CENTRO UNIVERSITÁRIO DE BRASÍLIA – UniCEUB
FACULDADE DE CIÊNCIAS EXATAS E TECNOLOGIA – FAET
CURSO DE ENGENHARIA DA COMPUTAÇÃO
PROJETO FINAL DE GRADUAÇÃO

CONTROLE DE ACESSO DE VEÍCULOS AUTOMOTORES

ALUNO: ANDRÉ LUIZ DA CÂMARA MUNIZ
ORIENTADOR: M.C. GLEYSON AZEVEDO DA SILVA

BRASÍLIA, 2009

ANDRÉ LUIZ DA CÂMARA MUNIZ

CONTROLE DE ACESSO DE VEÍCULOS AUTOMOTORES

Orientador:
Prof. M.C. GLEYSON AZEVEDO DA SILVA

Monografia apresentada ao Centro Universitário de Brasília – UniCEUB, Faculdade de Ciências e Tecnologia – FAET, para obtenção do Título de Bacharel em Engenharia da Computação.

Brasília – DF, Junho de 2009.

AGRADECIMENTOS

Dedico este trabalho ao meu pai, Dr. Nilton Muniz da Silva, por ser meu fã incondicional. Capaz de acreditar e investir em todos os meus projetos e sempre esperar mais de mim do que eu mesmo. E ao Senhor Jesus, pois tenho aprendido que dele, por Ele e para Ele são todas as coisas, e sem Ele, nada poderia fazer.

"... *sem mim nada podereis fazer.*"(João 15:5)

RESUMO

Este trabalho tem por objetivo explicar os principais conceitos de automação industrial, uma vez que o projeto a ser apresentado consiste num sistema automatizado. Assim, após breve explanação sobre a área de automação, passando pelo contexto histórico e seus principais conceitos, o trabalho segue falando sobre motores de passo, e tecnologia OCR, de modo a preparar o leitor a entender a implementação proposta.

A idéia central do trabalho consiste, como o próprio nome diz, num controle de acesso para veículos automotores. De modo que o veículo ao se aproximar de uma cancela de estacionamento, entrará em contato com um sensor capaz de captar a sua presença, o qual aciona uma *webcam.* O registro da placa é identificado e comparado com um banco de dados pré-existente. Isso quer dizer que caso a placa do veículo esteja cadastrada no banco de dados, será permitida a sua entrada, caso não esteja cadastrada, o condutor do veículo acionará um interfone para chamar os profissionais competentes para realizar o cadastro e permitir a entrada. É importante frisar que a proposta do projeto não compromete o autor à implementação de um interfone para os casos de não reconhecimento da placa, ficando, todavia, a sugestão para implementações futuras.

O capítulo de Implementação foi descrito de forma que, caso o leitor sinta o desejo de reconstruir o protótipo do projeto, possa fazê-lo conforme explicação.

ABSTRACT

This work has the objective of explaining the main concepts about automatic systems. After some explanation about this point, passing by the historical context and its main concepts, the work keeps talking about step motors, and OCR technology, preparing the reader to understand the author's proposal.

The project main idea is a vehicle access control. When the vehicle gets to the parking, a sensor, able to detect its presence, activates a webcam. The register from the camera is identified and compared to a data bank previously prepared. It means that if the car's plate can be found in the data bank, its access will be allowed. If the car's plate isn't registered in the data bank, the driver must use an intercom to call the employees of the parking to register and allow the entrance. Here is important to emphasize that the project's author doesn't need to perform an intercom in cases of prevented admission of the car's plate, what could be a theme to future researches.

The execution chapter was described in a way that the reader feels enable to reconstruct the project prototype, according to the explanation.

LISTA DE EQUAÇÕES

LISTA DE SIGLAS

AC/CA: *corrente alternada.*

API: *interface de programação aplicada.*

CI: *circuito integrado.*

CLP: *controlador lógico programável.*

CN: *controlador numérico.*

CNC: *controlador numérico computadorizado.*

CPU: *unidade central de processamento.*

D.B: diagrama de blocos.

DC/CC: *corrente contínua.*

DLL: biblioteca de acesso dinâmico.

E/S ou I/O: *entrada e saída.*

HB: motor *híbrido.*

IDE: *ambiente de desenvolvimento integrado.*

NA: *normalmente aberta.*

NF: *normalmente fechada.*

OCR: *reconhecimento óptico de caracteres.*

PM: motor *permanentemente magnético.*

POO: *programação orientada a objeto.*

RAD: *desenvolvimento de aplicações rápidas.*

VCL: *biblioteca de componentes visuais.*

VR: motor de *relutância variável.*

RT: regulador de tensão.

SUMÁRIO

LISTA DE FIGURAS

1 – INTRODUÇÃO

Desde o início da Revolução Industrial, no século XVIII, na Inglaterra, as máquinas passaram gradativamente a assumir o lugar das pessoas. Certamente, este evento sempre foi tachado como causa de desemprego e, logo no início, foi combatido com violência por organizações como as “*trade unions*”, sindicatos ingleses, ou o pior dos movimentos, denominado como "*ludista"*, ocorrido na França, conhecido como "os quebradores de máquinas", onde os partidários do movimento invadiam as fábricas e danificavam os equipamentos, (BRAICK, 2007).

Entretanto, a história tem demonstrado que os equipamentos eletromecânicos e computacionais têm trazido benefícios muito maiores do que se poderia imaginar, de modo que atividades repetitivas e insalubres não precisam mais ser executadas manualmente. Esta possibilidade trouxe grandes vantagens e segurança não só ao chão de fábrica, mas também a todo ambiente onde exista imerso algum equipamento dotado de automação e controle.

Seguindo essa tendência, este trabalho consiste em apresentar um sistema de controle de acesso para veículos automotores de forma a automatizar o ingresso e a saída de automóveis em ambientes privados ou controlados.

1.1. Justificativa

O que justifica o interesse por um sistema como o proposto neste trabalho é a necessidade de prover maior celeridade na identificação dos automóveis, no intuito de impedir os costumeiros congestionamentos nas entradas de *shoppings*, assim como, as longas e demoradas filas de saída. É consequência também da implantação deste sistema, a possível diminuição de mão de obra ocupada na execução de uma tarefa simples e possível de ser automatizada. Dessa forma, seria crível aumentar o nível de satisfação dos clientes, pois além de diminuir o tempo de espera, poder-se-ia oferecer um serviço com um preço melhor, devido à redução de operadores humanos do sistema.

1.2. Objetivo Geral

O objetivo deste trabalho é implementar um sistema de controle para entrada e saída de veículos em ambientes privados. Para isso, o sistema foi projetado de maneira que, quando um veículo se aproxima de uma cancela, um sensor de presença detecta o veículo. Essa detecção implica no acionamento de uma máquina fotográfica que registra a placa do veículo, permitindo em caso de reconhecimento, a sua entrada através do acionamento automático da cancela via porta paralela. Caso o carro não seja identificado, seria aberta uma oportunidade para um novo cadastramento.

O objetivo geral será alcançado através da construção de um protótipo, ressaltando que a placa fotografada será obtida através da fotografia da tela de um monitor simulando os caracteres de uma placa, e que a presença do automóvel será simulada pelo pressionamento de um sensor capaz de indicar a existência do automóvel.

1.3. Objetivos Específicos

Para alcançar o objetivo geral, foram traçados os seguintes objetivos específicos:

- pesquisar, na área de eletrônica, assuntos como circuitos integrados, portas-paralelas e sensores;
- desenvolver o sistema de reconhecimento baseado em DELPHI;
- estudar a teoria de normalização e modelagem de dados;
- desenvolver um protótipo capaz de simular a situação proposta.

1.4. Organização

O trabalho está organizado da seguinte forma: o capítulo 2 traz o referencial teórico, abordando assuntos como motores, sensores, banco de dados e porta paralela; o capítulo 3 versa sobre Automação e Controle; o capítulo 4 faz referencia à tecnologia de reconhecimento óptico de caracteres (*Optical Caracter Recognizing* - OCR); o capítulo 5 traz a implementação; o capítulo 6 apresenta a aplicação da solução proposta e os resultados obtidos; o capítulo 7 apresenta a conclusão e o fechamento geral do trabalho.

2 – REFERENCIAL TEÓRICO

Este capítulo visa trazer ao leitor uma maior interação com as tecnologias utilizadas nesse projeto. Oferecendo as condições teóricas mínimas para que seja possível entender a realização da parte prática, que será tratada mais a frente no capítulo 5 – Implementação.

Os motores elétricos representam um dos itens mais significativos e notórios já criados pelos homens. Sua aplicação, atualmente, é difundida de forma incomensurável, assim como também, ao menos teoricamente, ilimitada. As máquinas rotativas elétricas têm a vantagem de serem de baixo custo, não poluentes, e de construção simplificada, apesar da complexidade da idéia, na qual está apoiada a estruturação destas máquinas.

Podemos citar como principais fontes de utilização desses motores, desde o campo da eletrônica, passando pela informática, até utilizações mais pesadas como acionamentos de bombas de água, compressores, ventiladores industriais, inclusive os domésticos, assim como equipamentos para processamento e manuseio de carga pesada, (ROSÁRIO, 2005).

2.1. Motores Elétricos

Equipamentos conhecidos como geradores, transformam diferentes formas de energia em energia elétrica. Exemplo disso são as turbo-hélices, que transformam energia eólica em energia elétrica, as turbinas hidráulicas, que transforma a energia potencial gravitacional das quedas de água em energia elétrica, ou ainda as pilhas e baterias, que transformam energia química em energia elétrica, entre várias outras fontes de possibilidades, (FITZGERALD, 1975).

2.2.1. Visão Geral dos Motores Elétricos

Os motores elétricos se enquadram dentro da classificação de dispositivos receptores, pois transformam energia elétrica em outras formas de energias, seja ela mecânica, luminosa, sonora, entre outras. O princípio básico do funcionamento dos motores elétricos está ilustrado na Figura 2.1.

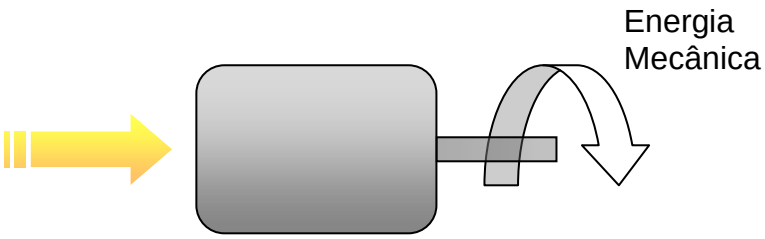


Figura 2.1 – Transformação de Energia.

Fonte: (ROSÁRIO, 2005)

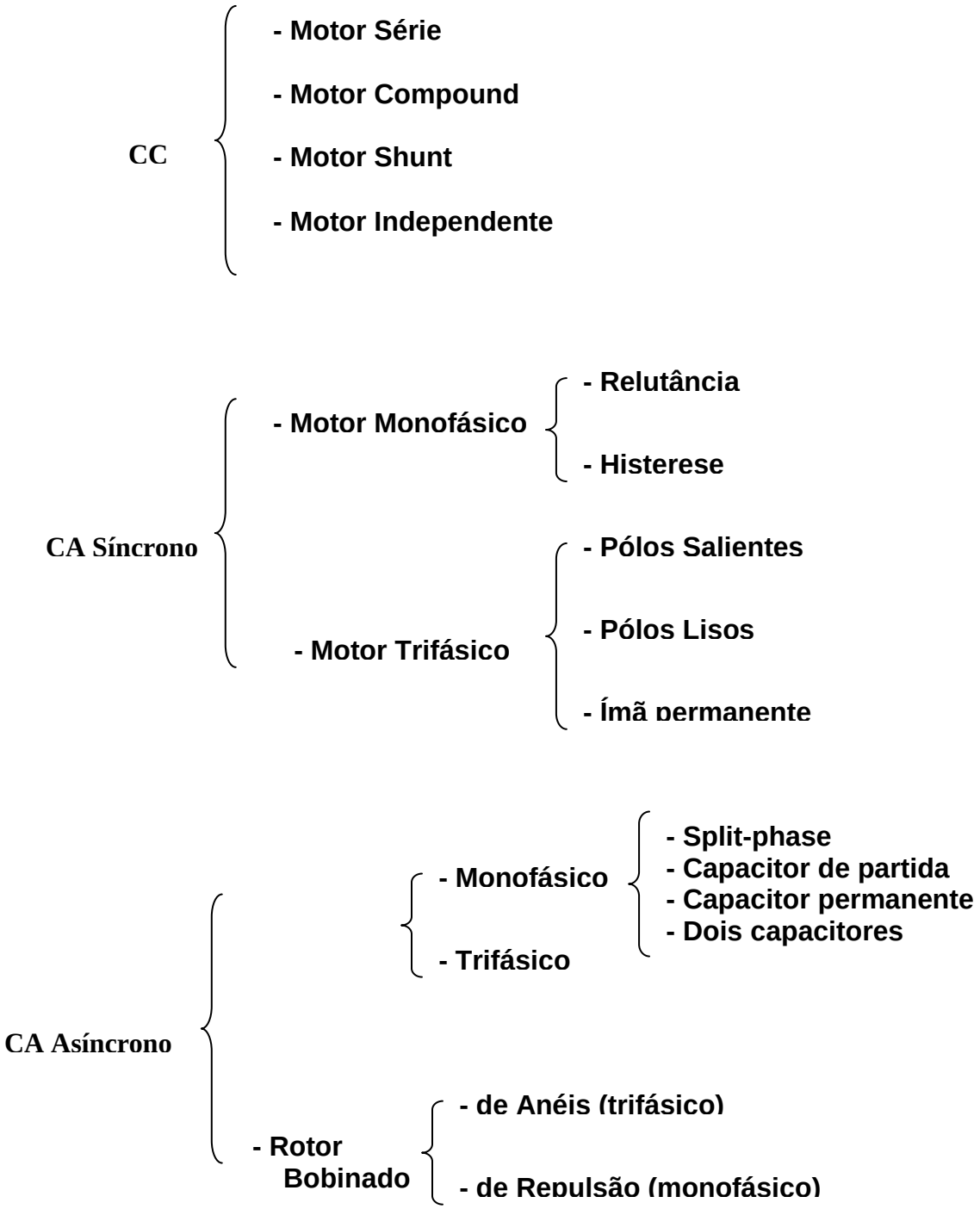
A ação motora obtida é utilizada em diversas modalidades de tecnologias distintas, como transporte de fluidos incompressíveis e compressíveis, processamento de materiais metálicos, manipulação de cargas, e transporte de cargas e passageiros.

Para esclarecer, podemos citar as bombas de água como máquinas acionadas através de motores elétricos que realizam o transporte de fluidos incompressíveis. Já os ventiladores, compressores e exaustores que também podem ser acionados por motores elétricos realizam o transporte de fluidos compressivos. E no ramo de materiais metálicos, existem várias máquinas operatrizes como: frezas, furadeiras, tornos e prensas.

Assim, devido ao seu largo campo de aplicação, os motores elétricos são as máquinas mais utilizadas tanto no meio industrial como rural e até mesmo comercial. E tudo isso devido a sua engenharia de fabricação que pode variar de equipamentos com potências mínimas de alguns poucos watts, como os motores de passo, a potências gigantescas de da ordem de megawatts. Outro fato importante e motivador para a sua utilização é o auto rendimento da transformação de energia, Enquanto os motores a combustão se estabelecem numa faixa de pouco mais de 30% de transformação efetiva de energia, os motores elétricos tem uma taxa mais elevada, o que acaba por torná-los mais potente e economicamente viável (ROSÁRIO, 2005).

2.2.2. Os Diversos Tipos de Motores Elétricos

Os motores elétricos são divididos em duas grandes famílias. A primeira são os motores acionados por corrente contínua CC e a segunda são os motores acionados pó CA. Podendo ser especificados conforme o seguinte esquema, (DEL TORO, 1999).



Dessa forma, os motores CA podem ser síncronos ou assíncronos. Quando o motor CA é assíncrono ele é denominado motor de indução, visto que seu funcionamento está baseado nos princípios de indução eletromagnética. Os motores assíncronos operam em alta tensão, podendo chegar a um índice superior a 2000 Volts e são destinados a aplicações de alta potência, podem chegar a centenas ou milhares de quilowatts de operação. São normalmente destinados a aplicações pesadas como siderurgias, petroquímicas, mineração e saneamento, entre outras.

Os motores de indução trifásicos e monofásicos representam 95% da totalidade dos motores instalados tanto nos setores industriais, rurais, comerciais e inclusive residenciais. E na representatividade geral de todos os motores elétricos, ocupam 75% de todos os equipamentos instalados. Esta enorme empregabilidade dos motores de indução monofásicos e trifásicos se deve não só ao índice de rendimento, mas também à longevidade do equipamento. Além disso, são capazes de serem acoplados mecanicamente à praticamente todos os tipos de equipamentos de engenharia moderna e ainda oferecem o menor custo de instalação e manutenção.

2.3. Motores de Passo

Os motores de passo têm o seu funcionamento semelhante aos motores síncronos. São projetados para girar um número específico de graus conforme os pulsos enviados pela unidade controladora. Assim, o motor recebe comandos em malha aberta, na forma de trem de pulsos, para girar um eixo.

Muito semelhante às máquinas elétricas convencionais, o projeto dos motores de passo é concebido através de um enrolamento do estator polifásico. Podendo assim, ensejar a origem de dois modelos diferentes do mesmo projeto, o unipolar e o bipolar, a depender da forma de enrolamento do seu estator. Dessa forma, os motores mais utilizados têm a característica unipolar, com quatro bobinas, onde cada fase consiste de um enrolamento com derivação central, ou mesmo de dois enrolamentos separados, contribuindo para que o campo magnético possa ser invertido sem a necessidade de se inverter o sentido da corrente. A Figura 2.2 demonstra o esquema sugerido:

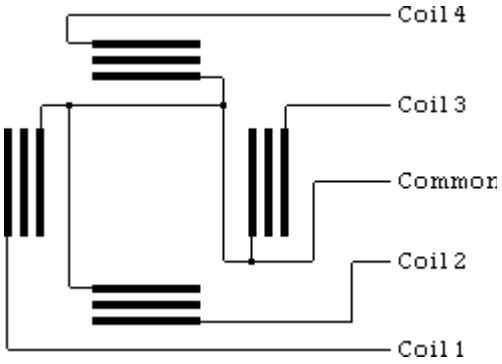


Figura 2.2 – Motor de Passo I

Fonte: (JRC, 2008)

A velocidade de rotação do motor está diretamente ligada à freqüência de pulsos elétricos e o comprimento angular da rotação está relacionado com a quantidade de pulsos aplicados, (JRC, 2008).

2.3.1. Estados do Motor

Os motores de passo podem se apresentar em três estados. São eles: desligado, parado, rodando. No caso de estarem desligados, não haverá alimentação suprindo o motor, o que implica no desligamento de todas as bobinas. Na maioria dos circuitos este estado ocorre quando a fonte de alimentação está desligada. Quando o motor está parado, pelo menos uma das bobinas assume o estado de energizada e o motor permanece estático num determinado sentido. Veja as Figuras 2.3, 2.4, 2.5.

Nesse caso, há consumo de energia. Já no estado de rotação, as bobinas são energizadas em intervalos de tempos determinados, impulsionando o motor a girar numa direção.

No Passo Completo 1, uma bobina é energizada a cada passo, tem a característica de menor torque e consecutivamente, menor consumo de energia e maior velocidade. No Passo Completo 2, duas bobinas são energizadas a cada passo, dessa forma, possuem consecutivamente, um maior torque, maior consumo de energia e velocidade. E finalmente no Meio Passo, ou *half-step*, há uma combinação do Passo Completo 1, com o Passo Completo 2, dessa forma, consome mais energias que os dois passos anteriores, mas tem a característica de maior precisão e porém com uma

menor velocidade, e com torque próximo ao apresentado pelo Passo Completo 2, (ROGERCOM, 2008).

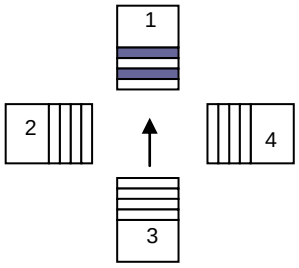


Figura 2.3 – Passo Completo 1

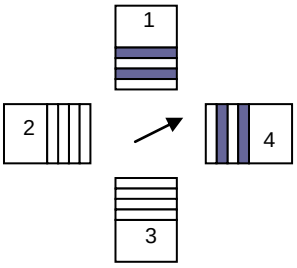


Figura 2.4 – Meio Passo

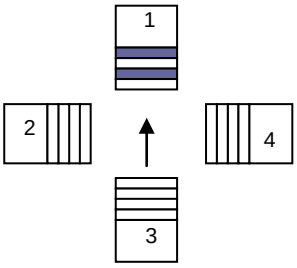


Figura 2.5 – Passo Completo 2

Fonte: (ROGERCOM, 2008)

2.3.2. Tipos de Motor de Passo

Quanto aos tipos de motores de passo, existem pelo menos 3, os de relutância variável (*variable-reluctance* - VR), ímã-permanente (*permanent-magnet* - PM), e os híbridos (*hybrid* - HB).

Os motores de relutância variável VR, do ponto de vista estrutural, são os mais fáceis de serem compreendidos. Conforme o corte transversal da Figura 2.6, percebe-se que ele tem um fio metálico enrolado em um rotor de pólos projetantes, e um estator que corresponde à camada externa do motor. Quando o estator está energizado com DC, os pólos permanecem magnetizados, dessa forma, a rotação ocorre quando o dente do motor é atraído pelo pólo energizado do estator.

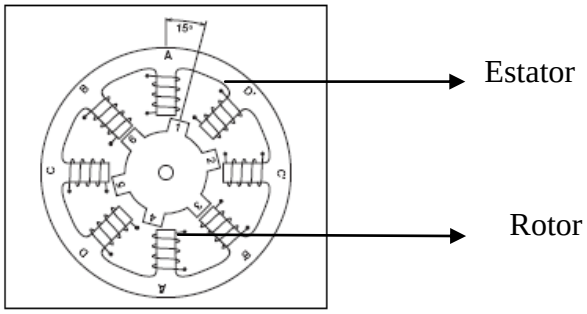


Figura 2.6 – Motor de Relutância Variável

Fonte: (JRC, 2008)

Os motores do tipo Ímã-Permanente ou PM perfazem as mesmas características do motor utilizado nesse projeto, no caso o PM35L-048. São popularmente conhecidos, no mercado americano, pelos nomes de *tin can* ou *canstock*. Essas expressões fazem referência à maneira como o motor é encapsulado, uma vez que o mesmo tem as mesmas características de um produto enlatado numa estrutura de estanho.

Os motores PM têm a característica de serem de baixo custo e baixa resolução, com ângulos típicos firmados em 7,5 º a 15º, Figura 2.7.

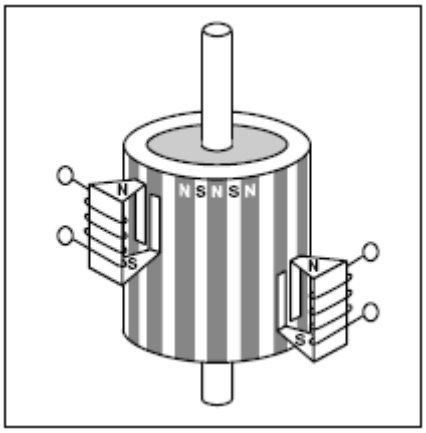


Figura 2.7 – Motor de imã Permanente

Fonte: (JRC, 2008)

Como o próprio nome diz, esses motores têm ímãs permanentes na sua composição estrutural. Nesse caso, o rotor, não possui pólos projetantes como o motor de relutância variável. Ao invés disso, o rotor é magnetizado com pólos alternantes entre norte e sul, os quais se situam numa linha paralela ao eixo do rotor. Esse pólo magnetizado do rotor fornece um aumento na intensidade do fluxo magnético e por isso, o motor PM apresenta um desempenho melhor se comparado com o torque do motor VR.

Os motores híbridos HB, representado em corte axial na Figura 2.8, são mais caros que os PM. Todavia, fornecem um melhor desempenho no que diz respeito à determinação do passo, velocidade e torque. Os ângulos típicos para esse motor são 3,6 º a 0,9 º (100 a 400 passos por ciclo ou rotação). Os motores de passo híbridos combinam a melhores características dos motores PM e VR.

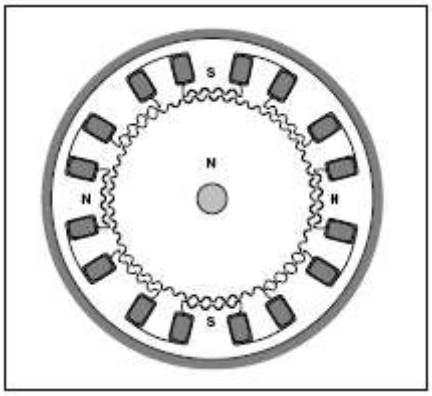


Figura 2.8 – Motor Hibrido.
Fonte: (JRC, 2008)

Dessa forma, o motor é multi-dentado, ou com vários pólos projetantes, assim como os motores VR, e contêm um ímã concêntrico e axialmente magnetizado em volta do eixo. O pólo projetante do rotor fornece o melhor caminho guiando o fluxo magnético através da lacuna existente entre o rotor e o estator, (JRC, 2008).

2.4. Sensores

Tal qual os 5 sentidos básicos dos seres humanos: tato, visão, audição, olfato e paladar, que nos proporcionam interagir e literalmente sentir o mundo que nos rodeia, a indústria utiliza também dispositivos denominados sensores. Estes têm o objetivo de mensurar a razão de medidas físicas no intuito de determinar a diferença do valor da variável de saída e o valor desejado como entrada.

2.4.1. Sobre Sensores

Os sensores são responsáveis por captar algum tipo de informação e remeter essa informação para um segundo dispositivo capaz de interpretá-la. A idéia, acima exposta, perfaz a característica dos transdutores. Por isso, são também chamados assim, pois toda estrutura, capaz de mudar suas características ou seu comportamento de maneira uniforme, mediante uma mesma excitação externa, pode ser classificada como transdutor. Todavia, voltando à questão dos sensores, eles têm larga aplicação na automação

indústria, onde identificam peças, através de medição e mensuração de posicionamento das mesmas. Na automação bancária e de escritório, podem ser utilizados nas leituras de códigos de barra, impressão digital. Atualmente têm larga aplicação na automação veicular, onde controlam temperatura, pressão do óleo, pneumáticos, autonomia relacionada à combustível, entre outros, (ROSÁRIO, 2005).

2.4.2. Critério para Especificar Sensores

Os sensores são classificados de acordo com o tipo de sinal que conseguem captar. Lembrando que: 1. SINAIS ANALÓGICOS – são aqueles que podem ser medidos dentro de um valor de escala e variam continuamente com o tempo, exemplo: termômetro e voltímetro, expressos em leitores analógicos; 2. SINAIS DIGITAIS – são uma especificação dos sinais binários, assumem valores finitos dentro de uma determinada escala, exemplo: balança digital e relógio digital, expressos em leitores digitais; 3. SINAIS BINÁRIOS – restringem sua atuação a sinais discretos como 0 e 1, precisando ser digitalizados para que possam ser expressos de forma compreensível.

Para controlar processos contínuos, são utilizados diferentes tipos de sensores, capazes de medir variáveis de controle como: velocidade, pressão, vazão, temperatura, entre outros. Nos processos discretos, os sensores são utilizados para controlar variáveis booleanas medindo, por exemplo, a proximidade entre objetos.

Os sensores podem ser classificados ainda conforme a sua linearidade e faixa de atuação. A linearidade é a correspondência entre o sinal gerado e a grandeza física medida, de forma que, quanto mais fiel a proporção ao estímulo, maior será a linearidade do sensor. Entretanto, seria impossível construir sensores absolutamente lineares, então, para solucionar essa impossibilidade, existe a medição através de faixa de atuação, que proporciona a leitura através de um intervalo de grandezas predeterminadas, sem, contudo, comprometer a interpretação do valor medido, (ROSÁRIO, 2005).

A Figura 2.9 mostra alguns exemplos clássicos de sensores industriais.

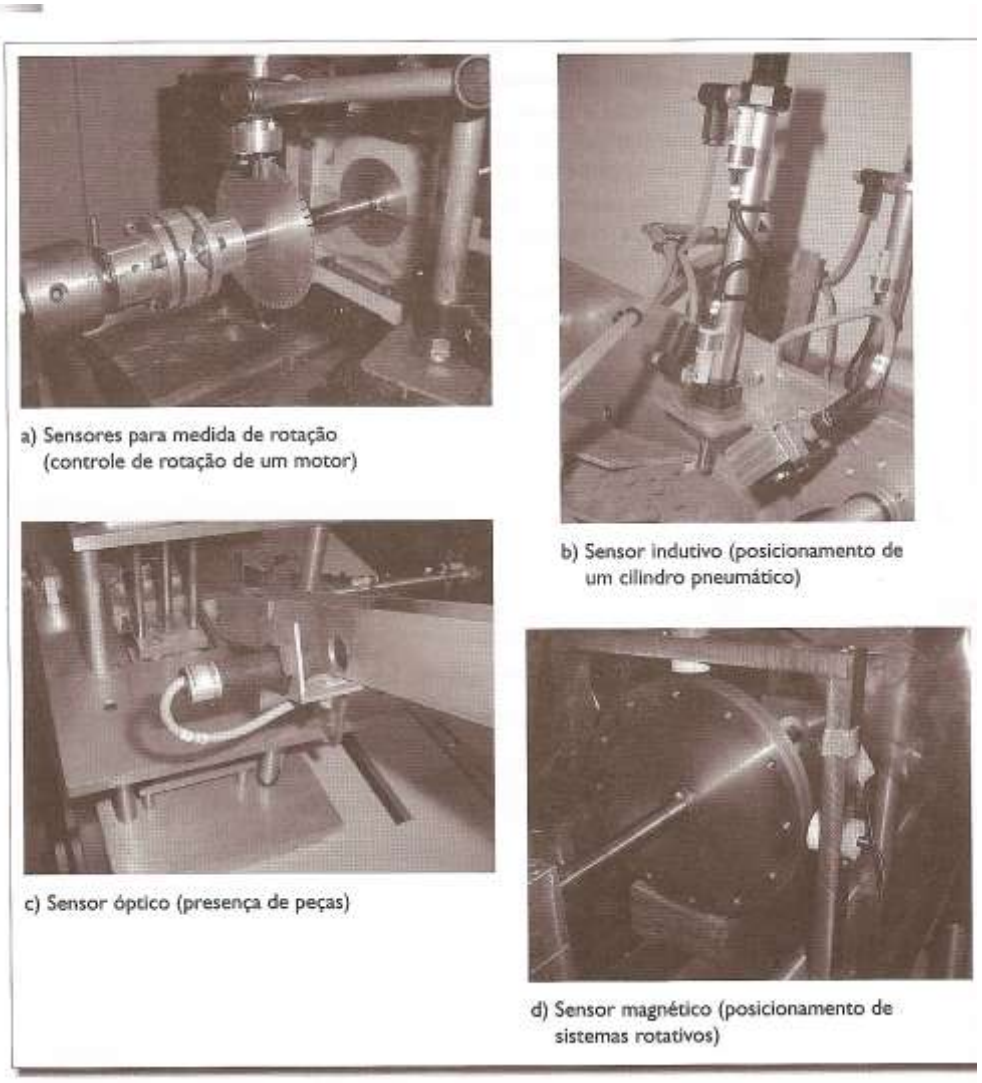


Figura 2.9 – Exemplo de Sensores.
Fonte: (ROSÁRIO, 2005)

No quesito industrial, as expressões mais utilizadas para a especificação de sensores são: 1. acurácia: razão entre o valor real e o valor medido pelo sensor; 2. resolução: grandeza relacionada ao grau de precisão do sensor; 3. repetibilidade: variação dos valores lidos quando uma mesma quantidade é medida várias vezes; 4. range: limite superior e inferior da variável a ser lida pelo sensor; 5. sensibilidade e linearidade: índice associado aos itens anteriores, acurácia, repetição e range.

2.4.3. Sensores utilizados como transdutores

A princípio, todo sensor é um transdutor. Denomina-se transdutor, todo dispositivo que receba alguma resposta de saída igual ou diferente do sinal de entrada. Na sua maior parte, os sensores são transdutores elétricos, pois convertem a grandeza de entrada, seja ela: som, luz, calor, pressão em grandezas elétricas. Algumas grandezas elétricas, como corrente elétrica, tensão elétrica e resistência elétrica, apresentam variação proporcional à indicação medida pelo sensor.

São exemplos de aplicação para transdutores e suas principais características frente à um estímulo elétrico:

1. FOTOCÉLULA: modifica a sua resistência;
2. FOTOTRANSISTOR: modifica o fluxo de corrente;
3. SENSOR INFRAVERMELHO: modifica sua tensão de saída.

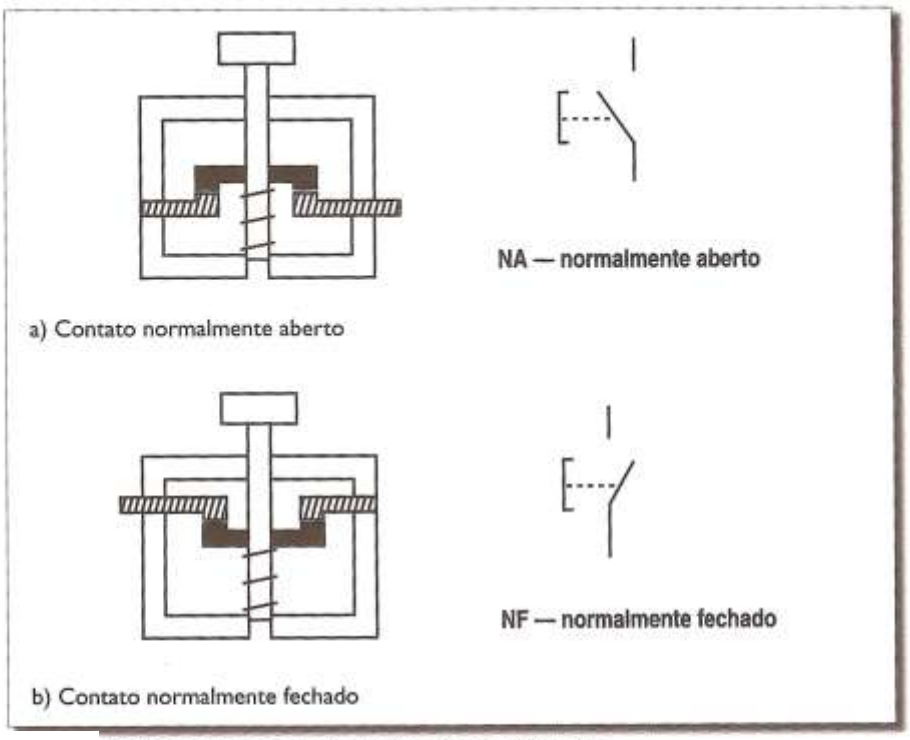
2.4.4. Sensores como elementos de comandos

Normalmente, nas aplicações de automação industrial, os sensores fazem o papel de elementos de comando, uma vez que são estabelecidos como os elementos de entrada dos CLPs. Todo elemento de comando, deve ser considerado um sensor. A Figura 2.10 mostra exemplos das botoeiras mais utilizadas.

1. Botão de acionamento: ao pressionar o botão, o elemento de comando atua contra uma força de mola, permitindo que uma conexão NA (normalmente aberta), torne-se NF (normalmente fechada), e assim acionando o circuito, a mola atua como força regeneradora, a fim de que o botão assuma sua posição normal, voltando ao *status* NA.

2. Botão inversor ou comutador: os dois tipos de contatos estão conjugados, NA e NF num único dispositivo. E para alternar a funcionalidade de execução dos botões, basta pressionar uma vez para executar a primeira função, e pressionar uma segunda vez, o mesmo botão, para obter a segunda função. Uma vez que o botão tenha sido pressionado duas vezes, um novo pressionamento retorna para a função NA.

3. Interruptor com trava: neste caso, o elemento memoriza sua condição de pressionamento. Porém, para abrir o circuito é preciso pressionar o botão correspondente à função NA, e para fechá-lo o botão NF, ou o circuito ficaria permanentemente aberto.



Funcionamento de um botão de acionamento.

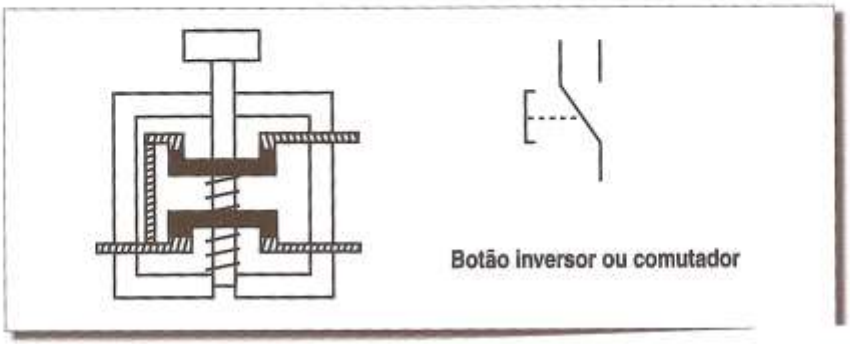


Figura 2.10 – Botoeiras ou Acionadores.

Fonte: (ROSÁRIO, 2005)

2.5. Bancos De Dados

Os bancos de dados, através dos SGBD (Sistemas Gerenciadores de Bancos de Dados), correspondem a um dos itens mais importantes de todo o espectro da computação. Basta pensar que o sentido da grande maioria dos sistemas informatizados, sejam eles pertencentes a universidades, bancos ou grandes empresas, encontram a sua verdadeira razão de ser, na medida em que as suas respectivas informações possam ser armazenadas de forma confiável, segura e acessível em algum ambiente virtual. A este ambiente chama-se banco de dados, e o meio pelo qual é possível acessá-lo e gerenciá-lo, chama-se sistema gerenciador de banco de dados, (KORTH, 1995).

2.5.1. SGDBs

Para realizar operações sobre os dados, é necessário dominar as linguagens específicas de definição e manipulação. São elas as DDL (*Data Definition Language*) e MDL (*Manipulation Data Language*). As DDL, Linguagem de Definição de Dados, são formadas por um conjunto de tabelas que armazenam os arquivos mais importantes dos SGDBs, chamado dicionário de dados, ou ainda, diretório de dados. O diretório de dados é responsável por armazenar os metadados, que são as informações sobre os dados contidos. Num entendimento mais simples e direto, os metadados armazenam “dados sobre dados”. Por isso, antes que quaisquer dados sejam lidos ou modificados dentro de um SGDB, os metadados serão obrigatoriamente consultados. Quanto as MDLs, Linguagem de Manipulação de Dados, são as responsáveis pelas principais execuções do operador sobre o SGDB, são elas: manipulações, inserções, eliminações e modificações de dados. Logo, as MDLs, permitem ao usuário fazer manipulações ou acessos ao banco de dados, e são subdivididas em dois grupos básicos, (KORTH, 1995).

DML procedural – requer do usuário uma especificação detalhada do dado a ser manipulado e como obtê-lo;

DML não-procedural – requer do usuário a especificação do dado, porém não é necessário indicar o caminho para obtê-lo.

2.5.2. SQL

Dentre as diversas linguagens comerciais capazes de executar operações sobre dados como Quel, QBE, o autor deste artigo, escolheu falar sobre SQL (*Structured Query Language*), Linguagem de Consulta Estruturada, tendo em vista a sua utilização na parte prática do projeto, pois o DELHI fará uso da SQL para conversar com o Banco de Dados Access.

Dentro da própria SQL, existem muitas versões, pois cada fabricante adapta o conceito da linguagem da forma mais adequada, segundo o entendimento dos fabricantes, para o seu produto. A versão original, foi desenvolvida pela IBM em San Jose na década de 70. Em 1986 a ANSI (*América National Standart Institute),* Instituto de Padronização Nacional Americana, estabeleceu o padrão de utilização para a SQL. Isto acabou por favorecer a utilização da linguagem e a mesma acabou por se firmar como a linguagem padrão para bancos de dados relacionais.

Algumas composições da linguagem SQL e suas particularidades:

- D*ata definition language* - DDL – fornece comandos para definir, modificar e remover esquemas de relação e criar índices;
- *Interctive data manipulation language* – DML – utilizada para manipular dados, podendo inserir, modificar e remover dados;
- *Embedded data manipulation manipulation language* – Linguagem de manipulação de dados embutida – como o próprio nome diz, é a forma embutida de SQL, utilizada para acoplar o SQL com as linguagens como Cobol, Pascal, Fortan e C;
- *View definition* – Definição de visões – pertence às DDLs e inclui comandos para definição de visões;
- *Authorization* – Autorização – também pertencentes às DDLs, inclui comandos para formalizar acessos a usuários;
- *Integrity* – Integridade - conforme o novo padrão ANSI, utilizado pela SQL, é utilizado uma forma limitada de integridade. Há uma perspectiva de que os próximos produtos e padrões sejam compostos por recursos mais avançados para aferir a integridade;

Transaction control – Controle de transações – são comandos capazes de especificar o início e fim de cada transação, (KORTH, 1995).

2.5.3. Estrutura Básica da SQL

As Instruções SQL são compostas de três cláusulas básicas: *select*, *from*, *where*.

- *SELECT*: é a cláusula responsável por selecionar o atributo, ou os atributos sobre os quais se processarão as pesquisas. Na matemática corresponde à operação projeção da álgebra relacional;
- *FROM*: é a cláusula responsável por selecionar uma ou mais tabelas envolvidas na operação de pesquisa. Corresponde a operação produto cartesiano da álgebra relacional;
- *WHERE*: é responsável pela associação de chaves primárias entre uma tabela e outra. A cláusula *where*, é responsável por evitar o produto cartesiano estabelecido pela cláusula *from*. Lembrando que produto cartesiano dentro de um banco de dados é uma condição essencialmente inviabilizadora de qualquer tipo de consulta, tendo em vista a inclusão de inconsistência e incerteza nos dados pesquisados, (KORTH, 1995).

Exemplo de aplicação da linguagem SQL:

select placa
from tbl_automovel
where placa=”JGI-2345”

De acordo com o exemplo, a pesquisa sugere a existência de um banco de dados hipotético, onde existe uma tabela chamada tbl_automovel, cuja seleção vem através da cláusula *from*. Já a cláusula *select*, é responsável por selecionar o atributo placa, que por sua vez pertence à tabela tbl_automovel. E por fim, a cláusula *where*, e responsável por separar o dado “JGI-2345”, dentre todos os demais dados possíveis pertencentes a esta tabela.

2.5.4. Modificações em Banco de Dados

Os bancos de dados costumam ter estruturas estáticas, embora isso não seja regra. Contudo, a massa de dados normalmente é mutante, pois a todo tempo há novas inclusões e consecutivamente necessidade de retirada ou até mesmo atualização dessas inclusões. Sendo assim, as próximas citações referem-se a esta possibilidade de retirar, atualizar e inserir dados no banco.

2.5.4.1. Remoção

O primeiro item de modificação que poderíamos citar seria a REMOÇAO. O comando utilizado quando algum dado e suas características não precisam mais fazer parte de um banco de dados. Imagine que temos um sistema que gerencia um estacionamento, e toda vez que um carro entra no estacionamento, sua placa é inserida com as demais numa massa de dados. Porém, quando o carro deixa o estacionamento, sua placa é retirada do banco de dados. Para isso usaríamos o seguinte comando:

delete placa
from tbl_automovel
where placa=”JGI-2345”

A cláusula *DELETE* é suficiente para retirar do banco a placa JGI-2345.

2.5.4.2. Atualização

Podem acontecer situações onde desejamos modificar um único valor de um registro, sem ter que inseri-lo novamente. Para isso, optamos pelo comando de atualizar o registro. Imagine que a tabela tbl_automovel, que é composta de um único atributo, placa, fosse composta por 30 atributos, dentre eles cor, modelo, ano, proprietário, etc.... Caso fosse necessário atualizar apenas a placa, devido a uma digitação errada, os outros atributos ficariam intactos sem ter que passar por uma nova inserção, onde teriam que ser novamente digitados todos os 30 dados referentes a cada atributo. Para evitar esse tipo de situação, muito comum em bancos de dados, é feito o processo de atualização do banco, com os seguintes comandos:

update placa
set placa=”JGL-2345”
where placa=”JGI-2345”

Assim, foi possível modificar apenas o caractere L na placa, que passou de ”JGI-2345” para ”JGL-2345”, sendo evitada a retirada do dado e uma consecutiva nova inserção.

2.5.4.3. Inserção

Utilizado sempre que se deseja inserir um dado novo na tabela. Como por exemplo a inserção de um novo veículo no banco de dados.

Insert into tbl_automovel

values (”JGH-3456”)

Este comando é capaz de inserir a nova placa (”JGH-3456”) na tabela tbl_automovel. Todavia, se na tabela houvesse 30 atributos, os mesmos deveriam ser inseridos da seguinte maneira:

Insert into tbl_automovel

values (”JGH-3456”, ”vermelho”, ”João”, ”VW”, ”FOX”, ”2008”, até o 30º termo) – representando respectivamente os atributos: placa, cor, proprietário, marca, modelo, ano, até o 30º termo.

2.5.5. Visões

Outro aspecto que deve ser citado nos bancos de dados relacionais são as visões. Em SQL as visões são definidas com o comando *create view*. As visões, mais conhecidas como *views*, são responsáveis pela visualização de campos predeterminados da massa de dados. A princípio são muito parecidas com a cláusula *select*. No entanto, para utilizar o *select* é preciso montá-lo para cada nova utilização. Já as *views*, uma vez montadas, uma única vez, basta chamá-la pelo nome para obter a visualização requerida. Uma *view* pode ser montada com a seguinte linha de comando.

Create view mostra_placa

select placa

from tbl_automovel

Este comando monta permanentemente uma *view* que exibe todas as placas cadastradas no sistema. A teoria que envolve SQL se estende muito além do que se possa contemplar neste trabalho. Há ainda o caso das TRIGGERS, conhecidas como gatilhos, são de fundamental importância dentro do processo de administração e manutenção dos bancos de dados, pois são capazes de automatizar muito das tarefas rotineiras de um SGDB.

2.6. Porta Paralela

A porta paralela como o próprio nome sugere, é um dispositivo de entrada e saída do computador. Atualmente são conhecidos dois modelos básicos de portas paralelas, as unidirecionais e as bidirecionais, (ROGERCOM, 2008).

2.6.1. Porta Paralela de Transmissão Unidirecional

Este modelo já não é tão comum. São conhecidas também como porta paralela SPP (*Standard Parallel Port*), ou ainda, Porta Paralela Padrão. Podem chegar a uma taxa de transmissão razoável, a tingindo níveis da ordem de dados a 150KB/s. Comunica-se com a CPU utilizando um BUS de dados de 8 bits. E quando trabalham na via inversa, ou seja, transmitindo dados para o periférico, chegam a fazê-lo a uma taxa de 4 *bits* por vez, (ROGERCOM, 2008).

2.6.2. Porta Paralela de Transmissão Bidirecional

Esta é a mais comum e a mais avançada também das modalidades de porta avançada EPP (*Enhanced Parallel Port*) no seu melhor estágio de transmissão pode atingir uma taxa de transferência de 2 MB/s. Esta porta comunica-se com a CPU utilizando um BUS, ou barramento de dados de 32 bits. Agora, já na transmissão de dados entre os periféricos utiliza 8 bits por vez. É importante salientar que tanto para portas uni ou bidirecional, o cabo de transmissão deve ter o limite máximo de 8m, para evitar os efeitos da resistência do próprio cabo e uma consecutiva transferência inapropriada dos dados, (ROGERCOM, 2008).

2.6.3. Endereço da Porta Paralela

O computador por operação padrão nomeia as Portas Paralelas como: LPT1, LPT2, LPT3, em diante. Porém, o que passa de LPT1 são portas virtuais, pois a porta física padrão é a LPT1, e seus endereços são:

- 378h - para enviar um byte de dados pela Porta;
- 378+1h - para receber um valor através da Porta;

- 378+2h - para enviar dados.

Caso haja necessidade de fazer implementações sobre a LPT2, os endereços são: 278h, 278+1h e 278+2h, com as mesmas funções dos endereços da porta LPT1, (ROGERCOM, 2008).

2.6.4. Interfaceamento Físico da Porta Paralela

A porta paralela se comunica com os periféricos, através do conector DB25. É um conector bastante conhecido, presente em praticamente todos os microcomputadores e localizado na parte de trás do gabinete do computador.

A Figura 2.11, ilustra o conector DB25.

Figura 2.11 – Conector DB25 Macho.

Fonte: (ROGERCOM, 2008)

Este é o exemplo concreto da porta paralela, através dessa conexão o computador se liga com os seus respectivos periféricos principalmente *scanners* e impressoras, Todavia, hoje os fabricantes preferem projetar equipamento para porta USB, pois além de serem mais abundantes nas máquinas, apresentam um tempo de ação e reação muito melhor. Voltando à porta paralela, no DB25, quando um pino está em nível lógico 0 a tensão elétrica no mesmo está entre 0 à 0,4v. Se o pino se encontrar em nível lógico 1 a tensão elétrica nele será acima de 3.1 e até 5v.

3 – AUTOMAÇÃO E CONTROLE

Embora o controle e automação façam parte da última palavra em tecnologia, desde o princípio o homem tenta estabelecer o domínio sobre os objetos à sua volta. Um exemplo prático consiste no homem primitivo que, através da sua tecnologia rudimentar, mantinha a qualidade dos seus artefatos, a fim de obter o êxito na caça, pesca ou guerras, (ROSÁRIO, 2005).

3.1. O Início Da Automação

Ao longo do tempo a noção de controle evoluiu e foi tomando uma forma característica de automação. Com o advento das máquinas-ferramenta, também chamadas de operatrizes, passou a ser possível a execução de tarefas como furação, soldagem e torneamento. A partir de então não mais através do homem propriamente dito, mas, sim, através de uma máquina fazendo a interface entre o homem e a matéria-prima e gerando o produto acabado.

Na década de 50, a máquina-ferramenta foi aprimorada e constituiu-se o CN (controle numérico). Na seqüência evoluiu-se para o CNC (controle numérico computadorizado), onde através de microprocessadores, as informações eram inseridas por uma interface de entrada, processadas e finalmente era obtida a concepção do produto ou tarefa acabada.

3.1.1. Controle Moderno

Atualmente existem dois tipos básicos de controle para processos industriais, os quais dependem do tipo de variável que se deseja controlar. Quando as variáveis são do tipo analógica ou contínua, tem-se o controle contínuo. Porém, se as variáveis forem do tipo discreto ou digital, tem-se o controle discreto.

O controle do tipo discreto iniciou-se com a utilização de dispositivos eletromecânicos, como os relés, onde as chaves de contatos simulam os níveis lógicos baseados na lógica binária. Na década de 60 os relés deram lugar aos

CLP (controladores lógicos programáveis), onde se utilizava a mesma forma de programação binária oriunda do sistema anterior.

3.2. Automação Industrial

A automação significa um conjunto de técnicas pelas quais se constroem sistemas ativos que atuam com eficiência ótima pelo uso de informações recebidas do meio sobre o qual atuam. Baseando-se nas informações, o sistema calcula a ação mais apropriada para execução da ação o que perfaz a característica de sistema de malha fechada ou sistema de realimentação. Esse sistema objetiva manter uma relação expressa entre o valor da saída em relação ao da entrada de referência do processo. Essa relação corrige valores na saída que por ventura não estejam entre os valores desejados. Assim, são usados controladores, os quais comparam o valor atual com o valor desejado (*setpoint*), efetuando o cálculo para ajuste e correção, através da execução algorítmica de um programa ou circuito eletrônico, (ROSÁRIO, 2005).

Por estar associada ao conceito de software, a automação é um processo flexível. O recurso de *software* traz ao sistema dotado de automação a possibilidade de alterar radicalmente todo o comportamento automatizado com o intento de produzir resultados diferentes, a cada nova implementação ou mudança no software.

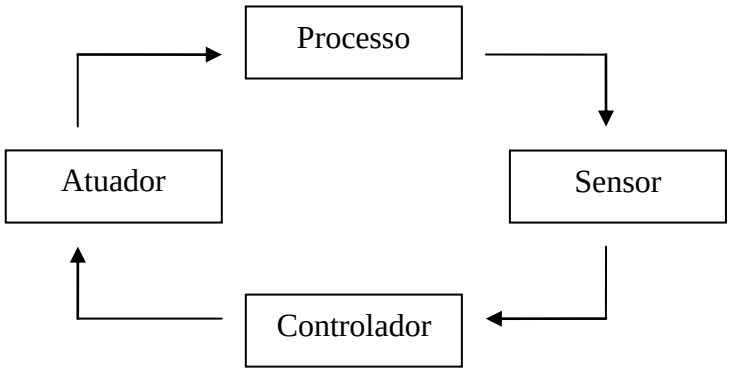


Figura 3.1 – D.B, Sistema de Automação.
Fonte: (ROSÁRIO, 2005)

Uma verdade prática no que diz respeito a automação é que todo sistema dotado de retroação e controle implica na presença de três componentes básicos, são eles: sensor, controlador e atuador, onde a principal característica é a realimentação das informações, Figura 3.1.

O Sensor é um dispositivo sensível a fenômenos físicos como temperatura, luz e pressão. Através dessas características de sensibilidade os sensores enviam um sinal para os dispositivos de medição e controle. Ou caso exista a necessidade de medir um fenômeno elétrico como corrente, a partir de um fenômeno físico qualquer envolvendo grandezas físicas que não seja de natureza elétrica, pode-se utilizar um transdutor, também conhecido como conversor de sinais. O transdutor é um dispositivo que da mesma forma que os sensores é capaz de responder a estímulos. Converte a magnitude aferida em um sinal elétrico conhecido, proporcional à amplitude desse estímulo.

Os Atuadores são dispositivos que após o acionamento são capazes de executarem uma determinada força, Figura 3.2.

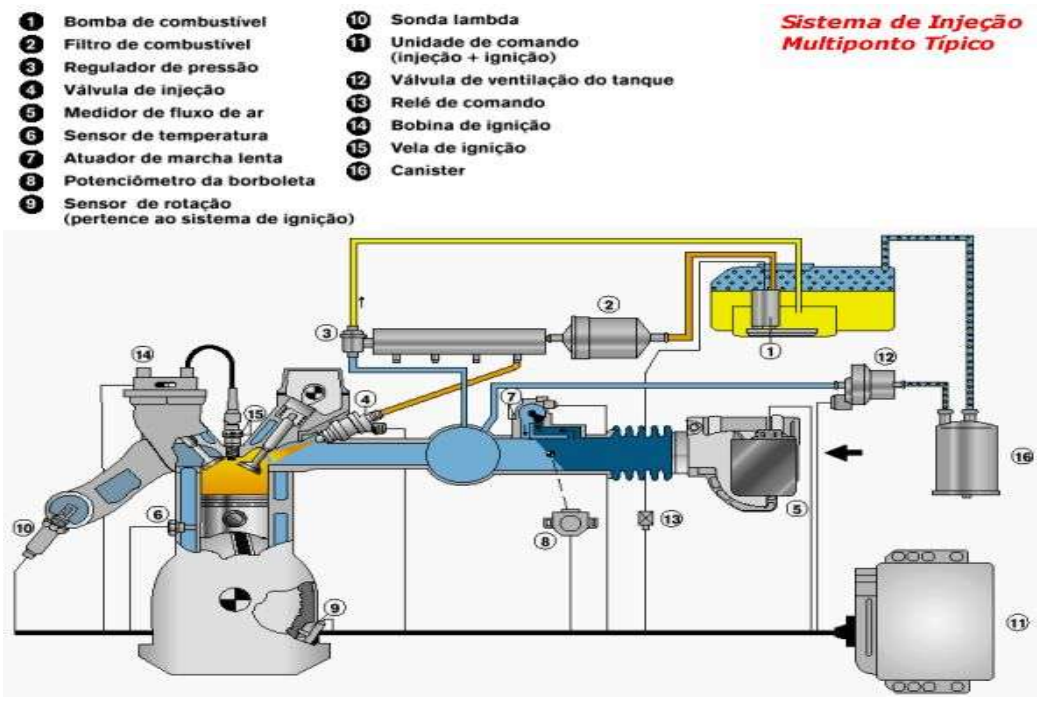


Figura 3.2 – Sistema de Atuadores e Válvulas.
Fonte: (CRUZ, 2005)

Essa força, também chamada de força de deslocamento é definida pelo sistema controlador por meio de uma ação de controle. Os atuadores podem ser dos tipos: magnéticos, hidráulicos, pneumáticos, elétricos ou de acionamento misto. Como exemplo, temos válvulas e cilindros pneumáticos, motores, aquecedores, entre outros.

Dessa forma, o atuador é produzido para atuar ou fazer atuar energia mecânica sobre uma máquina, levando a mesma a realizar trabalho. Se for levada em conta a natureza do fluído, os atuadores são classificados como pneumáticos quando utilizam ar comprimido, ou hidráulico se utilizam óleo sob pressão. Assim, são utilizados fluidos sob pressão para produzir a energia mecânica desejada, (CRUZ, 2005).

Figura 3.3 – Compressor de Ar.
Fonte: (CRUZ, 2005)

Figura 3.4 – Bomba Hidráulica.
Fonte: (CRUZ, 2005)

Quanto ao movimento que realizam, os atuadores podem ser classificados como lineares ou rotativos. Os atuadores lineares mais comuns são os pistões, que realizam trabalho deslocando-se sempre na mesma direção, variando apenas o sentido. Os pistões pneumáticos e hidráulicos encontram grande aplicação no ambiente industrial e em equipamentos de grande porte como guindastes, escavadeiras, dentre outros.

Os pistões são produzidos de forma, sugestivamente cilíndrica, por isso são também conhecidos como cilindros, dentro do corpo há uma estrutura a qual se denomina pistão, responsável pelo movimento oscilatório que realizará trabalho. Dentro do corpo da estrutura então, é colocado o fluido sob pressão, com especial cuidado com relação à vedação para que não haja extravasamento desse fluido, ou a entrada de impurezas que possam contaminá-lo, Figura 3.5.

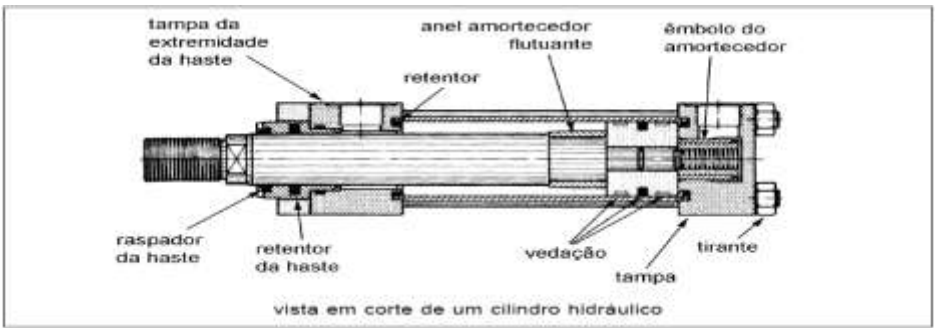


Figura 3.5 – Cilindro Hidráulico.

Fonte: (CRUZ, 2005)

Os atuadores rotativos têm essa denominação, devido à natureza rotatória do movimento produzido por esses equipamentos. Podem ainda ser subdivididos em angulares e contínuos.

Os atuadores rotativos angulares são responsáveis por produzirem um movimento de rotação. Embora utilizem pistão, não podem ser classificados como lineares, pois neste caso o pistão é apenas transladado de um ponto ao outro, e não rotacionado, Figura 3.6.

Os atuadores rotativos contínuos são também conhecidos como motores pneumáticos ou hidráulicos, a depender do fluido que os acionem Figura 3.7.

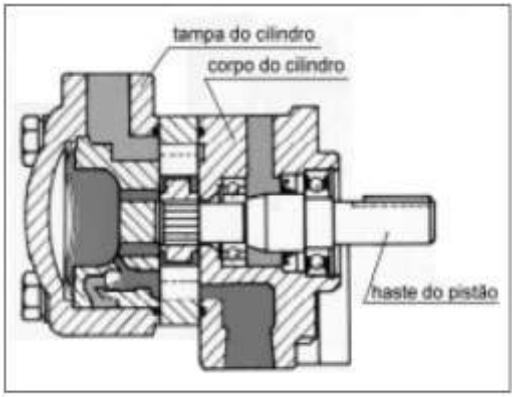


Figura 3.6 – Atuador Rotativo Angular.
Fonte: (ROSÁRIO, 2005)

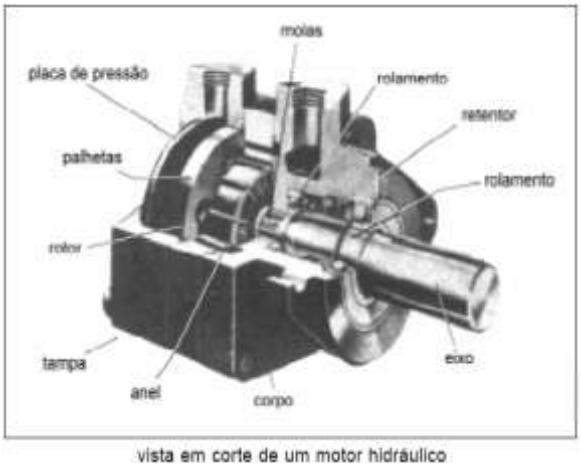


Figura 3.7 – Atuador Rotativo Contínuo.
Fonte: (ROSÁRIO, 2005)

Já os controladores, são modelados matematicamente através do conhecimento de sua planta. Esses modelos matemáticos representam os sistemas do mundo real, desenvolvidos através da aplicação de regras conhecidas de comportamento para os elementos do sistema. A Teoria Clássica de Controle estabelece os critérios de estabilidade para obter os parâmetros necessários para o correto projeto do controlador. Assim, será possível obter a efetiva ação de controle (ROSÁRIO, 2005).

Então, o controlador e o atuador são componentes do sistema de controle. O controlador compara a saída efetiva da planta com o comando de entrada e propicia um sinal de controle que reduz o erro a zero ou ao mais próximo de zero possível. O controlador pode ser modelado com a representação de um ponto de soma, um dispositivo de controle, amplificadores de potência, e dispositivos de hardware. O ponto de soma é a derivação onde os sinais de entrada e saída são comparados. O dispositivo de controle determina a ação de controle. Os amplificadores de potência e dispositivos de hardware são associados para realizar a ação de controle na planta, (CRUZ, 2005).

O atuador converte a ação de controle em movimento físico do manipulador. A Figura 3.8 ilustra um diagrama de blocos geral dos componentes do sistema de controle para uma junta robótica.

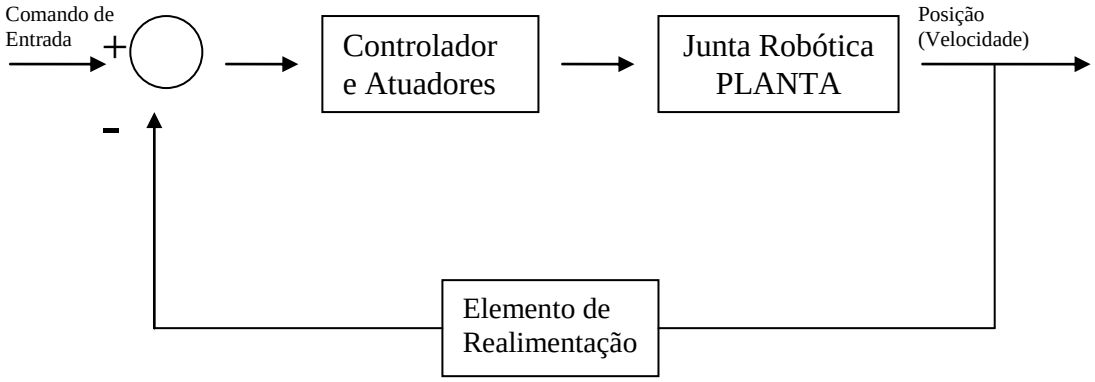


Figura 3.8 – D.B Sistema de Controle I.

Fonte: (CRUZ, 2005)

O esquema da Figura 3.8, representa um sistema dotado de realimentação ou retroação, o que caracteriza um sistema de malha fechada. O produto deste sistema é a variável de saída, também chamada de resposta, que é ajustada conforme o sinal de erro. O erro *e*(*t*), que consta na Figura 3.9, é calculado através da diferença entre a resposta real do sistema, obtida por um sensor, e uma resposta de referência, que seria a resposta ideal desejada pelo projetista.

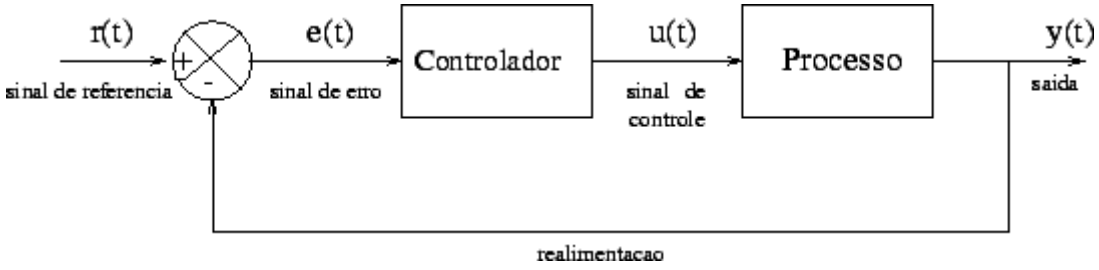


Figura 3.9 – D.B. Sistema de Controle II.

Fonte: (CRUZ, 2005)

O sentido da retroação consiste na necessidade de entender e controlar o sinal emitido pela saída. Então, a entrada é realimentada com o sinal de saída, onde é realizada a operação de comparação, já citada, entre este sinal, erro – *e*(*t*), e o *set-point* (sinal de referência). Na sequência o controlador, que também é chamado de compensador, emite o sinal de controle para que possa ter um sinal o tanto mais próximo quanto possível do sinal referencial.

O comportamento do sistema da Figura 3.9, pode ser analisado matematicamente através de uma equação diferencial linear, onde o sinal de entrada *r(s)*, será comparado com o sinal de saída *y(s)*. Fazendo uso da Transformada de Laplace, obtêm-se uma função entrada-saída, também chamada de função de transferência.

$$G(s) = \frac{y(s)}{u(s)} = \frac{N(s)}{D(s)}$$

FONTE: (DA SILVA, 2009)

Equação 3.1 – Função de Transferência

Onde *N(s)* e *D(s)* são polinômios em *s*. Supondo que as raízes de *N(s)* são todas diferentes das raízes de *D(s)*, define-se o seguinte:

- *pólos de G(s):* raízes de: *D(s)* .
- *zeros de G(s):* raízes de: *N(s)* .
- *ordem do sistema:* grau de: *N(s)*
- *tipo do sistema:* número de pólos da *G(s)*em *s=0*.

Para exemplificar, um sistema representado pela seguinte função de transferência, contida na Equação 3.2, teria a seguinte interpretação:

$$G(s) = \frac{s^2 + 5s + 6}{s^3 + 15s^2 + 50s} = \frac{(s+2)(s+3)}{s(s+5)(s+10)}$$

FONTE: (DA SILVA, 2009)

Equação 3.2 – Função de Transferência

- pólos: 0, -5, 10 (números que zeram a função);
- zeros: -3, -2 (raízes do numerador);
- tipo do sistema: 1 (apenas um pólo na origem) ;
- ordem do sistema: 3 (potência de maior ordem);

Através da transformada de Laplace de uma determinada entrada *u(t)* e da função de transferência do sistema, é possível, através da transformada inversa de Laplace, determinar a resposta temporal do sistema, ou seja:

$$y(t) = \mathcal{L}^{-1}[G(s)u(s)]$$

FONTE: (DA SILVA, 2009)
Equação 3.3 – Transformada de Laplace

A função de transferência a malha fechada pode ser representada também da seguinte forma:

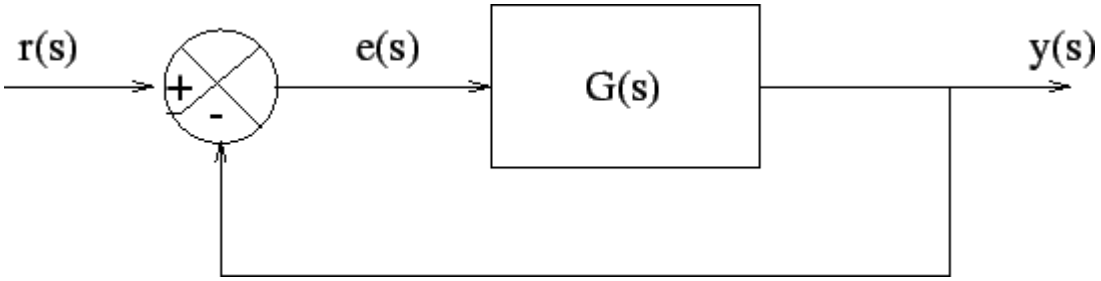


Figura 3.10 – Função de Transferência a Malha Fechada
FONTE: (DA SILVA, 2009)

3.2.1. CLPs

A palavra de ordem no mundo da automação industrial consiste nos CLPs, Controladores Lógicos Programáveis. São responsáveis pela função de controle num sistema de automação industrial. O CLP pode ser descrito como um dispositivo eletrônico dotado de uma memória interna programável, onde é possível armazenar as instruções lógicas que acabam por coordenar o comportamento físico das máquinas. O controlador lógico programável é um computador especializado que desempenha funções de controle baseados em um microprocessador, em virtude disso, é um equipamento eletrônico digital

com hardware e software voltados para aplicações industriais, fazendo uso de sua estrutura para armazenar as instruções na sua memória programável.

Para manter este nível de funcionamento, o CLP possui um microprocessador responsável por interpretar as sequências de instruções e executar as ações estabelecidas, que acontecem em função das suas variáveis de estado, (WEG, 2009).

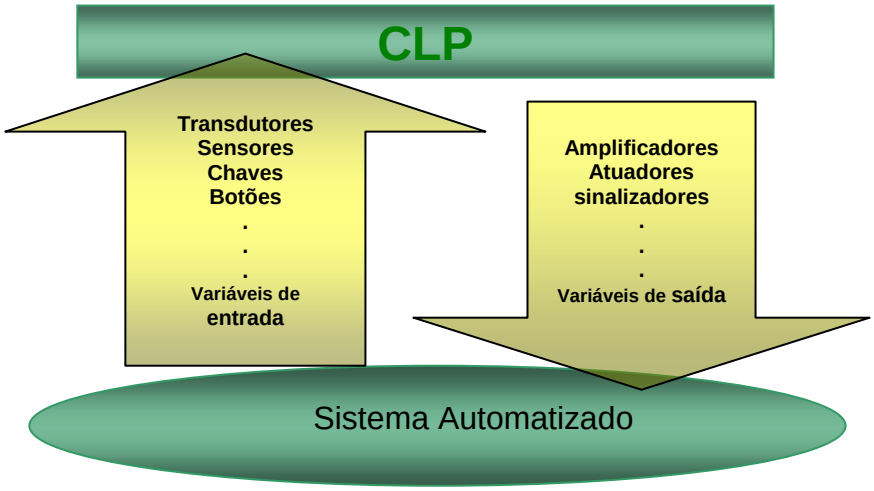


Figura 3.11 – Controle Básico.

Fonte: (WEG, 2009)

As variáveis de entrada correspondem aos sinais externos recebidos pelos CLPs, podendo partir tanto do controlador, assim como, de fontes pertencentes ao próprio sistema controlado. Normalmente são emitidas por sensores, chaves, botoeiras, transdutores e etc.

As variáveis de saída correspondem aos dispositivos controlados pelos pontos de saída dos CLPs, sendo representada pelo controle de dispositivos tais como solenóides, displays, chaves, ou o envio de sinais para outros CLPs.

Essa troca de sinais de entrada e saída através dos CLPs, está representada na Figura 3.11.

3.2.2. Princípio de funcionamento e arquitetura

O funcionamento do CLP inicia-se dentro da CPU (unidade central de processamento). Depois, é iniciado o processo de leitura das variáveis de entrada pelo módulo de entrada do CLP. Essa leitura é feita de acordo com a

execução do programa desenvolvido pelo usuário, e destinado a realizar o controle e monitoramento das tarefas destinadas ao dispositivo.

Então, a CPU é responsável por fazer o tratamento das instruções inseridas pelo programador, e também por gerenciar as entradas e saídas do CLP. Mas para isso, a CPU conta com o recurso da memória que é responsável por armazenar todo o programa desenvolvido, assim como, as informações internas do próprio fabricante que tornarão possível o funcionamento do CLP.

Os CLPs possuem uma arquitetura cujo funcionamento é semelhante ao representado na Figura 3.12.

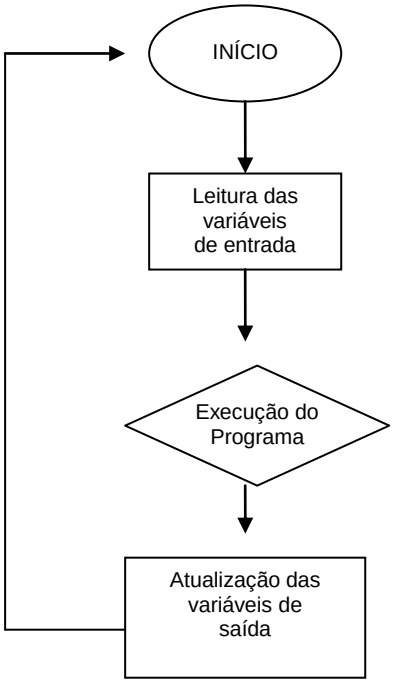


Figura 3.12 – Funcionamento de um CLP I.

Fonte: (WEG, 2009)

- CPU: Unidade Central de Processamento – composta pelo processador, cujas funções são dedicadas ao microcontrolador, possui ainda um sistema de memória baseada em RAM e ROM e circuitos internos;

- Fonte de alimentação: como o próprio nome sugere, fornece o suprimento de energia para o processador e os demais circuitos de entrada e saída;
- Circuitos de entrada e saída, ou ainda E/S ou I/O: circuitos destinados para o recebimento e entrada de sinais;
- Base: provê a conexão mecânica e elétrica entre a CPU, os módulos de entrada e saída e a fonte de alimentação.

Para que a conexão entre a CPU e o CLP seja viável, é preciso conectar as portas do PC com o CLP. Existe uma porta padrão que faz a comunicação entre os dois equipamentos, é a porta serial padrão RS-232, utilizada quando a transmissão dos dados se limita a uma distância de 15 m. Todavia, para transmissão até 1.200m, é utilizado o padrão RS-422, que tem as características de maior imunidade a ruídos, maior velocidade e comunicação *full-duplex*. Dentro do software vem embutido um protocolo de comunicação que irá determinar a forma de transmissão dos dados. Estes protocolos não têm relação com os protocolos de rede, são normalmente protocolos proprietários e fornecidos pelos fabricantes do hardware.

Já a comunicação homem-máquina, pode se dar através da própria interface homem máquina, IHM, através de botões de navegação e tela de cristal líquido, como na Figura 3.13.

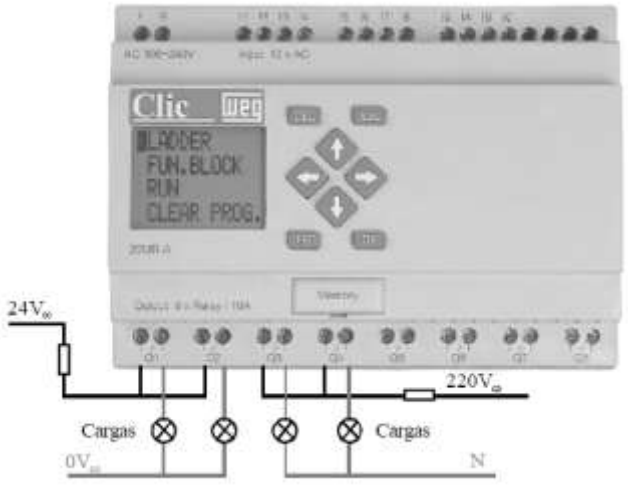


Figura 3.12 – Funcionamento de um CLP II.

Fonte: (WEG, 2009)

Enfim, uma vez tendo a posse de toda essa estrutura física, é necessário a utilização de um software para gerenciá-la. Então, existem os aplicativos que fazem a interação entre o usuário e a linguagem de máquina que será processada pelo equipamento. E para fazer esta intervenção, ou seja, para programar o dispositivo, pode-se utilizar um microcomputador comum ou ainda um equipamento, conhecido no mercado como programador de mão. O programador de mão é um aparelho portátil muito útil em intervenções *in locu* e porta toda tecnologia necessária para que se possa programar o CLP, utilizando normalmente a porta serial para fazer essa interconexão.

4 – RECONHECIMENTO DE PADRÕES

O grande poder de processamento do nosso cérebro nos possibilita realizar operações complexas, como o reconhecimento de padrões ou caracteres. Para isso utiliza-se a conjunção de todos os sentidos e obtém-se resultados em frações de segundos. As diferenças entre a inteligência do ser humano e a inteligência artificial consistem na rapidez do processamento e na capacidade de aprendizado natural. Os sistemas dotados de inteligência artificial, não possuem grande poder de generalização, pois são escritos para propósitos específicos e também não agem de maneira espontânea. A inteligência artificial tem a sua concretização baseada nas redes neurais artificiais, assemelhando-se ao conhecimento humano na aquisição de conhecimentos por etapas de aprendizagem e no armazenamento desse conhecimento, (GUINGO, 2003).

Para um ser humano a execução de uma tarefa simples não necessita grandes cálculos, o que para as máquinas, já não acontece. O processamento num sistema computadorizado normalmente tende a uma complexidade maior, devido ao número de variáveis a serem controladas no processo, tornando-o mais lento e com grande gasto de energia, o que por fim pode resultar em dificuldades na execução do projeto, assim como, imprecisão nos resultados obtidos.

O maior objetivo das pesquisas que envolvem inteligência artificial é contribuir para que os equipamentos possam realizar tarefas comuns às pessoas, utilizando raciocínio e conhecimento prévio. Contudo, a maior dificuldade em implementar essas características provem da própria falta de conhecimento sobre o real funcionamento do cérebro humano, (DE MENDONÇA, 2008).

4.1. Redes Neurais Artificiais (RNA)

Baseado no funcionamento da estrutura neural e na maneira de aprendizado do cérebro humano, as redes neurais artificiais (RNA), são uma proposta para implementação de sistemas inteligentes. Tal qual o cérebro humano, a RNA procura simular uma rede de neurônios e suas ligações. Essas ligações ou conexões são chamadas de sinapses, termo biológico para

conexões entre neurônios. Então, uma RNA é um modelo matemático que toma como base a estrutura cerebral, (GUINGO, 2003).

Da mesma forma como um neurônio biológico, as RNAs são constituídas de forma a apresentarem camadas de entrada, camadas intermediárias e de saída, conforme Figura 4.1. As camadas de entrada são responsáveis por receberem os dados iniciais do processamento, e de enviá-los para a camada intermediária. A camada intermediária, que também é conhecida como camada escondida, processa esses dados e envia a resposta para a camada de saída, que por sua vez apresenta os dados processados como resposta. Ressalta-se que a depender da complexidade dos dados a serem processados, a camada escondida pode ser multiplicada em várias outras camadas, onde uma camada passa a ser responsável pelo refinamento dos dados recebidos no interior da própria camada, até chegarem a uma resposta adequada.

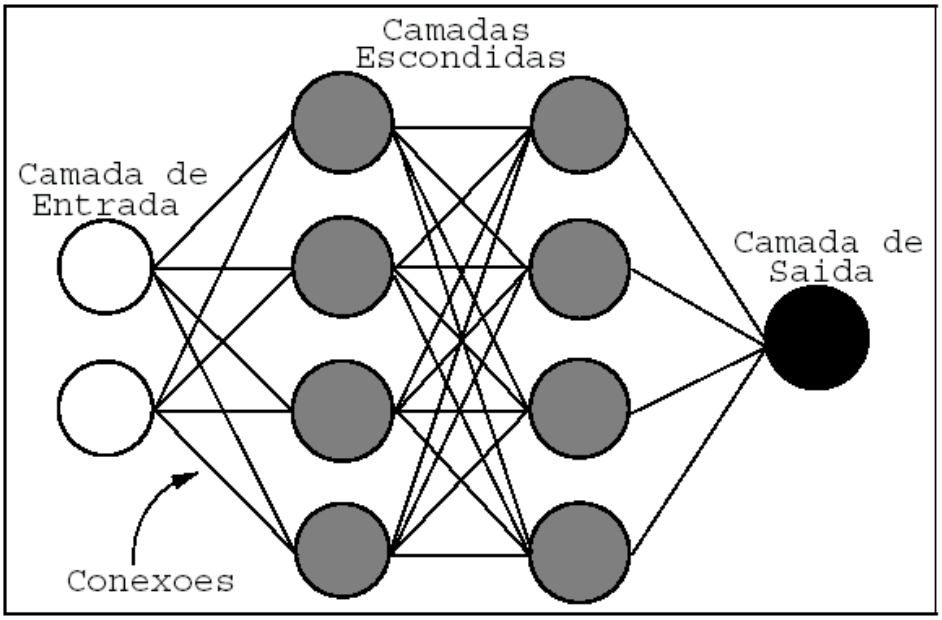


Figura 4.1 – Rede Neural Artificial em Camadas.

Fonte: (BOURCHARDT, 2006)

Então, ao contrário dos computares tradicionais, que funcionam conforme a arquitetura de Von Neumann, possuindo apenas um processador sequencial, as RNAs são subdivididas em vários processadores mais simples chamados de nós de RNA e funcionam como neurônios artificiais. Embora as

RNAs tenham como base o funcionamento dos neurônios humanos, o seu desempenho está muito aquém das redes neurais biológicas.

A inteligência artificial possui três componentes fundamentais, são eles: representação, raciocínio e a aprendizagem. A representação é a utilização de uma linguagem simbólica para referenciar tanto o conhecimento genérico, assim como, o conhecimento específico a respeito de um dado problema. O conhecimento pode ser declarativo ou procedimental. Na representação declarativa, o conhecimento se apresenta como uma coleção estática de fatos, e na representação procedimental, o conhecimento está acoplado a um código executável. Quanto ao raciocínio, se aplica à habilidade para tratar e solucionar problemas. O sistema deve ser capaz de controlar e determinar os respectivos procedimentos relacionados a cada problema, gerenciando o momento de início dessa solução e também ser capaz de perceber o momento de encerrar a intervenção, visto que o problema já se encontra resolvido. A aprendizagem é o componente, onde o ambiente fornece informações para a RNA, o conhecimento é retido num determinado banco de aplicações para ser executado em situações futuras. Todavia, pode acontecer do ambiente de aprendizagem fornecer informações imperfeitas para a rede, pois cada situação tem suas particularidades. Então o sistema tem o recurso de operar sobre suposições, neste caso, a rede é realimentada e o pretenso ato de execução passa a ser revisado, caso a hipótese tenha coerência o ato é executado, se não houver sentido para a medida, a rede toma outra caminho de decisão, (BOURCHARDT, 2006).

O processo de aprendizagem possui duas formas de processamento de informação, o indutivo e o dedutivo. No processamento indutivo, as determinações são oriundas dos dados brutos e da experiência anterior da rede. No dedutivo, utiliza-se todo o banco com os conhecimentos gerais e semelhantes para que se possam determinar os fatos específicos.

O diferencial quantitativo ente uma RNA e um cérebro, é que uma RNA pode possuir uma ordem de milhares de unidades de processamento, o cérebro reside na casa de bilhões de neurônios. O neurônio responsável por transmitir o pulso, faz o seu controle através da intervenção na polaridade da membrana pós sináptica, aumentando ou diminuindo a frequência do sinal transmitido. As RNAs, fazem esse controle não pelo chaveamento da

frequência, mas pela emissão de sinais positivos e negativos, o que se assemelha ao chaveamento por amplitude de fase. Outro diferencial entre essas redes, é que as RNAs são uniformes, já as redes naturais, são uniformes em apenas algumas partes do organismo.

4.2. Treinamento de RNA

Para que a RNA possua o conhecimento correspondente ao seu objetivo de atuação, é necessário que a rede passe por um período de treinamento. Tendo em vista que a velocidade desse período dependerá diretamente da complexidade do problema o qual a RNA se propõe a solucionar. O número de treinamentos possíveis para uma RNA está intrinsecamente ligado ao número de neurônios da camada de entrada. A rede neural recebe dados do ambiente sem nenhum filtro de suas influências e fatores adversos, então uma rede deve ser bem treinada, para que ela possua um maior potencial de generalização, e só assim, se poderá obter respostas a contento. Ressaltando que um treinamento deficiente da rede pode gerar desde a inexistência de respostas a respostas incorretas, que seria o pior caso a se tratar.

O treinamento da RNA pode ser do tipo supervisionado, ou não supervisionado. No primeiro caso, a rede recebe as entradas e simultaneamente as respectivas repostas para essas entradas, o que contribui para que a rede vá norteando o princípio das suas decisões. No treinamento não supervisionado, os parâmetros livres da rede são ajustados através de um algoritmo de aprendizagem não supervisionado, então se utiliza a competição entre os neurônios de saída, de modo a se buscar a melhor representação do padrão da entrada, ou seja, a resposta mais apropriada para aquela entrada.

O maior problema relacionado ao sucesso no treinamento das RNAs, é a dificuldade em associar o processo de decisão gerado a partir de uma determinada entrada, porque a RNA não possui um representação explícita do conhecimento armazenado. Isso quer dizer que existe um nível de abstração muito alto entre o momento do input, entrada de informação, e seu output, resposta processada, e o esforço incidi em compreender e relacionar essas variáveis de entrada e saída, (BOURCHARDT, 2006).

As maiores vantagens de uma RNA treinada são: 1. capacidade de entendimento pelo usuário do conhecimento adquirido pela rede; 2. identificar

e, se necessário excluir potenciais saídas que possam originar resultados errados; 3. identificar o tamanho mais adequado para a estrutura de uma RNA.

A arquitetura das redes neurais pode ser classificada como estática, dinâmica e ainda como de camada única ou múltiplas camadas. Sendo que a arquitetura dinâmica também é conhecida como *Fuzzy*, (BOURCHARDT, 2006).

4.3. Lógica Difusa (*Fuzzy*)

Nos ambientes de programação, a conduta normal é trabalhar com valores exatos, porque são baseados na lógica binária. Porém, na inexistência de valores precisos, muitos resultados serão expressos de forma errada quando o processamento ou a busca recair sobre um valor inexato.

Assim a lógica difusa pretende representar uma forma inovadora para manuseio de informações imprecisas, tentando fazer com que os computadores possam compreender e traduzir as intenções humanas, criando regras que utilizem valores aproximados ou subjetivos, tais como: quase bom, meio alto, razoável, entre outros. A *fuzzy* pode ser integrada ao sistema de RNAs aumentando a capacidade de aprendizado da rede através de interfaces com dados numéricos. Assim, ao emular a capacidade de compreensão humana, a lógica *fuzzy* ganhou grande notoriedade, sobretudo no mundo industrial devido as suas aplicações como ferramenta de controle industrial, comunicação homem-máquina e em sistemas de tomadas de decisões, (DE MENDONÇA, 2008).

4.3. Visão Computacional

Dentre as várias metas de pesquisa da inteligência artificial, a visão computacional constitui um objetivo de grande importância. Para isso, é preciso integrar *hardware* e *software*, conjugados em sistemas que os possibilitem perceber o mundo que os rodeia. Todavia, automatizar a visão de um equipamento, é um processo complexo e que requer muito empenho dos profissionais envolvidos.

Assemelhando-se à visão humana, a visão computacional tem a mesma ordem de funcionamento, onde primeiro se captura a imagem, em segundo lugar a imagem é processada, logo após vem a fase de identificação dessa

imagem. Porém, embora o princípio de funcionamento seja o mesmo, a maneira como uma máquina interpreta a imagem, é muito diferente da maneira como os seres humanos o fazem. As máquinas não percebem o encadeamento lógico ou familiar de uma cena, mas, sim, a identifica através de características peculiares e próprias, sendo uma identificação absolutamente impessoal, e também não têm capacidade para filtrar atributos alheios que não contribuem para a consecução do fato em evidência. Já os seres humanos, funcionam sob uma lógica muito mais complexa, onde é possível ignorar todas as circunstâncias que não têm participação ativa com fato a ser identificado, filtrando e armazenando apenas aquilo que é útil para o entendimento da ocorrência, (BOURCHARDT, 2006).

Então, a simples observação de algumas poucas cenas já seria razão suficiente para congestionar a capacidade de memória de um computador, visto que as máquinas não possuem a mesma capacidade de processamento paralelo que o cérebro humano possui. Assim pequenas modificações como sombras, cores e movimentos, entre outros são razões suficientes para sobrecarregar ainda mais a memória do computador. Para resolver este tipo de deficiência, os softwares baseados em visão computacional têm que ser destinados a aplicações específicas, diferentemente da pluralidade de aplicações da função visual humana.

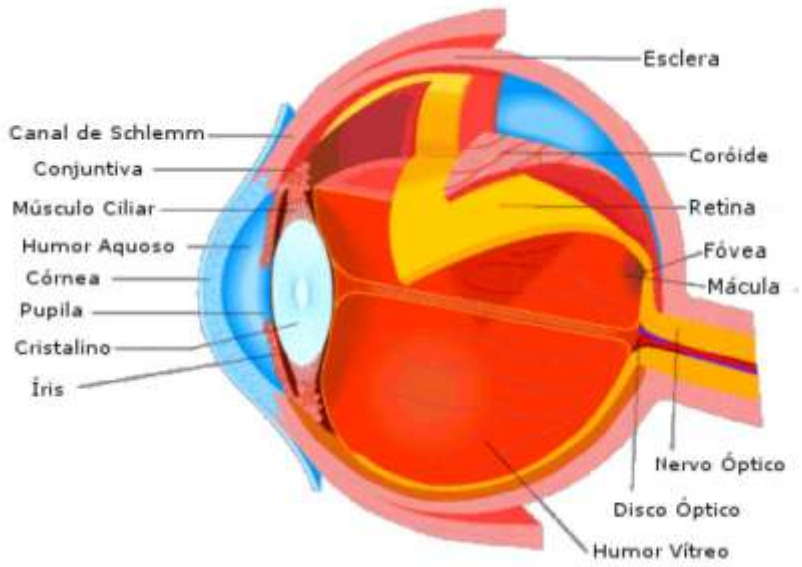


Figura 4.2 – Visão Humana.
Fonte: (BOUCHARDT, 2006)

Da mesma forma como demonstrado na Figura 4.2, a córnea é uma lente, sendo o elemento responsável pela focalização da imagem. A luz refletida pelos corpos atravessa o sistema de visão, e na sequência, agindo como transdutores, as células sensíveis à luz, presentes na retina, transformam os sinais luminosos em impulsos nervosos, que serão transmitidos e interpretados pelo cérebro, (BOURCHARDT, 2006).

Já no sistema computacional, a função visual pode ser perfeitamente substituída por uma *webcam*, reservando a este equipamento a função de captura da imagem, tal qual o olho humano. É preciso ressaltar também, que quanto mais simples for a imagem, melhor para o sistema processamento, pois detalhes e cores variadas tornam o processo de reconhecimento mais lento, por isso, imagens preto e branco, ou seja, tons de cinza são mais indicadas para o sistema de visão computacional.

4.4. Técnicas de Visão Computacional

Os seres humanos são capazes de abstrair a imagem a ser verificada do fundo no qual está imagem está emersa. Isso proporciona a observação apenas daquilo que importa. A visão computacional ideal deveria ser capaz de realizar o mesmo filtro.

Hoje, para que o sistema de visão computacional seja capaz de excluir a cena do ambiente ao fundo, é utilizada a técnica de segmentação por subtração de fundo. Esta metodologia consiste em tornar estática a imagem da paisagem, até que haja alguma nova modificação no fundo da cena. Logo, a imagem passa a ser extraída do fundo estático durante períodos maiores de tempo o que proporciona uma maior economia de processamento e armazenamento desta imagem. Assim, a lógica do sistema funciona da seguinte forma: uma vez que a captação é tratada em tons de cinza, sempre que houver uma variação estimável nos pixels, a imagem atual passa a ser a imagem de referência no lugar da imagem antiga, caso não haja uma variação estimável, continua da mesma forma, (BOURCHARDT, 2006).

4.5. Reconhecimento de Caracteres

O reconhecimento de caracteres tem sido uma excelente fonte para a aplicação da visão computacional. Atualmente, tomando por base a necessidade da engenharia de tráfego moderna, uma das grandes aplicações para reconhecimento de caracteres, se aplica ao reconhecimento de placas de veículos automotores. Sendo assim, os órgãos competentes para o controle de tráfego, buscam essas tecnologias, a fim de otimizar o processo de identificação dos veículos, e também baratear o custo das operações. De outra forma, a tecnologia de identificação de placas de veículos pode também ser utilizada por organizações não-governamentais, com intuito de controlar e gerenciar o acesso de veículos ao interior de suas dependências, (GUINGO, 2003).

Então, logo após o processo de obtenção da imagem, é utilizado o processo de segmentação para que se possa fazer o reconhecimento efetivo da placa. A Figura 4.3, retrata uma imagem original obtida através de uma câmera monocromática.

Figura 4.3 – Imagem Original de uma Placa.

Fonte: (GUINGO, 2003)

Na metodologia da segmentação, são criados sete arquivos a parte, para que cada caractere da placa seja exclusivo do seu respectivo arquivo. Porém, pode acontecer do processo de segmentação não ser capaz de separar os caracteres nos correspondentes arquivos. Isso pode gerar um número de arquivos, neste caso, também chamados de segmentos, maior ou menor do que os sete inicialmente previstos. Alguns fatores podem contribuir para a imprecisão dessa identificação, como placas em mau estado de conservação, placas enlameadas, iluminação deficiente ou outros fatores que impossibilitem

uma melhor identificação. A Figura 4.4, demonstra uma imagem segmentada, onde cada caractere está representando um arquivo em separado.

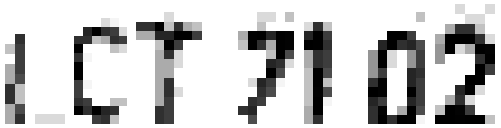


Figura 4.4 – Imagem DE UMA Placa Segmentada.

Fonte: (GUINGO, 2003)

O próximo passo, seguinte à segmentação, seria fazer esse reconhecimento utilizando uma das técnicas referenciadas na Figura 4.5. A metodologia chama-se projeção poligonal, a qual deriva dos algoritmos de detecção de controle. Na Figura 4.5, pode-se perceber alguns exemplos do processo de extração de características da projeção poligonal.

a. Método do quadrado **b. Método do hexágono** **c. Quadrado rotacionado**

Figura 4.5 – Exemplo de Metodologias Utilizadas na Projeção Poligonal.

Fonte: (GUINGO, 2003)

O item da Figura 4.5a. é extraído pelo método poligonal do quadrado. Nesse caso, é feita uma varredura a partir de um dos lados do quadrado até o lado oposto, o processo repete-se para cada lado. Assim que a varredura percebe o contorno externo da imagem, é medida a distância em pixels desse contorno até o final da linha poligonal. O valor medido é armazenado no vetor de características. Se a varredura chegar ao lado oposto e não encontrar nenhuma imagem, assumirá o valor 0 no vetor de características. A Figura 4.6, mostra em detalhes a realização do método do quadrado.

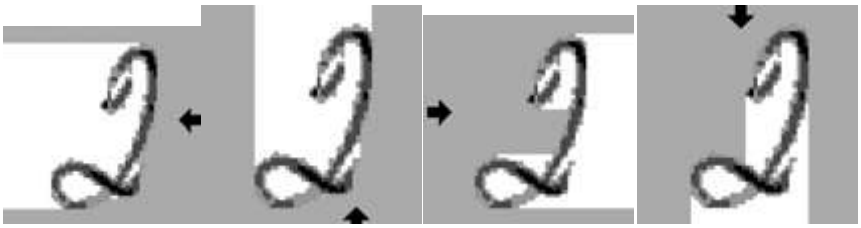


Figura 4.6 – Extração Pelo Método do Quadrado.
Fonte: (GUINGO, 2003)

Na Figura 4.4b, é utilizada a extração pelo método do hexágono, segue a mesma metodologia do quadrado, extraindo-se as medidas de cada lado do hexágono, até atingir o contorno externo da figura em análise, assim, da mesma forma o resultado final será inserido no vetor de características. A Figura 4.7, demonstra o processo de reconhecimento pelo método do hexágono de forma mais detalhada, onde cada linha de reconhecimento varre o hexágono em cada um dos seus seis lados, até encontrar o contorno da imagem, ou a outra margem do polígono. A vantagem desse método em ralação ao quadrado é que devido ao maior número de varreduras, a imagem será captada com uma precisão maior.

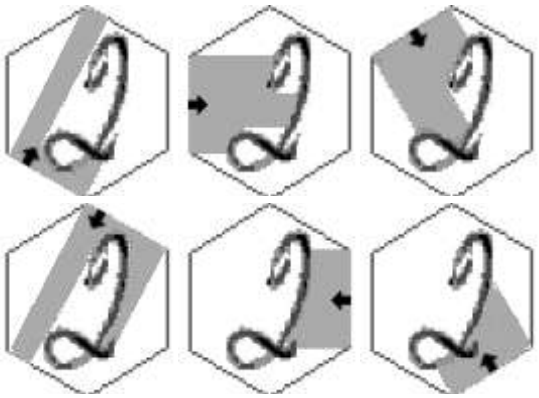


Figura 4.7 – Extração Pelo Método do Hexágono.
Fonte: (GUINGO, 2003)

Na Figura 4.4c, o método do quadrado rotacionado, tenta melhorar os resultados obtido pelos métodos do quadrado e do hexágono. O método do quadrado é mais eficiente se comparado com o método do hexágono, pois as linhas de reconhecimento varrem o polígono sem redundância. No método do hexágono, basta imaginar a faixa de reconhecimento viajando de cada

extremidade do polígono até o outro lado, é possível perceber a quantidade de faixas idênticas que essas linhas vão cobrir, neste caso, a isso se da o nome de redundância, pois são ordens emanadas de processos diferentes, mas que vão identificar a mesma informação. Assim, a proposta de otimização refere-se à possibilidade de rotacionar o quadrado em 45º, com isso tornou-se possível identificar detalhes que apenas o método do hexágono poderia identificar. Logo, este método tem um menor número de redundância se comparado ao hexágono, mas com um número de informações e detalhes muito maior, se comparado ao método do quadrado. Então, o método do quadrado rotacionado, une o melhor das duas tecnologias, conforme representado na Figura 4.8.

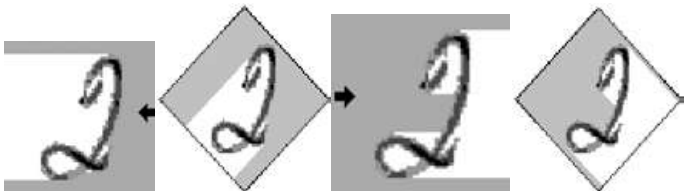


Figura 4.8 – Extração Pelo Método do Quadrado Rotacionado.

Fonte: (GUINGO, 2003)

A Figura 4.9, representa a metodologia de extração de caracteres, através da matriz de bits. Nesse caso, a imagem é transformada em dígitos binários onde as cores, branca e preta, são respectivamente 0 e 1, no caso de imagens em tons de cinza, a lógica do sistema prevê uma forma de calcular as matizes a fim de classificá-la com a binarização correspondente. Na sequência, os bits são alocados na respectiva matriz de bits.

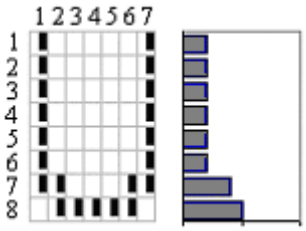


Figura 4.9 – Exemplo de Matriz de Bits

Fonte: (GUINGO, 2003)

Da mesma forma que a matriz de bits, a projeção horizontal, representada na Figura 4.10, utiliza um mapa de bits para poder extrair o caractere da imagem segmentada. Porém, o vetor de características é

composto de forma a se alocar o somatório dos pixels pretos em cada linha da matriz, ou dos tons de cinza desses pixels em cada linha da matriz. Da mesma forma, o método pode ser calculado como uma matriz transposta e implementar a chamada projeção vertical, onde o somatório dos bits pretos se dará através do somatório das colunas da matriz, e não mais das linhas.

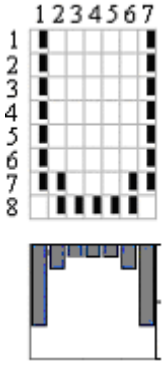


Figura 4.10 – Exemplo de Projeção Vertical

Fonte: (GUINGO, 2003)

4.6. Reconhecimento Óptico de Caracteres - OCR

Outra opção prática que tem potencial para resolver a proposta desse projeto, é o OCR. OCR é um acrônimo para *Optical Character Recognition,* ou seja, Reconhecimento Óptico de Caracteres. É uma tecnologia que reconhece caracteres de texto em imagens, transformando-os em texto editável. Os *scanners* são acompanhados de pelo menos um programa OCR, utilizado para obter texto de páginas impressas, o que vem, aos poucos, substituindo a digitação manual. A atuação dessa tecnologia também está relacionada às máquinas fotográficas e *webcam*, onde se utiliza a mesma tecnologia voltada para os *scanners* (DE MENDONÇA, 2008).

Os primeiros relatos sobre OCR foram feitos por David Shepard e Louis Tordella, quando tinham o objetivo de automatizar dados da Agência de Segurança dos Estados Unidos. Em 1950, juntamente com Harvey Cook eles construíram o primeiro software de OCR, o "Gismo". Após, Shepard fundou uma empresa, a *Intelligent Machines Research Corporation* (IMR), fabricando os primeiros softwares OCR comerciais. A IBM, em 1953, obteve uma licença da IMR e desenvolveu seu próprio software*,* dando–lhe o nome de *Optical*

Character Recognition, tornando, dessa forma, o termo OCR um padrão para essa tecnologia.

No princípio, os OCRs utilizavam implementações de redes neurais em software, o que causava lentidão e formas imprecisas no reconhecimento, principalmente quando lidando com uma língua diferente da inglesa, por exemplo o Português, onde o software se vê obrigado a confrontar estruturas como cedilhas, acentos graves, agudos, circunflexos e o extinto trema, entre outras peculiaridades da língua. Hoje a inserção é no hardware.

O princípio de reconhecimento dos caracteres via redes neurais não é simples. Em primeiro plano, existe uma estrutura chamada *perceptron*, uma rede neural, responsável pela entrada do sinal que recebe o estímulo e o passa para o neurônio. O neurônio recebe este sinal de entrada e repassa para os demais neurônios, causando uma subdivisão, a fim de concluir o processamento num tempo melhor. Para cada neurônio existe um peso diferente desse estímulo recebido, que pode determinar o processamento ou não do sinal. Em havendo processamento, será gerada uma saída que pode ser a saída final, ou então, ser redistribuída para novos neurônios se necessário, até o esgotamento total da necessidade de processamento. No caso da aplicação OCR, o estímulo é a matriz de bits, como são vários neurônios recebendo a entrada da matriz, o peso final calculado determina aproximadamente qual a letra que está ali. A Figura 4.11, mostra o princípio desse reconhecimento óptico.

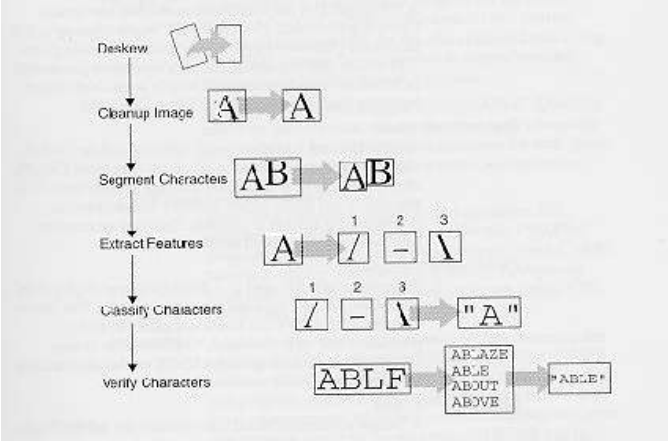


Figura 4.11 – Reconhecimento Óptico

Fonte: (DE MENDONÇA, 2008)

5 – IMPLEMENTAÇÃO

Antes de iniciar a implementação, o primeiro passo deve ser desbloquear a porta paralela, pois caso contrário, é impossível fazer qualquer tipo de experiência que utilize este recurso do computador. Para tanto, foi utilizado um programa chamado UserPort, na sua versão 1.0, cuja tela está representada na Figura 5.1. Todavia, caso conheça outro programa, ou não queira usar o indicado, há vários outros executáveis que podem fazer a mesma função.

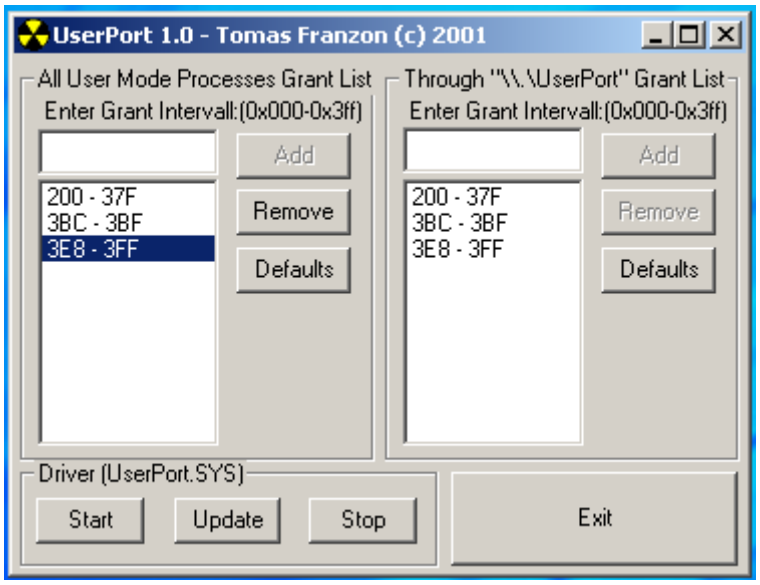


Figura 5.1 – Interface USERPORT

Fonte: Projeto Final, 2009

A manipulação do programa é simples. Basta digitar na caixa de texto do primeiro *groupbox* o número da porta que se deseja desbloquear, no caso $378 e $379, mandar adicionar através do botão *ADD* ainda no primeiro *groupbox*. Para finalizar, basta mandar executar o processo no botão *START*. Com esse procedimento, a porta paralela estará desbloqueada. Vale lembrar a necessidade de adicionar a biblioteca *simport* no campo correspondente ao arquivo *dpr* no DELPHI. Na falta dessa declaração o programa manifesta erro no momento de sua compilação. Veja o Apêndice A nas declarações de biblioteca, logo abaixo da palavra *uses*.

5.1. Montagem da Placa de Controle

No controle de um motor elétrico via porta paralela é indispensável construir um *driver* que seja capaz de fazer o controle físico desse motor. O *driver é* um *hardware* específico que tem a função de interfacear a comunicação entre o motor e a porta paralela. Neste projeto foi utilizado o CI ULN2003. A Figura 5.2 mostra o desenho da placa projetada.

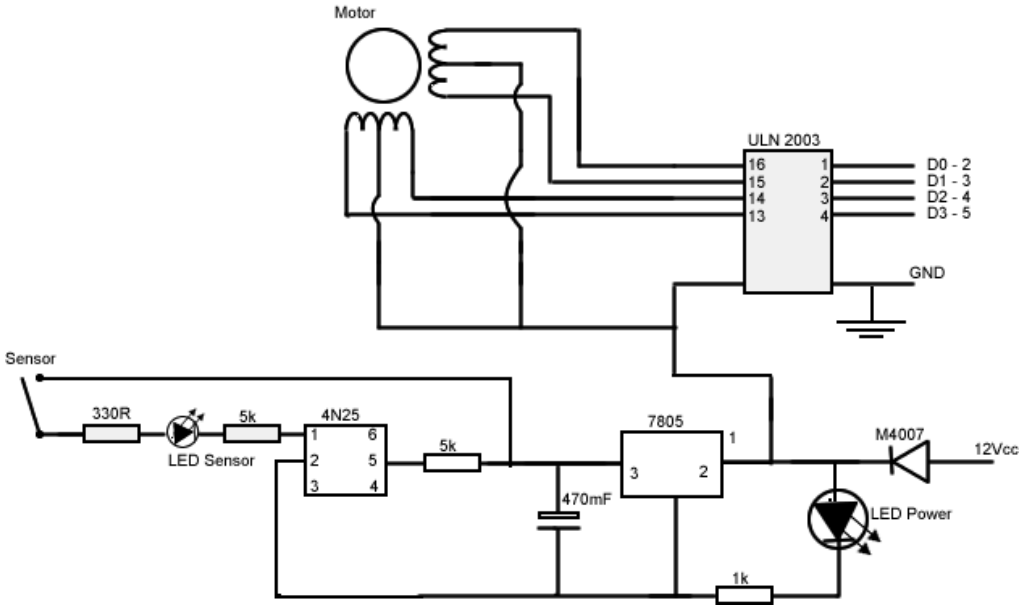


Figura 5.2 – DRIVER do Motor de Passo

Fonte: Projeto Final, 2009

O CI ULN2003 controla os pulsos enviados pela porta paralela para as bobinas do motor de passo. Dos pinos 2 até o 9, qualquer um deles pode ser utilizado, visto que estes pinos são os responsáveis por fazerem a transmissão paralela de dados. A este circuito integrado são ligados os fios do motor de passo, e os fios pertencentes ao cabo da porta paralela.

O circuito integrado LM 7805 (Figura 5.3) é um regulador de tensão.

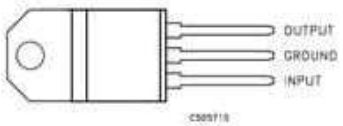


Figura 5.3 – Regulador de Tensão LM.

Fonte: Projeto Final, 2009

Os reguladores de tensão têm por princípio reduzir os valores de tensão CC. Eles executam essa operação às custas de dissipação térmica. No caso deste projeto, o LM 7805, recebe a tensão de 12 VDC da fonte e a transforma

em 5 VDC. A Figura 5.4 mostra o mesmo regulador de tensão com diferentes formas de encapsulamento.

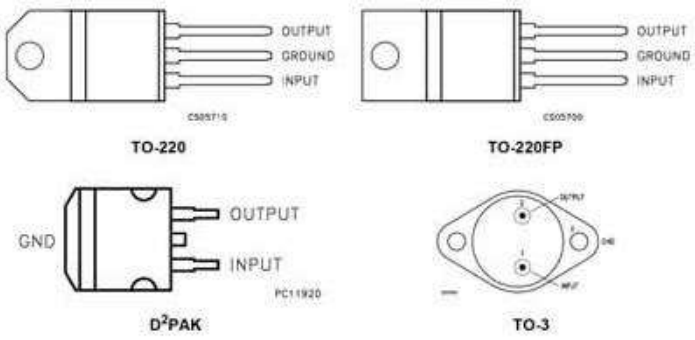


Figura 5.4 – Encapsulamento RT

Fonte: Projeto Final, 2009

Também constituinte do circuito da Figura 5.2, o opto acoplador 4N25, possui a função de receber a tensão de 5 VDC e enviá-la para a porta paralela toda vez que o sensor é pressionado.

Figura 5.5 – Opto acoplador 4N25

Fonte: Projeto Final, 2009

O resistor de 5 k é um limitador de corrente que tem a função de manter a tensão no pino 5 do opto acoplador quando a chave do sensor não estiver pressionada. O capacitor de 470 também age como limitador de corrente, mantendo a tensão de 5 VCC e reduzindo possíveis ruídos gerados pela fonte e pelo 7805. O diodo 4007 é um retificador de corrente, protegendo a placa contra possíveis inversões de polaridade. O diodo que corresponde ao LED verde está ligado em série com o opto acoplador, por isso apresenta uma indicação visual sempre que o circuito é fechado, ou seja, quando o sensor é pressionado. Ficando a placa após montada com a conformação retratada na Figura 5.6.

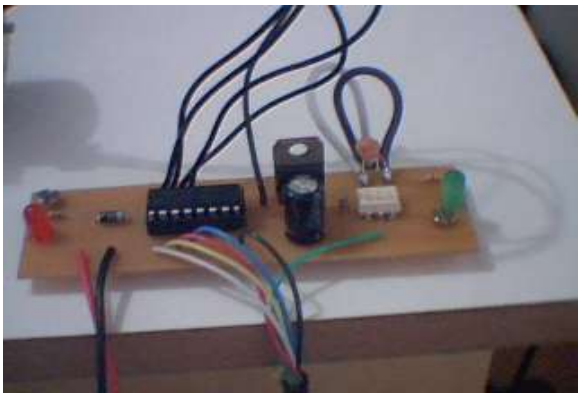


Figura 5.6 – Placa de Controle

Fonte: Projeto Final, 2009

A Figura 5.7 mostra o motor de passo que será controlado pela placa desenvolvida.

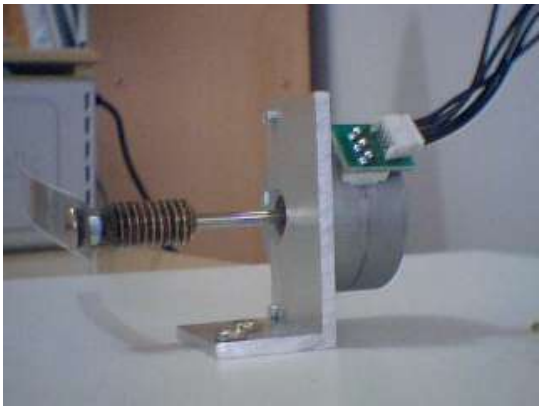


Figura 5.7 – Motor de Passo

Fonte: Projeto Final, 2009

A Figura 5.8 retrata o Motor de Passo e a Placa.

Figura 5.8 – Motor de Passo e Placa

Fonte: Projeto Final, 2009

5.2. Sistema de Controle de *Software*

A interface de controle da Figura 5.9 foi desenhada em DELPHI 7.0.

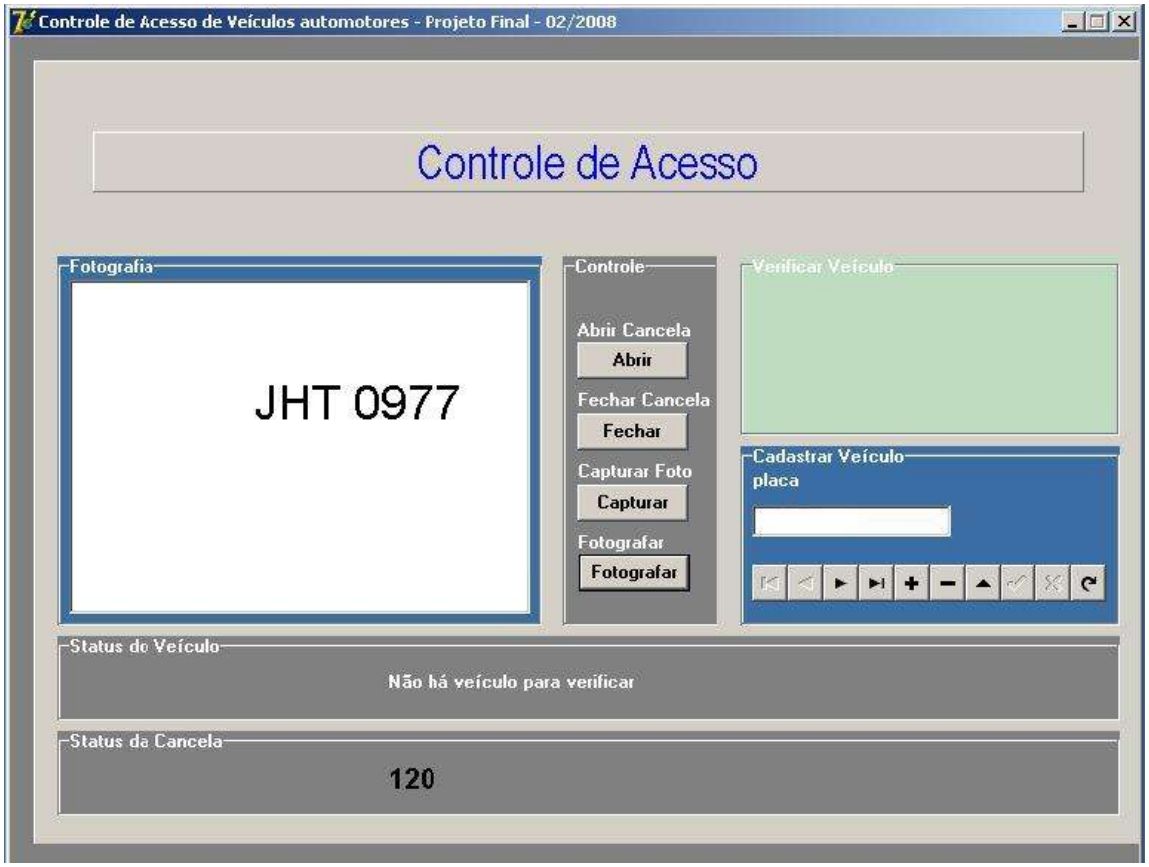


Figura 5.9 – Interface de Controle do Sistema.

Fonte: Projeto Final, 2009

Através desta interface, é possível obter o controle tanto do *software* como do *hardware*. O sistema é projetado para funcionar automaticamente. Ao pressionar o sensor, a *webcam* é disparada, a imagem é convertida via OCR em texto, e este dado é armazenado no banco de dados. Na sequência, é feito um reconhecimento da placa verificada com relação às demais placas já cadastradas. Em caso de reconhecimento positivo, é permitida a passagem e, em caso de não reconhecimento, a cancela permanece fechada.

Porém, como existe a possibilidade do sistema falhar, assim como se deparar com situações imprevistas e acabar por prover respostas inadequadas, foi implementado um modo manual para fazer o controle automático do sistema. Assim, é possível realizar o controle manual para interagir em caso de pane, ou situações adversas.

Segue uma descrição detalhada de todos os campos que constituem a interface apresentada na Figura 5.9.

- O campo Fotografia é responsável por realizar a identificação da placa e prosseguir no processamento automático do sistema.
- O campo Controle possui os botões Abrir e Fechar, que são responsáveis pelo controle manual do equipamento.
- O campo Verificar Veículo é utilizado para exibir um texto avisando se o veículo está ou não cadastrado.
- O Campo Cadastrar Veículo deve ser utilizado sempre que um veículo tiver acesso pela primeira vez ao sistema. É responsável pela inserção dos dados no banco, e só após esta inserção, os mesmos poderão ser verificados.
- O Campo Status do Veículo retorna a informação (Verificação em processamento... AGUARDE), ou (Não há veículo a VERIFICAR), caso o sensor esteja, respectivamente pressionado ou não.
- Finalmente, o campo Status da Cancela, informa se a cancela encontra-se aberta ou fechada.

Toda a codificação que dá origem a esta interface encontra-se no Apêndice A deste trabalho.

6 – APLICAÇÃO DA SOLUÇÃO COM RESULTADOS

Este capítulo inclui as condições nas quais pode ser realizada a aplicação da solução proposta e uma descrição da demonstração, evidenciando o cumprimento dos objetivos propostos e a validação dos resultados obtidos.

6.1. Ambiente de Simulação

O ambiente de simulação é composto por 2 computadores, um para rodar a aplicação e outro para poder digitar os caracteres correspondentes à placa, conforme as Figuras 6.1 e 6.2 a seguir.

Figura 6.1 – Computador 1

Figura 6.2 – Computador 2

Deve-se colocar uma *webcam* cuidadosamente instalada na frente do computador 2, a fim de se obter a melhor imagem. O protótipo deve estar ligado a uma fonte de 12 VCC e também à porta paralela. Cabe ressaltar que não se deve deixar a placa muito tempo ligada pois, mesmo sem trabalhar, as bobinas do motor continuam recebendo energia, podendo esquentar até o comprometimento total do motor elétrico.

6.2. Utilização do Protótipo

Para utilizar o protótipo, deve-se iniciar com o reconhecimento da placa. A princípio, é possível fazer esse reconhecimento de duas maneiras, uma manual, e outra automática. Independente de qual opção seja executada, no momento do reconhecimento, o DELPHI chamará para primeiro plano uma aplicação responsável pela identificação dos caracteres da placa. Neste

trabalho, a aplicação escolhida para executar esta tarefa foi um software chamado Top OCR. Assim, no instante da execução, aparecerá a imagem conforme a Figura 6.3.

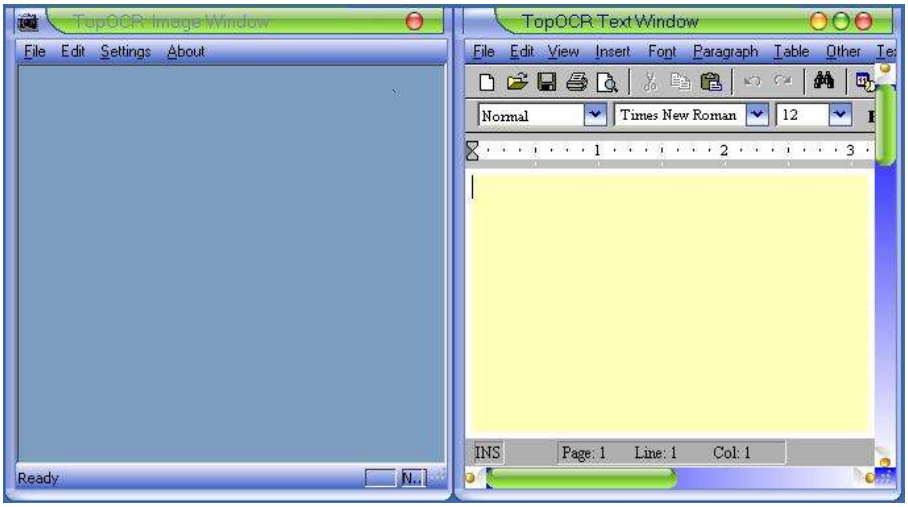


Figura 6.3 – Top OCR

Fonte: Projeto Final, 2009

Na modalidade manual, basta clicar no botão fotografar, uma vez, a partir de então, deve-se soltar o mouse, pois existem controles sobre o mouse para que o mesmo receba a *string* captada pelo OCR.

Para o controle automático basta pressionar e soltar apenas uma vez o sensor do protótipo, circulado em vermelho na Figura 6.4.

Figura 6.4 – Protótipo

Fonte: Projeto Final, 2009

Uma vez pressionado o sensor, ele não deve ser pressionado novamente ou mantido pressionado. Não se deve manipular o sistema até

findar a operação de reconhecimento, e da mesma forma, não tocar no mouse. Na sequência, a *string* é obtida e inserida no banco de dados para que o sistema possa concluir a operação.

6.3. Resultados Práticos Obtidos

É possível fazer o controle de acesso, porém, o ambiente de simulação tem algumas restrições para ser montado, caso contrário, a identificação não ocorre de forma apropriada.

A primeira restrição diz respeito à luz. No caso do computador 2, Figura 6.2, que será fotografado, não poderá haver uma incidência de luz acentuada sobre a tela, pois isso pode prejudicar a fotografia. Caso a imagem não seja satisfatória, o OCR retornará uma resposta errada. A fonte utilizada em todos os testes foi a fonte ARIAL. O tamanho da fonte pode variar de 20 a 40. Fora deste espectro, causa deturpação na aquisição da *string* do OCR. A distância entre a Câmera e o monitor, pelo que se pode observar, deve estar entre 20 e 30 cm. Da mesma forma, o que não estiver estabelecido nessas medidas também dá origem a uma identificação errada.

Os testes correspondentes ao monitor 2 foram feitos num FLATRON W1952TQ, da LG, e também sobre um monitor de 15'' de um notebook ACER. O monitor da LG apresenta um índice maior de resultados satisfatórios, o notebook também funciona, porém necessita de mais ajustes e cuidados no momento de obtenção da fotografia.

Sendo assim, desde que a captura da imagem seja obtida sem erros, não há problemas quanto ao desempenho do sistema. Porém, caso a captura não ocorra, inviabiliza parcialmente o desempenho do programa. Foi utilizada uma *webcam* de qualidade básica, fabricada pela GOLDSHIP.

6.4. Sugestões para Implementações Futuras

O sistema tem alguns aspectos que podem ser repensados, ou até melhorados. O primeiro ponto é o próprio OCR. Deve-se dominá-lo melhor, talvez até desenvolver uma própria solução para este produto. A fotografia só funciona se os caracteres fotografados estiverem estáticos, o que deve ser objeto de estudo e melhoria. Enquanto protótipo, esta solução pode ser mostrar funcional, mas é absolutamente inviável numa situação real. Outro ponto a ser

observado seria implementar um sistema mecânico para a subida e descida da haste da cancela. No protótipo aqui apresentado, a haste está presa ao motor elétrico por meio de um parafuso. Um controle mecânico se aproximaria mais da situação real.

7 – CONCLUSÃO

Sistemas automatizados fazem parte de um processo irrevogável. As vantagens trazidas pela automação são benéficas às pessoas, pois além de trazer conforto e segurança, no caso atividades insalubres, proporciona a possibilidade para se dedicar a outras atividades inteligentes e variadas. Embora o processo de automação possa gerar o desemprego de algumas categorias, este mesmo processo também é dotado de uma grande capacidade de empregabilidade para profissionais especializados e capazes de dominar e gerenciar equipamentos e sistemas de automação.

Seguindo a mesma linha dos processos automatizados, a proposta desta monografia é teoricamente exequível, podendo ser plenamente executada no campo prático, desde que seja seguido o programa de fabricação do *hardware* e *software*, assim como, a viabilização dos ajustes sugeridos no tópico 6.3 – Implementações Futuras. Lembra-se que as implementações futuras trazem melhoras substanciais ao projeto, porém não são essenciais para a sua demonstração enquanto protótipo.

Os ganhos trazidos pela proposta deste projeto são evidentes, pois estão relacionados com aceleração de providências, tais como a necessidade de identificar rapidamente as pessoas que podem ou não ter acesso a um ambiente, ou ainda, a redução do número de pessoas cuja única ocupação é anotar placas e operar uma cancela. Dessa forma, o sistema provê uma nova atividade para este empregado, que ao invés de controlar uma única cancela, a partir de então, passa a gerenciar um sistema que opera várias cancelas, o que pode significar uma economia razoável para o empregador e consecutivas melhoras não só na formação do preço do serviço oferecido, mas também, no salário dos operadores deste sistema.

REFERÊNCIAS

BOURCHARDT, F. E. Identificação de Veículos Utilizando Técnicas de Visão Computacional e Inteligência Artificial. Monografia, Rio Grande do Sul, 2006.

BRAICK, P. R. e MOTA, M. B. História: das cavernas ao terceiro milênio. Moderna, São Paulo, 2005.

BRAIN, M. Como funcionam os Motores Elétricos. Disponível em: <http://ciencia.hsw.uol.com.br/motor-eletrico.htm>. Acesso em 16 de Outubro de 2008.

Controladores Programáveis e de Posicionamento. Disponível em: <http://www.weg.net/br/Produtos-e-Servicos/Automacao/Sistemas-de-Automacao-e-Controle-de-Processos/Controladores-Programaveis-de-Medio-e-Grande-Porte>. Acesso em 22 de Maio de 2009.

CRUZ, T. A. L. Controle e Acionamento de Máquinas. Nota de Aula, Pernambuco, 2005.

CHAVES, A. S. Física II. Reichman & Afonso Editores, Rio de Janeiro, 1996.

CREDER, H. Instalações Elétricas. LTC, Rio de Janeiro, 1991.

DA SILVA, J. M. G. Funções de Transferência. Disponível em: <http://www.ece.ufrgs.br/~jmgomes/pid/Apostila/apostila/node8.html>. Acesso em 10 de Maio de 2009.

DE MENDONÇA, L. L. F. Proposta de Arquitetura de um Sistema com Base em OCR Neuronal para Resgate e Indexação de Escritas Paleográficas do Sec. XVI ao XIX. Dissertação de Mestrado em Engenharia Elétrica, Brasília, 2008.

DE PÁDUA, A. Placa POWER DRIVER UNIPOLAR. Disponível em: <http://www.iorobotics.com/>. Acesso em 20 de Outubro de 2008.

DEL TORO, V. Fundamentos de Máquinas Elétricas. LTC, São Paulo, 1999.

FITZGERALD, A. E., KINGSLEY C. e KUSKO A. Máquinas Elétricas. McGraw-Hill do Brasil. LTDA, Recife, 1975.

GUINGO, B. C. Reconhecimento Automático de Placa de Veículos Automotores. Curso de Mestrado, Rio de Janeiro, 2003.

IDOETA, I. V., CAPUANO, G. Elementos de Eletrônica Digital, Érica, São Paulo 1988.

JRC. Stepper Motor Basics. Disponível em: <files.me.com/mirp06/e4wh2a>. Acesso em 22 de Outubro de 2008.

KLANDER, L, JAMSA, K. Programando em C/C++ A Bíblia. Makron Books, São Paulo, 1999.

KORTH, H. F., SILBERSCHATZ, A. Sistemas de Banco de Dados, Makron Books, São Paulo, 1995.

LEÃO, M. DELPHI 7: Curso Completo. Axcel Books, São Paulo, 2003.

MALVINO, A. P. Eletrônica: volume 1. Makron Books, São Paulo, 1995.

ROGERCOM, Introdução à Porta Paralela. Disponível em: < http://www.rogercom.com/>. Acesso em 30 de Outubro de 2008.

ROSÁRIO, J. M. Princípios De Mecatrônica. Prentice Hall, São Paulo, 2005.

SILVA. R. P. Eletrônica Básica: um enfoque voltado à informática. UFSC, Florianópolis, 1995.

SPANGHERO, A. Aprendendo DELPHI 7. Futura, São Paulo, 2003.

TORRES, G. Fundamentos de Eletrônica, Axcel Books, Rio de Janeiro, 2002.

WEG. Controladores Lógicos Programáveis. Disponível em: <http://pt.wikipedia.org/wiki/Controlador_l%C3%B3gico_program%C3%A1vel>. Acesso em 22 de Maio de 2009.

WOLSKI, B. Fundamentos de Eletromagnetismo. Ao Livro Técnico, Rio de Janeiro, 2005.

Manual de Ar Comprimido. Disponível em: <http://www.metalplan.com.br/pdf/br2/manual_de_ar_comprimido.pdf>. Acesso em 22 de Maio de 2009.

APÊNDICE A – CÓDIGO FONTE

Este código, escrito em DELPHI, na sua versão 7.0, é capaz de controlar o motor de passo, receber o estímulo enviado pelo sensor de presença, cadastrar e verificar o cadastramento da placa a ser identificada.

```
unit untPrincipal;

interface

uses

  Windows, Messages, SysUtils, Simport, Variants, Classes,
Graphics, Controls, Forms,

  Dialogs, ExtCtrls, StdCtrls, Menus, ShellApi, DB,
DBCtrls, Mask, ADODB;


type

  TfrmPrincipal = class(TForm)

    pnlFundo: TPanel;

    gbbFotografia: TGroupBox;

    gbbControle: TGroupBox;

    gbbVerificarVeiculo: TGroupBox;

    gbbCadastrarVeiculo: TGroupBox;

    gbbStatusVeiculo: TGroupBox;
```

```
gbbStatusCancela: TGroupBox;

pnlTitulo: TPanel;

Memo1: TMemo;

btnAbrir: TButton;

btnFechar: TButton;

lblAbrir: TLabel;

lblFechar: TLabel;

Timer1: TTimer;

lblStatusDaCancela: TLabel;

lblStatusDoVeiculo: TLabel;

btnCapturar: TButton;

lblCapturar: TLabel;

btnFotografar: TButton;

conexao: TADOConnection;
ADOTable1: TADOTable;
ADOTable1idt_placa: TAutoIncField;

ADOTable1placa: TWideStringField;
```

```
    Label1: TLabel;
    DBEditPlaca: TDBEdit;
    DBNavigator1: TDBNavigator;
    DataSource1: TDataSource;
    labelResposta: TLabel;
    ADOQuery1: TADOQuery;
    lblFotografar: TLabel;
   {Controla o tempo de envio de pulsos para as bobinas do
motor de passo}
    procedure Timer1Timer(Sender: TObject);
   {Abre a Cancela}
    procedure btnAbrirClick(Sender: TObject);
   {Fecha a Cancela}
    procedure btnFecharClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    {1. Abre o OCR;
     2. Grava a string em placa.txt;
     3. Exibe a string no Memo1;
     4. Envia para o banco de dados}
    procedure btnCapturarClick(Sender: TObject);
    procedure btnFotografarClick(Sender: TObject);
```

```
    procedure abrirTopOcr;
    procedure moverMouse(x : integer; y : integer);
    procedure pressionarMouse;
    procedure liberarMouse;
    function retirarEspaco(texto : string) : string;
    function estaCadastrado(placa : string) : boolean;
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  frmPrincipal: TfrmPrincipal;
  x:integer;
  y:integer=10;  // 10 é o valor de atraso entre os envios
implementation

{$R *.dfm}
```

```
//*****************************************************

////************** Manipula arquivos ******************

//*****************************************************

////LER arquivos ***************************************

function percorreArquivoTexto ( nomeDoArquivo: String ):String;

var arq: TextFile;

texto: String;

begin

AssignFile ( arq, nomeDoArquivo);

Reset (arq);

ReadLn (arq, texto);

while not Eof ( arq ) do

begin

{ Processe a linha lida aqui. }

{ Para particionar a linha lida em pedaços, use a função Copy. }

ReadLn (arq, texto);

end;

CloseFile ( arq );

Result := texto;

end;
```

```
////////////////////////////////////////////////////////

//Controla os pulsos enviados ao motor elétrico

procedure TfrmPrincipal.Timer1Timer(Sender: TObject);

begin

 x:=port[$379];       //coloca o valor de 379 na variavel x

 lblStatusDaCancela.caption:=inttostr(x); //mostra o  valor
de 379 no label

// if (Sensor pressionado x := 120)

// if (Sensor não pressionado x := 104)

if x = 104 then

begin

 lblStatusDoVeiculo.caption:=('Iniciar    verificação    de
veículo ... AGUARDE');  // se o valor de 379 é 111  escreva
carro no label

 DBEditPlaca.Clear;

 Memo1.Clear;

 btnCapturarClick(Sender);

end;

if x = 120 then

begin  lblStatusDoVeiculo.caption:=('Não  há  veículo  para
verificar');  // se o valor de 379 é 127 escreva sem carro
no label

end;

end;
```

```
////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////
////Abre a Cancela
procedure TfrmPrincipal.btnAbrirClick(Sender: TObject);
begin
    port[$378]:=12;       // sequencia que aciona o motor
para subir
   sleep (y);           // pause entre os pulsos do motor
    port[$378]:=6;
   sleep (y);
    port[$378]:=3;
   sleep (y);
    port[$378]:=9;
   sleep (y);


    port[$378]:=12;
   sleep (y);
    port[$378]:=6;
   sleep (y);
    port[$378]:=3;
   sleep (y);
```

```
    port[$378]:=9;
   sleep (y);


    port[$378]:=12;
   sleep (y);
    port[$378]:=6;
   sleep (y);
    port[$378]:=3;
   sleep (y);
    port[$378]:=9;
   sleep (1000);
     port[$378]:=0;
end;
////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////
////Fecha a Cancela
procedure TfrmPrincipal.btnFecharClick(Sender: TObject);
begin
   port[$378]:=3;   // sequencia que aciona o motor para
descer
   sleep (y);      // pause entre os pulsos do motor
```

```
port[$378]:=6;

sleep (y);

port[$378]:=12;

sleep (y);

port[$378]:=9;

sleep (y);

port[$378]:=3;

sleep (y);

port[$378]:=6;

sleep (y);

port[$378]:=12;

sleep (y);

port[$378]:=9;

sleep (y);


port[$378]:=3;

sleep (y);

port[$378]:=6;

sleep (y);

port[$378]:=12;

sleep (y);

port[$378]:=9;
```

```
   sleep (1000);
   port[$378]:=0;
end;
////////////////////////////////////////////////////////
////Cria o formulário e seta 0 na porta 378
procedure TfrmPrincipal.FormCreate(Sender: TObject);
begin
port[$378]:=0;
end;
////////////////////////////////////////////////////////
//////Lê a string no notepad  *************************
procedure TfrmPrincipal.btnCapturarClick(Sender: TObject);
begin
   DBEditPlaca.Clear;
   Memo1.Clear;
   btnFotografar.Click;
end;
////////////////////////////////////////////////////////
//////Captura a imagem no TOP_OCR **********************
procedure          TfrmPrincipal.btnFotografarClick(Sender:
TObject);
 var placa : string;
```

```
begin
    DBEditPlaca.Clear;
    Memo1.Clear;

    abrirTopOcr;
    sleep(3000);
    //File
    moverMouse(265,220);
    pressionarMouse;
    liberarMouse;
    sleep(3000);
    //Aquire
    moverMouse(280,315);
    pressionarMouse;
    liberarMouse;
    sleep(3000);
    ///Instantaneo
    moverMouse(560,520);
    pressionarMouse;
    liberarMouse;
    sleep(2000);
    //OCR
```

```
moverMouse(450,220);

pressionarMouse;

liberarMouse;

sleep(2000);

//Posicionar cursor para que ele possa copiar string

moverMouse(700,520);

pressionarMouse;

liberarMouse;

sleep(3000);

//Acessar o Edit - janela_2 TOP_OCR

moverMouse(640,220);

pressionarMouse;

liberarMouse;

sleep(1000);

//Acessar a opção select all - janela_2 TOP_OCR

moverMouse(640,380);

pressionarMouse;

liberarMouse;

sleep(2000);

//Acessar o Edit - janela_2 TOP_OCR

moverMouse(640,220);

pressionarMouse;
```

```
    liberarMouse;
    sleep(1000);
    //Acessar o Copy - janela_2 TOP_OCR
    moverMouse(640,260);
    pressionarMouse;
    liberarMouse;
    sleep(2000);
////////////////////////////////////////////////////////
    //Essa função trás o DELPHI novamente para primeiro
plano
    SetForegroundWindow(handle);

    Memo1.Clear;
    Memo1.PasteFromClipboard;

    sleep(2000);

    placa := retirarEspaco(Memo1.Text);

    if estaCadastrado(placa) = true then
       begin
           labelResposta.Caption := 'Cadastrado!';
```

```
            btnAbrir.Click;


            sleep(10000);
            btnFechar.Click;
        end
    else
        begin
            labelResposta.Caption := 'Não cadastrado!';
        end;
    //DBNavigator1.BtnClick(nbInsert);
    //DBEditPlaca.Text := retirarEspaco(Memo1.Text);
    //DBNavigator1.BtnClick(nbPost);
    //ShowMessage('Placa salva!');
end;
////////////////////////////////////////////////////////
//////Função responsável por abrir o TOP_OCR ***********
procedure TfrmPrincipal.abrirTopOcr;
var caminho : string;
begin
 caminho := 'C:\Arquivos de programas\TopOCR\topocr.exe';
 ShellExecute(Handle,  nil,  Pchar(caminho),  nil,  nil,
SW_NORMAL);
```

```
end;

//////////////////////////////////////////////////////

//////Função responsável movimentar o mouse ************

procedure TfrmPrincipal.moverMouse(x, y: integer);

begin

   //Mouse_Event(MOUSEEVENTF_ABSOLUTE  or  MOUSEEVENTF_MOVE,
x, y, 0, 0);

   SetCursorPos(x, y);

end;

//////////////////////////////////////////////////////

//////Função responsável liberar o mouse ***************

procedure TfrmPrincipal.liberarMouse;

begin

   Mouse_Event({MOUSEEVENTF_ABSOLUTE                    or}
MOUSEEVENTF_LEFTUP, 0, 0, 0, 0);

end;

//////////////////////////////////////////////////////

//////Função responsável fazer o mouse clikar **********

procedure TfrmPrincipal.pressionarMouse;

begin

    Mouse_Event({MOUSEEVENTF_ABSOLUTE                    or}
MOUSEEVENTF_LEFTDOWN, 0, 0, 0, 0);

end;
```

```
//////////////////////////////////////////////////////

//////Função responsável por retirar os espaçoes da placa
capturada

function    TfrmPrincipal.retirarEspaco(texto:    string):
string;

   var resp : string;

   i : integer;

begin

      resp := '';

      for i := 1 to length(texto) do

         begin

            case texto[i] of


'A','B','C','D','E','F','G','H','I','J','K','L','M','N'


,'O','P','Q','R','S','T','U','V','W','X','Y','Z'

               ,'0','1','2','3','4','5','6','7','8','9':
resp := resp + texto[i];

            end;

         end;

      result := resp;

end;
```

```
//////Função responsável por verificar se a placa consta no
BCO_DADOS

function    TfrmPrincipal.estaCadastrado(placa:    string):
boolean;

begin

     with ADOQuery1 do

       begin

           SQL.Clear;

           SQL.Add('select * from tbl_placa where placa =
''''+placa+'''');

           Open;

       end;

     if ADOQuery1.RecordCount > 0  then

        begin

           result := true;

        end

     else

        begin

           result := false;

        end;

end;

end.
//////////////////FIM*********************************
```