AMBIENTE COMPUTACIONAL PARA SUPERVISÃO E CONTROLE DE PROCESSOS INDUSTRIAIS[‡]

Rodrigo MAIA; Ginalber SERRA; Luís NUNES
Laboratório de Inteligência Computacional e Controle - LabICC
CEFET/MA-DEE, Avenida Getúlio Vargas, nº 04 - Monte Castelo - São Luis-MA - CEP 65030-000.
ginalber@dee.cefet-ma.br , rodrigoframa@hotmail.com

RESUMO

Neste trabalho é proposta uma metodologia para o desenvolvimento de um Ambiente Computacional, via linguagem de programação MATLAB, com Interface Homem-Máquina para supervisão e controle de processos industriais. Esta metodologia permite adquirir dados da planta e especificações de projeto de controladores PI e PID, resultando em visualizações de gráficos acessados a partir de uma interface gráfica que permite ao usuário supervisionar, analisar e interagir com os dados dos processos.

Palavras-chave: Automação, Controle Digital, Sistemas Supervisórios, Processos Industriais.

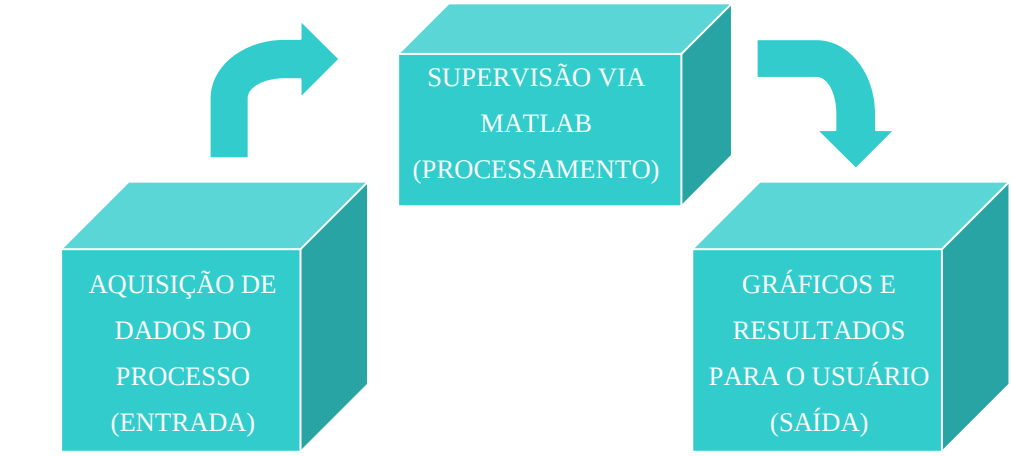
1. INTRODUÇÃO

No mercado de desenvolvimento existe uma área de muito impacto científico-tecnológico, que é a de aplicações para supervisão e controle de processos industriais. A grande maioria dos analistas e programadores estão habituados ao desenvolvimento de aplicações comerciais, utilizando banco de dados e linguagens convencionais, e desconhecem essa área de aplicação. As aplicações de supervisão e controle de processos industriais podem ser encontradas em todas as empresas que produzem um produto ou serviço, em que existe a necessidade de um controle rígido dos processos industriais envolvidos. Por isso os sistemas supervisórios permitem que sejam monitoradas e rastreadas informações de um processo produtivo ou instalação física. Tais informações são coletadas através de equipamentos de aquisição de dados e, em seguida, manipulados, analisados, armazenados e, posteriormente, apresentados ao usuário. Estes sistemas também são chamados de SCADA (Supervisory Control and Data Aquisition) ou IHM (Interface Homem-Máquina) [1]. Os primeiros sistemas SCADA, basicamente telemétricos, permitiam informar periodicamente o estado corrente do processo industrial, monitorando sinais representativos de medidas e estados de dispositivos, através de um painel de lâmpadas e indicadores, sem que houvesse qualquer interface com o operador. Atualmente, os sistemas de automação industrial utilizam tecnologias de computação e comunicação para automatizar a supervisão e controle dos processos industriais [2][3]. Com a evolução da tecnologia, os computadores passaram a ter um papel importante na supervisão e controle dos sistemas, coletando e tornando disponíveis os dados do processo. O acesso remoto aos dados facilita tanto o monitoramento quanto o controle do processo, fornecendo, em tempo útil, o estado atual do sistema através de gráficos (interfaces homem-máquina), previsões ou relatórios, viabilizando tomadas de decisões, seja automaticamente ou por iniciativa do operador [4][5]. Neste trabalho é proposta uma metodologia para o desenvolvimento de um Ambiente Computacional, via linguagem de programação MATLAB, com Interface Homem-Máquina para supervisão e controle de processos industriais. Esta metodologia permite adquirir dados do processo através de equipamentos de aquisição de dados, via microcontrolador, e torná-los disponíveis para os usuários em visualizações de gráficos acessados a partir de uma interface gráfica que permite ao usuário supervisionar, analisar e interagir com os dados dos processos [6][7][8][9], conforme mostra a **Figura 1**. Os sistemas supervisórios têm se mostrado de fundamental importância na estrutura de gestão das empresas e indústrias, fato pelo qual deixaram de ser vistos como meras ferramentas operacionais, ou de engenharia, e passaram a ser vistos como uma relevante fonte de informação. Os

[‡] Projeto de pesquisa com apoio financeiro PIBIC/Júnior do Centro Federal de Educação Tecnológica do Maranhão.

sistemas ou ambientes computacionais de supervisão e controle de processos industriais automatizados desempenham três atividades principais:

- Comunicação amigável entre Homem-Máquina;
- Supervisão;
- Controle;



.

Figura 1 – Diagrama do Ambiente Computacional

2. SISTEMA SUPERVISÓRIO BASEADO EM MATLAB

Os sistemas supervisórios permitem que sejam monitoradas e rastreadas informações de um processo produtivo ou instalação física. Tais informações são coletadas através de equipamentos de aquisição de dados e, em seguida, manipulados, analisados, armazenados e, posteriormente, apresentados ao usuário. Estes sistemas também são chamados de SCADA (Supervisory Control and Data Aquisition). Os primeiros sistemas SCADA, basicamente telemétricos, permitiam informar periodicamente o estado corrente do processo industrial, monitorando sinais representativos de medidas e estados de dispositivos, através de um painel de lâmpadas e indicadores, sem que houvesse qualquer interface aplicacional com o operador. Atualmente, os sistemas de automação industrial utilizam tecnologias de computação e comunicação para automatizar a monitoração e controle dos processos industriais, efetuando coleta de dados em ambientes complexos, eventualmente dispersos geograficamente, e a respectiva apresentação de modo amigável para o operador, com recursos gráficos elaborados (interfaces homem-máquina) e conteúdo multimídia. Internamente, os sistemas SCADA geralmente dividem suas principais tarefas em blocos ou módulos, que vão permitir maior ou menor flexibilidade e robustez, de acordo com a solução desejada.

Em linhas gerais, podemos dividir essas tarefas em:

- Núcleo de processamento;
- Comunicação com PLCs/RTUs;
- Gerenciamento de Alarmes;
- Históricos e Banco de Dados;
- Lógicas de programação interna ou controle;
- Interface Gráfica;

A regra geral para o funcionamento de um sistema supervisório parte dos processos de comunicação com os equipamentos de campo, cujas informações são enviadas para o núcleo principal do software. O núcleo é responsável por distribuir e coordenar o fluxo dessas informações para os demais módulos, até

chegarem na forma esperada para o operador do sistema, na interface gráfica ou console de operação com o processo, geralmente acompanhadas de gráficos, animações, relatórios, etc., de modo a exibir a evolução do estado dos dispositivos e do processo controlado, permitindo informar anomalias, sugerir medidas a serem tomadas ou reagir automaticamente. As tecnologias computacionais utilizadas para o desenvolvimento dos sistemas supervisórios têm evoluído bastante nos últimos anos, de forma a permitir que, cada vez mais, aumente sua confiabilidade, flexibilidade e conectividade, além de incluir novas ferramentas que permitem diminuir cada vez mais o tempo gasto na configuração e adaptação do sistema às necessidades de cada instalação.

2.1. Interface gráfica

Um programa de computador é um conjunto de instruções que representam um algoritmo para a resolução de algum problema. Estas instruções são escritas através de um conjunto de códigos (símbolos e palavras). Este conjunto de códigos possui regras de estruturação lógica e sintática própria. Dizemos que este conjunto de símbolos e regras formam uma linguagem de programação. Um programa de computador possui várias utilidades de acordo com a necessidade do programador. Atualmente a maioria dos programas possui uma interface gráfica amigável, ou seja, uma tela com ícones clicáveis que dão acesso as funções do sistema. Portanto, o usuário não precisa conhecer a linguagem do computador para executar determinados comandos, basta clicar sobre o comando escolhido, na forma de uma palavra ou de uma figura, ou digitar uma combinação específica de teclas para que o programa envie uma mensagem para o processador causando a execução de uma certa tarefa. Uma interface gráfica com o usuário é o ponto de contato ou método de interação entre uma pessoa e um computador ou programa de computador. É o método usado pelo computador e pelo usuário para trocarem informações. O computador exibe informações e imagens gráficas na tela para a visualização do usuário, já este se comunica com o computador utilizando um dispositivo de entrada, como um teclado, mouse ou microfone. A interface com o usuário define a aparência e a facilidade de uso do computador. Neste trabalho utilizaremos à linguagem de programação MATLAB, pois além de ser pouco explorada, é uma linguagem de programação com características mais avançadas e é mais prática de usar como sistema supervisório do que linguagens de programação tais como BASIC, Pascal ou C por apresentar um ambiente rico para a visualização de dados, graças a sua vantagem de ter uma poderosa capacidade gráfica;

2.2. Guide do MATLAB

GUIDE é uma ferramenta GUI (Interface gráfica do utilizador) fornecida pelo MATLAB que foi projetada para que o usuário construa suas próprias interfaces com maior facilidade e rapidez. A função guide contém ferramentas para criar, instalar, alinhar e alterar o tamanho de objetos uicontrol, um editor de propriedades e inspetor que lista propriedades de objetos e que permite ao usuário modificar essas propriedade interativamente, um menu editor para a edição interativa e para modificar os menus pull-down e contextuais definidos pelo usuário, conforme mostra o fluxograma na **Figura 2**. GUIDE proporciona uma abordagem interativa ao desenvolvimento de interfaces e é ótima para ajustar a geometria de uma interface. Para criar uma GUI através do GUIDE, o usuário deve entrar com o comando guide no prompt do MATLAB, em seguida aparece uma tela na qual o usuário deve escolher dentre as opções a que é de seu interesse.

O GUIDE armazena uma interface em dois arquivos, que são gerados quando o usuário salva e interpreta uma interface:

• Arquivo-FIG : um arquivo com a extensão .fig, contêm uma descrição completa sobre as figuras presentes na interface, tais como sliders, caixas de texto, menu popup e assim por diante. Quando você deseja fazer alguma modificação na aparência da interface, esta deve ser feita no arquivo.fig.

• Arquivo-M : um arquivo com a extensão .m, contem o código que controla a interface, incluindo as callbacks dos componentes. A princípio este arquivo está em branco, e cabe ao usuário inserir os comandos que sejam necessários para o funcionamento da interface.

Uma interface gráfica construída no MATLAB, pode se tornar uma ferramenta extraordinária na demonstração e exploração de uma ampla variedade de dados técnicos, e isto se deve ao fato que o MATLAB dispõe de poderosos recursos numéricos e gráficos.

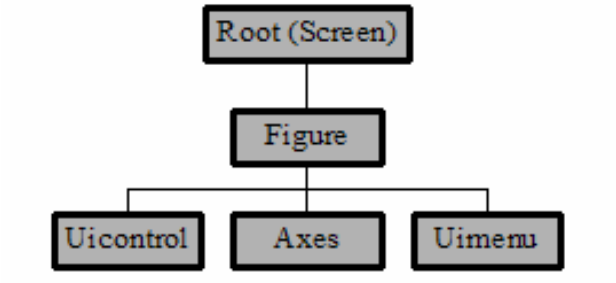


Figura 2 – Fluxograma do funcionamento do Guide

2.3. Handle Graphics

Handle Graphics é o nome dado a uma coleção de rotinas de baixo nível que cumprem a tarefa de gerar gráficos no MATLAB. Handle Graphics proporciona um alto grau de controle sobre os gráficos. Suas funções e opções normalmente estão ocultas dentro de arquivos M de nível mais alto. Handle Graphics permite que o usuário personalize aspectos de gráficos que não podem ser gerados por meio dos comandos e funções de alto-nível. Handle Graphics se baseia na idéia de que cada componente de um aspecto visual do MATLAB é um objeto, que cada objeto possui um único identificador, ou handle, associado a ele e que cada objeto possui propriedades que podem ser modificadas quando se desejar.

2.3.1. Handle de objetos

Suponha que o usuário tenha nas três figuras abertas, com gráficos em duas delas, o usuário quer alterar a cor de uma linha em um dos eixos dos gráficos. Como o usuário identificaria a linha que desejar alterar? No MATLAB, cada objeto possui um identificador associado, chamado Handle. Cada vez que um objeto é criado, um Handle único é criado para ele. Todas as funções do MATLAB que criam objetos fornecem um handle ou um vetor coluna de Handles para cada objeto criado. Alguns gráficos são compostos por mais de um objeto.

A seguir mostra as funções de Handle Graphics do MATLAB conforme a **Tabela 1**:

Tabela 1 – Tabela das funções de Handle Grafics do MATLAB

Função	Descrição
Get	Obtêm as propriedades do objeto
Set	Define as propriedades do objeto
Gcf	Obtêm a figura atual
Gca	Obtêm os eixos atuais
Gco	Obtêm o objeto atual
Findobj	Encontra os objetos com propriedades especificadas
Findall	Encontra objetos ocultos e não-ocultos com propriedades especificadas
Allchild	Retorna Handles de descendentes ocultos e não-ocultos de um objeto
Copyobj	Faz cópia de objetos
Root	Objeto raiz (root) do computador, handle = 0
Figure	Cria uma janela Figure
Axes	Cria eixos
Line	Cria linhas
Text	Cria textos
Patch	Cria um retalho
Rectangle	Cria um retângulo
Surface	Cria uma superfície
Image	Cria uma imagem

Light	Adiciona efeitos de luz
Uicontrol	Cria controle de interface com usuário
Uimenu	Cria menu de interface com usuário
Uicontextmenu	Cria menu de interface contextual com usuário
Clf	Apaga a figura atual
Cla	Apaga os eixos atuais
Ishandle	Verdadeiro para argumentos que são handles de objetos
Delete	Remove um objeto
Close	Fecha a janela Figure
Refresh	Redesenha a Figura atual
Gcbo	Ativa a manipulação do objeto atual
Gcbf	Ativa a manipulação da figura atual
Closereq	Fecha a figura
Newplot	Cria eixos a partir da propriedade NextPlot
Reset	Restabelece propriedades de objeto aos valores originais

2.4. Estilos de Controle

O MATLAB aceita dez tipos de uicontrol (controle de interface com usuário) também chamados de objetos, vejamos quais são:

a. *Botões simples (pushbuttons)* - Os pushbuttons, às vezes chamados de botões de comando ou simplesmente botões, são objetos de tela pequenos, retangulares que normalmente contém um rótulo de texto. Selecionado um botão simples com o mouse movendo-se o ponteiro para cima o objeto e pressionando-se o botão do mouse, faz-se com que o MATLAB execute a ação definida pala callback string do objeto. Imediatamente depois que um botão simples é acionado, ele retorna a seu estado inicial. Os botões simples costumam ser usados para executar uma ação e não para mudar um estado ou fixar um atributo.

b. Botões de chave (toggle Buttons) - Os botões de chave são idênticos aos botões simples exceto pelo fato de alternarem entre dois estados, para cima e para baixo, quando pressionados. A propriedade "value" de um botão de chave é fixada como igual ao valor especificado pela propriedade "Max" quando o botão está para baixo ou pressionado. E fixada como igual ao valor da propriedade "Min" quando o botão esta para cima.

c. *Botões de rádio (radio buttons)* - Os botões de rádio consistem em botões contendo um rótulo e um pequeno círculo ou diamante à esquerda do texto de rótulo. Quando selecionado, o círculo ou diamante é preenchido e a propriedade "Value" é fixada como o valor especificado pela propriedade "Max", que por padrão é 1; quando não selecionado, o indicador é removido e a propriedade "Value" é fixada como o valor especificado pela propriedade "Min", que é 0 por padrão. Rádio buttons costuma ser usados para selecionar uma de um grupo de opções mutuamente exclusivas. Para reforçar essa exclusividade, todavia, a callback string para cada botão de rádio deve desativar todos os outros botões de grupo fixando o "Value" de cada um deles em 0 ou com o valor atribuído a propriedade "Min". Isso, no entanto,e apenas uma convenção. Os botões de rádio podem ser usados em alternância coma as caixas de controle, caso haja interesse.

d. *Caixas de controle (Checkbox)* - As caixas de controle consistem em botões com um rótulo e uma pequena caixa quadrada à esquerda do texto do rótulo. Quando ativado, o controle alterna entre preenchido e limpo. No primeiro caso, a caixa é preenchida ou contem um "X", dependendo da plataforma e a propriedade "Value" recebe o valor especificado pela propriedade "Max", que por padrão é 1; quando limpo, a caixa fica vazia e a propriedade "Value" recebe o valor especificado pela propriedade "Min", que por padrão é 0. As caixas de controle costumam ser usadas para indicar o estado de uma opção ou atributo. Normalmente são objetos independentes, mas podem ser usados em alternância com botões de radio, caso haja interesse.

e. *Caixas de edição (Edit Text)* - Caixas de texto editáveis que exibem texto em uma caixa, de maneira que o usuário possa modificar ou substituir a string de texto dinamicamente. A nova string de texto se torna então, disponível na propriedade "string" do uicontrol. Caixas de texto editáveis costumam permitir que o usuário introduza textos ou um valor, e podem conter uma ou mais linhas de texto. Uma caixa de texto editável de linha única aceita uma linha de texto do usuário enquanto uma caixa de texto é concluída pressionando a tecla Return. A entrada de textos de múltiplas linhas é concluída com as teclas Control-Return. As caixas de texto de múltiplas linhas são criadas fixando-se os valores das propriedades "Max" e "Min" em números tais que Max − Min > 1. A propriedade "Max" não especifica o numero máximo de linhas. As caixas de texto de múltiplas linhas podem ter um número ilimitado de linhas. Strings de múltiplas linhas podem ser especificadas como vetores celulares de texto ou vetor de caracteres.

f. *Caixas de texto estáticas (Static Text)* - Caixas de texto estáticas são controles que simplesmente exibem uma string de texto como determinado pela propriedade "String". Caixas de texto estáticas costumam ser usadas para exibir rótulos, informação ao usuário ou valores atuais. Caixas de texto estáticas são estáticas; o usuário não pode mudar dinamicamente o texto exibido. O texto só pode ser alterado mudando-se a propriedade "String". Strings de texto são centradas no topo da caixa de texto. Strings de texto maiores que a largura da caixa de texto passam por word-wrap; ou seja, linhas múltiplas são exibidas com as linhas quebradas entre palavras quando possível. Se a altura da caixa de texto é pequena demais para a string de texto, parte do texto não será visível. Strings de múltiplas linhas podem ser especificadas como um vetor celular de texto ou com um vetor de caracteres.

g. *Sliders* - Sliders, ou controles deslizantes, consistem em três partes distintas: o canal, ou a área retangular representando o intervalo de valores de objetos válidos; o indicador dentro do canal representando o valor atual do slider; e setas em cada extremidade do canal. Sliders costumam ser usados para selecionar um valor de um intervalo de valores. Muitas vezes os sliders são acompanhados de objetos de texto separados usados para exibir rótulos, o valor atual do slider e limites de intervalo.

h. *Caixas de listagem (List Box)* - As caixas de listagem parecem caixas de texto de múltiplas linhas que permitem que os usuários selecionem itens individuais ou múltiplos de uma lista com um clique de mouse. Itens de lista individuais são especificados por vetores celulares de texto, por um vetor de string com espaços em branco, como separações, ou por uma única string com um caractere de barra vertical usado para separar as entradas da lista. Quando um item de lista é selecionado com um clique de mouse, a propriedade "value" é atualizada com o índice do item selecionado.

i. *Menu popup (Popupmenu)* - Os menus popup costumam ser usados para apresentar uma lista de escolhas mutuamente exclusivas ao usuário. Quando fechado, um menu popup aparece um retangulo ou um botão contendo o rótulo da seleção atual com um pequeno retângulo ou seta apontando para baixo ao lado do rótulo para indicar que o objeto é um menu popup. Quando o ponteiro está colocado sobre um controle popup e o botão do mouse é pressionado, outras escolhas aparecem. Movendo-se o ponteiro para uma escolha diferente e soltando-se o botão do mouse, o menu popup se fecha e exibe uma nova seleção.

j. *Axes* – Este objeto é usado para visualização de gráficos.

2.4.1. Property Inspector

Quando o usuário clica duas vezes sobre um dos uicontrol, têm-se a possibilidade de modificar várias propriedades do mesmo. As propriedades mais importantes são:

a. *Callback* – Sem dúvida, a configuração mais importante. É aí que se define o que será feito quando o objeto (uicontrol) for acionado. Se o usuário deseja colocar uma função para calcular o valor de uma variável qualquer e que se queira executar esta função após o usuário clicar em um botão simples(Push Button), e ainda que esta função tenha o nome de ''cálculo''. Deste modo, o usuário deve colocar no callback do botão simples o nome da função a ser executada.Assim, quando acionado o botão, chama-se e executa-se a função ''cálculo''. Mais de uma função pode ser chamada por um botão. O callback pode conter linhas de código e o usuário pode colocar o que quer pra ser acionado no botão;

b. *Tag* – É a segunda opção mais importante. A Tag é o nome do objeto. Inúmeras vezes faz-se necessário fazer referência a um objeto. Para isto utiliza-se a Tag, ou nome deste objeto. Ao se abrir uma tela, por exemplo, deseja-se que um determinado botão fique desativado. Assim, a programação é feita referindo-se à desabilitação do objeto que possui uma determinada Tag, nome.

c. *Backgroundcolor* – Define-se a cor de fundo do objeto.

d. *Enable* – Habilita/desabilita objeto.

e. *Fontsize* – escolhe o tamanho da fonte do texto escrito no objeto.

f. *FontWeight* – escolhe-se letra normal ou tipo negrito (demi), por exemplo.

g. *ForeGroundColor* – define-se a cor do texto que vai no objeto.

h. *ListboxTop* – No caso das caixas de lista de opções, define-se quantas opções podem ser selecionadas no máximo e no mínimo.

i. *Position* – Local onde são definidos o tamanho (altura e largura) e a posição do objeto (relacionada aos eixos x e y).

j. *String* – É o que fica escrito no campo do objeto.

k. *Style* – Neste local pode-se mudar a funcionalidade do objeto para qualquer outra das mostradas na barra de ferramentas. Um botão pode virar um frame, por exemplo.

l. *TooltipString* – Define-se o texto que deve aparecer quando o cursor do mouse se aproximar do objeto. Normalmente, é um resumo da função do mesmo.

m. *Visible* – Pode deixar o objeto visível ou invisível.

3. RESULTADOS COMPUTACIONAIS

Para a elaboração dessa interface gráfica, fez-se uso do referido ambiente de programação MATLAB pelo estudo do funcionamento do GUIDE. O funcionamento desta interface está intrinsecamente relacionado com o projeto de controle PI/PID baseado nas especificações das margens de ganho e fase. Por meio desta, é possível projetarmos o controlador a ser utilizado (PI/PID), uma vez conhecida as especificações das margens de ganho e fase e a planta a ser controlada. Dessa forma, podemos obter os parâmetros para um controlador PI (proporcional e integral) ou PID (proporcional, integral e derivativo), cujos parâmetros devem ser ajustados de modo a proporcionar um desempenho satisfatório da planta de primeira ou segunda ordem, respectivamente. A partir da **Figura 4**, ao entrar com os valores das margens de ganho e fase, o controlador a ser projetado e os parâmetros da planta envolvida, o usuário obterá como resposta os parâmetros do controlador selecionado. Ao obter tais dados, é possível ainda visualizar o desempenho do controlador e a planta que é controlada, por meio do botão “Resposta Temporal”, que traça o gráfico da resposta desse sistema a partir do Diagrama de Bode. Para complementar esse método, é viável a simulação desse sistema de controle a partir do botão “SIMULINK”, que por meio de diagramas em bloco, esquematiza o sistema de controle em malha-fechada, formado pelo controlador (PI/PID) já projetado e uma dada planta. Na **Figura 5** tem-se um exemplo do funcionamento da interface, onde o usuário entra com os valores das margens de ganho e fase e escolhe o tipo de controlador a ser projetado, neste caso foi utilizado o controlador PI (proporcional e integral); então disponibilizou-se a entrada de dados dos parâmetros da planta de primeira ordem: Ganho(Kp), Constante de tempo(t), Atraso de Transporte (L), ficando impossibilitado a entrada de valores da planta de segunda ordem. Após definir os valores da planta de primeira ordem, habilitando-se o botão “Cálculo”, a interface carrega os valores que o usuário especificou e calcula, mostrando os resultados dos parâmetros: Ganho(Kc) e Tempo Intergral (Ti), tendo ainda a opção de visualizar o desempenho do controlador e a planta que é controlada, por meio do botão “Resposta Temporal”, que traça o gráfico da resposta desse sistema. A simulação desse sistema de controle pode-se ter também a partir do botão “SIMULINK”, que por meio de diagramas em bloco, esquematiza o sistema de controle em malha-fechada, formado pelo controlador PI já projetado. Na **Figura 6**, o usuário entra com os valores das margens de ganho e fase, e opta pelo controlador PID (proporcional, integral e derivativo), tendo assim disponível a entrada de parâmetros da planta de segunda ordem: Ganho(Kp), Constante de tempo(t), Constante de tempo(t’), Atraso de Transporte(L) não podendo entrar com valores da planta de segunda ordem. Habilitando-se o botão “Cálculo”, a interface carrega os valores que o usuário especificou e calcula, mostrando os resultados dos parâmetros para controlador PID: Ganho(Kc) e Tempo Intergral (Ti) e Tempo Derivativo(Td), tendo ainda a opção de visualizar o desempenho do controlador e a planta que é controlada, por meio do botão “Resposta Temporal”, que traça o gráfico da resposta desse sistema. A simulação desse sistema de controle pode-se ter também a partir do botão “SIMULINK”, que por meio de diagramas em bloco, esquematiza o sistema de controle em malha-fechada, formado pelo controlador PID já projetado e a planta de segunda ordem.

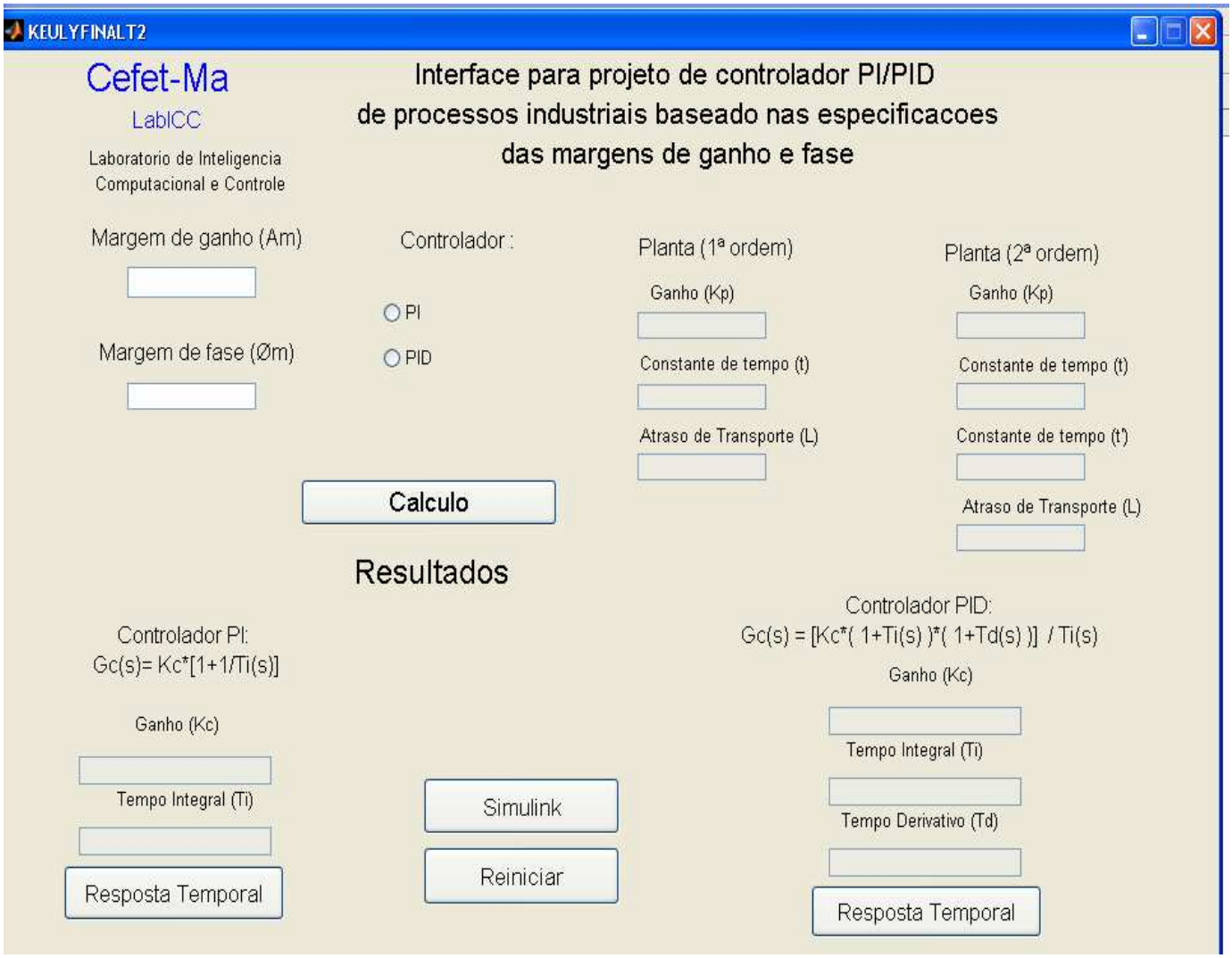


Figura 4 - Visualização supervisório via MATLAB.

Figura 5 - Visualização supervisório para projeto de controle PI.

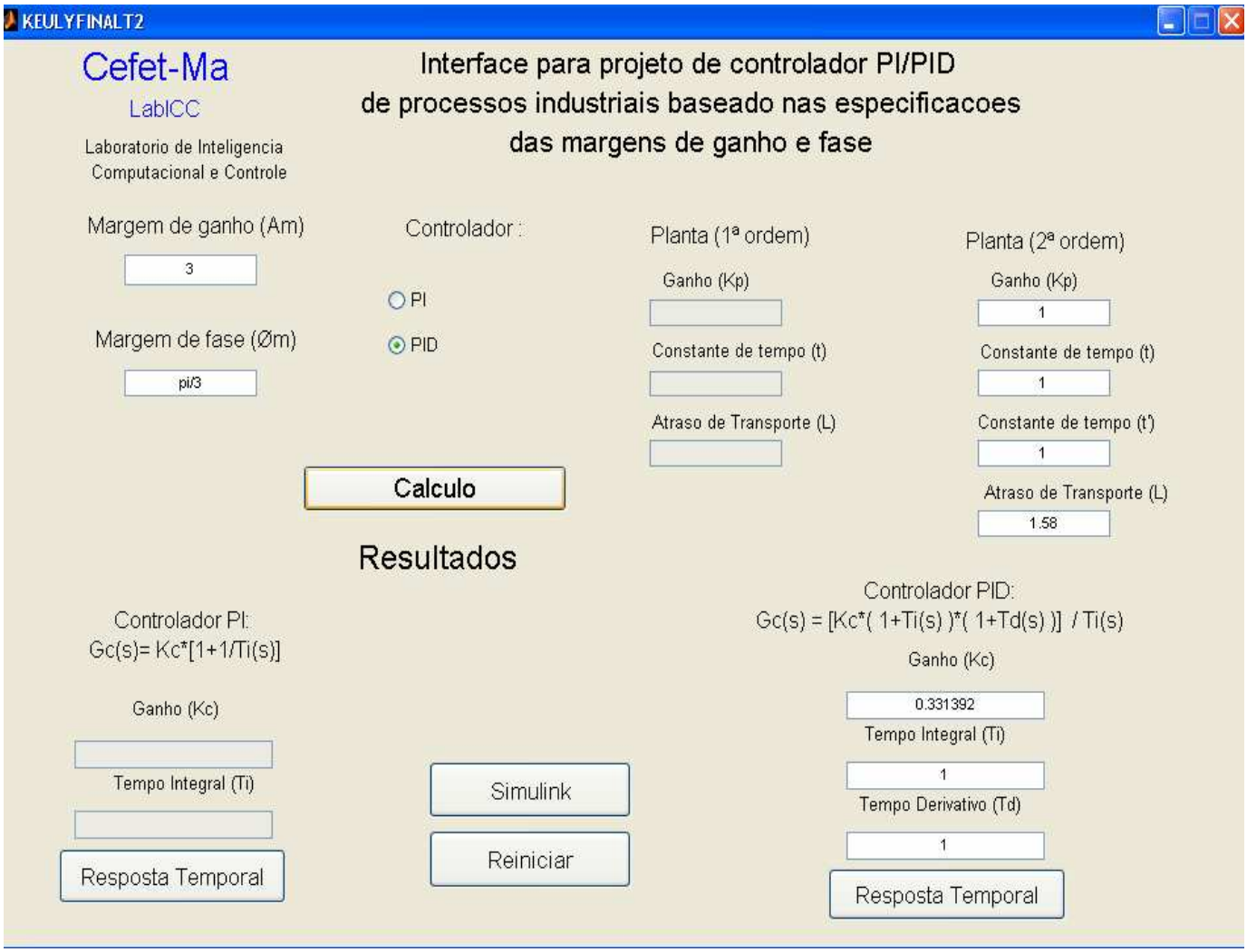


Figura 6 - Visualização supervisório para projeto de controle PID.

4. CONCLUSÃO

Neste trabalho foi proposta uma metodologia para o desenvolvimento de um Ambiente Computacional, via linguagem de programação MATLAB, com Interface Homem-Máquina para supervisão e controle de processos industriais. Esta metodologia permitiu inserir dados da planta, bem como os critérios de desempenho para projeto de controladores PI e PID. Ainda, através de botões, pode-se visualizar a resposta temporal do sistema de controle e a simulação em ambiente SIMULINK para visualizações de gráficos acessados a partir de uma interface gráfica que permite ao usuário supervisionar, analisar e interagir com os dados da planta.

REFERÊNCIAS

[1] SILVA, A. P. G., SALVADOR, M., ***O que são sistemas supervisórios?*** http://www.elipse.com.br/download/artigos/rt025.04.pdf <Acessado em 20/05/2008>.

[2] GONZÁLEZ, D. J. D.; HERRERA, M. M.; BENÍTEZ, S. G.; LOPÉZ, V. V. ***Automación y control: prácticas de laboratorio***. McGraw-Hill, 2004.

[3] N. JUNIOR, C. L.; YONUJARAMA, T. ***Inteligência artificial em controle e automação***. 1.ed., São Paulo: Edgard Blücher, 2000.

[4] BOTTURA, C. P. ***Princípios de controle e servomecanismos***. Rio de Janeiro: Guanabara Dois, 1982.

[5] KUO, B. C. ***Sistemas de Control Automático***. 7ª ed., Prentice-Hall, México, 1996.

[6] CHAPMAN, S.J. ***Programação em MATLAB para Engenheiros***. 1ªed., Thomson, 2003,

[7] GILAT, A. ***MATLAB an Introduction with Applications.*** 2ª ed., John Wiley, 2004.

[8] TONINI, A. M., SCHETTINO, D. N., ***MATLAB para Engenheiros***. 1ª ed, Centro universitário de belo horizonte 2002, http://academicos.cefetmg.br/admin/downloads/2101/ApostilaMatLab_UNI.pdf <Acessado em 20/05/2008>.

[9] MATSUMOTO, E. Y. ***MATLAB 7 : Fundamentos.*** 1ª ed., Èrica, 2004.

AGRADECIMENTOS

Os autores agradecem ao PIBIC/Júnior do Centro Federal de Educação Tecnológica do Maranhão (CEFET-MA) pelo apoio financeiro.