-5;
                        if (kpt<0) kpt = 0;
                        tecla = 0; break;
    case 69:  case 101: kpt = 1;     tecla = 0; break;

    case 49: imenu = 1; tecla = 0; break;
    case 50: imenu = 2; tecla = 0; break;
    case 51: imenu = 3; tecla = 0; break;

    case 70: case 102: Lajf = !Lajf; tecla = 0; break;
  }

  } while(tecla!=27);   /* Interrupção do programa pela tecla ESC */

  VOut = 0.0;
  conv_da(); //Ajusta a saida para 0V.
  asm sti;

}
```

APÊNDICE B

Artigo publicado



Electric Power Components and Systems
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/uemp20

Implementation of a Simulator for the Most Commonly Found Industrial Motor Loads Based on Pulse-width Modulation Inverters and Torque Estimator

José Carlos Grilo Rodrigues [a] , Ângelo José Junqueira Rezek [a] , Manuel Luiz Barreira Martinez [a] , Délvio Franco Bernardes [a] & Nery de Oliveira Júnior [a b]

[a] Universidade Federal de Itajubá/Instituto de Sistemas Elétricos e Energia (UNIFEI/ISEE), Federal University of Itajubá, Electrical Systems and Energy Institute, Itajubá, Minas Gerais, Brazil

[b] Nery Trade and Representations Engineering Ltd, Minas Gerais, Brazil

Version of record first published: 11 Jan 2013.

To cite this article: José Carlos Grilo Rodrigues , Ângelo José Junqueira Rezek , Manuel Luiz Barreira Martinez , Délvio Franco Bernardes & Nery de Oliveira Júnior (2013): Implementation of a Simulator for the Most Commonly Found Industrial Motor Loads Based on Pulse-width Modulation Inverters and Torque Estimator, Electric Power Components and Systems, 41:3, 345-364

To link to this article: http://dx.doi.org/10.1080/15325008.2012.742946

Electric Power Components and Systems, 41:345–364, 2013

ISSN: 1532-5008 print/1532-5016 online
DOI: 10.1080/15325008.2012.742946

Taylor & Francis
Taylor & Francis Group

Implementation of a Simulator for the Most Commonly Found Industrial Motor Loads Based on Pulse-width Modulation Inverters and Torque Estimator

JOSÉ CARLOS GRILO RODRIGUES,[1]
ÂNGELO JOSÉ JUNQUEIRA REZEK,[1]
MANUEL LUIZ BARREIRA MARTINEZ,[1]
DÉLVIO FRANCO BERNARDES,[1] and
NERY DE OLIVEIRA JÚNIOR[1,2]

[1]Universidade Federal de Itajubá/Instituto de Sistemas Elétricos e Energia (UNIFEI/ISEE), Federal University of Itajubá, Electrical Systems and Energy Institute, Itajubá, Minas Gerais, Brazil
[2]Nery Trade and Representations Engineering Ltd, Minas Gerais, Brazil

Abstract *The aim of this article is to present the results of the implementation of a simple generic load simulator using pulse-width modulation inverters. The implementation of this load simulator took place at Universidade Federal de Itajubá's electrical drives laboratory. Different types of load can be simulated to evaluate the performance of electric motors in electrical drives. The implemented system shows satisfactory results, even at low speeds. The system arrangement utilizes two pulse-width modulation inverters (one for the motor and the other for the generator), a PC microcomputer, and a data acquisition board installed in a free slot of the microcomputer. A tachogenerator is also used to provide an analog speed signal as well as LabVIEW data acquisition (Hungary) for filtering speed and torque signals for better visualization and interpretation of the results. The torque signal is originated from the motor pulse-width modulation inverter torque transducer.*

Keywords AC drives, AC generators, AC machines, AC motors, control systems

1. Introduction

AC drives [1, 2] are widely used in industry due to their robustness, simplicity, ease of maintenance, and lower acquisition and operating costs, which characterize a three-phase induction motor with squirrel-cage rotors as compared to other types of electrical motors [2–6].

At the Universidade Federal de Itajubá (UNIFEI's electrical drives laboratory), workbenches are equipped with Simovert pulse-width modulation (PWM) inverters manufactured by Siemens [7] and feeding squirrel-cage induction motors. These motors, however, are operating at no load due to the lack of a load simulation system.

Received 28 July 2012; accepted 19 October 2012.
Address correspondence to Prof. José Carlos Grilo Rodrigues, UNIFEI/ISEE, Federal University of Itajuba, Electrical Systems and Energy Institute, Av. BPS 1303 P.O. Box 50, Itajubá, 37500903, MG, Brazil. E-mail: jcgrilo@unifei.edu.br

To overcome this lack, implementation of a universal load simulator [8–13] has been proposed, which uses two PWM inverters—one to feed the motor to be tested and the other to feed another induction machine—operating as a generator, coupled to the first by means of an elastic coupling and acting as load to the motor.

The proposed system, which will be described in detail in what follows, will be of great importance to enable tests on motors with different types of load, such as constant torque loads (*e.g.*, belt conveyors and elevators), loads with torque proportional to the square of the speed (*e.g.*, pumps, fans, and compressors), loads with torque proportional to speed (*e.g.*, calenders), or loads with torque proportional to the inverse of the speed (*e.g.*, winders). These types of loads account for the vast majority of industrial loads, and they can all be simulated in order to assess the performance of the drive.

2. Methodology and Overall Description of the Proposed Technique

Figure 1 shows the experimental arrangement that will be used. In this experimental arrangement, the PWM 1 inverter is feeding the motor and the PWM 2 inverter is connected to the generator stator windings, which acts as the load to the motor. The coupling of the two machines is an elastic one. If the inverter capacitors were not coupled, the PWM 2 inverter machine working as a generator would feed the capacitor, and it would charge up to the point when the resulting over-voltage would cause a fault in the inverter (DC link). Since the capacitors are coupled, however, the active power from the PWM 2 generator will flow to the PWM 1 inverter, and there will be no over-voltage (and thus no fault) in the PWM 2 inverter.

The inverter 1 frequency will always be higher than that of inverter 2. Thus, as the machines are coupled shaft to shaft, the mechanical rotation of the generator will always be greater than the synchronous speed of its rotating field. Therefore, the operation as a generator will be always assured, feeding energy back to the DC bus (DC link). This energy will always be circulating in the loop formed by the machines, since the DC-link capacitors of the two inverters will be connected in parallel, enabling this circulating

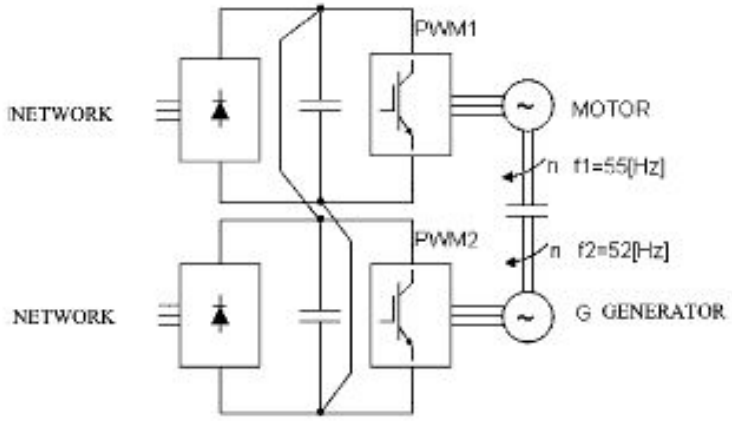


Figure 1. Experimental arrangement proposed.

energy condition between the machines. Thus, the motor will absorb the energy coming from the generator through the DC bus (DC link of the interconnected inverters).

The frequency of inverter 2 is controlled through a dedicated computer program written in C++ [14]. A PC microcomputer (Lenovo TEC, Limitada, Brazil) is used to run this program and display the input and output data. A data acquisition board is made with a PCL711 dedicated card [15] from Advantech (Taiwan) installed in a free slot of the PC, with eight analog inputs (analog-to-digital [A/D] converters) and one analog output (digital-to-analog [D/A] converter). The digital control output of the developed program is the input to the D/A converter of the board. The analog signal from the board will control the frequency of inverter 2, corresponding to the desired load torque, selected in the program load selection menu (MENU).

One of the analog inputs in the acquisition board receives the signal from the speed transducer, which is generated in the tachometer coupled to the machines shaft; another receives the signal generated in the inverter 1 torque transducer.

These signals, after being converted to digital form, will be the inputs to the control program. The A/D conversion is accomplished by a conversion board data acquisition (DAQ) A/D NI USB-6212 (National Instruments, Hungary), with 14-bit (1.2-mV) resolution. Its sampling rate is 400 kS/s, enabling operation with signals up to 200 kHz. Internally, the DAQ has a signal treatment unit dealing with input voltages ranging from -10 to $+10$ V.

In this way, if the motor speed varies, the inverter 2 frequency will also vary, complying with the choice made in the input MENU of the control program. For example, consider a case where the chosen load has a torque initial condition in which the frequencies in inverters 1 and 2 are 60 and 57 Hz, respectively. If the inverter 1 frequency is shifted to 55 Hz, the inverter 2 frequency will be automatically altered to a smaller value, which results from the calculations performed by the dedicated program. The equations utilized in these calculations will be described later.

It is seen that different loads can be simulated, even at low speeds and also in cases where it is necessary to vary the speed of the tested motor.

Figure 2 shows the driver circuit for the frequency adjustment of inverter 2. The driver circuit has an input stage consisting of two op amps in a cascaded configuration, with unity gain in the first and adjustable in the second, resulting in a voltage input to the

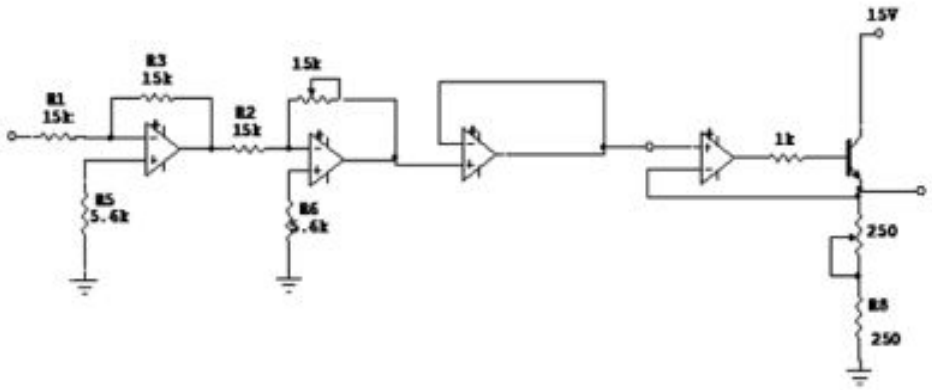


Figure 2. Driver circuit for the frequency adjustment of inverter 2.

second stage without inversion and with adequate value. This stage consists of a buffer circuit with high input impedance and low output impedance, with the final stage having feedback voltage and current source sufficient to supply the analog input voltage of the PWM 2 inverter to enable frequency variation. Thus, the output voltage of the driver to supplying the analog input of PWM 2 may vary from 0–10 V, allowing variation of the frequency controlled inverter in the range of 0–60 Hz.

When the driver circuit is added the undesirable voltage drop in the supply to the PWM 2 inverter, analog input is eliminated. The D/A output of the data acquisition board is the input to the driver circuit, and the output voltage of 0–10 V is analog input 1 to inverter 2 for frequency adjustment of this inverter. The inverter 2 frequency adjustment is made according to the load selection menu, which can be:

1. loads with constant torque (constant torque load),
2. loads with torque proportional to the speed (linear torque load), and
3. loads with torque proportional to the square of the speed (quadratic torque load).

The part of the C++ program where this selection is made is given below:

```
switch(imenu)
        {
           case 1:
                        VOut_ = (1-sn*fc)*nrum;
             tr = kt;
                    break;

           case 2: VOut_ = (1-sn*fc*nrum)*nrum;
             tr = kt * nrum;
             break;

           case 3: VOut_ = (1-sn*fc*pow(nrum,2))*nrum;
tr = kt * (nrum*nrum);
                  break;
         }
```

where

VOut is the p.u. value of the acquisition board output, varying from 0 to 1 p.u.; the input to the circuit of Figure 2 (in volts) varies from 0 to 10 V for a range of frequencies of inverter 2 from 0 to 60 Hz;

sn is the rated slip of the load generator;

fc is the load factor varying from 0 to1 p.u.; this factor represents the maximum load torque of the motor;

kt is the torque constant and also represents the maximum motor torque and is therefore is equal to fc:

$$kt=fc; \tag{1}$$

nrum is the medium value (in p.u.) of the motor speed; the program acquires this value from the speed transducer (tachogenerator) and calculates the mean value of 100 samples, ensuring good accuracy even in the presence of ripple variations of the output signal of the tachogenerator; the value of nrum is obtained by dividing the

signal (in volts) of the speed transducer by the value of the tachogenerator voltage when the speed is 1 p.u. (that is, 1800 rpm); and

tr is the reference torque; this torque value (also in p.u.) will be used for the implementation of a fine adjustment in the frequency of inverter 2, as will be described next.

A fine adjustment procedure has been implemented for a more precise adjustment of the inverter 2 frequency according to the load torque menu selected. Therefore, the value of VOut_ in cases 1, 2, and 3, described above, only represents a rough frequency adjustment. The part of the C++ program that performs the fine adjustment procedure is presented below:

```
if (Lajf)
{
 terr = tr - mtrealu;
if(fabs(terr)>mnerr)
{
 ajf += terr * kpt;
}
VOut = VOut_ - ajf;
}
else
{
VOut = VOut_;
ajf  = 0;
}

conv_da();
```

where

(Lajf) shows that if the F key is pressed, the fine adjustment procedure is executed; otherwise, only the rough adjustment of cases 1, 2, or 3 is made;

ajf denotes fine adjustment;

terr denotes error torque;

tr denotes reference torque;

mtralu denotes feedback torque in p.u., obtained through the inverter 1 torque transducer (also calculated as the mean value of 100 samples);

mnerr denotes values of the window used to calculate the fine adjustment; if the absolute value of the error torques are within the window, the fine adjustment procedure is not executed;

kpt denotes gain constant for the calculation of the fine adjustment procedure given as kpt = 0.3e-5; this constant value is very low, because at each loop of the program, the fine adjustment procedure is increased, as can be seen by the instruction:

```
ajf=terr*kpt
```
(2)

conv_da() is the call of the subroutine of D/A conversion for the inverter 2 frequency adjustment.

3. Results Obtained

Figures 3, 4, and 5 show the results obtained using fine adjustment for loads of the constant torque, linear torque, and quadratic torque types with load factor fc (or torque constant Kt) equal to 0.6, with 1 p.u. torque = 22 Nm and 1 p.u. speed = 1800 rpm. It can be observed that the results using the fine adjustment procedure are significantly improved compared to the previous results.

Figures 6, 7, and 8 show the results obtained for the three types of load using the fine adjustment procedure and also a dynamic linear variation of the speed from 300 to 1800 rpm in 40 sec for load factor fc (or torque constant Kt) equal to 0.6 with 1 p.u. torque = 22 Nm and 1 p.u. speed = 1800 rpm.

In Figures 6, 7, and 8, the upper curves are the non-filtered signals of the tacho-generator speed and the inverter 1 torque. The lower curves show the corresponding filtered signals (filtering is performed by a data acquisition system using LabVIEW (Hungary), which processes an average of 50 values of speed and torque, respectively; the description of this system will be presented later). Maximum torque was taken equal to 0.6 p.u. Figure 9 shows the flowchart of the developed C++ program.

LabVIEW is software for graphical programming based on data flow instead of command lines. It was created and is mainly used for signal acquisition and processing. Because of this and because it can be programmed very quickly, this software was used in all data acquisitions and result displays.

After the signal is captured, it must be processed in order to extract from it the information that really matters, in this case, the medium value of the voltage of the two input channels, one of them proportional to the motor shaft speed and the other proportional to the torque.

On the front panel, there are three graphics and some control buttons. The graphics labeled "unfiltered" and "filtered" display the measured values of the torque and speed

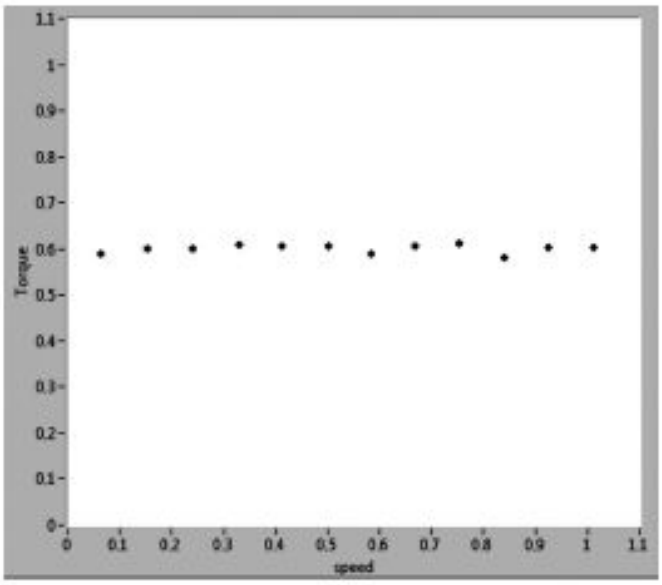


Figure 3. Torque versus speed for constant torque load using fine adjustment procedure with 1 p.u. torque = 22 Nm and 1 p.u. speed = 1800 rpm.

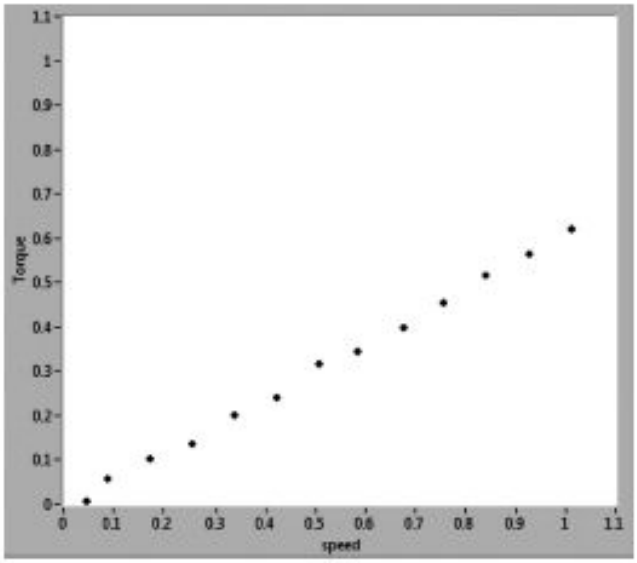


Figure 4. Torque versus speed for linear torque load using fine adjustment procedure with 1 p.u. torque = 22 Nm and 1 p.u. speed = 1800 rpm.

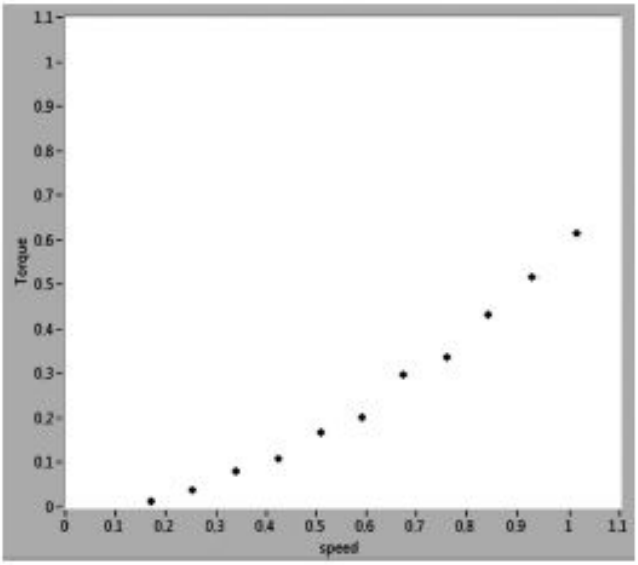


Figure 5. Torque versus speed for quadratic torque load using fine adjustment procedure with 1 p.u. torque = 22 Nm and 1 p.u. speed = 1800 rpm.

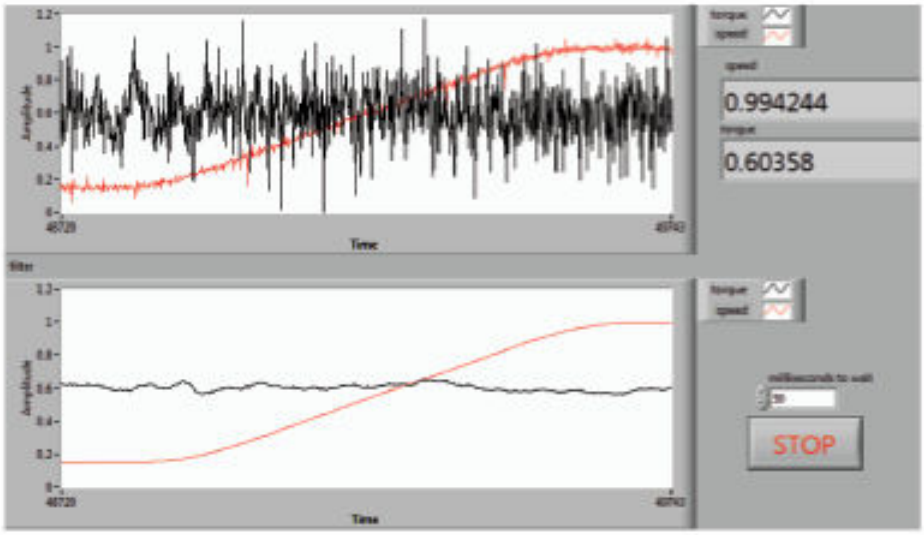


Figure 6. Results obtained for constant torque load using the fine adjustment procedure and dynamic linear variation of speed from 300 to 1800 rpm. Black curve represents torque, and red curve represents speed (maximum torque 0.6 p.u.). (color figure available online)

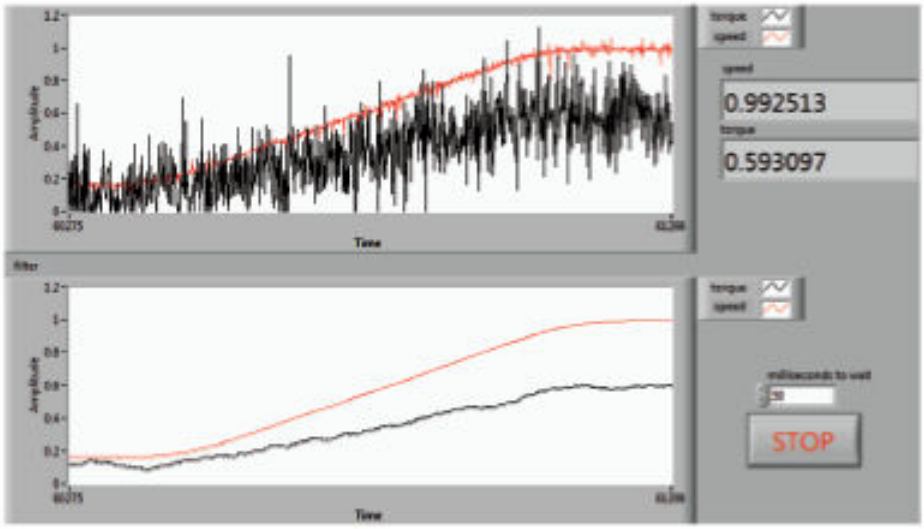


Figure 7. Results obtained for linear torque load using the fine adjustment procedure and dynamic linear variation of speed from 300 to 1800 rpm. Black curve represents torque, and red curve represents speed (maximum torque 0.6 p.u.). (color figure available online)

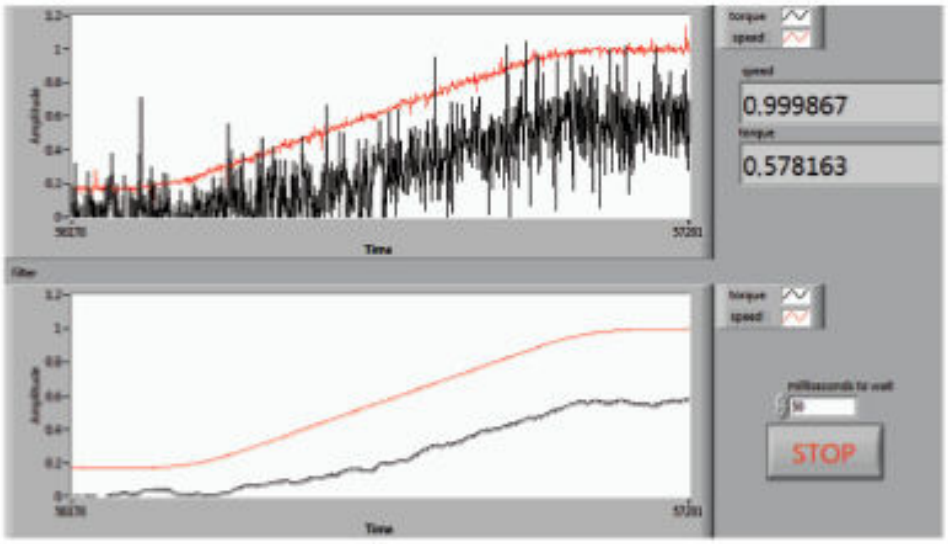


Figure 8. Results obtained for quadratic torque load using the fine adjustment procedure and dynamic linear variation of speed from 300 to 1800 rpm. Black curve represents torque, and red curve represents speed (maximum torque 0.6 p.u.). (color figure available online)

versus time, but one of them shows the values without any processing. The graphic on the right shows the values of the torque versus speed, as described in each of its axes. Under the graphic, the indicators "rotation" and "torque" indicate the medium values of their respective measurements. Finally, the two fields that can be filled serve to indicate the location and name of the file where the data will be saved.

The first part of the program creates communication channels between LabVIEW and the DAQ. The signal is then captured and displayed in the graphics. In one of them, only the signal is shown, while a window of 100 points "runs through it" while calculating the average value of these points. With this treatment, the signal obtained is slightly out of phase but almost without any noise. The other part of the program is used to perform the tests with the motor already stabilized (rather than during the starting process). The program plots a point every time the user clicks "OK." This point is included in the graphic on the right. The data can also be saved in a txt file in table form when the user clicks "save," and the fields are filled with valid values.

4. Considerations on Voltage Fluctuations in the Intermediate DC Link

In the case of continuously varying torque, the case of loads that vary continuously in linear or quadratic form with speed, oscillation problems do not occur. The experimental test using the workbench has consisted of applying to the motor, in a time interval smaller than 1 sec, an abrupt variation of torque of magnitude 1 p.u. positive and negative. As shown in Figure 10, registers of the voltage and speed in the intermediate DC link were obtained with no voltage fluctuations observed; in fact, the voltage and the speed remained quite steady and almost constant. In the case of a positive torque abrupt change, the energy supplied by the generator is absorbed by the motor via an intermediate DC link.

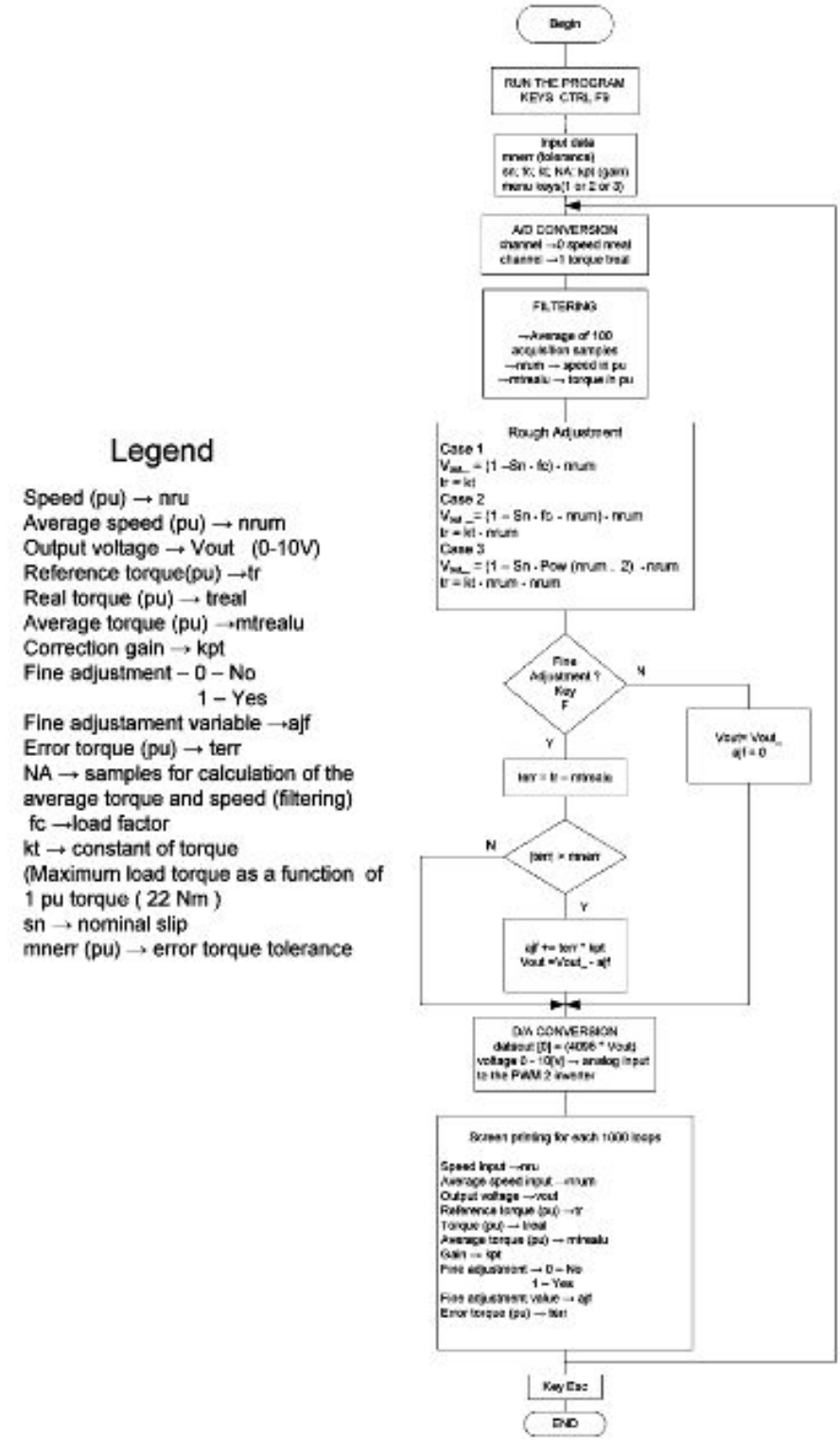


Figure 9. Flowchart of the developed C++ program.

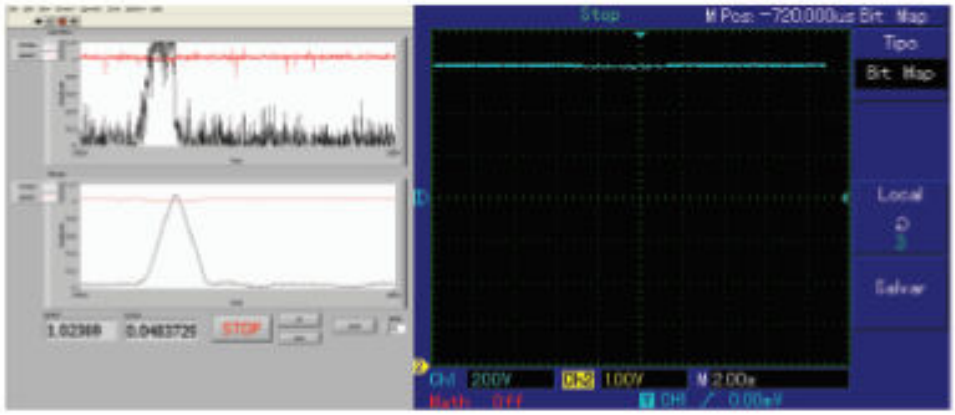


Figure 10. Abrupt change from positive to negative direction torque applied to the motor (left) and intermediate DC-link voltage (right). (color figure available online)

In the case of a negative torque abrupt change, the braking energy must be dissipated in the intermediate DC link. This is a critical condition if the torque achieves negative values. The motor does not have the capacity to absorb this energy, and the machine coupled always operates as a generator, and therefore not having also the ability to consume energy. Thus, three options to dissipate this braking energy can possibly be employed.

Option 1: use a DC-link converter thyristor bridge to return the braking energy to the grid, since the bridge rectifier diodes of the inverter does not have the ability to return the energy to the grid;

Option 2: use a braking resistor in the DC intermediate link to dissipate this braking energy;

Option 3: use an inverter with transistor active rectifier technology type AFE (active front end), for returning braking energy to the network instead of conventional inverters using diode bridge rectifiers, which do not allow bi-directional power flow; Figure 10 shows the abrupt changes of the torque, first in a positive and then in a negative direction, and the intermediate DC link voltage (scales are 200 V/div in the vertical axis and 2 s/div in the horizontal axis).

The red curve represents the speed and the black one the torque. Values are for unfiltered (upper curves) and filtered signals (lower curves); the bus voltage in the DC intermediate link was registered, as shown in Figure 10 (right). Conventional inverters were used in this work, so second option is more easily implemented because the braking resistors are commonly found for this type of inverter.

When the short-circuit power at the converter AC bus is greater or equal to 100 times the converter power, the voltage fluctuations or the DC bus is negligible [16]. In the case of the implemented system, the following values are found:

three-phase transformer feeder (laboratory): 13.800/220 V, power 150 KVA, impedance percentage: 4.5%; feeder network (concessionaire of energy);

three-phase short-circuit current in the bus of 13.800 V = 2640 A (power short circuit = 63.1 MVA); and

autotransformer to supply the workbench: 220/440 V, power 50 kVA, impedance Z autotransf = 2.25% or 0.0225 p.u.

Thus adopting a power base of 50 kVA or 0.05 MVA and a voltage base of 440 V (autotransformer data) gives:

system impedance: Zsist = 0.05/63.1 = 0.000792 p.u., therefore negligible and
transformer impedance (corrected to the base values of power and voltage, respectively, 50 KVA and 440 V): Ztranf = 0.045 × (50/150) × (220/440)2 = 0.00375 p.u., also negligible with respect to the autotransformer impedance.

So the power short circuit in the bus converters is (Ssc - pu) = 1/Zautotransf = 1/0.0225 = 44.44 p.u.. Therefore, the short-circuit power in the bus converters is Ssc = 44.44 × 50 = 2222 KVA. The power converters (workbench) is Sconv = 19.43 KVA, and the relationship is Ssc/Sconv = 2222/19.43 = 114.36. It is seen that this last relationship is greater than 100, and Figure 10 shows that no fluctuations were in fact observed.

The parameterization of the Siemens master drives PWM inverters utilized is described next. Figure 11 shows the flowchart for the procedure according to its operation's manual. The adjusted values were

P050 = 1
P051 = 2
P052 = 5
P071 = 440
P100 = 0
P101 = 380
P102 = 16
P104 = 0.86
P107 = 60
P108 = 1760
P165 = 0 (linear torque)
P420 = 60
P452 = 60
P453 = 60
P462 = 40
P463 = 0
P464 = 20
P465 = 0
P052 = 6

For the PWM 2 inverter (generator), the procedure was identical to that for the PWM 1 inverter. Only the parameter P462 was modified and set to zero to avoid delay in the torque output.

A potentiometer with 10 turns (analog input 1) was used to vary the speed of the motor of inverter PWM 1. Thus, the parameterization was done with PWM 1 inverter as P443 = 1003 (analog input 1), for borns 27 (signal) and 28 (mass) of the borns plate X102 of the PWM 1 inverter (input ranging from 0 to 10 V in the ten-turn potentiometer).

For the analog output of the motor PWM 1 inverter, a parameter P655.1 = 264 was used (Iq real, signal considered as proportional to torque, the inverter was parameterized for the motor working with constant flux [constant V/f], according to parameter

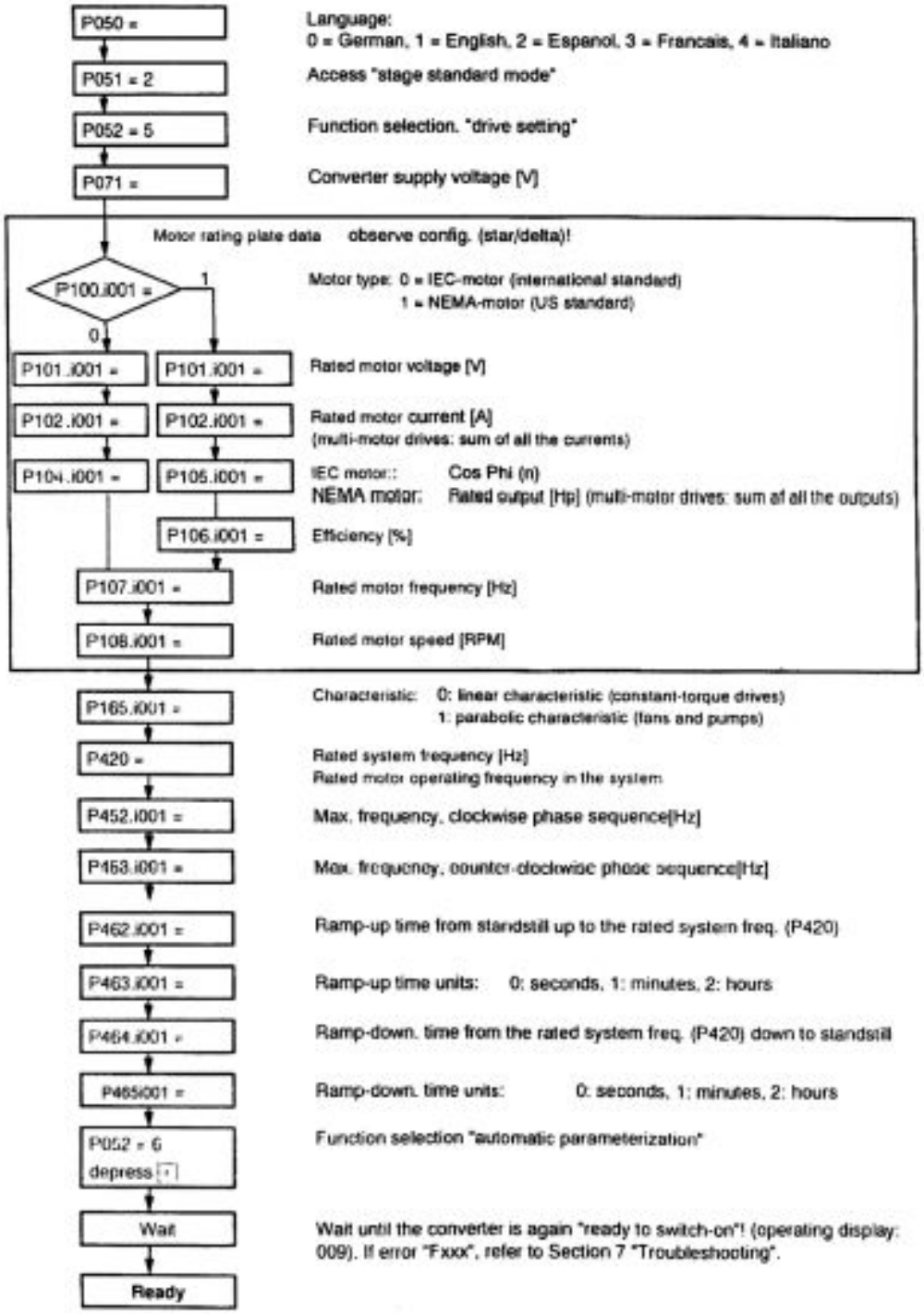


Figure 11. Parameterization flowchart.

Figure 12. Overview of the workbench utilized. (color figure available online)

P165 = 0). The borns of the analog output (that is, torque transducer) were those numbered 33 (mass) and 34 (signal), plate X102.

For the adjustment of the frequency of the PWM 2 inverter, the parameterization P443 = 1003 (analog input 1) was also used for borns 27 (signal) and 28 (mass), plate X102 of the PWM 2 inverter, and analog signal from circuit driver of Figure 2.

The Advantech PCL 711 data acquisition board therefore received two analog inputs, one corresponding to speed and the other corresponding to the inverter torque transducer (A/D converters), and the output of the board (D/A converter) was utilized to adjust the frequency of the PWM 2 generator converter, going through a dedicated driver, as shown in Figure 2.

Figure 12 shows an overview of the workbench utilized. The inverters are seen in the front, the computer on the right, and the machines in the rear.

The inverters are Siemens Master Drives VC (Germany), 380/460 V, 25.5 A; the voltage used was 440 V. The machines are a WEG (generator; Jaragua do Sol [SC], Brazil) and EBERLE (motor; Caxias do Sul [RS], Brazil) and three-phase induction motors of 10 CV, 380 V, and 16 A. The microcomputer is Pentium 3 with 600 MHz.

5. Pre-filtering of the Samples and On-line Adjustments

A pre-filter for the speed samples was implemented in the program so that if the obtained sample is outside a window, it is considered as noise and discarded; this pre-filtering feature, although implemented in the program, was not activated, because the value of the variable threshold (Ths) was considered equal to zero, as can be seen in the program listing shown in the Appendix. The filtering of speed and torque was activated only by an average of 100 samples for each sampling loop, and the results were satisfactory with this action.

As can be seen in the program listing, the following on-line adjustments are possible for variables kpt (gain), fc (load factor), and kt (constant of torque) by the keys:

D incrementkpt (+0.01 e-5),
C decrementkpt (−0.01 e-5),
S incrementkt (+1 e-3),
X decrementkt (−1 e-3),
A incrementfc (+1 e-3),
Z decrementfc (−1 e-3), and
Menu keys:
1. constant torque,
2. linear torque, and
3. parabolic torque

6. Conclusion

The implemented system showed quite satisfactory results. The estimation of the current isq of the motor inverter has been used as torque transducer, since it is proportional to the motor torque [17].

As a continuation of this work, a torque transducer installed on the motor shaft will be used. The results obtained will be compared with those obtained with the torque estimation made with the frequency inverter. In addition, loads with torque inversely proportional to speed (*e.g.*, winders) will be considered, so that nearly all loads found in industrial plants will be able to be simulated for the evaluation of the performance of the motor.

The fine adjustment procedure presented accurate results in the static analysis of torque variation with speed (constant torque, linear torque, and quadratic torque), verified as seen in the results shown in Figures 3, 4, and 5.

Also, in the dynamic torque regime with linear variation of the speed from 300 to 1800 rpm, the results were quite satisfactory, as can be seen in Figures 6, 7, and 8. The LabVIEW signal acquisition system was very useful for filtering signals from the speed and torque transducers for better evaluation of the results. In the continuation of this work, it is intended to use LabVIEW to replace the acquisition board PCL711.

Acknowledgments

The authors would like to thank Dr. Eben Ezer Prates da Silveira for help in the development of the used control program in C++ language. They would like to thank undergraduate students Paulo Silva Lima, João Paulo Barbosa Orsi, and Gustavo Retuci Pinheiro for the aide in the implementation of the hardware and software of the work, including the use of the LabVIEW acquisition. Also, Dr. Júlio César Tibúrcio is gratefully acknowledged for translation of this article into English, along with laboratory technicians Túlio Pimentel, Luciano de MesquitaTeles, Luiz Otávio Campos de Medeiros, and Luiz Sérgio Ferreira for development of the cards of the used driver circuit and mounting of the developed workbench. Finally also, the authors would like to thank all people that directly or indirectly contributed to the success of this research work.

References

1. Rodríguez, J., Kennel, R. M., and Espinoza, J. R., "High-performance control strategies for electrical drives: An experimental assessment," *IEEE Trans. Ind. Electron.*, Vol. 59, No. 2, pp. 1208–1216, February 2012.
2. Bose, B. K., *Modern Power Electronics and Variable Frequency AC Motor Drives*, IEEE Press, 1997.

3. Fitzgerald, A. E., Kingsley Jr., C., and Kusko, A., *Electrical Machines*, McGraw Hill do Brasil, 1977 (in Portuguese).
4. Kosow, I. L., *Electrical Machines and Transformers*, Brazil: Globo S.A., 1987 (in Portuguese).
5. Rezek, A. J. J., *Electrical Machines Basic Fundamentals: Theory and Tests*, Brazil: Acta and Synergia, 2011 (in Portuguese).
6. Simoes, M. G., and Farret, F. A., *Alternative Energy Systems: Design and Analysis with Induction Generators* (2nd ed.), Boca Raton, FL: Taylor & Francis, CRC Press, 2008.
7. Simovert, *Master Drives, Siemens: Operation Manual*.
8. Hakan Akpolat, Z., Asher, G. M., and Clare, J. C., "Experimental dynamometer emulation of non-linear mechanical load," *IEEE Trans. Ind. Electron.*, Vol. 46, No. 2, pp. 532–539, 1998.
9. Vodyakho, O., Steurer, M., Edrington, C. S., and Fleming, F., "An induction machine emulator for high-power applications utilizing advanced simulation tools with graphical user interfaces," *IEEE Trans. Energy Conversion*, Vol. 27, No. 1, pp. 160–172, March 2012.
10. Hakan Akpolat, Z., Asher, G. M., and Clare, J. C., "Experimental dynamometer emulation of nonlinear mechanical loads," *IEEE Trans. Ind. Appl.*, Vol. 35, No. 6, pp. 1367–1373, November/December 1999.
11. Hassania, B., Sicard, P., and Ba-razzouk, A., "Solutions to typical motor load emulation control problems," *Electrimacs*, 18–21 August 2002.
12. Fernão Pires, V., Martins, J. F., and Amaral, T. G., "Web based teaching of electrical drives using a mechanical load simulator," *Proc. 34th IEEE IECON*, pp. 3545–3550, 10–13 November 2008.
13. Pedrasa, M. A. A., and Delfin, V. L. S., "Low cost mechanical load emulator," *IEEE Region 10 Conference*, pp. 1–3, 14–17 November 2006.
14. Pappas, C. H., and Murray, W. H., *Turbo C++ Total and Complete*, Macron Books do Brasil, 1991 (in Portuguese).
15. Advantech Co., Ltda., *PCL-711B—PC-MultiLab User's Manual*, Taiwan: Advantech Co., Ltd., August 1993.
16. Nery Trade and Representations Engineering Ltd, available at: www.nery.com.br
17. Davari, S. A., Khaburi, D. A., Wang, F., and Kennel, R. M., "Using full order and reduced order observers for robust sensorless predictive torque control of induction motors," *IEEE Trans. Power Electron.*, Vol. 27, No. 7, pp. 3424–3432, July 2012.

Appendix

```
/*
     ********************************************************************
  * Program      : LOADSIM
Description  : Generic motor load simulator, using the PCL 711B board
Version    : 3
Date              : 09/2012*
     ********************************************************************
/*
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <dos.h>
#define  DATABUFNUM 40
extern "C" pcl711(int, unsigned int *);
unsigned int param[20];                          /* Parameter Table
*/
unsigned int datain[DATABUFNUM], dataout[DATABUFNUM];/* Buffer of data to be converted
*/
unsigned int far * datin, * datout;
int i,tecla;                                /* Keyboard reading              */
float  nReal=0.0, tReal=0.0, VOut=0.0, nReal_ =0.0;
float  DataBuf[DATABUFNUM];
void conv_ad(void);                              /* A/D Conversion               */
void conv_da(void);                              /* D/A Conversion               */
void conv_ad(void);
```

```
{
        unsigned int i;
        datin = datain;
        param[0]  = 0;                          /* Board number                  */
        param[1]  = 0x220;                      /* I/O Base Address              */
        param[5]  = 10;                         /* Board sampling frequency = 2M /
                                                           (10 * 10) = 20 kHz */
        param[6]  = 10;
        param[7]  = 0;                          /* Trigger Mode, 0 : pacer triger */
        param[10] = FP_OFF(datin);              /* A/D Buffer A Offset           */
        param[11] = FP_SEG(datin);              /* A/D Buffer A Segment          */
        param[12] = 0;                          /* Buffer B Address,             */
        param[13] = 0;                          /* if not used, set at 0.        */
        param[14] = 1;                          /*Number of A/D conversions      */
        param[15] = 0;                          /* A/D conversion initial channel */
        param[16] = 0;                          /* A/D conversion final channel  */
        param[17] = 0;                          /* Channel gains, 0 : +/- 5V     */
           /* param[45] : Error code                                             */

        pcl711(3, param);                       /* Func 3 : BOARD PCL-711B
                                                   INITIALIZATION  FAILURE       */
        if (param[45] != 0) {
            clrscr();
           printf("A/D CONVERTER INITIALIZATION  FAILURE!");
            getch();
           exit(1);
           }
        pcl711(4, param);                       /* Func 4 : A/D Converter
                                                   Initialization                */
        if (param[45] != 0) {
            clrscr();
           printf("A/D CONVERTER INITIALIZATION  FAILURE!");
            getch();
             exit(1);
            }
        pcl711(5, param);                       /* Func 5 : A/D conversion begins */
        if (param[45] != 0) {
            clrscr();
           printf("A/D CONVERTER SOFTWARE DATA TRANSFER FAILURE ");
            getch();
             exit(1);
            }
        for (i = 0; i < param[14]; i++)         /* Sampled data                  */
        {
            DataBuf[i] = datain[i] & 0xFFF;
            DataBuf[i] =( (5.0 - (-5.0)) * DataBuf[i] / 4096) + (-5.0);
            /* (5 - (-5)) : A/D input range (-5V to 5V)
                    4096 : Range in A/D scale - 12 bit
                 DataBuf : A/D input data
                    (-5) : A/D scale beginning "-5" V                          */
        }
      nReal=(DataBuf[0]);
       tReal=(DataBuf[1]);
}
void conv_da()
{
        /*  D/A Conversion
        */
        datout=dataout;
        param[0]=0;                             /* Board number                  */
        param[1]=0x220;                         /* I/O Base Address              */
        param[20] = FP_OFF(datout);             /* D/A Buffer A Offset           */
        param[21] = FP_SEG(datout);             /* A/D Buffer A Segment          */
        param[22] = 0;                          /* Buffer B Address              */
        param[23] = 0;                          /* if not used, set at 0.        */
        param[24] = 1;                          /* Number of D/A conversions     */
        param[25] = 0;                          /* D/A conversion initial channel */
        param[26] = 0;                          /* D/A conversion final channel  */
```

```
        pcl711(3, param);                          /* Func 3 : ("PCL-711B CARD
                                                      INITIALIZATION FAILURE !")     */
        if (param[45] != 0) {
         clrwcr();
        printf("D/A CONVERTER INITIALIZATION FAILURE!");
            exit(1);
           getch();
            }
        pcl711(12, param);                         /* Func 12 : D/A converter
                                                      initialization                 */
        if (param[45] != 0) {
       clrwcr();
        printf("D/A CONVERTER INITIALIZATION FAILURE !!");
          exit(1);
         getch();
           }

        pcl711(13, param);                         /* Func 13 : D/A converter output */
        if (param[45] != 0) {
       clrwcr();
        printf("D/A CONVERTER OUTPUT FAILURE !");
         exit(1);
        getch();
           }
        dataout[0]=(4095*VOut);

}
/* main program */


void main(void)   /*Ensures that global variables do not return values */
{                 /* Declaration of variables */
   int   k1=0, k2=0;

   float wn = 0.0277, fc = 0.6, nru, nrum = 0.0,Thw = 0.0e-2, VOut_=0.0,
wnru=0.0;
   float wtrealu=0, ntrealu=0;

   //-------------------------------------------------------------
   int   NA = 100; // <---- Samples for calculation of the average(filter)
   //-------------------------------------------------------------
   int   NS = 1000; // <---- Screen interval update
   //-------------------------------------------------------------
   float kt = 0.6, tr = 0, terr=0, nnerr = 1e-4, kpt=0.051e-5; // <----
   //-------------------------------------------------------------
   float ajf = 0; // <----
   //-------------------------------------------------------------
   unsigned char Lajf = 0; // <----

   int   imenu = 2;

   auh cli;

   clrwcr();


   VOut = 0.0; //output voltage 0V.
   conv_ad();       /* A/D conversion */
   conv_da();


if (kbhit()) tecla=getch();
     else tecla = 0;
if (fabs(nReal-nReal_)>=Thw)
        nReal_ = nReal;

     nru      = nReal_ / 4.88;
     wtrealu += tReal;
```

```
      wnru    += nru;

if (++k1>=NA)
      {
         nrum  = wnru /(float) NA;
         if (nrum<0) nrum = 0.0;
         wnru    = 0.0;
         ntrealu = (wtrealu / (float) NA) / 1.45;
         wtrealu = 0;
         k1      = 0;
      }
      switch(imenu)
      {
         case 1:
                VOut_ = (1-wn*fc)*nrum;
                tr = kt;
                break;
         case 2: VOut_ = (1-wn*fc*nrum)*nrum;
                tr = kt * nrum;
                break;
         case 3: VOut_ = (1-wn*fc*pow(nrum,2))*nrum;
               tr = kt * (nrum*nrum);
               break;
      }
   clrscr();

   gotoxy(1,18);
   cprintf("fc  (+) = <A> / fc  (-) = <Z> / fc  <- default = <Q>\n\r");
   cprintf("kt  (+) = <S> / kt  (-) = <X> / kt  <- default = <W>\n\r");
   cprintf("kpt (+) = <D> / kpt (-) = <C> / kpt <- default = <E>\n\r");
   cprintf("Fine adjustment (on/off) = <F>\n\r");
   cprintf("imenu <1>, <2>, <3>");

   do {

      if (kbhit()) tecla=getch();
       else tecla = 0;

      conv_ad();       /* A/D Conversion */


      if (fabs(nReal-nReal_)>=Thw)
        nReal_ = nReal;

      nru      = nReal_ / 4.88;
       wtrealu += tReal;
      wnru    += nru;

      if (++k1>=NA)
      {
         nrum  = wnru /(float) NA;
         if (nrum<0) nrum = 0.0;
         wnru    = 0.0;
         ntrealu = (wtrealu / (float) NA) / 1.45;
          wtrealu = 0;
         k1      = 0;
      }
      switch(imenu)
      {
         case 1:
                VOut_ = (1-wn*fc)*nrum;
                  tr = kt;
               break;
         case 2: VOut_ = (1-wn*fc*nrum)*nrum;
                  tr = kt * nrum;
                  break;
         case 3: VOut_ = (1-wn*fc*pow(nrum,2))*nrum;
                  tr = kt * (nrum*nrum);
```

```
            break;
      }
//-------------------------------------------------------------------------
      if (Lajf)
      {
         terr = tr - mtrealu;
         if (fabs(terr) > mmerr)
         {
            ajf += terr * kpt;
         }
      }
      else
      {
         VOut = VOut_;
         ajf = 0;
      }

      VOut = VOut_ - ajf;

      if (VOut>1) VOut=1;
      if (VOut<0) VOut=0;

            conv_da();    /* D/A Conversion */
//-------------------------------------------------------------------------
      if (++k2>=NS)
      {
         k2 = 0;
         gotoxy(1,1);
         cprintf("imenu=%d,   \n\rfc=%2.3f,   \n\rwm=%2.3f      \n\r",  imenu, fc, wm);
          cprintf(" Speed Input voltage     = %2.3f     \n\r", nru);
          cprintf(" Average Speed Input voltage     = %2.3f     \n\r", nrum);
          cprintf("Output voltage    = %2.3f     \n\r", VOut  * 10.0);
         cprintf("Reference torque    = %2.3f     \n\r, tr);
         cprintf("Torque (V)    = %2.3f     \n\r, treal);
         cprintf(" Average Torque  pu    = %2.3f     \n\r, mtrealu);
         cprintf("Kpt                  = %2.7f     \n\r, kpt);
         cprintf("Fine adjustment   = %d     \n\r, Lajf;
         cprintf(" Fine adjustment          = %2.6f        \n\r, ajf);
         cprintf("Error torque   = %2.6f     \n\r, terr);
          }
      switch (tecla)
      {
         case 65:  case 97:  fc += 1e-3; tecla = 0; break;
         case 90:  case 122: fc -= 1e-3; tecla = 0; break;
         case 81:  case 113: fc = 1;     tecla = 0; break;

         case 83:  case 115: kt += 1e-3; tecla = 0; break;
         case 88:  case 120: kt -= 1e-3; tecla = 0; break;
         case 87:  case 119: kt = 0;     tecla = 0; break;

         case 68:  case 100: kpt += 0.01e-5; tecla = 0; break;
         case 67:  case 99 : kpt -= 0.01e-5;
                             if (kpt<0) kpt = 0;
                             tecla = 0; break;
         case 69:  case 101: kpt = 1;     tecla = 0; break;

         case 49: imenu = 1; tecla = 0; break;
         case 50: imenu = 2; tecla = 0; break;
         case 51: imenu = 3; tecla = 0; break;

         case 70: case 102: Lajf = !Lajf; tecla = 0; break;
}
  } while(tecla!=27);   /* program interruption by ESC */

  VOut = 0.0; //Output voltage adjustment 0V.
  conv_da();
  asm sti;
}
```

APÊNDICE C

Procedimento para operação do implemento

UNIFEI *LEPCH*	*INSTRUÇÃO DE TRABALHO*	*IT-121*
	Operação da bancada – Avaliação de MIT – Inversores de frequência	Emissão: 11/02/2013 Revisão: Nº 01 Folha 89 de 93

1-OBJETIVO

Este documento estabelece o procedimento necessário para a correta operação da bancada – MIT x Inversor Siemens.

2-ABRANGÊNCIA E RESPONSABILIDADE

Este documento aplica-se na operação da bancada montada no LEPCH - UNIFEI, sob a responsabilidade do gerente técnico.

3-DEFINIÇÕES

O Painel de ligação QG2 tem a finalidade de alimentar com tensões variadas as bancadas do laboratório, permitindo alimentar com tensões de 127 a 1300 V, permitido que sejam realizados diversos tipos de testes com múltiplas tensões.

4-TEXTO NORMATIVO

4.1- Equipamento Necessário:

Painel – QG2

4.2- Manutenção Preventiva:

Realizada 01 vez a cada semestre.

4.3- Procedimento de Verificação:

NÃO APLICÁVEL.

4.4- Procedimento de Ensaio:

4.4.1. Energização da bancada de testes:

Antes de começar a operação, verificar se o transformador TR1 (Fig. 1) esta conectado na conexão entrada 220V – tap 1 e saída 440V – tap 3

- Pressionar a Botoeira [AC] – [Liga]
- Colocar a chave [AC LIGA] localizada no interior do painel na posição [A], Fig. 3
- Pressionar a botoeira [CEMIG] (Fig. 4)

- Pressionar a botoeira com indicação [> 220] (Fig. 4)

4.4.2. Desligamento da bancada de testes:

- Pressionar a Botoeira [AC] e a botoeira [Desliga] (Fig. 4)

4.4.3. **<u>Atuação no PC</u>**, onde se encontra instalado a placa de conversão Advantech e o software em linguagem C++:

4.4.3.1 Após a energização do PC executar os seguintes passos:

a) Entrar no modo DOS pressionar **[F2]**, em seguida pressionar continuamente a tecla **[F8]**, em seguida selecionar a opção **[5];**
b) Digitar no prompt DOS: **TC**;
c) Para entrar no programa, Pressionar: **[File] [Open] [CMS] [NC.CPP]**.

4.4.3.2 Execução do software C++

a) Compilação e execução – **[ctrl] [F9]**
b) Escolha do tipo de variação de carga: No menu de opções

disponível no rodapé da tela aberta, escolher as opções a seguir:

1 – Torque Constante;
2 – Torque Linear;
3 – Torque Quadrático.

4.4.3.3 Alteração do ganho

Atuar através da tecla "D"(kpt) no teclado do PC

4.4.4 Aquisição e amostragem de dados pelo software LabView

a) Ligar o Desktop, carregando o Windows
b) No Desktop selecionar o programa **[Rotação 8]**

4.4.4.1 Execução do programa

a) [OPERATE], barra superior;
b) [RUN].

4.4.4.2 Gravação

a) Tecle [Ok], grava ponto no gráfico;
b) Tecle em [clear] para limpar os pontos;
c) Entra no [PACH]
d) Clicar no botão corrente folder, no espaço em branco digitar um nome do arquivo a ser salvo.

4.4.5 Atuação nos inversores (Liga)

a) Ligar fonte de alimentação 1 e 2
b) Verificar e ajustar o potenciômetro da **chave 1** para a posição **0**;
c) Ajustar a **chave 2** para a posição automático;
d) Fechar o disjuntor dos capacitores;
e) Ligar o ventilador
f) Ligar o **inversor 1**;
g) Na **chave 1** atuar no potenciômetro para seleção das frequências desejadas: Para aumentar a frequência atuar livremente com a velocidade que desejar. Para diminuir a frequência atuar lentamente;
h) Atuar na tecla **[F]** do **PC**, colocando no software a posição ajuste fino.
i) Ligar o **inversor 2**;
j) Monitorar e aquisitar através do software **labview.**

4.4.6 Atuação nos inversores (Desliga)

a) Na **chave 1,** caso a frequência estiver no valor máximo, atuar lentamente no potenciômetro, reduzindo a frequência para o mínimo possível;
b) Desligar os inversores simultaneamente;
c) Abrir o disjuntor dos capacitores.

5-REFERÊNCIA

NÃO APLICÁVEL

6-ANEXOS

NÃO APLICÁVEL

7-HISTÓRICO DE ALTERAÇÕES

REVISÃO	DATA	HISTÓRICO
01	25/03/2013	Documento inicial.

8-ELABORAÇÃO

_____
xxxxxxxxxxxxxxxxxxxxxxxxxxxx
Gerente Técnico

9-APROVAÇÃO

_____
xxxxxxxxxxxxxxxxxxxxxxxx
Gerente da Qualidade