FELIPE ISSAMU KITADANI ODAGUIL
MAURÍCIO MIRARA PIESCO

MONITORAMENTO DE MANUFATURA VIA INTERNET COM ROBÔ MÓVEL

Monografia apresentada à Escola Politécnica da Universidade de São Paulo para a Conclusão de Curso de Engenharia Mecatrônica.

São Paulo
2010

FELIPE ISSAMU KITADANI ODAGUIL
MAURÍCIO MIRARA PIESCO

MONITORAMENTO DE MANUFATURA VIA INTERNET COM ROBÔ MÓVEL

Monografia apresentada à Escola Politécnica da Universidade de São Paulo para a Conclusão de Curso de Engenharia Mecatrônica.

Área de Concentração:
Engenharia Mecatrônica

Orientador:
Prof. Dr. Fabrício Junqueira

São Paulo
2010

Aos meus pais Thais e Roberto Vaz Piesco.

Aos meus pais Zilda e Evandro Kunio Odaguil.

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DA ENGENHARIA MECÂNICA/NAVAL DA ESCOLA POLITÉCNICA (EPMN) – USP.

Odaguil, Felipe Issamu Kitadani
Monitoramento de manufatura via internet com robô móvel / F.I.K. Odaguil; M.M. Piesco. -- São Paulo, 2010.
45 p.

Trabalho de Formatura - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos.

1. Robôs 2. Redes e comunicação de dados 3. Middleware I. Piesco, Maurício Mirara II.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos III. t.

Agradecimentos

Ao nosso orientador, Prof. Dr. Fabrício Junqueira, pela sua constante orientação e incentivo para o desenvolvimento deste trabalho.

À Escola Politécnica da USP, em especial ao Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos, que institucionalmente viabilizaram este trabalho.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

BEEP - *Blocks Extensible Exchange Protocol*

FTP - *File Transfer Protocol*

HTTP - *Hypertext Transfer Protocol*

OPC - *Object Linking and Embedding for Process Control*

RCPs - *Remote Procedure Calls*

SFC - *Sequential Flow Chart*

SIT - Sistema Inteligente de Transporte

SOAP - *Simple Object Access Protocol*

UDDI - *Universal Description, Discovery and Integration*

WSDL - *Web Service Description Language*

XML - *eXtensible Markup Language*

Resumo

A robótica tem apresentado importância crescente na sociedade: inicialmente, colaborando para o aumento da produtividade nas empresas e, recentemente, na exploração do espaço e na realização de tarefas domésticas. Nesse contexto, propõe-se utilizar robôs móveis para a monitoração da produção em células de manufatura. Para tanto, será utilizado o Robotino, um robô móvel dotado de câmera, sensores (de contato, ópticos e indutivos), e rede de dados sem fio, que permite sua operação à distância (teleoperação). O sistema produtivo a ser monitorado é a célula de manufatura didática do Laboratório de Sistemas Automatizados (MiniCIM). Utilizando-se WebServices, o Robotino deve ser movimentado para posições predefinidas do MiniCIM, permitindo o acompanhamento da produção.

Palavras-chave: Robôs móveis, teleoperação, WebService

Abstract

Robotics has presented increasing importance in the society: initially, collaborating with the growth of the productivity in the factories and, recently, with the exploration of the space and helping with domestic duties. In this context, we suggest the usage of mobile robots for monitoring the production in manufacturing cells. We will use the Robotino, a mobile robot equipped with a camera, sensors (inductive, optic and contact) and wireless connection, which allows us to operate it remotely (teleoperation). The productive system to be monitored is the didactic manufacturing cell of the of Automatized System laboratory (MiniCIM). Using WebServices, the Robotino will move to predefined positions of the MiniCIM, allowing the accompaniment of the production.

Keywords: mobile robots, teleoperation, WebService

1. Introdução

A maioria das pessoas se perguntam a respeito do que é um robô e qual sua função.

Segundo o dicionário Aurélio, robô é: um aparelho automático, geralmente em forma de boneco, que é capaz de cumprir determinadas tarefas. Pessoa que procede como um robô, isto é, que executa ordens sem pensar.

Segundo a R.I.A. (*Robotics Industries Association*), robô é um manipulador reprogramável e multifuncional projetado para mover materiais, partes, ferramentas ou dispositivos especializados através de movimentos variáveis programados para desempenhar uma variedade de tarefas.

Ainda há outra definição que diz que um robô é um dispositivo que permite realizar trabalhos mecânicos, normalmente associados a seres humanos, de uma maneira muito mais eficiente e sem a necessidade de pôr em risco a vida humana.

Na verdade não há uma definição que satisfaça todos. Joseph F. Engelberger, considerado o “pai da robótica”, declarou: “Eu não posso definir um robô, mas sei reconhecer um quando o vejo”.

A palavra “robô” deriva da palavra checa *robota*, usada pela primeira vez pelo checo Karel Capek numa Peça de Teatro – R.U.R. (*Rossum's Universal Robots*) – estreada em 1921 (Praga), no qual existiam trabalhadores capazes de fazer o trabalho de qualquer homem. Esses trabalhadores eram chamados de *robota*, que significa trabalho forçado em checo. Isso fez com que até hoje a maioria das pessoas relacionem robôs com uma máquina com aparência humana, porém um robô não precisa ser humanóide para ser considerado robô, por exemplo, um robô que aspira pó precisa parecer com um aspirador de pó e não um faxineiro (MURPHY, 2000).

O desenvolvimento inicial dos robôs baseou-se no esforço de automatizar as operações industriais. Este esforço começou no século XVIII, na indústria têxtil, com o aparecimento dos primeiros teares mecânicos. Com o contínuo progresso da revolução industrial, as fábricas procuraram equipar-se com máquinas capazes de realizar e reproduzir, automaticamente, determinadas tarefas. No entanto, a criação de verdadeiros robôs não foi

possível até a invenção do computador em 1940, e dos sucessivos aperfeiçoamentos das partes que o constituem (PAZOS, 2002).

O primeiro robô industrial foi o *Unimate* (Figura 1), desenvolvido por George Devol e Joe Engleberger, no final da década de 50, início da década de 60. As primeiras patentes de máquinas transportadoras pertenceram a Devol, máquinas essas que eram robôs primitivos que removiam objetos de um local para outro.

Figura 1 – *Unimate*

Pode-se afirmar hoje que a robótica é uma realidade tecnológica cujo progresso científico ainda apresenta diversos desafios a serem vencidos e um grande potencial a ser explorado (SHIROMA, 2004).

1.1. Objetivo

O objetivo deste trabalho é realizar o monitoramento das estações de trabalho da célula de manufatura didática do Laboratório de Sistemas Automatizados (MiniCIM) com uma câmera instalado no Robotino e por meio de comandos pela internet.

1.2. Estrutura

No capítulo 2 é apresentada a fundamentação teórica, no qual foram abordados temas importantes para a execução deste trabalho como robôs móveis, o próprio Robotino, teleoperação, WebService e MniCIM.

No capítulo 3 é desenvolvida a programação do Robotino, incluindo as estratégias utilizadas para a realização do objetivo; e a interface com o servidor OPC e a página da internet.

No capítulo 4 são apresentadas a conclusão e as recomendações para trabalhos futuros.

2. Revisão Bibliográfica

2.1. Robôs móveis

Segundo McComb (1987), existem duas vertentes na definição de robôs móveis: uma que diz ser o robô completo, autocontido, autônomo, que necessita instruções de seu mestre apenas ocasionalmente. Já a segunda é aquela que define robô como sendo qualquer equipamento que se mova por seus próprios meios, com o objetivo de executar tarefas próximas às humanas. As duas definições de McComb levam a aceitar que um robô móvel é capaz de manobrar livremente em seu ambiente, alcançando metas enquanto desvia de outros obstáculos (SPENCE & HUTCHINSON, 1995).

Uma vez que necessitam ser livres, ou seja, sem conexões e fios, a aplicação de robôs móveis sofre limitações, basicamente à armazenagem e geração de energia para o próprio robô (MCKERROW, 1991).

O primeiro robô móvel criado foi o *Shakey* (Figura 2), projetado entre 1966 e 1972 pelo *Stanford Research Institute*. *Shakey* foi o primeiro robô móvel a ter a capacidade de "pensar" em suas ações. Ele tinha uma variedade enorme de sensores, incluindo uma câmara de vídeo e sensores de toque, binários, (GROOVER *et al.*, 1988) e navegava entre as salas do laboratório, enviando sinais de rádio ao seu cérebro, um computador DEC PDP-10, que permitia efetuar algumas tarefas como empurrar caixas e evitar obstáculos (NITZAN, 1985). Deve-se atentar para o fato que a unidade de processamento embarcada no robô apenas coletava os sinais sensoriais e os enviava para um computador remoto que executava o processamento, transmitindo ao robô o comando que geraria a ação desejada.

Desde então a robótica móvel tem encontrado muitas aplicações, dentre elas pode-se citar (AUGUSTO, 2007):

- Execução de tarefas em ambientes inóspitos, nos quais a presença de seres humanos seria perigosa. Tem-se como exemplos: inspeção de tubulações industriais, de tanque de resíduo, e de usinas nucleares, busca de minas terrestres, aplicação de produtos químicos e aplicações militares (ex: veículos de reconhecimento);

- Exploração de ambientes para os quais enviar um operador humano seria muito difícil ou demandaria muito tempo ou custo, como, por exemplo, exploração espacial e submarina;
- Tarefas repetitivas com alto fator de fadiga ou degradáveis aos seres humanos, como transportes de materiais em ambientes de manufatura, limpeza comercial e domiciliar;
- Recreação e diversão.

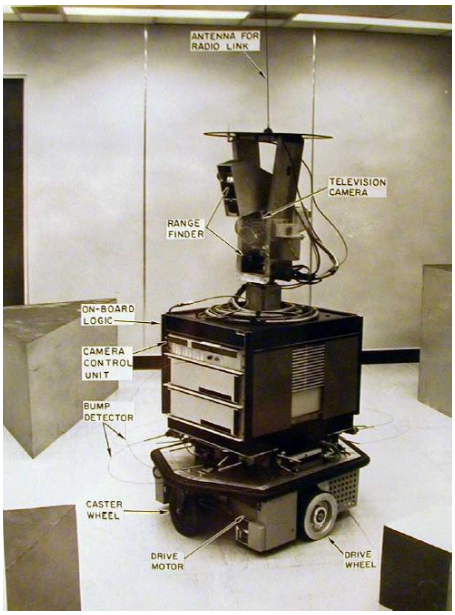


Figura 2 – *Shakey*

Como o próprio nome indica, os robôs móveis são muito mais versáteis, pois não precisam estar fixados a uma célula de trabalho, podendo ser utilizados em tarefas onde não existam limites geográficos, movimentando-se por meio de pernas ou rodas (MCKERROW, 1991).

2.2. Robotino

O Robotino (Figura 3) é um robô móvel desenvolvido pela FESTO e tem o propósito didático de oferecer ao aluno um primeiro contato com uma tecnologia que pode ser aplicada principalmente na área de automação industrial.

Figura 3 – Robotino

2.2.1. Unidade Motora

O Robotino é equipado com um sistema de direção omnidirecional (Figura 4), composto por 3 rodas posicionadas a 120° uma da outra, o que permite a movimentação do robô em todas as direções – para frente, para trás e para os lados – além de ter a capacidade de rotacionar em torno de si mesmo.

1 – Motor DC
2 – Encoder
3 – Roda omnidirecional
4 – Caixa de engrenagem
5 – Correia dentada

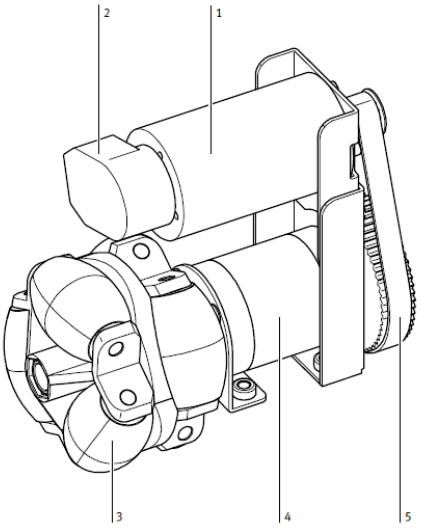


Figura 4 – Sistema de direção

2.2.2. Câmera

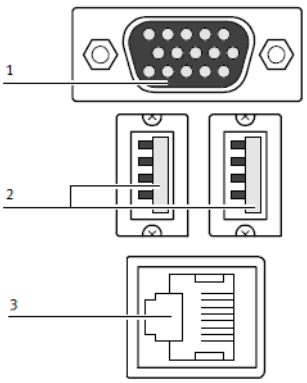
O Robotino vem equipado com uma câmera cuja altura e inclinação pode ser ajustada. Com a câmera é possível exibir imagens em tempo real com a ajuda do *software* Robotino View, que ainda oferece uma séria de opções de processamento de imagem que podem ser usadas para o controle do robô. A câmera utiliza conexão USB e tem resolução máxima para vídeo de 640x480 pixels e de 1024x768 para captura de imagem, que são armazenadas em formato BMP ou JPG.

2.2.3. Unidade de controle

A unidade de controle do Robotino consiste de 3 componentes:

- Processador PC 104, compatível com MOPSlcdVE, 300 MHz, e sistema operacional Linux com núcleo em tempo real, SDRAM 128 MB;
- *Compact flash card* (256 MB) com interface para programação de aplicativos em C++ para controle do Robotino;
- Ponto de acesso de rede sem fio.

Além disso, possui interfaces Ethernet, 2 entradas USB e saída VGA (Figura 5), que podem ser usadas para conectar um teclado, um mouse e/ou um monitor.



1 – Conector VGA
2 – Entradas USB
3 – Interface Ethernet

Figura 5 – Interfaces

2.2.4. Sensores

O Robotino vem montado com seguintes sensores:

- 9 sensores infravermelhos (Figura 6) para a medição de distância montados com um ângulo de 40° um com o outro que permitem o reconhecimento de algum objeto localizado entre 4 cm e 30 cm do sensor;
- 1 detector de colisão (Figura 6) que envolve todo o perímetro do robô;
- 3 *encoders* incrementais, um em cada motor.

Além desses, a FESTO oferece os seguintes sensores:

- Sensor fotoelétrico, que utiliza um cabo de fibra ótica, usado para a detecção de superfícies com graus de reflexão da luz diferentes (Figura 7a);
- Sensor indutivo analógico capaz de detectar objetos metálicos (Figura 7b).

2.2.5. Entradas e saídas

Para conectar sensores e atuadores adicionais, o Robotino possui as seguintes entradas e saídas posicionadas conforma a Figura 8:

- 8 entradas analógicas (0 a 10 V) (AIN0 a AIN7);
- 8 entradas digitais (DI0 a DI7);
- 8 saídas digitais (DO0 a DO7);
- 2 relês para atuadores adicionais (REL0 e REL1).

2.2.6. Robotino View

O Robotino View é a interface de programação de Robotino, nele a programação é feita com blocos, no qual cada bloco representa uma função cujos parâmetros podem ser modificados. Com ele é possível acompanhar os dados e a imagem da câmera em tempo real quando conectado com o robô.

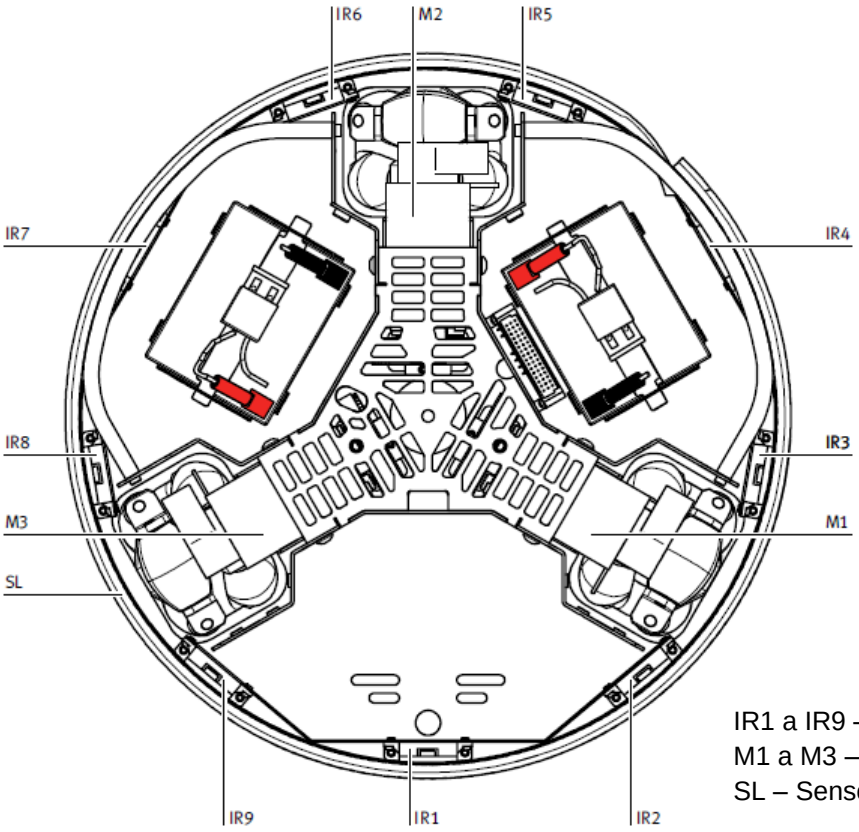


Figura 6 – Sensores infravermelhos e de colisão

(a) (b)

Figura 7 – (a) Sensor ótico; (b) Sensor indutivo

Na Figura 9 é apresentada a interface gráfica do Robotino View com um programa que controla o Robotino com um painel de controle:

O usuário pode desenvolver programas a partir do Robotino View e em seguida carregá-lo no Robotino ou pode optar por programar em outras linguagens como C++, Java e MATLAB.

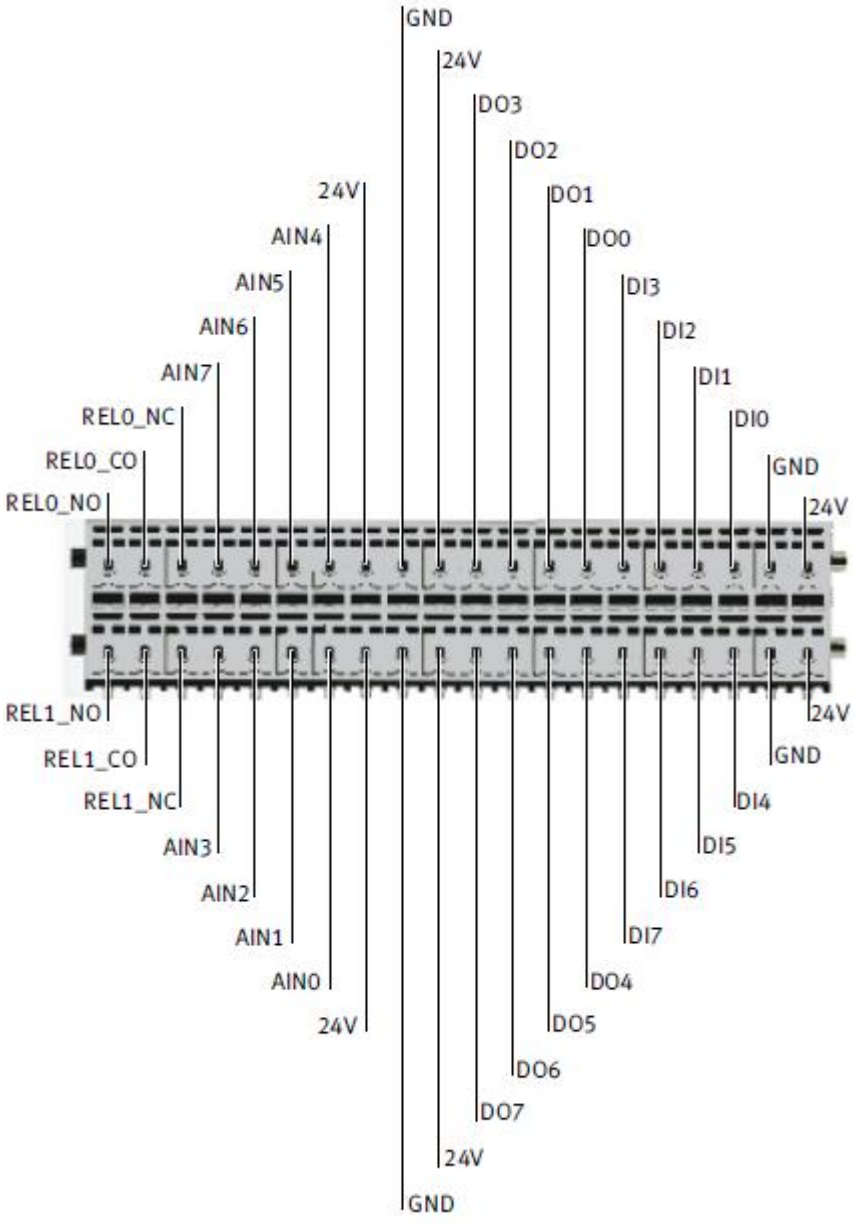


Figura 8 – Entradas e saídas

2.3. Teleoperação

A teleoperação é definida (TOURINO, 2000) como o controle contínuo e direto de um teleoperador (máquina remota). Inicialmente desenvolvida para a manipulação de materiais radioativos, a teleoperação permite que um operador exerça força e realize movimentos por meio de dados visuais, sonoros ou táteis. Com a introdução da tecnologia de teleoperação, foi possível o desenvolvimento de interfaces capazes de prover uma interação satisfatória

entre homem e máquina, permitindo que serviços de grande destreza fossem realizados. Ela pode ser classificada (ZHAI, 1991), como:

- Controle manual sem auxílio computacional;
- Controle manual com significativo auxílio ou transformação computacional;
- Controle supervisório com predomínio do controle realizado pelo operador humano;
- Controle supervisório com predomínio do controle realizado pelo computador;
- Controle completamente automático, onde os operadores humanos observam o processo sem interferir.

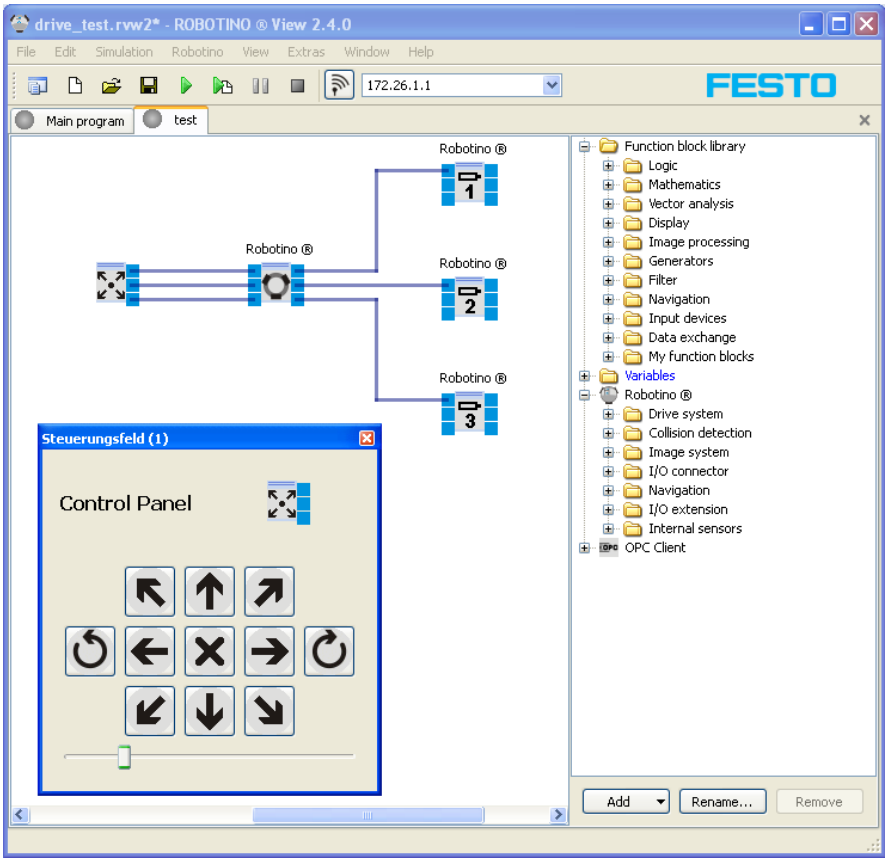


Figura 9 – Interface gráfica do Robotino View

A teleoperação proporciona para as indústrias agilidade e flexibilidade, uma vez que o operador não necessita estar presente fisicamente no local para poder executar alterações nas funções (KANEKO, 2008). Prevenir, deter,

detectar e responder são as ações que definem a importância de uma monitoração remota em um sistema que envolve uma rede de comunicação (ROCHA, 2003).

Segundo SOUZA (2002), as redes de comunicação estão presentes na sociedade, interligando desde computadores pessoais em uma casa até os dispositivos computacionais de uma empresa. Por facilitar o desenvolvimento de trabalhos e permitir o acesso a informações *on line* na internet, as redes de comunicação são cada vez mais objeto de novos serviços oferecidos pelas empresas e instituições, por isso torna-se vital mantê-las sempre disponíveis e com bom desempenho.

A monitoração remota é empregada em diversos casos, desde o apoio para as operações de máquinas ferramentas até em serviços de bancos e hospitais (CHRIST *et al.*, 2006).

2.4. WebService

O *WebService* é qualquer tipo de serviço disponível na internet que utilize do empacotamento XML (*eXtensible Markup Language*) e não possua vínculos com nenhum sistema operacional ou linguagem de programação. Está se tornando a técnica dominante para representação e distribuição de informações por meio de sistemas múltiplos, inclusive para sistemas que não foram inicialmente planejados ou desenvolvidos para interagirem (BELL *et al.*, 2006).

É desejável que ele apresente duas propriedades (CERAMI, 2002):

- Autoexplicativo: para facilitar a integração de outros sistemas com o *WebService* é interessante que ele tenha sua documentação disponibilizada, contendo, por exemplo, uma lista com seus métodos, parâmetros e valores retornados;
- Localizável: quando um *WebService* é criado, é desejável que se tenha uma forma de publicá-lo, e para isso podem ser utilizados diversos sistemas de buscas.

A estrutura de um *WebService* pode ser separada em quatro camadas (CERAMI, 2002): serviço de transporte, mensagem XML, descrição de serviço e descoberta de serviço.

A camada de serviço de transporte é responsável pelo transporte da mensagem entre as aplicações. Atualmente, esta camada inclui HTTP (*HyperText Transfer Protocol*), SMTP (*Simple Mail Transfer Protocol*), FTP (*File Transfer Protocol*) e protocolos mais novos como BEEP (*Blocks Extensible Exchange Protocol*) (CERAMI, 2002).

Na camada da mensagem, o XML foi escolhido para a realização de troca de informação dos *WebServices* pois possibilita que diferentes sistemas possam trocar informações, independente do sistema operacional ou da linguagem de programação.

Para a comunicação têm-se dois protocolos padrões (CERAMI, 2002):

- *Simple Object Access Protocol* (SOAP): Este protocolo também se utiliza do XML para a troca de informação, seu foco é o transporte de RPCs (*Remote Procedure Calls*) via HTTP, apesar de poder ser utilizado nos mais variados sistemas de mensagens;
- XML-RPC: Protocolo que utiliza mensagens XML para realizar RPCs. Os pedidos são codificados em XML e enviados via HTTP POST (método de envio de informação HTTP). A resposta vem contida no corpo da resposta HTTP. Ele é mais simples que o SOAP e mais fácil de ser adotado, entretanto ele não suporta chamadas automáticas, o que impede sua integração em aplicações "*just-in-time*".

A camada da descrição do serviço é responsável por descrever a interface pública de um *WebService* específico, ela é descrita pelo WSDL (*Web Service Description Language*).

O WSDL utiliza XML para especificar uma interface pública. Esta interface pública pode conter informações de todas as funções e dados publicáveis, vinculando informações sobre o protocolo de transporte utilizado e o endereço específico para localizar o serviço (CERAMI, 2002).

A camada de descoberta de serviço é responsável por centralizar serviços em um registro comum e facilitar a publicação/busca por

funcionalidades, atualmente isso é feito pelo UDDI (*Universal Description, Discovery, and Integration*).

2.5 MiniCIM

O MiniCIM é um pequeno sistema de manufatura integrado ao computador (“Computer Integrated Manufactoring” - CIM). Este equipamento de treinamento da Festo visa simular o processo de montagem de diferentes peças.

O MiniCIM presente no laboratório de Sistemas de Automação consiste de 4 estações de treinamento descritas a seguir:

- Estação de Alimentação ou Distribuição: trata-se do sistema responsável por armazenar as peças, separar uma peça do magazine de distribuição e disponibilizar a peça para o processo inspeção;
- Estação de Inspeção ou Testes: trata-se do sistema responsável pela execução de testes com a aquisição de informação e comparação de características específicas;
- Estação de Montagem com Unidade de Execução da Montagem: trata-se do sistema responsável por montar um conjunto de peças em um produto determinado pelo usuário;
- Sistema Inteligente de Transporte (SIT): trata-se do sistema responsável por transportar as peças entre as estações de trabalho.

Uma representação esquemática do Mini CIM pode ser visualizada na Figura 10.

Neste trabalho, o objeto de estudo é a monitoração das estações do MiniCIM por meio do Robotino, por isso é necessário um melhor entendimento a respeito do funcionamento de cada uma das quatro estações.

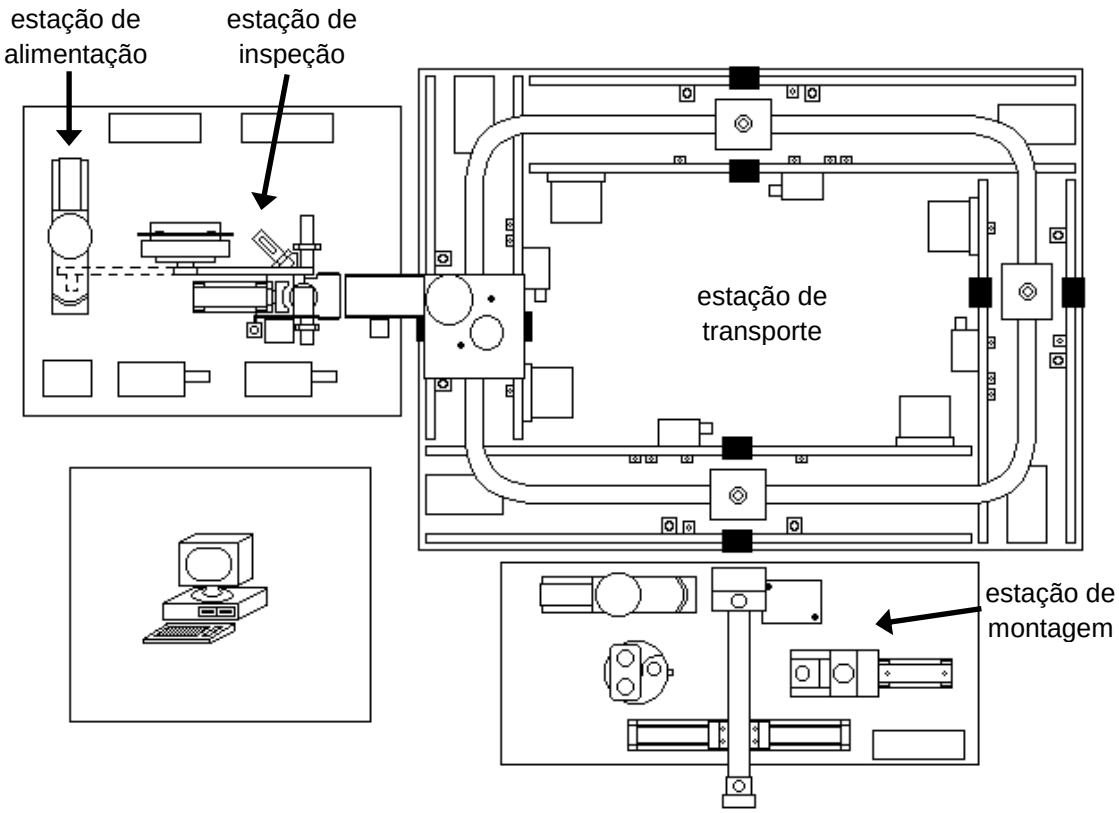


Figura 10 - Esquema Ilustrativo do MiniCIM

2.6. OPC

O OPC (OLE *for Process Control*) é composto por uma série de padrões de interface aberta para comunicação entre aplicações de Automação/Controle, Sistemas/Dispositivos de campo e aplicações de Negócios/Escritórios na indústria de controle de processo, e suas especificações são estabelecidas pela OPC *Foundation* (organização sem fins lucrativos, patrocinada pela Intellution, Microsoft e outras empresas).

O OPC é baseado nas tecnologias Microsoft de OLE (*Object Linking and Embedding*), COM (*Component Object Model*) e DCOM (*Distributed Component Object Model*) e é caracterizado por uma estrutura cliente/servidor, que funciona independente de linguagem de programação e suporta a integração e a operação em um sistema heterogêneo ou distribuído.

Quanto à transferência de dados, o servidor OPC atua como um “Portal de Software”, lendo informação dos dispositivos de campo e transformando-as

em dados OPC, com o propósito de integração dos diversos sistemas e dispositivos (LING,CHEN,YU, 2004).

Neste projeto, a comunicação entre o Robotino e a página da internet pode, portanto, ser feita utilizando um servidor OPC como um intermediador.

2.7. SFC

O SFC é uma linguagem gráfica de programação usada para CLPs. Ele pode ser usado para os processos de programas que podem ser dividido em etapas.

Adequada para controle de sistemas a eventos discretos que envolvam *steps* (passos, condições) e *transitions* (transições) interconectados por *links*. Foi desenvolvido baseado na rede de Petri, que são técnicas de representação efetiva do funcionamento dos sistemas, não importando sua complexidade (MIYAGI, 1996).

O *step* representa uma etapa de uma sequência e pode permanecer em um dos estados lógicos: ON (ativo) ou OFF (inativo). O estado da sequência é determinado em qualquer instante pelo conjunto dos *steps* ON e pelos valores das variáveis internas e de saída destes *steps*.

A *transition* indica a condição para que o estado ON dos *steps* antecedentes passe para os *steps* subsequentes. A *transition* é representada por um traço horizontal (perpendicular) ao *link*. A informação escrita ao lado da *transition* é a condição da *transition*, que é uma condição lógica para que a evolução ocorra, e é descrita conforme a Tabela 2.

As regras de evolução do SFC são apresentadas na Tabela 3.

Tabela 1 – Elementos do step

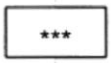
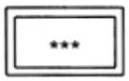
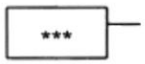
representação gráfica/textual	*observações*
 STEP ***: (conteúdo do step) END_STEP	step (representaçõ gráfica) • possui um link supeior e um inferior • "***" é o nome do step step (representação textual) • não possui link
 INITIAL _STEP ***: (conteúdo do step) END_STEP	step inicial (representaçõ gráfica) • também possui links (existem casos onde o link superior não existe) • "***" é o nome do step inicial step inicial (representação textual) • não possui link
***.X 	step flag (representação geral) • "***" é o nome do step step flag (conexão direta) • o step flag "***.X" é conectado no lado direito do step "***"
***.T	tempo transcorrido no step (representação geral)

Tabela 2 – Exemplo de transition condition

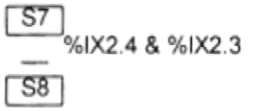
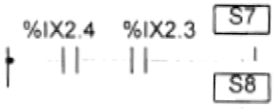
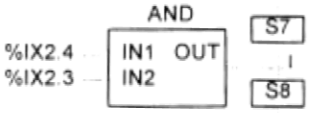
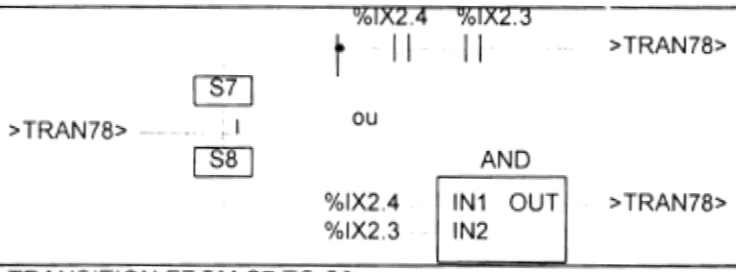
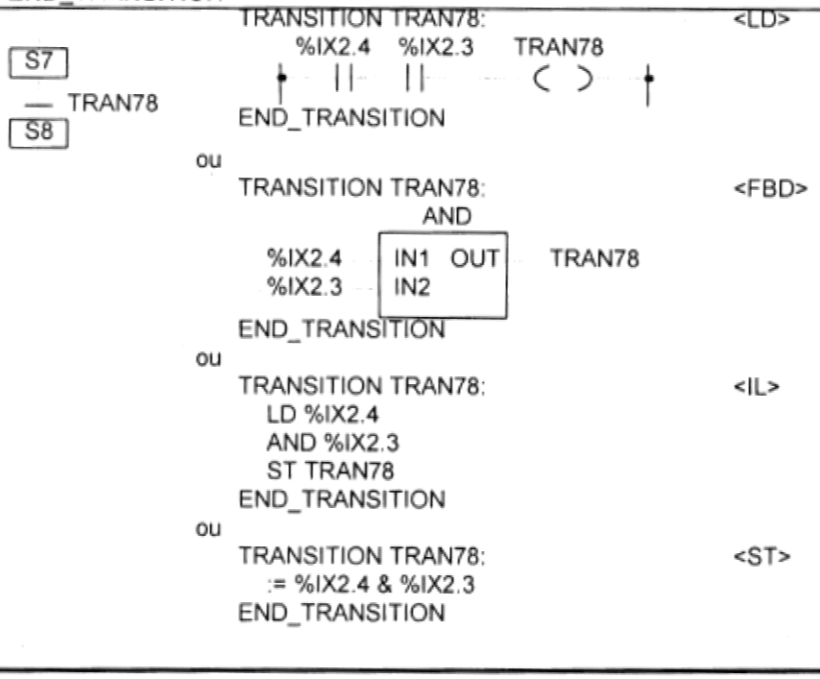
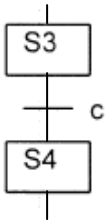
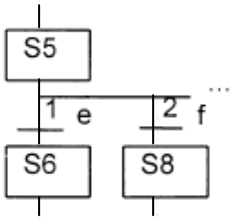
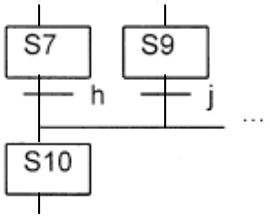
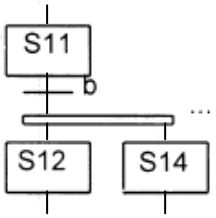
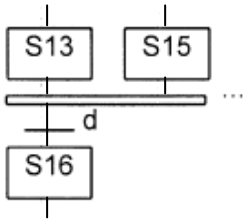
n.	*exemplo*
(a)	
(b)	
(c)	
(d)	
(e) <ST>	TRANSITION FROM S7 TO S8 := %IX2.4 & %IX2.3 END_TRANSITION
(f) <IL>	TRANSITION FROM S7 TO S8 LD %IX2.4 AND %IX2.3 END_TRANSITION
(g)	

Tabela 3 – Regras de evolução do SFC

exemplo	*regra*
	Seqüência simples • steps e transitions são conectados intercaladamente
	Início da seleção de uma seqüência • várias transições conectadas abaixo da linha horizontal representa o início da seleção • os números indicados na transição representam a ordem de prioridade quando existe conflito (quando estes números não estão indicados a prioridade é maior para as transições à esquerda)
	Fim da seleção de uma seqüência • várias transições conectadas abaixo por uma linha horizontal representa o fim da seleção
	Início de seqüências em paralelo • vários steps conectadas abaixo da linha dupla horizontal representa que estes são ativados simultaneamente • os outros steps abaixo destes são ativados independentemente
	Fim de seqüências em paralelo • vários steps conectadas abaixo por uma linha dupla horizontal representa o fim de seqüências em paralelo • todas as condições destes steps devem estar satisfeitas para ativar as transições seguintes

3. Projeto

O objetivo do projeto é o de monitorar o MiniCIM por meio do Robotino. Este percorre um caminho fixo ao redor do MiniCIM e poderá parar em 5 pontos predefinidos. Em cada ponto se posicionará de modo a transmitir imagens das estações de trabalho através da câmera para averiguar possíveis falhas e acompanhar o funcionamento da célula de manufatura.

O usuário terá a capacidade de mandar o robô para os pontos de monitoramento por meio de uma página da internet, que se comunica por meio de WebService com um servidor OPC, no qual o Robotino também está conectado. Esta página contém botões e a imagem da câmera. O usuário deverá escolher uma das 5 posições. Feita a escolha, o Robotino segue para o ponto de monitoramento desejado exibindo a imagem da câmera. Um esquema do fluxo de informações é ilustrado na Figura 11:

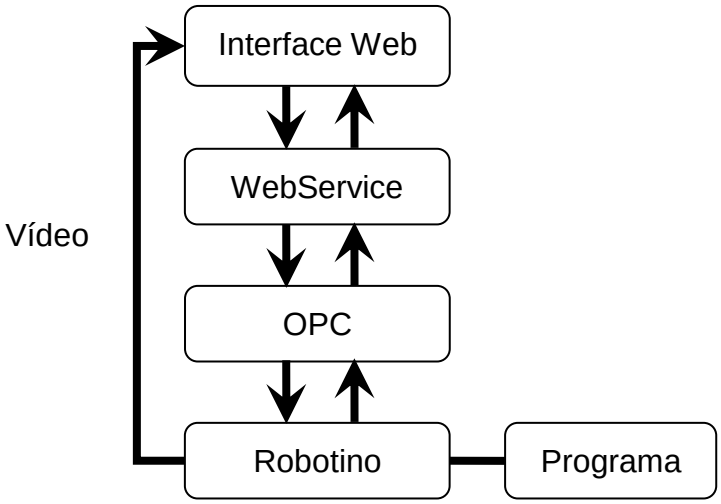


Figura 11 – Fluxo de informações

3.1. Pontos de monitoramento

Para a orientação e determinação da rota do Robotino, foi colada no chão uma fita prateada em volta do MiniCIM e uma fita isolante preta por cima desta. Para a determinação dos pontos de parada foram coladas fitas de cobre

conforme a Figura 12. Uma explicação mais detalhada do funcionamento do sistema de parada do Robotino é dada no capítulo 3.2.

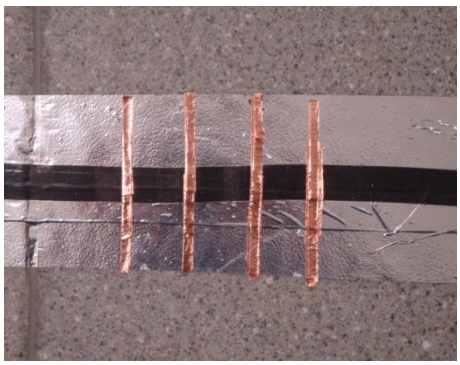


Figura 12 – Fitas coladas no chão

Para se ter uma visão geral da célula de manufatura MiniCIM, foram escolhidos 5 pontos de parada, demonstrados na Figura 13 com as respectivas orientações da câmera do Robotino.

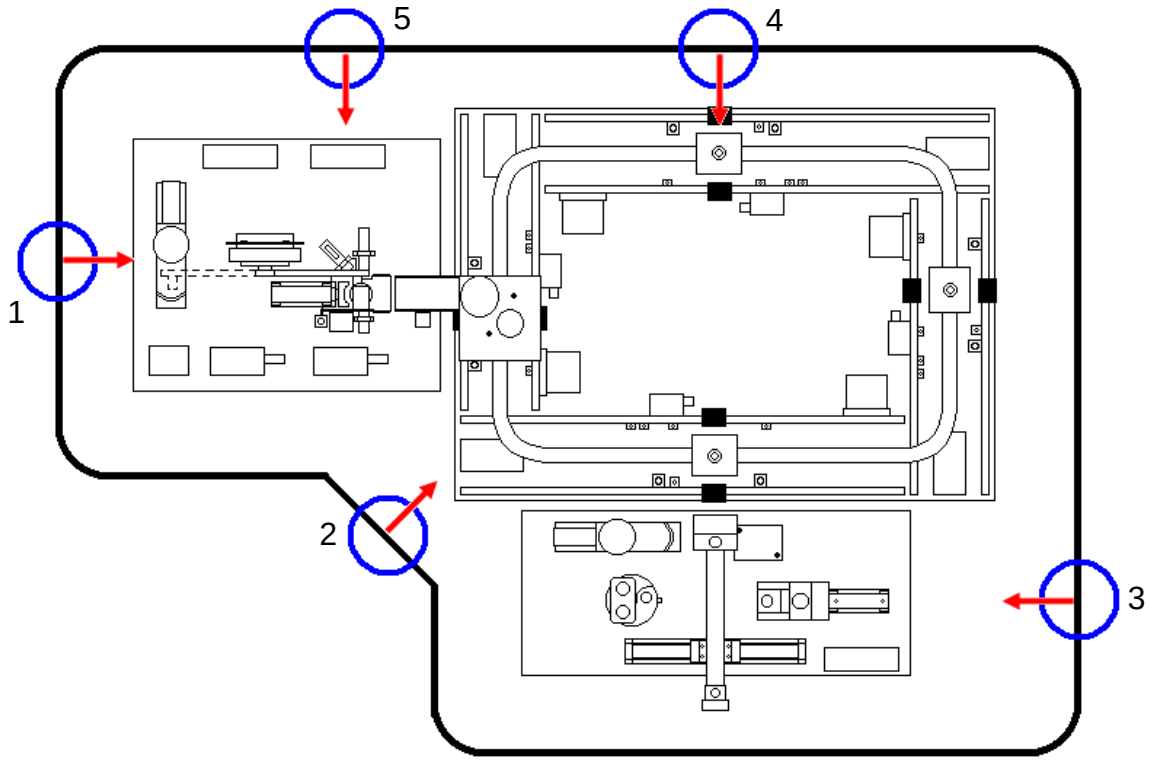


Figura 13 – Pontos de monitoramento

Na posição 1, o Robotino terá a visão da estação de alimentação, também chamada de estação de distribuição. Esta estação é composta do compartimento de armazenagem, sensores, um sistema de retirada e posicionamento e um mecanismo de transporte.

Na posição 5, a estação de inspeção ou testes pode ser monitorada. Esta estação apresenta dois andares, sendo composta por uma plataforma elevatória e diferentes tipos de sensores para a realização dos testes. No andar inferior, existem três sensores distintos: indutivo, óptico e capacitivo. O sensor indutivo identifica peças metálicas (prateadas), o sensor óptico identifica peças que refletem a luz (rosas ou prateadas) e o sensor capacitivo identifica as peças pretas. Uma visão detalhada dos componentes das estações de distribuição e de testes é ilustrada na Figura 14.

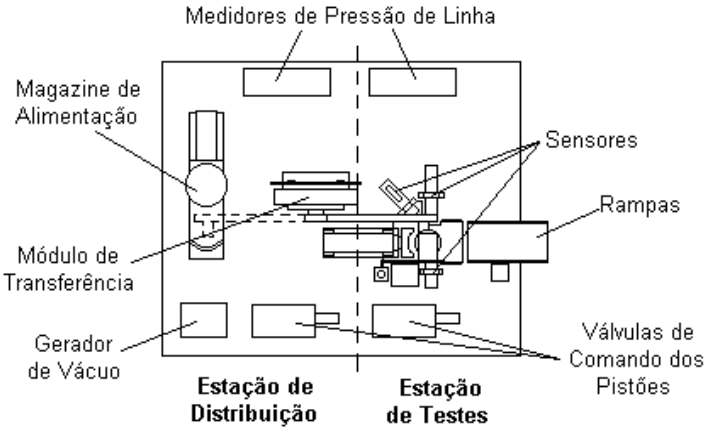


Figura 14 - Detalhamento dos componentes das estações de distribuição e de testes

O andar superior é responsável pelo teste de altura através de um sensor piezoelétrico. Na Figura 15 é possível ver os dois andares da estação de testes.

Após os testes, as peças aprovadas seguem para um carro transportador por meio de uma rampa (superior), ou são descartadas no andar inferior, por meio de outra rampa.

Nas posições 2 e 4, é possível verificar o funcionamento da esteira e o posicionamento de cada carro no Sistema Inteligente de Transporte (SIT).

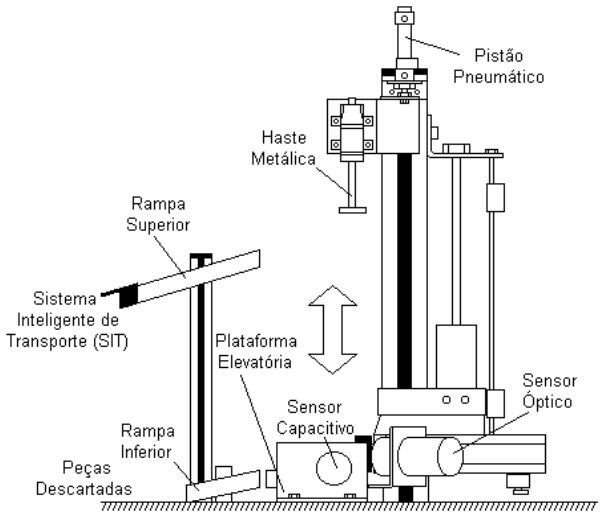


Figura 15 – Plataforma elevatória e rampas

Na posição 3, é possível monitorar a montagem da peça bruta, onde são adicionados um pino, uma mola e uma tampa. Na Figura 16 é possível ver os componentes da estação de montagem.

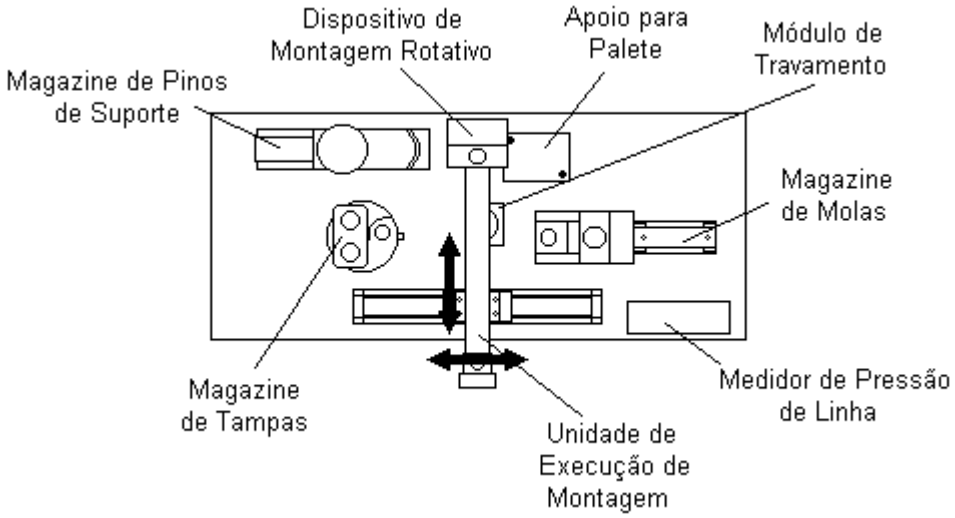


Figura 16 – Detalhamento dos componentes da estação de montagem

3.2. Preparação do Robotino

Para esse projeto foram utilizados dois sensores óticos e um sensor indutivo, instalados na parte inferior do Robotino (Figura 17). Eles foram conectados às seguintes entradas (Figura 18):

- Entrada analógica 6 (sensor indutivo);

- Entrada digital 1 (sensor ótico);
- Entrada digital 5 (sensor ótico).

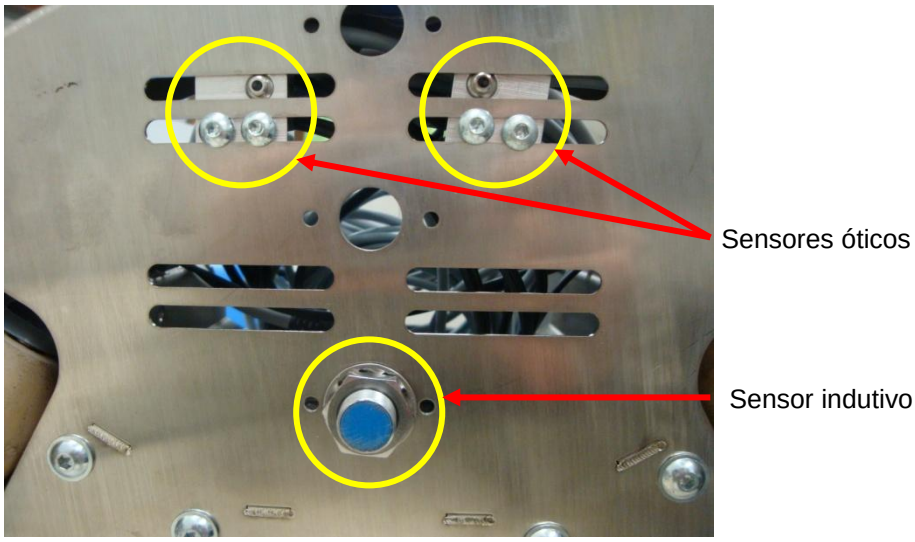


Figura 17 – Sensores

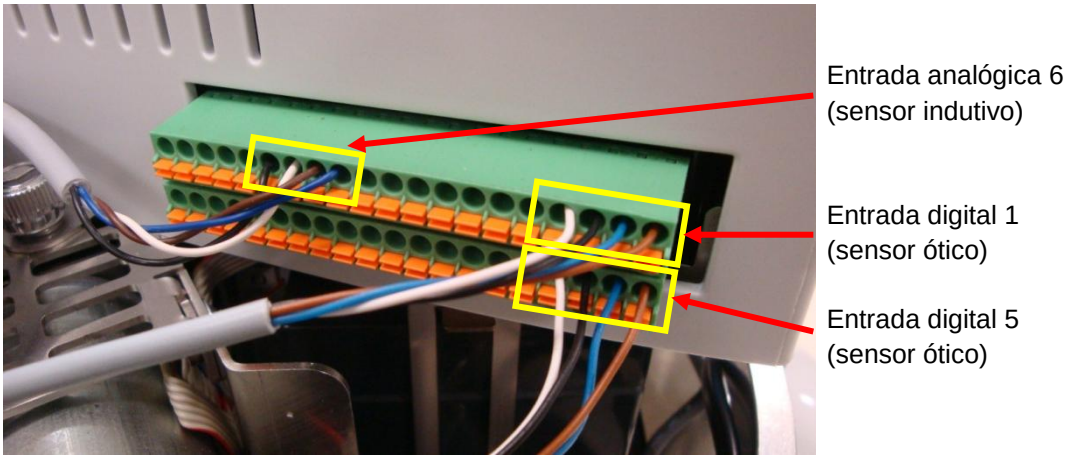


Figura 18 – Entradas dos sensores

3.2.1. Sensores óticos

Os sensores óticos no Robotino são utilizados para guiar o robô na linha feita ao redor do MiniCIM. Esses sensores geram um sinal digital, portanto foi muito importante a escolha da fita prateada e da fita isolante, pois eles apresentam um alto contraste entre o nível de reflexão da luz, minimizando assim possíveis erros na detecção da linha.

Esses sensores podem ser calibrados por meio de um parafuso localizado na parte lateral dele (Figura 19).

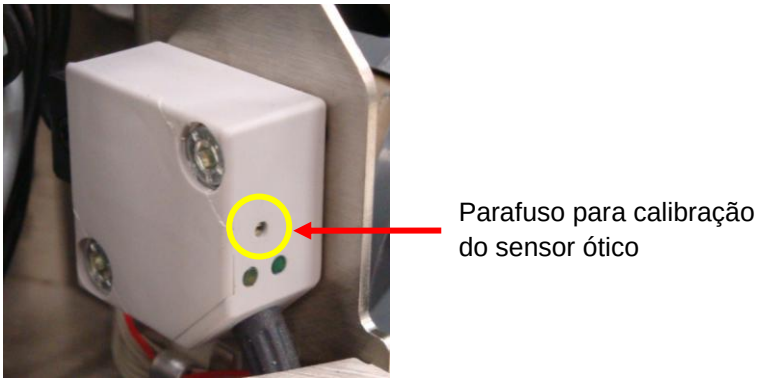


Figura 19 – Calibração do sensor ótico

3.2.2. Sensor indutivo

O sensor indutivo gera uma saída analógica que é função da distância de algum objeto metálico. Esse sensor não permite que a sensibilidade seja ajustada, assim o único parâmetro que foi possível mudar foi a distância do sensor ao chão, no qual a fita metálica foi colada.

3.2.3. Câmera

Para que o usuário tivesse uma visão boa da célula de manufatura, uma haste de madeira com a câmera na ponta teve que ser instalada no Robotino conforme a Figura 20.

Figura 20 – Haste para câmera

3.3. Programação do Robotino

O programa que controla o Robotino foi desenvolvido utilizando-se o Robotino View 2.4.0 e a estrutura criada é a apresentada na Figura 21.

Quando o programa é iniciado, os programas *Principal* e *Salva_imagem* são iniciados. Já o subprograma *Espera* é iniciado quando a condição *camera == 1* é satisfeita e, quando a condição *camera == 0* é satisfeita, o programa volta a executar o subprograma *Salva_imagem*.

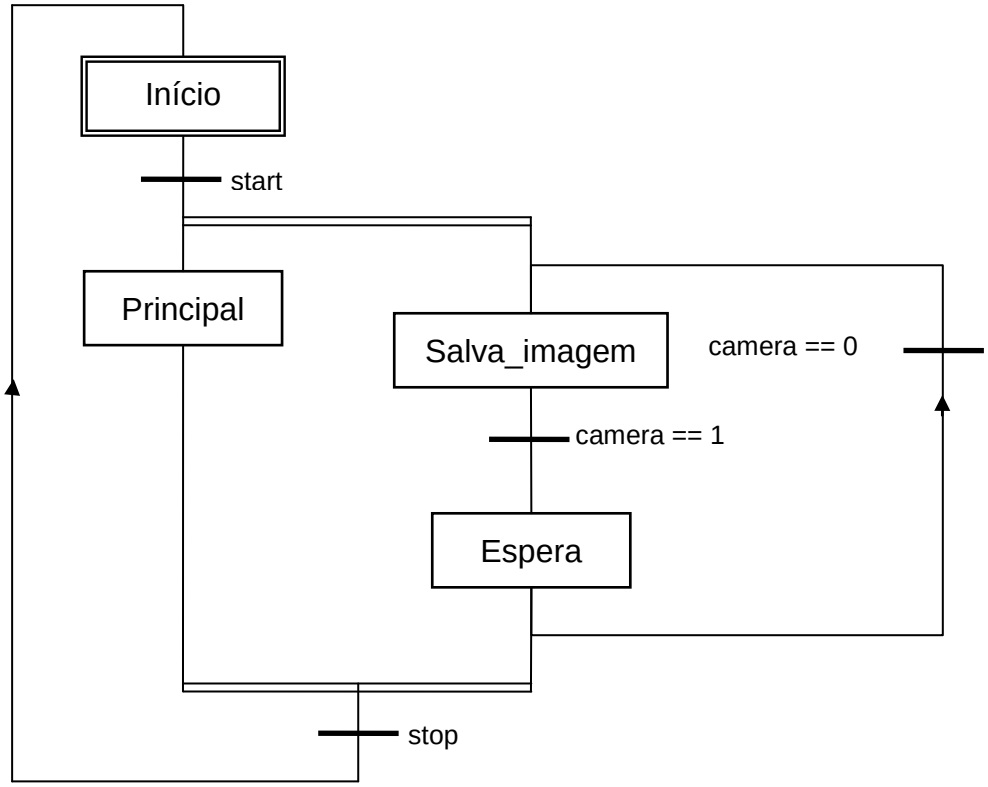


Figura 21 – Estrutura do programa

3.3.1. Programa *Principal*

Este programa (Figura 22) é responsável por fazer a comunicação com o servidor OPC, analisar os dados dos sensores e acionar os motores do robô.

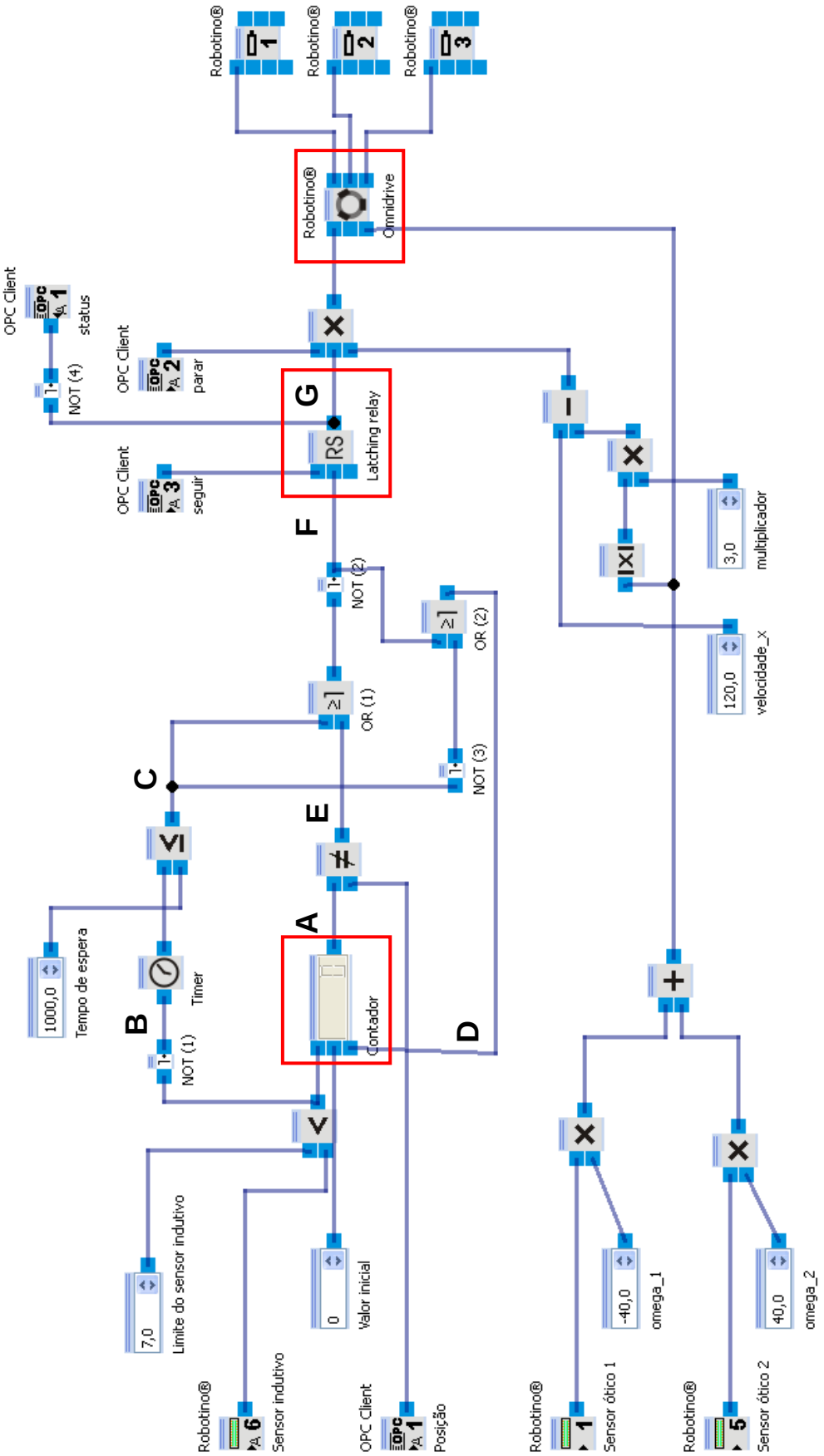


Figura 22 – Programa *Principal*

Para a explicação deste programa, foram tomados como base alguns blocos importantes:

- *Contador* (Figura 23):

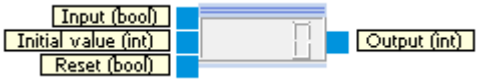


Figura 23 – Bloco *Contador*

Este bloco recebe como entrada um *valor inicial* (*Initial value*), que corresponde ao valor que o contador inicia quando ele é reiniciado. Para o projeto ele é igual a 0.

A entrada *Input*, adiciona 1 ao valor acumulado no contador quando um pulso é identificado. Essa entrada é dada pela comparação de uma constante (*Limite do sensor indutivo*) e o valor do sensor indutivo, e é usado para identificar quando o robô passa por uma das tiras de cobre coladas no chão.

Quando o robô passa por uma tira de cobre, ele também reinicia um *Timer*, usado para determinar o número de tiras da posição sendo percorrida. O *Timer* conta o tempo em milisegundos e é comparado com a constante *Tempo de espera* (1.000 ms). Este tempo foi determinado para que o robô tenha tempo de passar por todas as tiras de cobre, determinando de forma correta a posição pela qual o robô está passando.

Quando o valor do *Timer* for superior ao *Tempo de espera* e o valor do contador não corresponder ao valor da posição escolhida, um sinal é enviado à entrada *Reset* do contador, reiniciando a contagem a partir do *valor inicial*.

- *Latch RS* (Figura 24):

Figura 24 – Bloco *Latch RS*

Este bloco é usado para manter a saída *Q* em um estado de acordo com as entradas *Set (S)* e *Reset (R)*.

A variável *seguir*, conectada à entrada *S* do *Latch*, envia um pulso que faz com que a saída *Q* mude para *true* até que a entrada *R* seja *true* e a *S* esteja em *false*.

Um sinal *true* é enviado para a entrada *Reset (R)* quando o Contador apresentar um valor igual ao da variável Posição e o *Timer* um valor superior ao da constante *Tempo de espera*, ou seja, quando o robô encontrou a posição desejada. Então a saída *Q*, que é usada para acionar os motores do robô, vai para 0 (zero), fazendo com que o robô pare.

- *Omnidrive* (Figura 25)

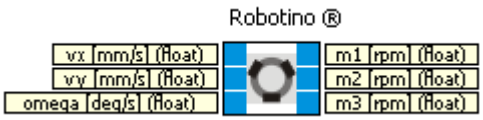


Figura 25 – Bloco *Omnidrive*

Este é um bloco disponibilizado pelo Robotino View que torna o acionamento dos motores mais fácil, transformando valores de velocidade nos eixos x e y e uma rotação *omega* em torno do robô nas velocidades correspondentes em cada um dos três motores m1, m2 e m3.

A velocidade longitudinal v_x do robô é determinada pela multiplicação de três valores: a variável *parar*, que é um booleano o qual permite o usuário parar o robô a qualquer momento; a saída *Q* do *Latch RS*, que também é um booleano; e o resultado da conta com as constantes *velocidade_x* e *multiplicador*. Quando nenhum dos sensores óticos está ativado, ou seja, quando eles estão em cima da faixa prateada, v_x recebe o valor da *velocidade_x* = 120, mas quando um dos sensores óticos detecta a linha preta, v_x recebe o valor zero, resultado da conta *velocidade_x* – 40 x *multiplicador*.

A velocidade de rotação *omega* também é dependente do valor dos sensores óticos. Quando estes detectam a linha preta, ou seja, quando o robô precisa corrigir a rota, o valor do sensor ótico, que é um booleano, é

multiplicado pelas respectivas constantes *omega_1* ou *omega_2*. Caso contrário, *omega* recebe o valor zero e o robô segue em linha reta.

A Figura 26 apresenta uma carta de tempos que ilustra a lógica usada para a determinação da posição requisitada. Para isso, foi usado um exemplo onde a variável *Posição* é igual a 3 e o robô passa pelas posições 2, 3 e 4. Os sinais das variáveis indicadas pelas letras A a G, indicadas no diagrama da Figura 22, são apresentados nesta carta de tempos.

Ao receber o comando de seguir, o robô começa a se movimentar na linha desenhada no chão. Ao passar por cada faixa de cobre, o *Timer* é reiniciado e quando o *Timer* passa de 1.000 ms, podemos determinar por qual posição o robô acabou de passar, já que o tempo para o robô passar entre duas faixas de cobre consecutivas é menor que 1.000 ms.

Esta carta de tempos foi feita com o intuito de mostrar o robô passando por uma posição anterior à posição desejada, pela própria posição e pela posição posterior, demonstrando assim a lógica utilizada no programa criado.

3.3.2. Subprograma *Salva_imagem*

Este subprograma (Figura 27) é responsável por salvar a imagem da câmera com o bloco *Image Writer*, no qual é especificado um local para o armazenamento das imagens.

Como esse bloco salva as imagens com um sufixo que é incrementado toda vez que uma nova imagem é gerada, foi necessário fazer com que o programa alternasse entre 2 subprogramas (*Salva_imagem* e *Espera*). Para isso foi usado uma variável global (*camera*), que faz com que o subprograma *Espera* seja ativado e o *Salva_imagem* desativado. O valor da variável *camera* é alterado para 1 após 5 ms nesse subprograma.

3.3.3. Subprograma *Espera*

Este subprograma (Figura 28) foi usado para alternar com o subprograma *Salva_imagem* para que a imagem fosse armazenada sempre com o mesmo nome.

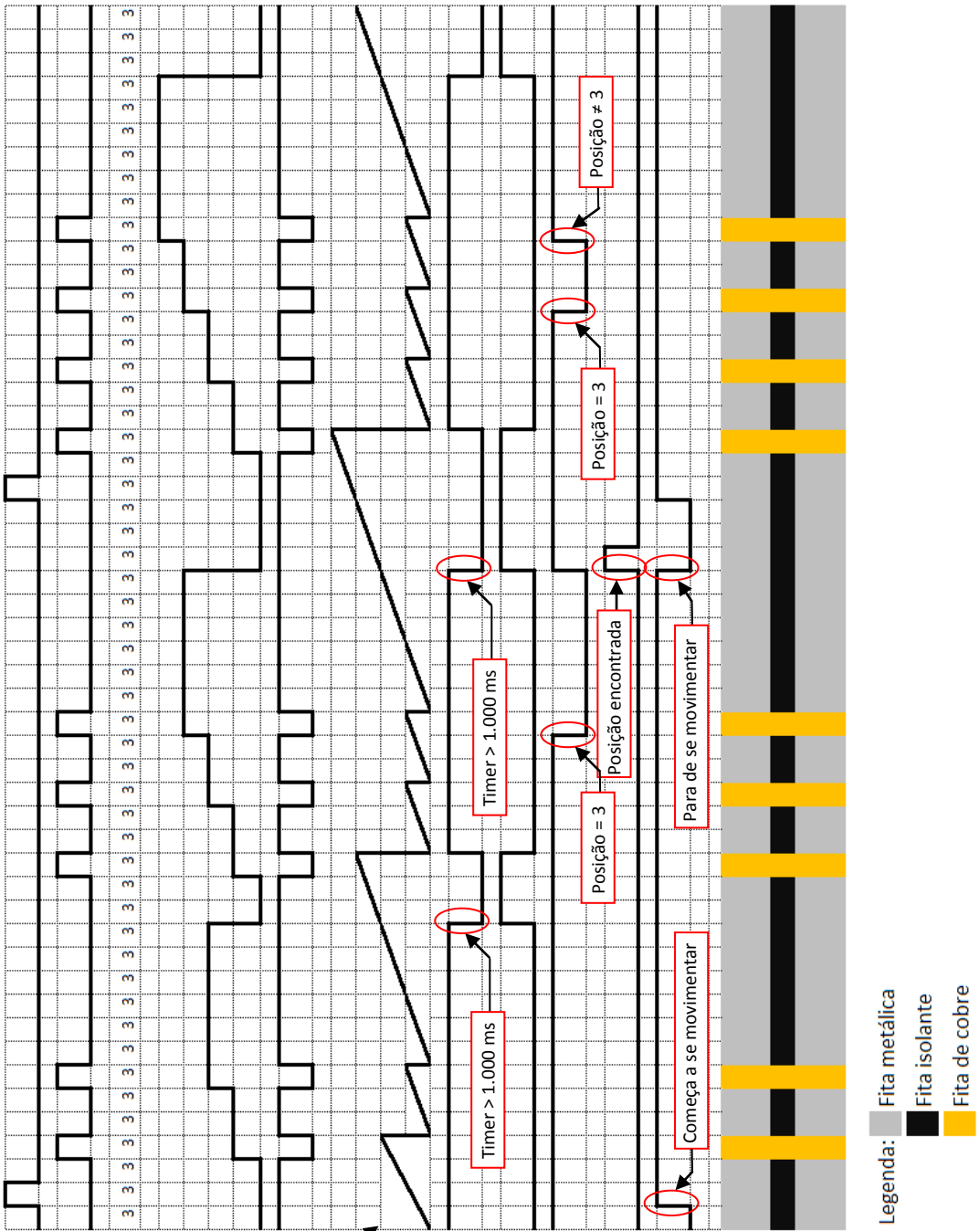


Figura 26 – Carta de tempos

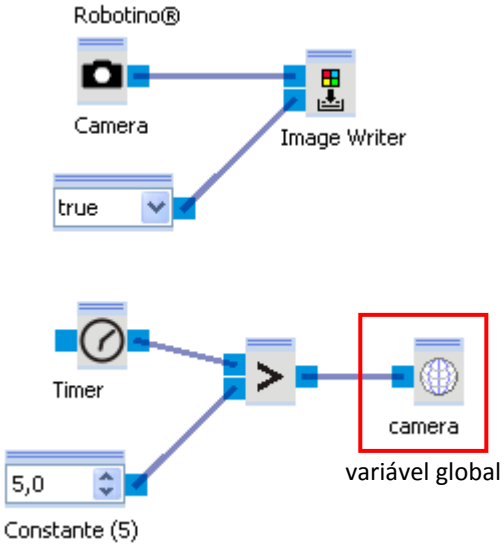


Figura 27 – Subprograma *Salva_imagem*

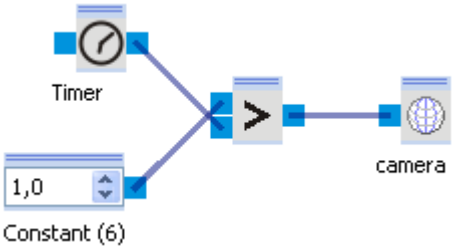


Figura 28 – Subprograma *Espera*

3.4. Servidor OPC

Para fazer a comunicação entre o Robotino View e a página da internet, foi utilizado um servidor OPC, criado com o auxílio do programa OPC Scout. Neste servidor foram criadas 4 variáveis (Figura 29):

- Posição: determina a posição para a qual o Robotino deve seguir;
- Seguir: determina quando o Robotino deve começar a se locomover em direção à posição escolhida;
- Parar: variável usada para requisitar a parada do robô durante o seu movimento;
- Status: variável utilizada para indicar ao usuário quando o robô chegar na posição desejada.

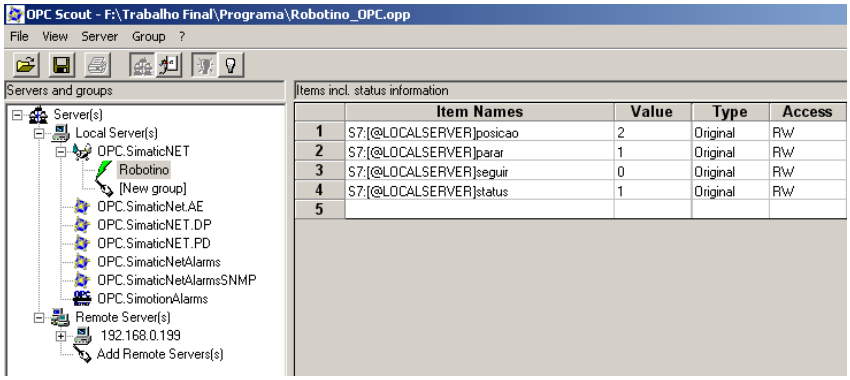


Figura 29 – Servidor OPC

Para que as informações fossem trocadas, deve-se primeiro conectar o Robotino View ao servidor OPC com o seguinte procedimento (Figura 30):

- 1. Duplo clique em OPC Client na barra de funções à direita;
- 2. Inserir as variáveis nas entradas e saídas analógicas e digitais correspondentes;
- 3. Escolher o servidor OPC;
- 4. Clicar em *Connect*.

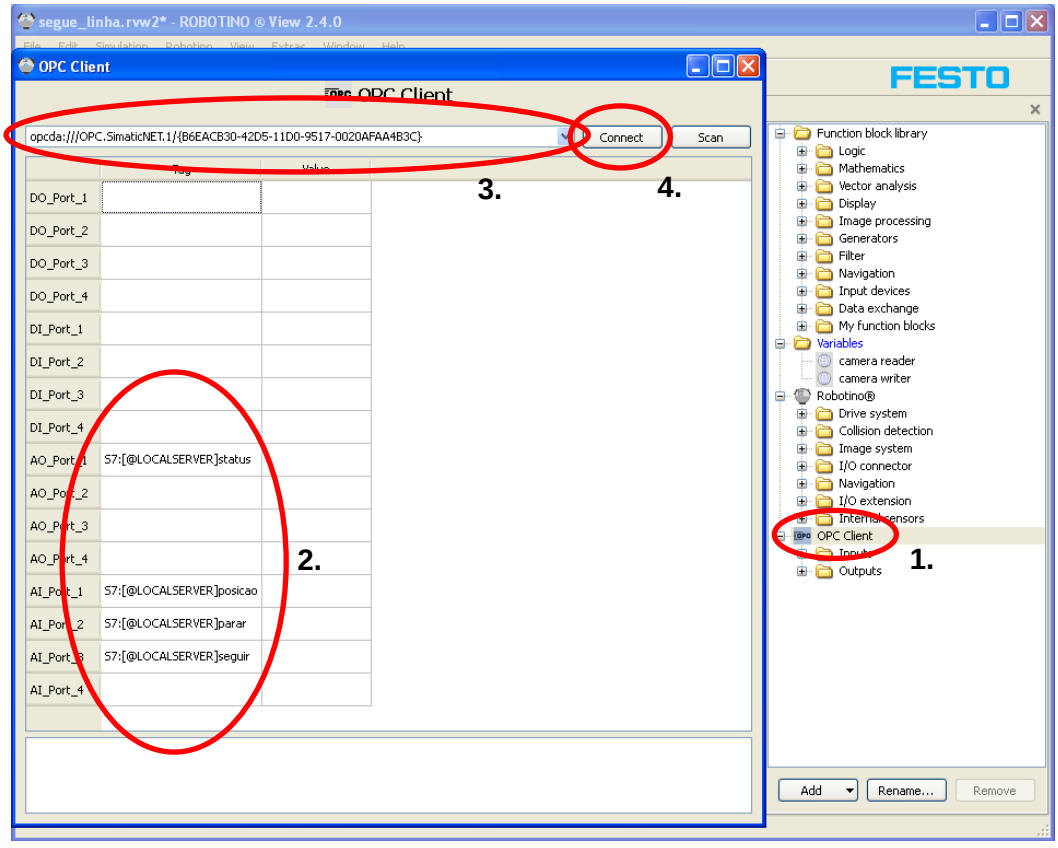


Figura 30 – Conexão Robotino View - Servidor OPC

3.5. Página de comando e monitoramento via internet

A interface de controle e monitoramento do Robotino é realizada via uma página na internet.

Esta página é composta por 8 botões, um ambiente para mostrar as imagens capturadas pela câmera e um botão que corresponde à variável *Status* do servidor OPC, indicada pelo botão “Chegou no destino”.

A Figura 31 mostra uma visão geral da página:

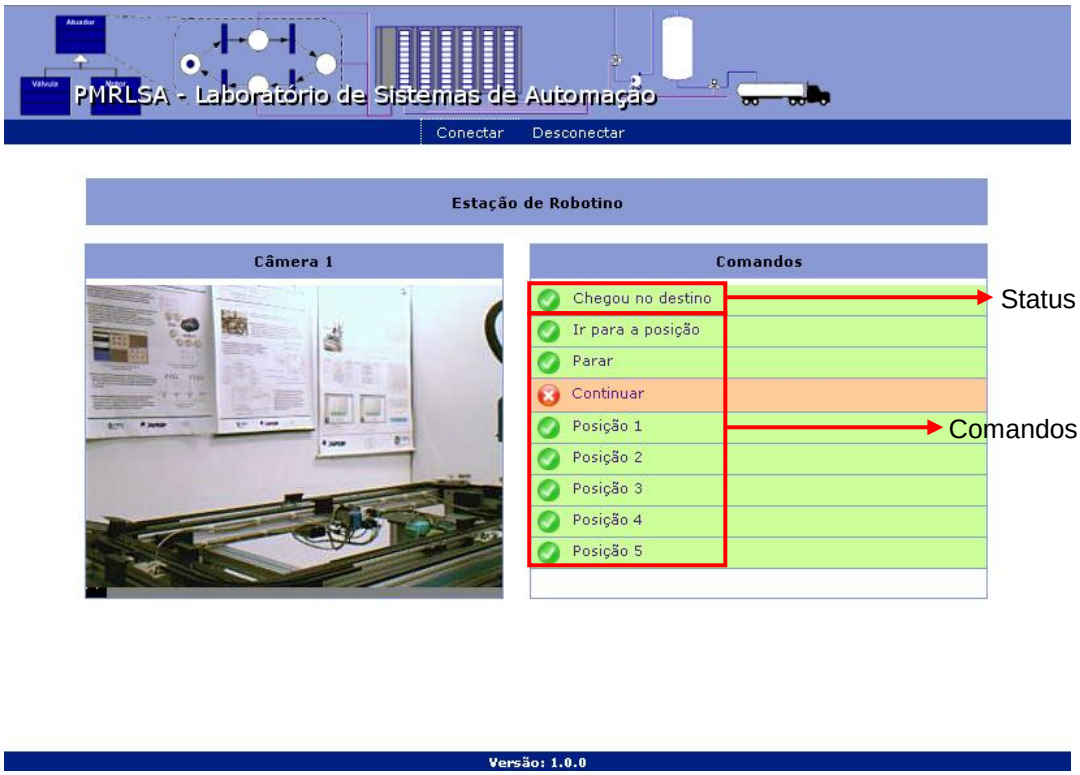


Figura 31 – Página de comando e monitoramento via internet

A região da esquerda denominada “Câmera 1” fica em constante atualização. O programa do Robotino escrito no Robotino View salva sucessivas imagens capturadas pela câmera numa determinada pasta do disco rígido. Esta região é responsável por ler essas imagens e mostrar na página. Devido à frequência de leitura e visualização das imagens serem relativamente altas, é possível ter a impressão de que estamos visualizando um vídeo em tempo real. Através dessas imagens é possível monitorar a célula de manufatura nos pontos predefinidos, sendo possível detectar possíveis falhas

de funcionamento. Esta câmera fica em funcionamento sempre que o Robotino estiver conectado ao Robotino View, sendo então possível monitorar não somente as regiões predefinidas, mas também a todo o trajeto do robô ao redor do MiniCIM.

A região da direita denominada “Comandos” possui oito variáveis de comando (botões) e uma variável de monitoramento (*Chegou no destino*), que é representado pela variável *status* no servidor OPC. Os 5 botões de posição são indicados pela variável inteira *posição* e os botões *Ir para a posição* e *Continuar* pela variável *seguir*.

A seguir será descrito a função de cada botão:

- *Ir para a posição*: Este botão é responsável por iniciar o movimento do robô, enviando um pulso para a variável *seguir* do servidor OPC;
- *Parar*: este botão é responsável por parar o movimento do robô, colocando o valor 0 (zero) na variável *parar* do servidor OPC. Ele pode ser usado por motivo de segurança;
- *Continuar*: este botão é responsável por dar continuidade à trajetória do robô após a pausa colocando o valor 1 (um) na variável *parar* do servidor OPC;
- *Posição 1* a *Posição 5*: estes botões são responsáveis por configurar o programa Robotino View para o destino final de acordo com a posição desejada alterando o valor da variável *posição* do servidor OPC para o valor da posição correspondente;

Por meio desses botões é possível determinar qual posição do MiniCIM deseja-se monitorar e enviar o Robotino para que realize a filmagem.

A variável de *Chegou no destino* é responsável por indicar se o Robotino está em movimentação ou se já atingiu a região desejada. Caso ela seja *false*, indica que o robô está em movimentação ao redor do MiniCIM. Caso seja *true*, indica que o robô atingiu a posição pedida e encontra-se parado monitorando o funcionamento da célula de manufatura.

3.6. WebService

O WebService é o responsável por realizar a integração entre a região “Comandos” da página da internet e o servidor OPC. Ele realiza a leitura da variável *Status* e altera o valor das variáveis de comando no servidor OPC de acordo com os comandos por meio da página da internet.

Tem-se então um fluxo bidirecional, onde o WebService é responável por escrita e leitura de variáveis no servidor OPC.

Por exemplo, ao clicar no botão posição 2, a variável *posição* no servidor é alterada para 2. Quando o Robotino atingir esta posicão, a variável de leitura *status* é alterada para *true* pelo Robotino View e será lida pelo WebService. Na Figura 29 é possível verificar como ficam as variáveis para o robô parado na posição 2:

4. Conclusão e considerações

O Robotino mostrou-se bastante capaz de realizar a tarefa de monitoração de uma linha de montagem devido a sua facilidade de programação e capacidade de expansão usando sensores.

Algumas sugestões de melhorias são listadas a seguir:

- A criação de um bloco no Robotino View que salve as imagens com um mesmo nome; não precisando então da criação de dois subprogramas como foi realizado no projeto;
- A adição de mais uma câmera de modo a se obter uma visão mais ampla da célula de manufatura, já que o Robotino possui duas entradas USB;
- Permitir que a rotação do Robotino na posição predefinida de modo a obter mais versatilidade no controle;
- Obtenção do vídeo em *streaming* de modo a obter imagens contínuas da câmera;
- Adicionar um objeto que possa ser identificado pelas ferramentas de tratamento de imagem disponíveis no Robotino View para um ajuste no posicionamento do robô nas posições predeterminadas;
- Caso deseja-se um monitoramento 24 horas, será necessários dois robôs, devido ao longo tempo de carregamanto de sua bateria;
- Melhoria na parte da interface gráfica da página da internet, fazendo com que o usuário, ao invés de clicar no botão posição x, pudesse clicar na posição indicada em uma figura com a própria célula de manufatura;
- Utilização de um sistema de posicionamento para mostrar a posição atual do robô na página da internet.

Referências Bibliográficas

ÁLVARES, A.J., TOURINO; S.R. Desenvolvimento de um Robô Móvel Autônomo Teleoperado Via Internet, Congresso Nacional de Engenharia Mecânica 2000, Natal, RN, 7−12 de agosto de 2000.

AUGUSTO, S.R. **Uma Plataforma Móvel para Estudos de Autonomia**. Tese de Doutorado, Escola Politécnica, São Paulo, 2007.

BELL, D., CESARE, S., IACOVELLI, N., LYCETT, M., MERICO, A., *A framework for derviving semantic web services*. In: **Springer Science and Business Media online**, LLC 2006.

CERAMI, E. ***Web Services Essentials – Distributed applications with XMLRPC, SOAP, UDDI & WSDL***. O'REILLY, 2002.

CHRIST, R.E.R., FIGUEREDO, M.V.M., BASSANI, T., DIAS, J.S., NIEVOLA, J.C. **Sistema de monitoração remota de pacientes em tempo-real através da Internet do hospital**. Disponível em: <http://www.sbis.org.br/cbis9/arquivos/603.pdf>. Acesso em Abril de 2010.

GROOVER, M.P., WEISS, M., NAGEL, R.N **Robótica: tecnologia e programação**. São Paulo: McGraw-Hill, 1988, 401p.

KANEKO, A.M. **Desenvolvimento de uma interface gráfica para supervisão de um sistema produtivo teleoperado**. Trabalho de Formatura. Escola Politécnica da Universidade de São Paulo. São Paulo. 2008.

LING, Z., CHEN, W., YU, J., Research and Implementation of OPC Server Based on Data Access Specification, **Apresentado no 5º Congresso Mundial de Controle e Automação Inteligente**, Hangzhou, P.R. China, Junho de 2004.

MCCOMB, G. ***The robot builder's bonanza: 99 inexpensive robotics projects***. New York: TAB Books, 1987. 326p.

MCKERROW, P.J. ***Introduction to robotics***. Sydney: Addison-Wesley, 1991. 811p.

MIYAGI, P. E. **Controle Programável**. São Paulo: Editora Edgard Blücher, 1996.

NITZAN, D. *Development of intelligent robots: achievements and issues*. **IEEE Journal of Robotics and Automation.** Vol. RA-1, N. 1, pp. 3-13, mar, 1985.

PAZOS, F. **Automação de sistemas & robótica.** Rio de Janeiro: Axcel Books do Brasil Editora, 2002.

ROCHA, L.F. **CSO e a importância da monitoração remota**. 11 de agosto de 2003. Disponível em: <http://www.cert.br/docs/reportagens/2003/2003-08-11.html>. Acesso em Abril de 2010.

SHIROMA, P. M. **Controle por visão de veículos robóticos.** Dissertação de Mestrado. UNICAMP. 2004.

SOUZA, E. M. **Desenvolvimento de componentes para facilitar a monitoração remota de redes de computadores usando a ferramenta WebManager**, 2002.

SPENCE, R., HUTCHINSON, S. *An integrated architecture for robot motion planning and control in the presence of obstacles with unknown trajectories*. **IEEE trans. on systems, man, and cybernetics**, Vol. 25, N. 1, pp. 100-110, jan., 1995.

ZHAI, S., MILGRAM, P. *A Telerobotic Virtual Control System***. Proceedings of SPIE**. Boston, 1991