

**UNIVERSIDADE FEDERAL DE VIÇOSA  
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS  
DEPARTAMENTO DE ENGENHARIA ELÉTRICA**

**RENAN FERNANDES BASTOS**

**CONSTRUÇÃO DE UMA PLATAFORMA EXPERIMENTAL  
ACOPLADA A UM SIMULADOR DE MANIPULADORES  
ROBÓTICOS**

**VIÇOSA  
2010**

RENAN FERNANDES BASTOS

CONSTRUÇÃO DE UMA PLATAFORMA EXPERIMENTAL  
ACOPLADA A UM SIMULADOR DE MANIPULADORES  
ROBÓTICOS

Monografia apresentada ao Departamento de Engenharia Elétrica do Centro de Ciências Exatas e Tecnológicas da Universidade Federal de Viçosa, para a obtenção dos créditos da disciplina ELT 490 – Monografia e Seminário e cumprimento do requisito parcial para obtenção do grau de Bacharel em Engenharia Elétrica.  
Orientador: Prof. Dr. Tarcisio de Assunção Pizziolo

VIÇOSA  
2010



RENAN FERNANDES BASTOS

CONSTRUÇÃO DE UMA PLATAFORMA EXPERIMENTAL  
ACOPLADA A UM SIMULADOR DE MANIPULADORES  
ROBÓTICOS

Monografia apresentada ao Departamento de Engenharia Elétrica do Centro de Ciências Exatas e Tecnológicas da Universidade Federal de Viçosa, para a obtenção dos créditos da disciplina ELT 490 – Monografia e Seminário e cumprimento do requisito parcial para obtenção do grau de Bacharel em Engenharia Elétrica.

Aprovada em 03 de dezembro de 2010

COMISSÃO EXAMINADORA

---

**Prof. Dr. Tarcisio de Assunção Pizzolo - Orientador**  
**Universidade Federal de Viçosa**

---

**Prof. Dr. Alexandre Santos Brandão - Membro**  
**Universidade Federal de Viçosa**

---

**Prof. Dr. Denílson Eduardo Rodrigues - Membro**  
**Universidade Federal de Viçosa**

## ***Resumo***

A utilização de simuladores para controle de trajetória de robôs manipuladores permite-nos o estudo e análise das condições de trabalho às quais um robô manipulador pode-se sujeitar ao executar uma dada tarefa. O espaço de trabalho, bem como a otimização da trajetória, podem ser analisados através de modelos matemáticos, os quais o simulador permite-nos uma visualização geométrica tridimensional da execução da tarefa. Este projeto propõe o estudo e desenvolvimento de uma plataforma simuladora para um manipulador robótico com o objetivo de controle do caminho percorrido pelo manipulador. Este sistema envolverá variáveis tais como as coordenadas de juntas, as dimensões dos elos e as posições inicial e final do efetuador do manipulador, onde o operador poderá fazer a programação do caminho percorrido pelo braço ou comandá-lo em tempo real .

Todo o projeto foi executado utilizando microcontroladores de última geração e componentes robustos que resistem a ambientes industriais. Utilizou-se a interface USB para comunicação com o computador; que torna o projeto versátil. Além de ser programada em linguagem C++, que é uma linguagem de alto nível muito fácil de ser trabalhada, todo o projeto foi simulado em computador antes de ser implementado e o circuito impresso foi confeccionado utilizando técnicas convencionais, com um ótimo resultado. Ao final, com o circuito confeccionado, foram feitos vários testes comprovando a eficiência do sistema.

## *Abstract*

The use of simulators for trajectory control of robot manipulators allows us to study and analysis the working conditions of a robot manipulator which can be bound to perform a given task. The workspace and the trajectory optimization can be analyzed using mathematical models, which allows us to view three-dimensional geometric task execution. This project proposes to study and develop a platform for simulating a robotic manipulator in order to control the path. This system involves variables, such as joints coordinates, links dimensions and the start and end positions of the manipulator, where the operator can do the programming of path taken by the arm or control it in real time.

The entire project is executed using the last generation of microcontrollers and robust components that resist industrial environments. We used the USB interface for communication with the computer that makes the versatile design. Besides being programmed in C language, which is a high level language very easy to work, the whole project was simulated in the computer before being implements and the printed circuit was built using conventional techniques, with an excellent result. And finally showing the efficiency of the system.

## Sumário

1. Introdução.....	8
1.1 Cinemática direta e cinemática inversa do manipulador.....	9
1.1.1 Cinemática direta .....	10
1.2.2 Cinemática inversa .....	10
1.2 Trajetória e caminho .....	11
1.3 Pixel e conectividade.....	12
1.4 Objetivos .....	12
2. Materiais e Métodos .....	13
2.1 Microcontrolador pic 18F4550.....	13
2.2 Motor de Passo .....	15
2.3 Comunicação USB .....	16
2.3.1 Topologia USB .....	17
2.3.2 Interface Elétrica .....	17
2.4 Montagem do circuito de acionamento.....	18
2.5 Simulador de circuitos .....	19
2.6 Desenvolvimento do circuito impresso .....	20
2.7 Construção final da placa .....	22
2.8 LCD .....	22
2.9 Software manipulador e interface USB – Serial .....	23
2.10 Programação do Microcontrolador .....	26
2.10.1 O microcontrolador 18f e a USB .....	26
2.10.2 Tratamento de erros .....	28
3. Interface com o MatLab .....	29
3.1 Mapeamento do caminho.....	32
3.2 Cinemática inversa aplicada ao caminho mapeado .....	34
3.3 O braço mecânico .....	35
4. Resultados Alcançados.....	37
4.1 Resultados com o software Manipulador .....	37
4.2 Resultados Alcançados com o MatLab .....	40
5. Conclusões e propostas futuras.....	42
6.Referências Bibliográficas .....	43
7.Anexos .....	44

## 1. INTRODUÇÃO

Um dos maiores desafios ao engenheiro de controle e automação é a modelagem e o controle de sistemas modernos, complexos e interligados, como sistemas de controle de tráfego, processos químicos e sistemas robóticos [2].

Um robô manipulador industrial é uma máquina que possui uma geometria similar a um braço humano e é caracterizado por ser um manipulador programável, multifuncional, projetado para mover materiais, peças, ou dispositivos especiais em movimentos variáveis com uma sequência lógica definida, para realização de uma infinidade de tarefas. Normalmente, o robô é programado para repetir os padrões de movimento até que seja reprogramado para realizar outra tarefa [1].

Robôs manipuladores são empregados na indústria aeroespacial, metalúrgica, produção de equipamentos eletrônicos de precisão, entre outros. A maior parte desses braços mecânicos são usados nas linhas de montagem industrial automobilística. No Brasil, segundo a Federação Internacional de Robótica em 2008 já se contabilizavam 7 mil deles, o que corresponde a 1% do total de robôs no mundo, quantidade muito baixa se comparado à de países industrializados. Entretanto, nenhum destes robôs utilizados na indústria brasileira é de fabricação nacional, eles vem principalmente do Japão, daí a necessidade deste tipo de estudo no Brasil, para o desenvolvimento de tecnologias na área da robótica e de automação, além de treinar mão de obra para operar o sistema. Outro ponto a se frisar é que os grupos de pesquisa que se dedicam a este fim criam uma abordagem muito teórica, baseada somente em modelos matemáticos e computacionais, o que torna este tipo de estudo tedioso, assim a disposição de uma plataforma real que simule o comportamento programado seria um grande estímulo. Neste sentido, este trabalho propõe o desenvolvimento de um sistema capaz de simular no mundo real os manipuladores robóticos.



## 1.1 Cinemática Direta e Cinemática Inversa do manipulador

Na maioria das aplicações industriais, a programação de tarefas de robôs, é realizada por aprendizagem. Assim sendo, a programação de trajetórias de um robô torna-se muito fácil, sendo a fase de aprendizagem basicamente uma operação de armazenamento de uma sequência de incrementos necessários para que o conjunto execute um movimento específico a partir de um perfil de trajetórias fornecido (robô controlado a partir do sistema de coordenadas de juntas). Desta maneira, sua trajetória é definida através de um conjunto de ângulos associados ao movimento angular de cada grau de liberdade do robô. Após uma interpolação, estes valores servirão de referência para o controlador de posição de cada junta, que realizará uma comparação com os sinais provenientes dos transdutores de posição de cada junta.

A cinemática trata do estudo dos movimentos dos robôs sem considerar as causas que lhes dão origem. Para tratar dos movimentos dos manipuladores é necessário desenvolver técnicas para representar a posição de determinado ponto do braço no tempo. Esta representação depende da posição das juntas e dos elos, sendo que é necessário ter a base do robô como ponto de referência. A Figura 1 mostra a forma geométrica de um braço articulado simples do tipo *RR* (duas juntas rotacionais) com dois graus de liberdade movendo-se em um plano.

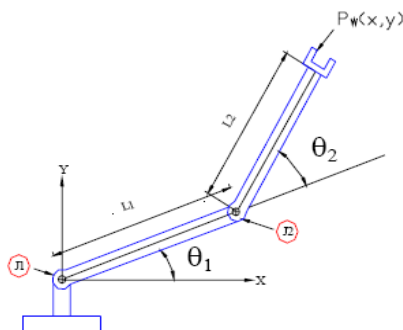


Figura 1 – Braço robótico com dois graus de liberdade.

As juntas são rotuladas com  $J_n$ , onde  $n$  começa com 1 na base do braço robótico, e os elos são rotulados por  $L_n$ , novamente sendo 1 o elo mais próximo da base. Sendo assim, é possível aplicar métodos matemáticos para calcular o posicionamento do braço robótico numa determinada posição, desde que se conheçam os ângulos das juntas num dado instante. Essa posição pode ser obtida por meio de sensores localizados nas juntas, os quais podem medir os ângulos  $\theta_1$  e  $\theta_2$ , e assim obter as posições  $x$  e  $y$  (cinemática direta) expressas em função desses ângulos. Isso é conhecido como a representação no espaço de junta. Contudo, muitas vezes, é necessário aplicar o inverso, isto é, calcular  $\theta_1$  e  $\theta_2$  em função de  $x$  e  $y$  (cinemática inversa), representado no espaço cartesiano.

### 1.1.1 Cinemática direta

A cinemática direta constitui da necessidade de ir do espaço de junta para o espaço cartesiano. O problema da cinemática direta consiste então em determinar, a partir dos ângulos das juntas, a posição cartesiana da extremidade do braço. Definindo um vetor para o elo 1 e outro para o elo 2, um braço  $RR$  tem como equações da cinemática direta:

$$x = l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2) \quad (1)$$

$$y = l_1 \sin(\theta_1) + l_2 \sin(\theta_1 + \theta_2) \quad (2)$$

### 1.1.2 Cinemática Inversa

Usando as equações da cinemática direta, é possível determinar as coordenadas  $x$  e  $y$  de um manipulador de dois graus de liberdade do tipo  $RR$ , uma vez conhecidos os ângulos das juntas  $\theta_1$  e  $\theta_2$ . Entretanto, em alguns casos é necessário calcular os ângulos das juntas a partir das coordenadas cartesianas. No caso do robô  $RR$ , fornecida a posição do órgão terminal no espaço cartesiano  $(x, y)$ , então é possível inverter as equações da cinemática direta e calcular os ângulos  $\theta_1$  e  $\theta_2$  pelos pontos conhecidos, de modo a posicionar o órgão terminal neste ponto. Este problema é mais complicado que a cinemática direta, pois a cinemática inversa apresenta duas soluções possíveis para este tipo de braço, conforme mostra a Figura 2, com o cotovelo acima ( $\theta_2 < 0$ ) e cotovelo abaixo ( $\theta_2 > 0$ ).

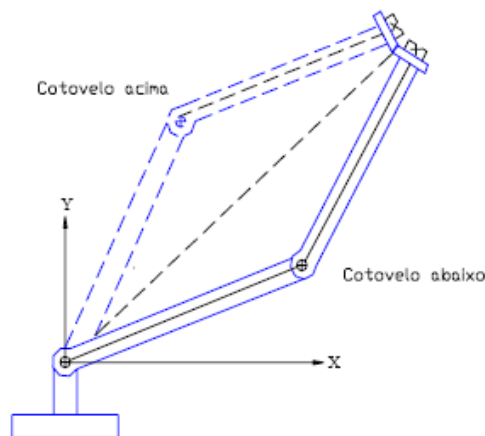


Figura 2 – Soluções possíveis para braço robótico de dois graus.

Aplicando-se a lei dos cossenos ao triângulo formado pelo centro das duas juntas e pela posição do órgão terminal, consegue-se obter o valor do ângulo  $\theta_2$ , dado por:

$$\theta_2 = \pm \arccos\left(\frac{x^2 + y^2 - l_1^2 - l_2^2}{2l_1l_2}\right) \quad (3)$$

Através de alguma trigonometria tem-se:

$$\tan\theta_1 = \frac{y(l_1 + l_2 \cos(\theta_2)) - xl_2 \text{sen}(\theta_2)}{x(l_1 + l_2 \cos(\theta_2)) + yl_2 \text{sen}(\theta_2)} \quad (4)$$

A equação 3 possibilita achar o valor de  $\theta_2$  tanto para a solução cotovelo acima quanto abaixo, e a última equação permite obter os valores de  $\theta_1$  associados às soluções de  $\theta_2$ .

## 1.2 Trajetória e caminho

O controlador de um robô se encarregará de interpretar os comandos recebidos de modo a aplicar ao longo do tempo os esforços de atuadores necessários para alcançar o objetivo especificado. O manipulador deve movimentar-se suavemente, da posição atual à posição desejada, de modo a não gerar acelerações muito altas que possam forçar os atuadores nem movimentos bruscos que possam danificar a estrutura mecânica do manipulador. Desta maneira, o controlador do robô deve computar uma trajetória suave que sirva de referência a ser seguida pelos motores controladores das juntas. Aqui faremos distinção entre a trajetória propriamente dita e o caminho percorrido pelo robô. Definimos **Caminho** como a descrição geométrica do conjunto dos pontos percorridos pelo manipulador da posição atual à posição alvo. É dado por uma curva contínua, com o maior número possível de derivadas contínuas. A **Trajétória** é um caminho sujeito a restrições temporais. A geração de trajetória envolve a descrição temporal da posição, velocidade e aceleração para cada grau de liberdade do manipulador (seja em espaço cartesiano ou em espaço de junta). A geração de trajetória pode ser realizada previamente à sua execução (geração off-line), ou em tempo real, a medida em que é executada.

### 1.3 Pixel e Conectividade

Um pixel ("picture element") é o elemento básico em uma imagem. A forma mais comum para o pixel é a forma retangular ou quadrada. O pixel é também um elemento de dimensões finitas na representação de uma imagem digital. Frequentemente, a organização de uma imagem sob a forma de uma matriz de pixels é feita em uma simetria quadrada, i.e., na forma de um tabuleiro de xadrez. Isto se deve a facilidade de implementação eletrônica, seja dos sistemas de aquisição, seja dos sistemas de visualização de imagens. É importante lembrar que este tipo de organização provoca o aparecimento de dois problemas importantes nas técnicas de processamento. Em primeiro lugar um pixel não apresenta as mesmas propriedades em todas as direções, i.e., ele é anisotrópico. Esta propriedade faz com que um pixel tenha 4 vizinhos de borda e 4 vizinhos de diagonal.

Esta propriedade nos força a definir o tipo de conectividade que vamos trabalhar, ou D4 (onde levamos em consideração apenas os vizinhos de borda) ou em D8 (onde levamos em consideração os vizinhos de borda e os de diagonal), Figura 3. O segundo problema é consequência direta do primeiro, ou seja, as distâncias entre um ponto e seus vizinhos não é a mesma segundo o tipo de vizinho (ela é igual a 1 para vizinhos de borda e  $\sqrt{2}$  para aqueles na diagonal).

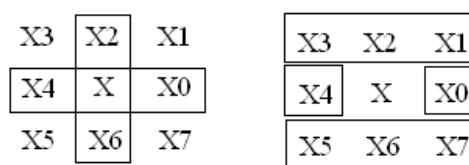


Figura 3 – Conectividade do tipo D4 e D8 respectivamente.

Alguns tipos de algoritmos que são sensíveis a este problema são: as operações morfológicas que usam uma matriz de análise do tipo 3x3, as operações de esqueletização em análise de formas e principalmente as transformações de Distâncias (na transformação de distância, cada ponto da imagem ao invés de representar uma intensidade luminosa, representa uma distância, de um dado ponto referência ao ponto calculado).

### 1.4 Objetivos

Este trabalho tem como objetivo a construção de uma plataforma simuladora de um manipulador robótico, baseado em um sistema microcontrolado interligado ao computador via porta USB utilizando motores de passo para o posicionamento do sistema, sendo esta plataforma reprogramável via software.

## 2. Materiais e Métodos

Nos tópicos a seguir são apresentados os conceitos básicos para se entender os processos envolvidos na implementação deste projeto. Aborda-se sucintamente algumas características de um microcontrolador, da arquitetura USB e dos motores de passo.

### 2.1 Microcontrolador pic 18F4550

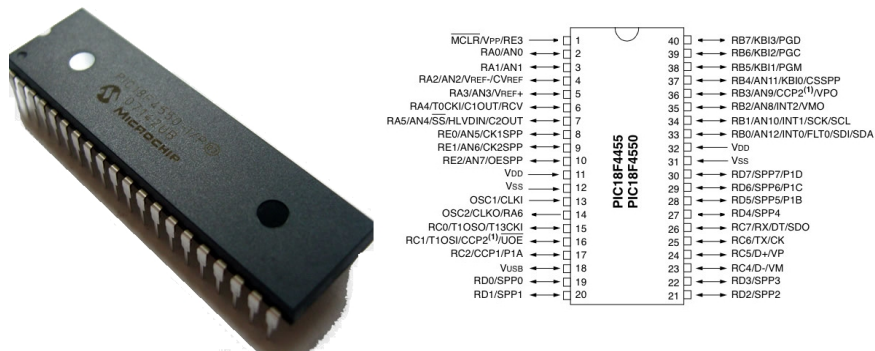


Figura 4 – Diagrama PIC 18F4550.

Defini-se microcontrolador como um componente eletrônico, dotado de uma “inteligência” programável utilizado no controle de processos lógicos. O microcontrolador é provido internamente de uma memória de programa, memória de dados, portas de entrada e/ou saída paralela, timers, contadores, comunicação serial, PWMs, conversores analógicos-digitais [4].

Os microcontroladores se diferenciam dos microprocessadores, pois além dos componentes lógicos e aritméticos usuais de um microprocessador de uso geral, o microcontrolador integra os elementos adicionais em sua estrutura interna que foram supracitados[3].

O PIC18F4550 é um microcontrolador da fabricante Microchip pertencente a família 18Fxxxx, à qual contém uma interface serial USB compatível com a USB de média velocidade e a de baixa velocidade que permite rápida comunicação entre qualquer USB PC e o microcontrolador PIC [5]. Esta interface pode utilizar um transceptor interno, ou pode ser conectado através de um transceptor externo ao PC. Um regulador interno de 3,3 V está também disponível para energizar o transceptor interno em aplicações de 5 V [6]. Esta família de microcontroladores foi escolhida devido à sua característica de possuir internamente o hardware para comunicação USB, que é uma comunicação extremamente versátil e veloz, além de possuir um número de entradas e saídas elevado.

A Figura 5 representa as configurações de clock do microcontrolador, onde o clock é a medida de processamento do microcontrolador, pois a cada 4 ciclos do clock o microprocessador realiza uma instrução em sua unidade lógica aritmética, ou seja, quanto maior o clock, maior a taxa de processamento. Para o nosso caso em que utilizamos a interface USB, uma exigência para que esta interface funcione corretamente é que o microcontrolador trabalhe em 48 Mhz, que é a máxima frequência do clock para esta família. Outra vantagem desta família é que, internamente eles possuem um dispositivo denominado por PLL que se trata de um multiplicador de clock, ou seja, basta que seja inserido uma frequência de 4Mhz ou múltipla superior que configurando corretamente o hardware, na saída haverá 48Mhz. Isto é uma grande vantagem, uma vez que quanto maior a frequência do cristal, maior o seu preço, assim é possível economizar utilizando um cristal de 4 Mhz, por exemplo.

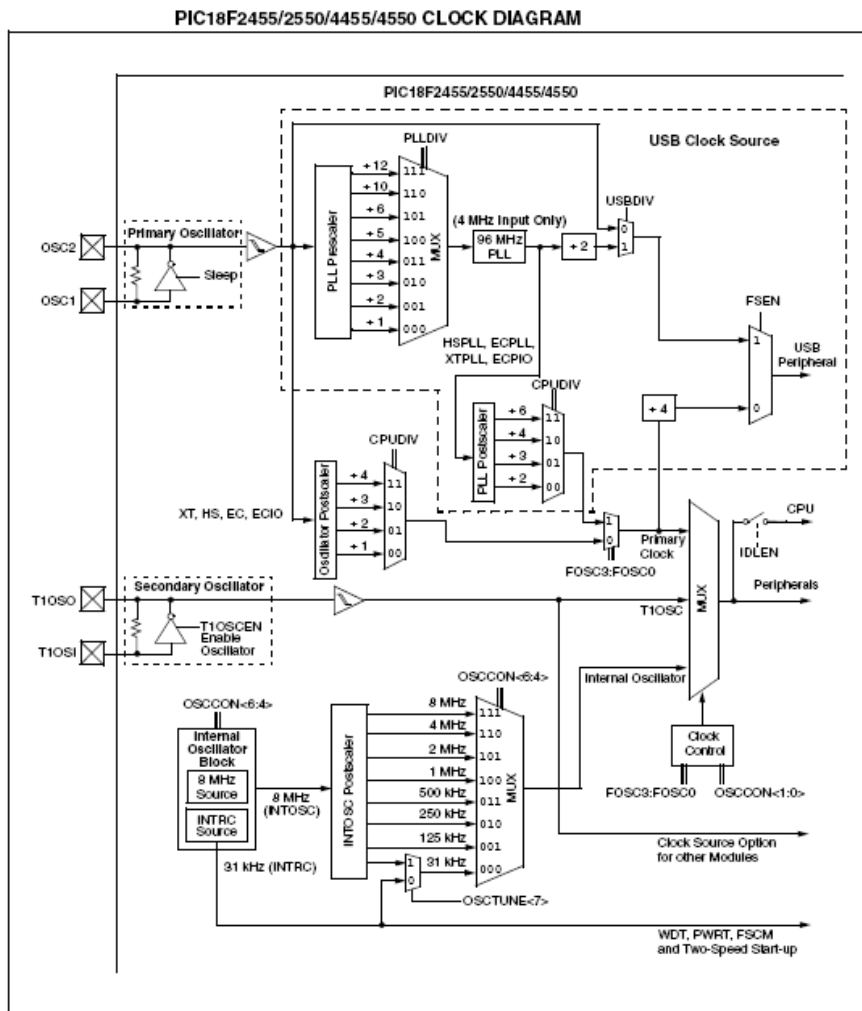


Figura 5 - Diagrama de configurações de clock do PIC 18Fxxxx [5].

## 2.2 Motor de passo

Um motor de passo é um tipo de motor elétrico usado quando algo tem que ser posicionado muito precisamente ou rotacionado em um ângulo exato. Neste tipo de motor a rotação do balansete é controlado por uma série de campos eletromagnéticos que são ativados e desativados eletronicamente[8].

O controle computadorizado de motores de passo é uma das formas mais versáteis de sistemas de posicionamento, particularmente quando digitalmente controlado[8]. Baseado nestas características do motor de passo, este tipo foi o escolhido para esta aplicação, pois alia as nossas necessidades sem perder a versatilidade.

As Figuras 6 e 7 ilustram bem o funcionamento do motor de passo unipolar em duas configurações, **meio passo** e **passo inteiro** (Figura 7) que é o utilizado neste trabalho .O motor funciona basicamente alternando as bobinas do estator, ou seja, energizando as bobinas de forma sistemática dependendo do tipo de passo. Esta alternância de bobinas gera o campo magnético variante que o rotor tende a seguir. A precisão do motor é alta, pois a cada conjunto de bobinas acionadas, o rotor gira uma quantidade fixa que varia de motor para motor devido às características construtivas. Esta quantidade é definida como o “Passo” do motor, que para este trabalho é de  $1,8^\circ$  por passo inteiro. Entretanto, o motor pode ser acionado no modo meio passo (Figura 6), desta forma a cada alternância das bobinas o motor gira metade do passo, daí vem o nome meio passo. Para esta configuração existe uma enorme vantagem que é a precisão, mas, com o aumento da precisão o torque tende a diminuir. Sabendo destes fatos a configuração escolhida foi a do passo inteiro, pois essa apresenta maior torque [9].

As características, pinagem e modos de ligação dos motores utilizados neste trabalho estão presentes no Anexo-1, no final deste trabalho.

## Meio Passo

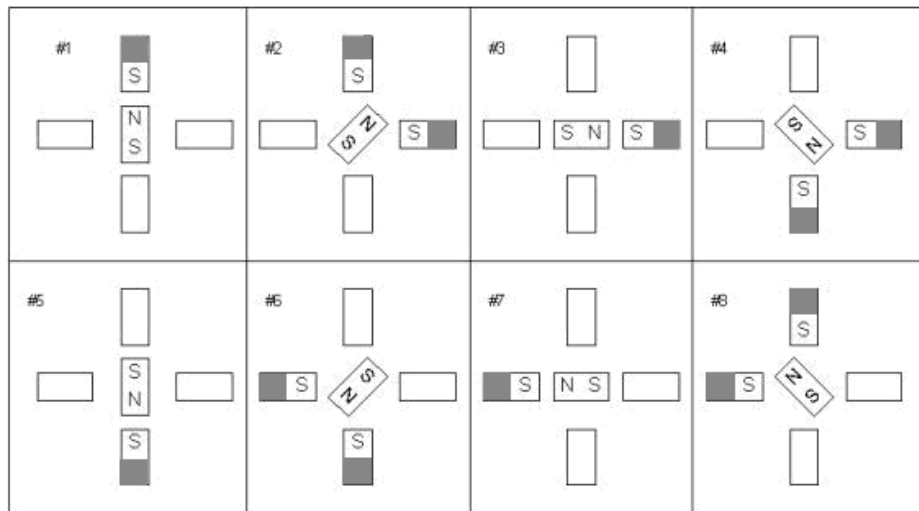


Figura 6 - Motor de passo – meio passo.

## Passo inteiro

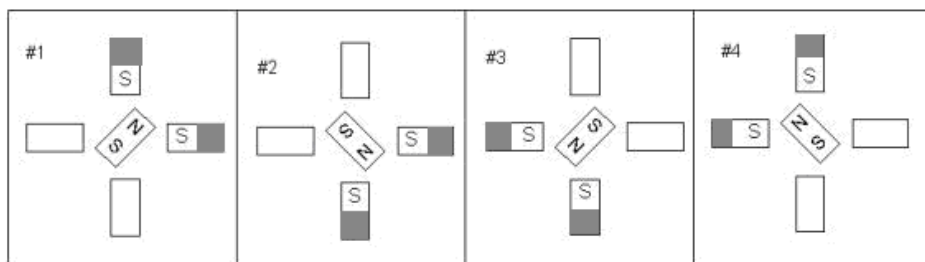


Figura 7 - Motor de passo – passo inteiro.

### 2.3 Comunicação USB (Universal Serial Bus)

A USB foi projetada para dar suporte a vários periféricos sem as limitações e frustrações das interfaces mais antigas. Das qualidades da USB pode-se destacar:

- Baixo custo.
- Fácil de usar.
- Confiável, erros são raros, e quando acontecem existem novas tentativas de automáticas de comunicação.
- Energeticamente econômica.
- Suportada pelo Windows e por outros sistemas operacionais.



### 2.3.1 Topologia USB

O barramento USB conecta dispositivos e hosts que suportam este padrão. A interconexão física da USB usa a topologia estrela disposta em níveis. Um hub, conector, é o centro de cada estrela. Cada segmento de cabo é uma conexão pontual entre o host e hub ou função, ou um hub conectado a outro hub ou função. A Figura 8 ilustra esta topologia.

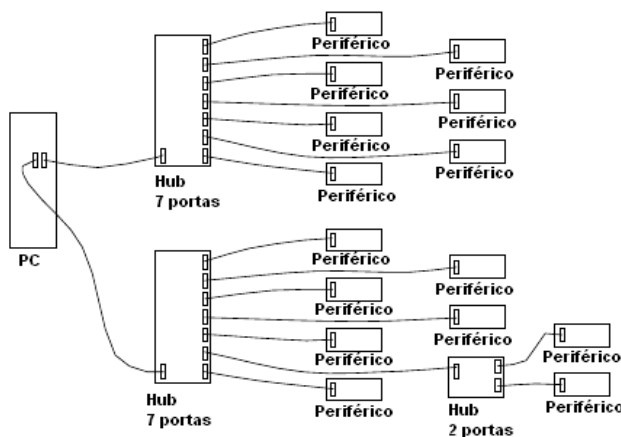


Figura 8 – Topologia estrela USB.

### 2.3.2 Interface Elétrica

As transferências de sinal e eletricidade são feitas através de um cabo, com quatro fios, como mostrado na Figura 9.

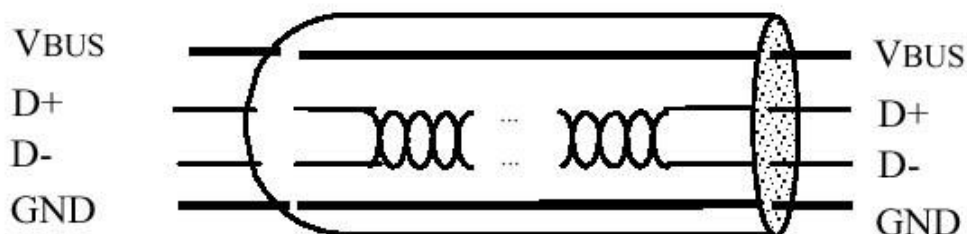


Figura 9 – Cabo USB.

O cabo possui os fios VBUS e GND em cada segmento para distribuir energia aos dispositivos. VBUS é nominalmente +5 V na fonte. A USB permite a utilização de segmentos de cabo variáveis até alguns metros, escolhendo-se um padrão adequado de condutores. Para fornecer níveis de voltagem de entradas garantidos e impedâncias apropriadas nas terminações, são utilizadas terminações parciais no fim de cada cabo. Estas terminações permitem a detecção da conexão e remoção em cada porta e a diferenciação entre dispositivos de alta e baixa velocidade.

## 2.4 Montagem do circuito de acionamento

Cada motor de passo é constituído de quatro enrolamentos definidos por A, A', B, B', sendo que para o ciclo completo do passo inteiro basta realizar a sequência da Figura 10[10].

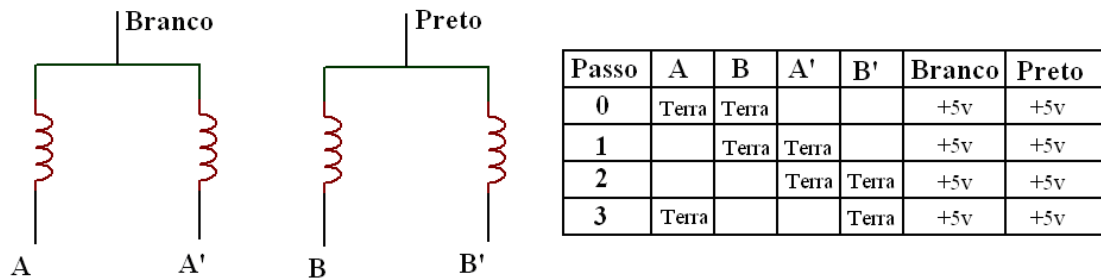


Figura 10– Enrolamento do motor de passo e sequência para execução do passo inteiro[10].

Para acionar um motor de passo utilizando microcontroladores, é necessário atenção, pois, como o motor é uma carga muito indutiva, existe o risco da ocorrência de arcos de tensão, pois os motores trabalham chaveando. Assim, o circuito da Figura 11 foi utilizado em todos os motores, e demonstrou-se extremamente eficiente e simples .

Quando o transistor é ligado, este funciona como um curto, ligando o enrolamento do motor, e quando o transistor desliga, o diodo funciona como uma roda livre para a corrente da tensão reversa gerada pelo chaveamento da carga indutiva. Este diodo é o que garante a integridade do microcontrolador, Figura 11 [11].

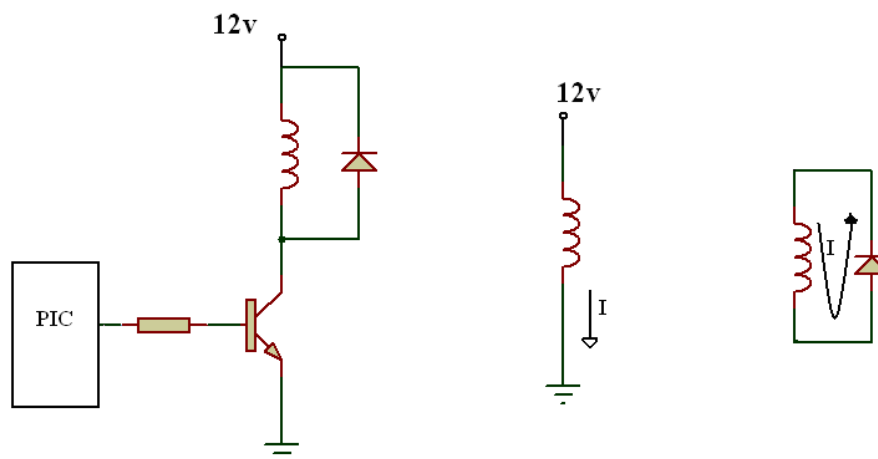


Figura 11 – Circuito de acionamento , transistor ligado e transistor desligado [10].

## 2.5 Simulador de circuitos

Para a criação e desenvolvimento de sistemas elétricos microcontrolados é indispensável a utilização de sistemas que simulem o funcionamento do circuito antes da implementação real, pois isto reduz a possibilidade de erros e aumenta a confiabilidade do sistema real. Além disso estes softwares podem ser utilizados no desenvolvimento das placas de circuito, pois apresentam ferramentas gráficas para este fim. Assim, o software participa de todas as etapas do projeto, desenvolvimeto, testes e construção.

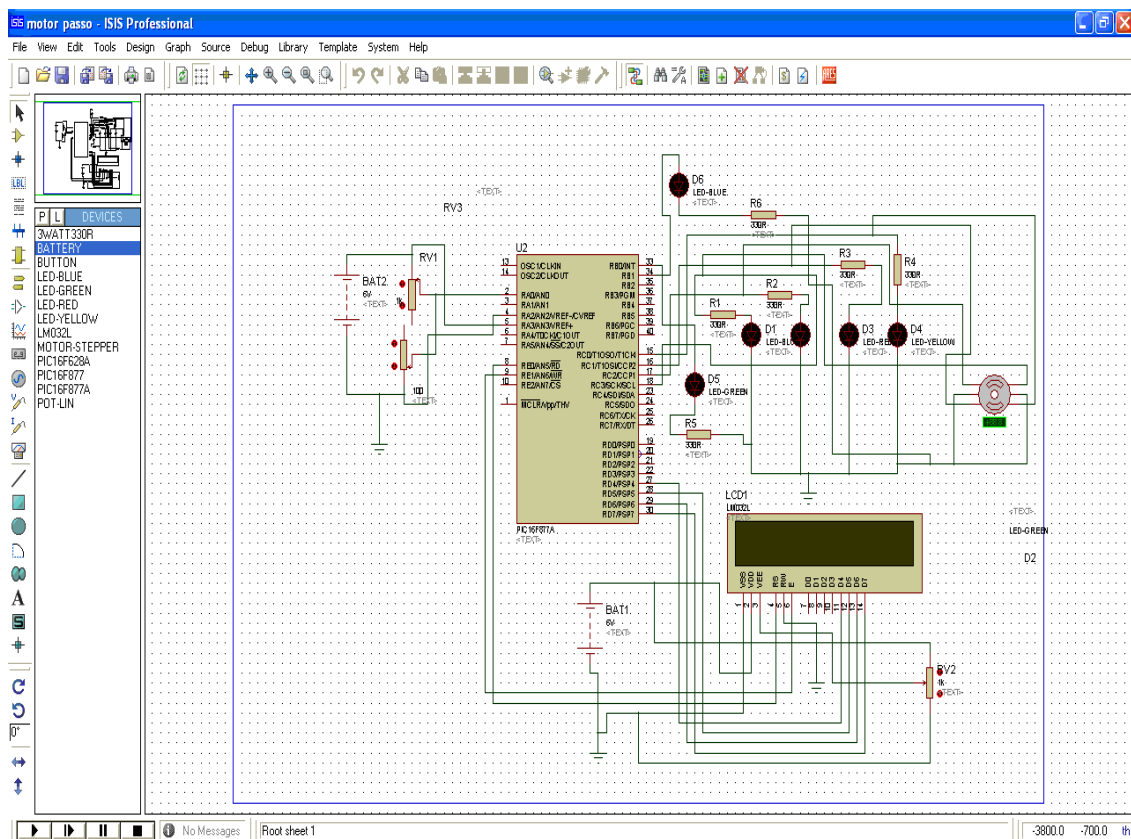


Figura 12 - ISIS Protheus v7.2 Exemplo demonstrativo do software.

Utilizou-se o ISIS Protheus v7.2 como ferramenta de simulação e planejamento do sistema. Este software possibilita inserir elementos como microcontroladores, LCDs e motores de passo. Todos os testes realizados antes da montagem do circuito impresso foram realizados neste software e constatou-se que realmente a simulação do sistema antes da implementação é de suma importância no planejamento e execução do projeto. Este software também foi utilizado na confecção da placa do circuito impresso, pois com ele podemos inserir todos os componentes necessários e visualizar o “layout” da placa.

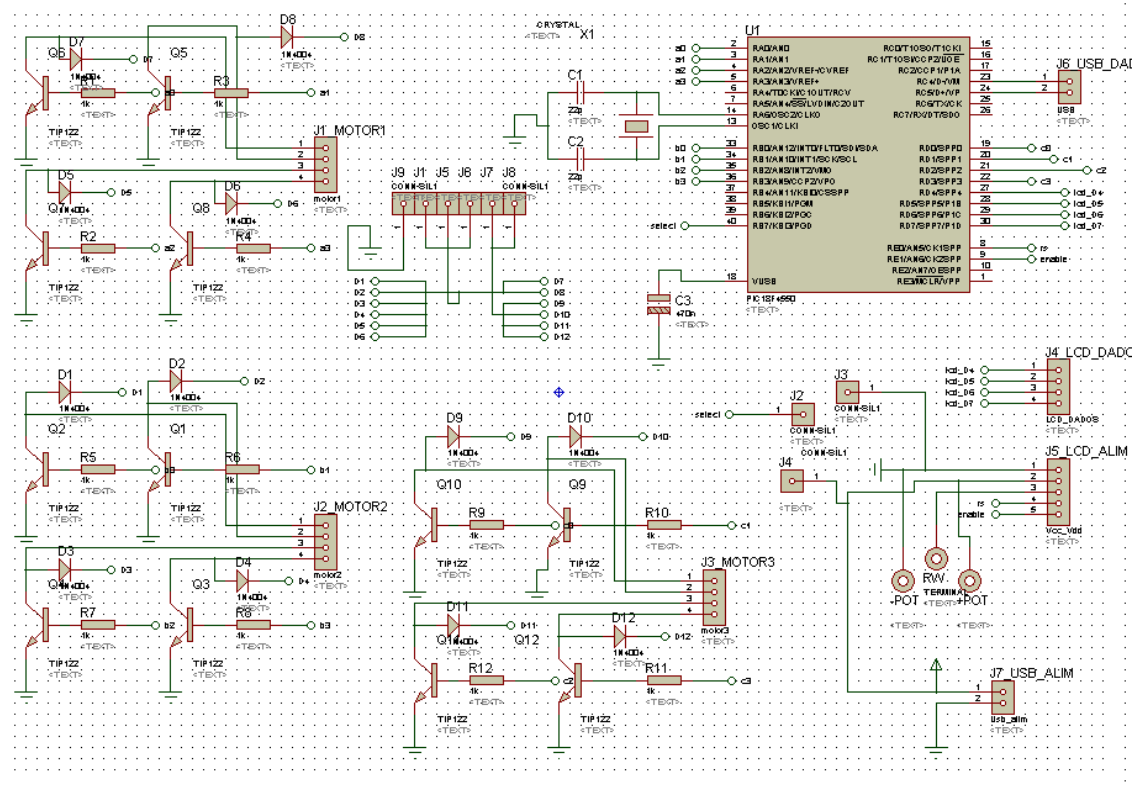


Figura 13 – Circuito desenvolvido no projeto.

## 2.6 Desenvolvimento do circuito impresso

Para que o resultado final apresentasse boa performance construiu-se uma placa de circuito impresso para o protótipo. Isto garantiu a confiabilidade e reduziu as chances de erros na execução das tarefas, principalmente em se tratando de sistemas microcontrolados com comunicação com o computador. Assim, após todos os testes e da definição final do circuito, importou-se o circuito para o ARES Professional, que é uma ferramenta do próprio software Protheus. Após a exportação do arquivo, montou-se os componentes de maneira que melhor se encaixasse no circuito e gerou-se o layout final da placa.

Com este layout utilizou-se uma técnica muito aplicada por “hobistas” que fazem circuitos impressos. Imprimiu-se o layout da Figura 14 em papel fotográfico utilizando uma copiadora a laser e transferiu-se o layout para a placa cobreada com uma fonte de calor. O calor faz com que o tonner impresso no papel solte-se e fixe na placa metálica, em seguida colocou-se a placa em solução de Percloroeto de Ferro até que a solução retirasse o cobre das áreas não cobertas com tonner, Figura 15 .

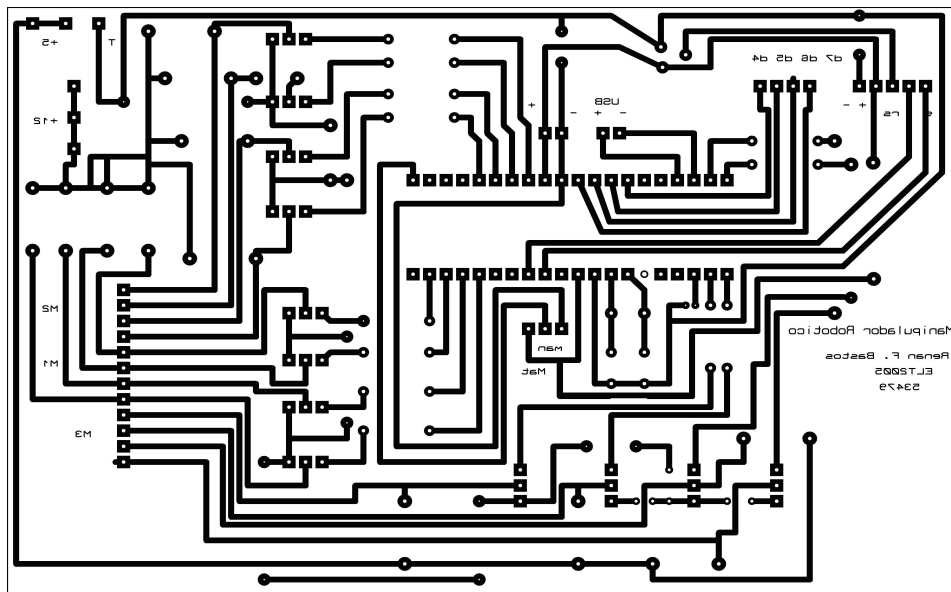


Figura 14 – Layout final da placa.

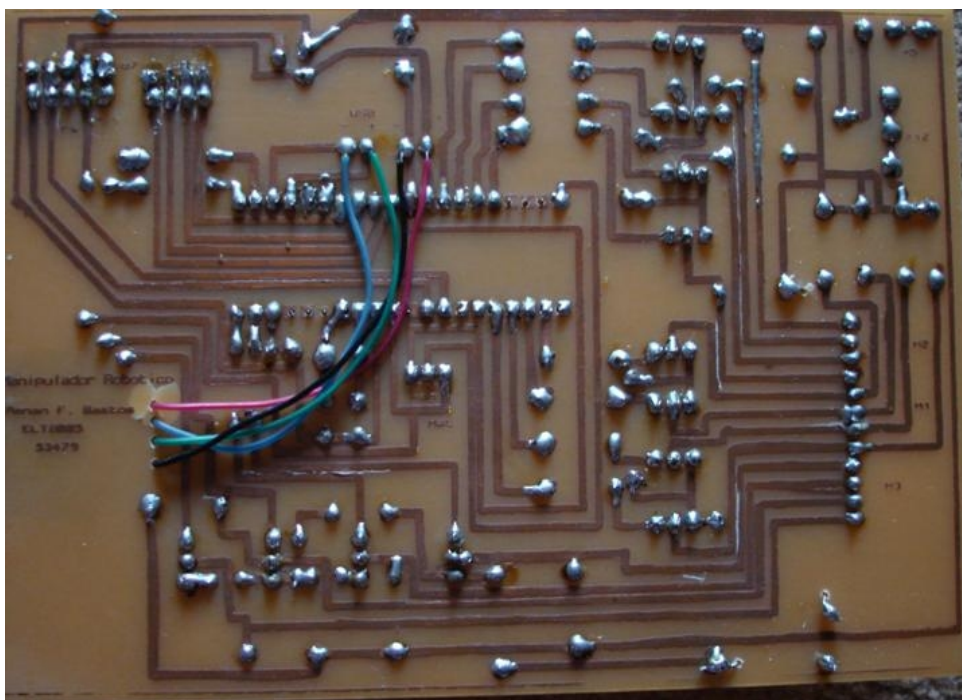


Figura 15 – Placa construída.

## 2.7 Construção final da placa

Após o desenvolvimento do layout, construção da placa e programação do microcontrolador foi feita a finalização da placa, Figura 16.

Cada motor de passo drena correntes muito altas[10], próximo a 1 ampér, assim para construir uma fonte desta magnitude, demandaria muitos componentes de alto preço e o resultado final não seria dos melhores, por isso optou-se em utilizar uma fonte de computador externa a placa do manipulador. Desta forma o circuito construído conta com as entradas para energização dos motores.

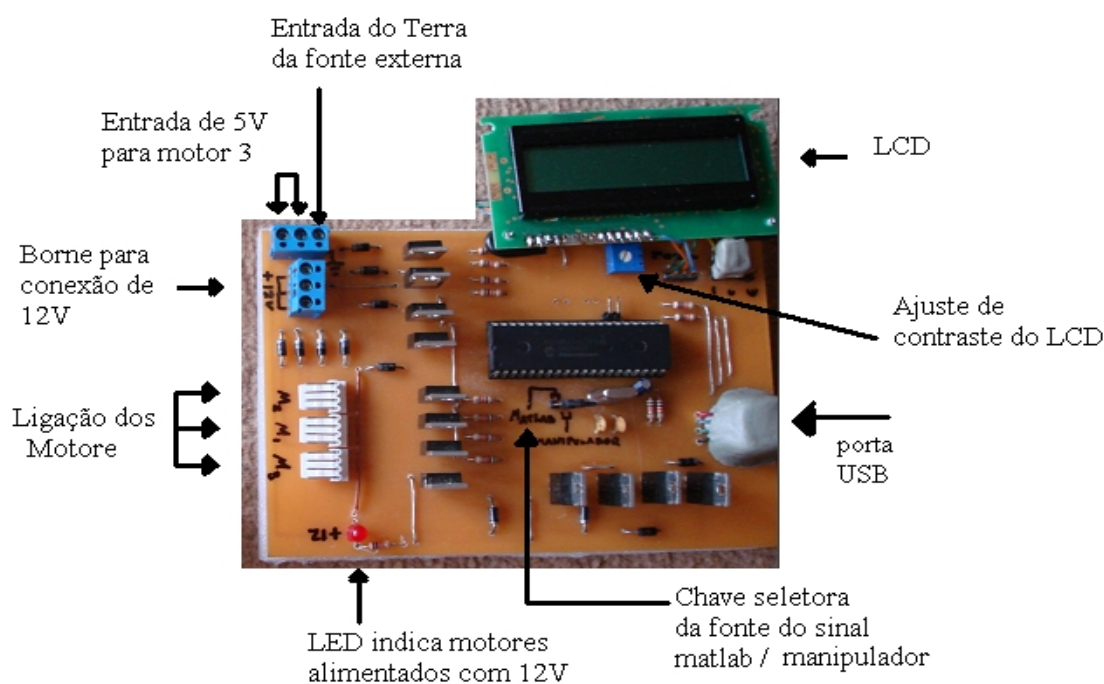


Figura 16 – Circuito final

## 2.8 LCD

Para uma melhor visualização dos dados e do estado da plataforma, foi instalado um display de LCD 16x2 de comunicação paralela, na plataforma manipuladora. Este LCD mostra se a plataforma está conectada ao computador além de mostrar a posição atual de cada motor em graus[4].

## 2.9 Software manipulador e interface USB – Serial

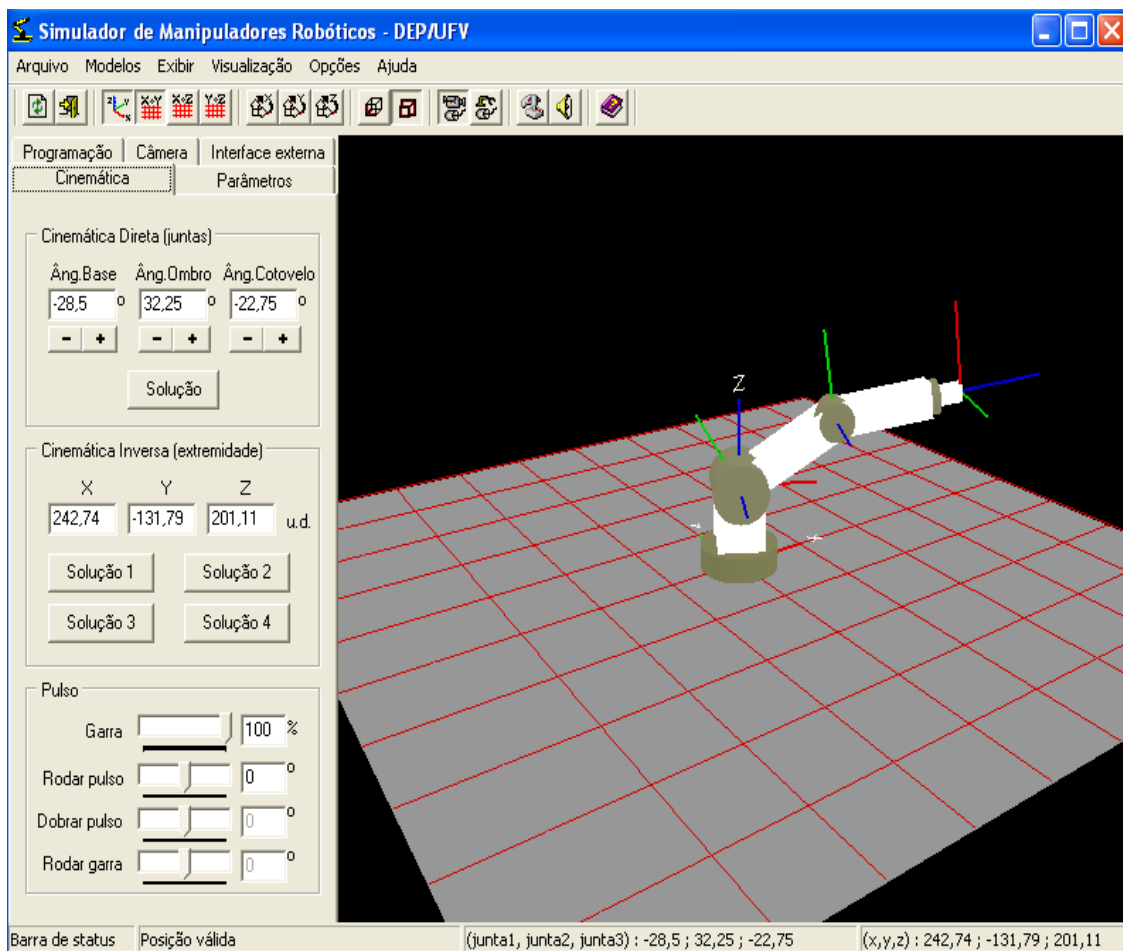


Figura 17 -Software manipulador

Este software possui todas as ferramentas para o estudo e simulação computacional de diversos tipos de manipuladores robóticos, além de possuir uma ferramenta de comunicação com o meio externo. Este software serviu de base para a plataforma simuladora, pois pode-se acompanhar em tempo real o sistema simulado e o sistema real simultaneamente. Uma das características deste programa é a existência de uma interface de comunicação com o mundo exterior através das portas seriais “COM”[12].

Como foi dito anteriormente, a plataforma pode se comunicar com o mundo externo, utilizando-se a comunicação serial, Figura 18, que é um padrão antigo e quase não encontrado em computadores da nova geração, assim utilizou-se de uma técnica muito vantajosa para a execução do software. Foi aplicado um artifício de programação em que o computador emula uma porta serial pela porta USB, Figura 19, ou seja, quando o dispositivo é introduzido no computador pela porta USB, o computador irá criar uma porta serial virtual, mesmo que este não possua a entrada serial física. Este tipo de artifício é muito versátil pois não necessita que o programa do computador seja alterado [13].

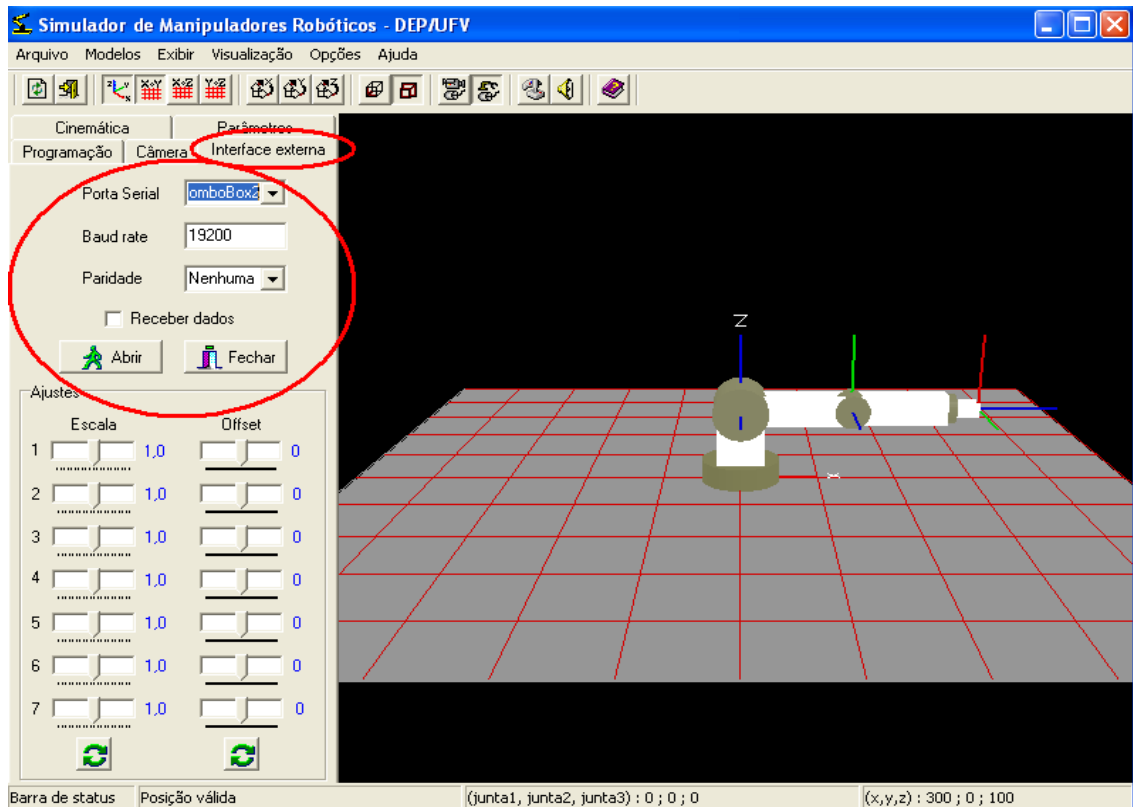


Figura 18 – Interface serial do Programa manipulador

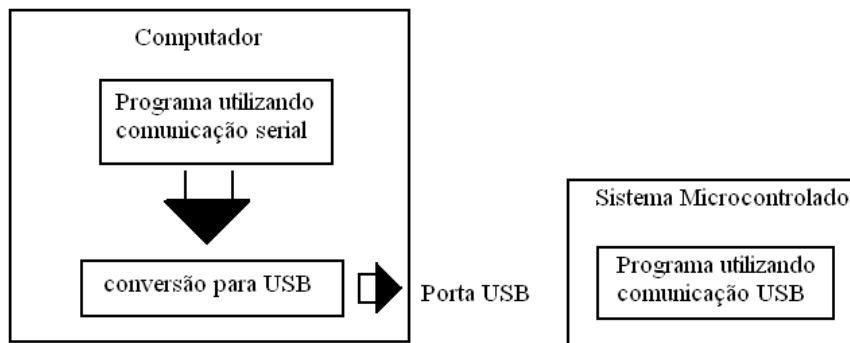


Figura 19 – Esquema utilizado na programação



Esta conversão de serial para USB foi feita pelo código do "driver" do sistema, ou seja, quando o dispositivo é inserido pela primeira vez, aparecerá um dispositivo USB no sistema, Figura 20, este dispositivo exige a instalação de um "driver " e a partir do momento que este "driver" estiver instalado, toda vez que o dispositivo for inserido, uma porta serial “COM ” será criada instantaneamente. Este "driver" para conversão é encontrado na internet livremente para ser feito o download [14] .



Figura 20 – Iniciação da plataforma USB – serial

Deve-se notar que antes do dispositivo ser conectado não existe porta serial, pois se trata de um computador portátil sem porta serial, e após a conexão do dispositivo a porta é criada, Figura 21.

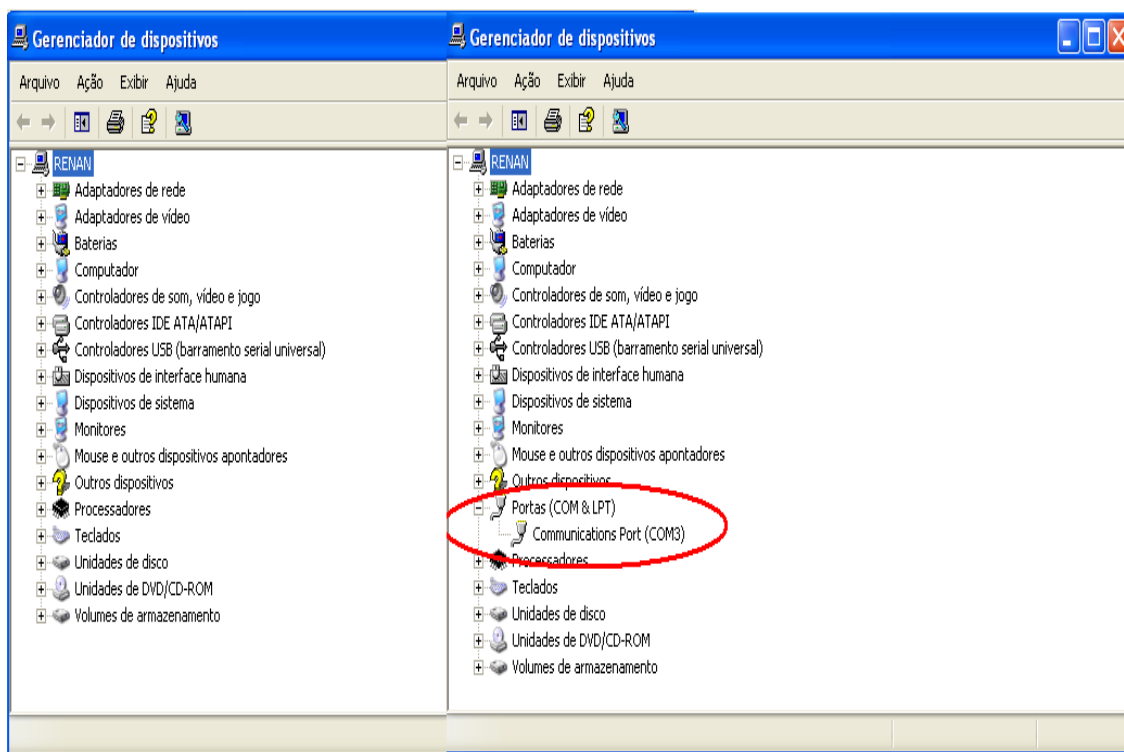


Figura 21 – Antes e depois de conectado o dispositivo

Outro ponto a se frisar é que, para cada porta USB do computador deve-se instalar o "driver", pois cada porta USB toda vez criará a mesma porta serial, por exemplo, se uma porta USB gera uma porta “COM3”, toda a vez que este dispositivo for conectado na mesma porta, sempre será criada a “COM3”.

## 2.10 Programação do Microcontrolador

A programação do microcontrolador pode ser considerada a parte mais importante deste projeto, pois um erro de programação acarretaria erros de comunicação e de posicionamento dos motores. Assim, esta foi a parte mais trabalhosa e que demandou mais tempo do projeto.

Uma das preocupações para que a plataforma manipuladora não ficasse restrita apenas ao programa manipulador, foi a de sua programação aceitar entradas do **MatLab**, que é atualmente um software muito utilizado por engenheiros e cientistas do mundo inteiro. Assim, caso o software manipulador estivesse limitado por algum motivo, o operador poderia com simples comandos do MatLab ou com qualquer outro tipo de programação controlar o sistema. Além de esta comunicação com o MatLab possibilitar o uso da plataforma juntamente com algum tipo de inteligência computacional ou visão artificial, fazendo desta plataforma mais versátil.

### 2.10.1 O microcontrolador 18f e a USB

Como mostrado anteriormente, os microcontroladores da família PIC 18Fxxxx possuem internamente o hardware para a comunicação USB[5], assim para utilizar esta comunicação basta adicionar a biblioteca USB-Serial na programação do microcontrolador e utilizar os comandos básicos. Esta biblioteca encontra-se livremente na internet[14].

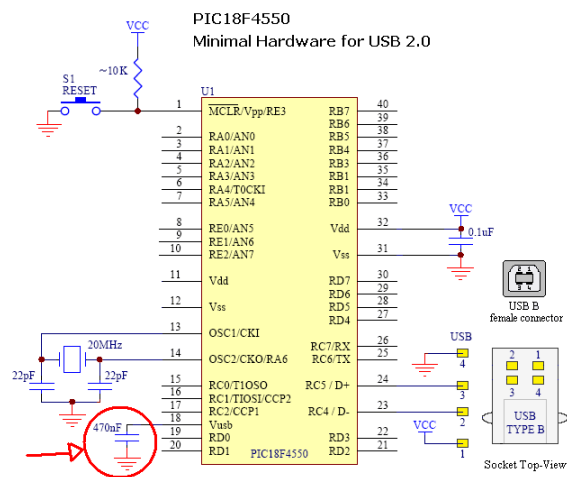


Figura 22 – Capacitor necessário ao funcionamento da USB-serial

## Diagrama de blocos do programa no Microcontrolador

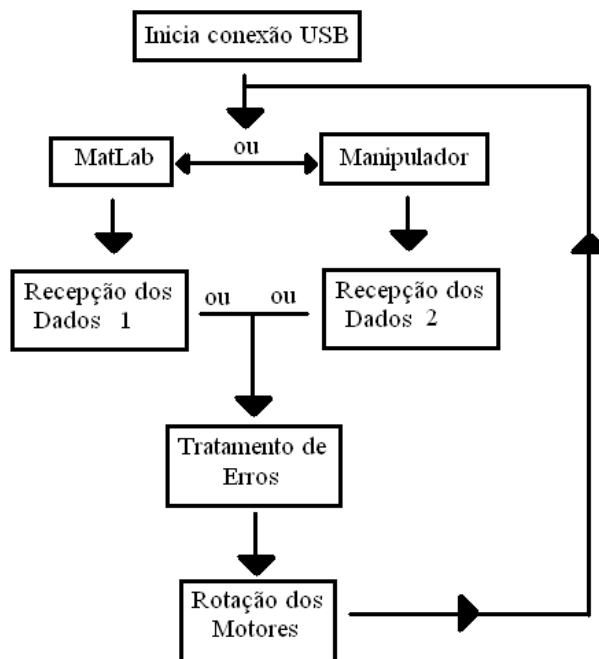


Figura 23 – Diagrama de blocos do programa

Após o programa iniciar a conexão USB, este vai detectar se os dados estão vindo pelo MatLab ou pelo programa Manipulador ( existe uma chave no circuito onde a fonte do sinal é selecionada ). Esta diferenciação é feita, pois os dados são enviados de forma diferente no MatLab e no Manipulador. Então cada dado tem que ser tratado antes de ir para rotina de rotação. Em seguida, é feito um tratamento de erros (será discutido em seguida), para que os mesmos não se acumulem. Após este tratamento de erros, os dados vão para a rotina de rotação, onde a rotina continua ativa até que o erro de posicionamento seja mínimo.

Para a plataforma não foi criada nenhuma retroalimentação da posição, ou seja, o sistema trabalha em malha aberta. Assim quando a rotina ordena que o motor se posicione em  $90^\circ$ , após o fim da rotina o software admite que o motor foi para  $90^\circ$ , mesmo se algum obstáculo impedir o movimento e a posição desejada não for alcançada. Isto foi feito, pois os motores de passo apresentam uma ótima precisão no seu posicionamento, além de que a adição de um sistema de retroalimentação da posição tornaria o projeto mais complexo sem a devida necessidade, uma vez que não é necessária esta precisão visto que se trata de um projeto didático.

### 2.10.2 Tratamento de erros :

Segundo a literatura[8], o motor de passo é um tipo de motor discreto, ou seja, caso o operador deseje uma posição que não seja múltipla do passo, essa posição não será alcançada com um erro de zero. Assim a plataforma deve apresentar alguma ferramenta que minimize esses erros. Desta forma foi criada uma metodologia para que o erro não se acumule .

Quando o operador aciona um comando à plataforma, por exemplo, 30°, Figura 24, a posição logo anterior à 30° com o mínimo erro, para um passo de 1,8° é 28,8°, pois a posição final pode ser descrita pela equação 5. Desta forma a plataforma fará o mesmo cálculo, o programa sempre para na posição anterior ao “setpoint” com o mínimo erro, independente do sentido de rotação, e armazena este valor na memória. Assim quando o operador acionar outro “setpoint”, a plataforma calculará seu deslocamento baseado na posição real de 28,8°, e novamente se o setpoint for dado de 60°, para um menor erro com um passo de 1,8° a posição final será de 59,4°, que será o valor armazenado na memória.

$$\text{Posição} = \text{Passo} * N \quad (5)$$

N= número de pulsos.

Onde “N” é dado pela parte inteira da divisão :

$$N = \text{Setpoint} / \text{Passo} \quad (6)$$

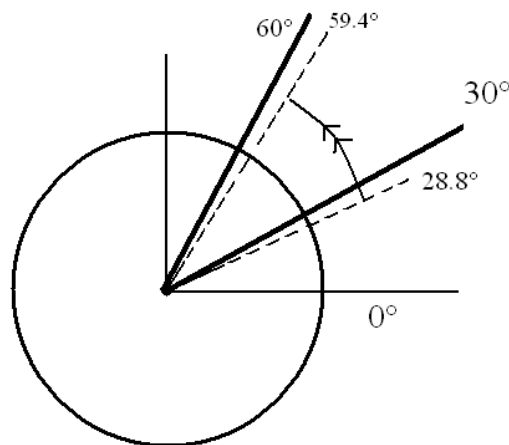


Figura 24 - Representa a programação realizada para a redução dos erros .

### 3 Interface com o MatLab

Foi também criada a possibilidade de comando do braço pelo MatLab, com o objetivo de eliminar limitação do software manipulador, além de criar novas possibilidades de utilização da plataforma. Uma vez que o MatLab já possui uma biblioteca para comunicação serial, este se torna bem simples de se trabalhar, bastando ao operado enviar os comandos a seguir :

```
s=serial('com3'); % cria porta serial "com3"  
fopen(s); % abre a porta para iniciar a comunicação  
fwrite(s,[a]); % envia um valor "a" qualquer
```

Assim, foi criada uma função para enviar os ângulos de duas juntas (a plataforma foi projetada para três motores de passo, mas a interface foi criada para um manipulador planar , considerando apenas as duas primeiras articulações) onde o ângulo de cada junta é fragmentado em três partes, centena, dezena mais unidades, além de duas casas depois da vírgula (isto é feito para evitar o envio de caracteres do tipo “float”), assim para cada grupo das duas juntas envia-se seis dígitos, Figura 25.

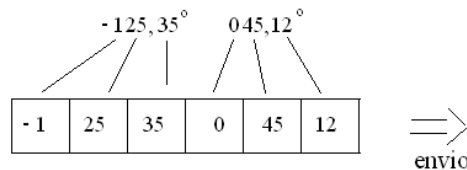


Figura 25 – Lógica de envio

De posse desta função de envio, foi criada a interface gráfica mostrado na Figura 26 com a ajuda da ferramenta “Guide” do MatLab. Nesta simples janela podemos fazer o comando pela cinemática direta ou inversa, a cinemática direta é executada na janela mostrada pela Figura 27, onde devemos simplesmente inserir os ângulos desejados e clicar em “setar”. Automaticamente os motores se posicionarão nos pontos desejados com um erro mínimo.

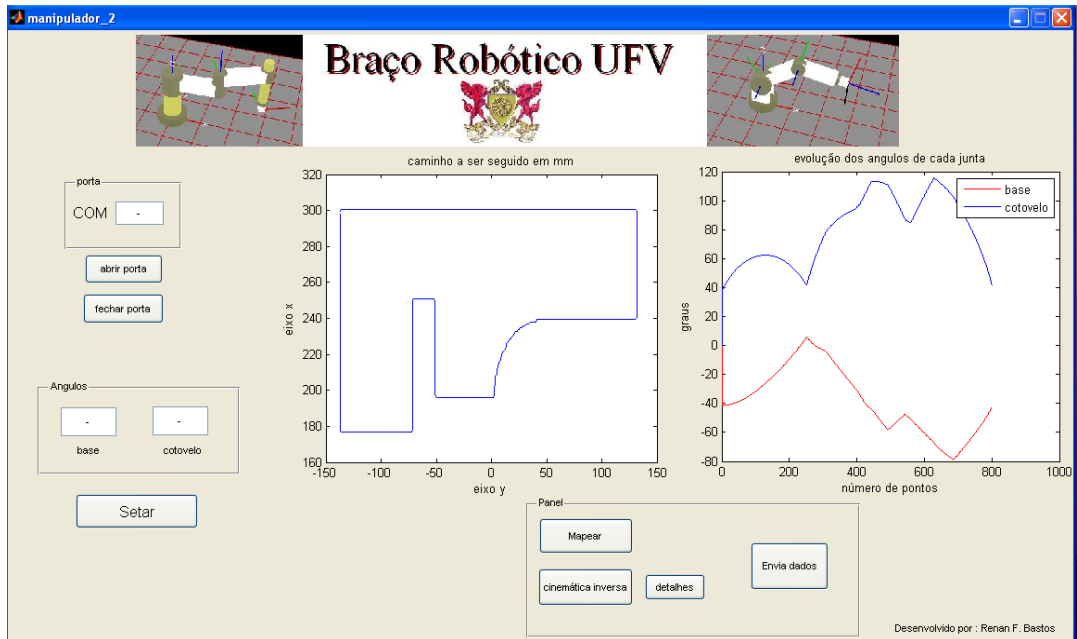


Figura 26 – Interface criada para MatLab

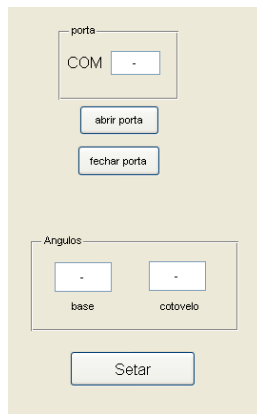


Figura 27 – Parte da janela destinada à cinemática direta

Para utilização da cinemática inversa, faz-se o uso de um sistema mais avançado. É possível inserir uma figura em formato “.bmp”, que represente a trajetória desejada, na pasta do arquivo executável, assim o programa vai gerar automaticamente através das equações (3) e (4) da cinemática inversas do respectivo modelo o caminho desejado, Figura 28.

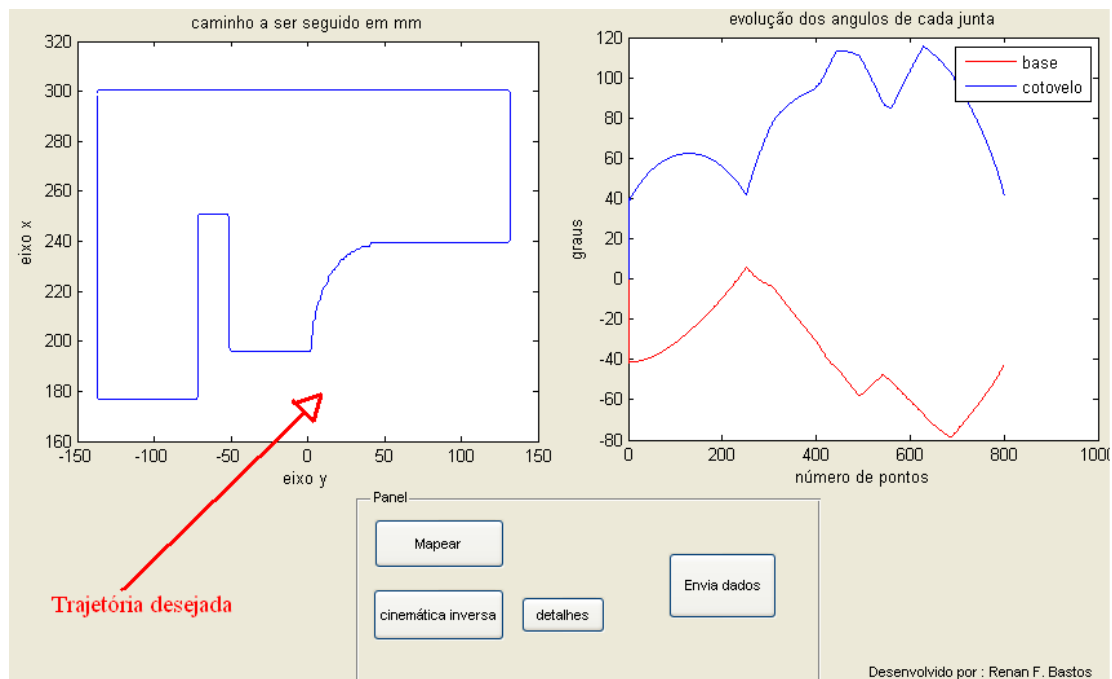


Figura 28 – Parte da janela destinada à cinemática inversa

Para gerar os ângulos das juntas a partir do arquivo de imagem, admitiu-se que o caminho encontra-se à uma distancia fixa, como da Figura 29. Assim, para um modelo articulado de 3 graus de liberdade, a coordenada “y” será constante e as coordenadas “x” e “z” serão definidas pela imagem. Para um modelo SCARA, que foi o modelo estudado nesta interface, a coordenada “z” será fixa e a imagem definirá as componentes “x” e “y” do caminho. Assim basta definir um comprimento à imagem como na Figura 30, desta forma cada pixel representa um ponto no espaço.

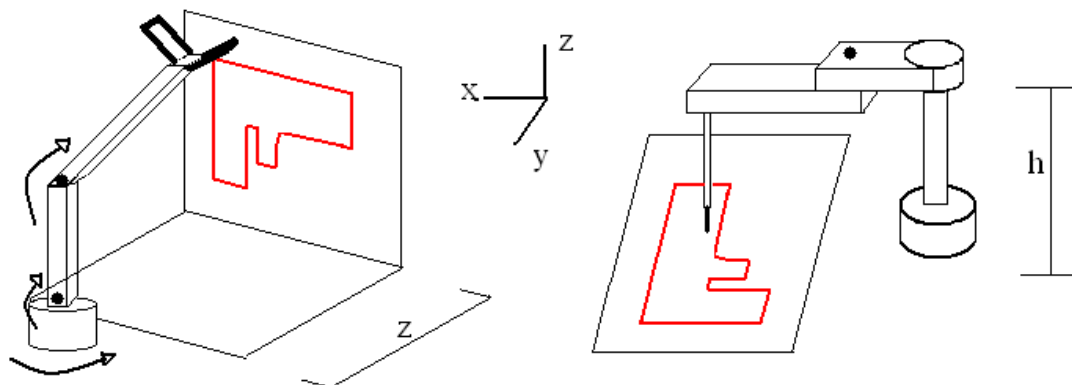


Figura 29 – Sistemas robóticos estudados, Articulado 3graus de liberdade e SCARA.

Para a transformação de uma imagem em um sistema de coordenadas cartesianas, não foi utilizada uma relação Pixels/mm constante, definiu-se um comprimento em milímetros à imagem, e de acordo com o número de pixels da imagem esta relação é estabelecida.

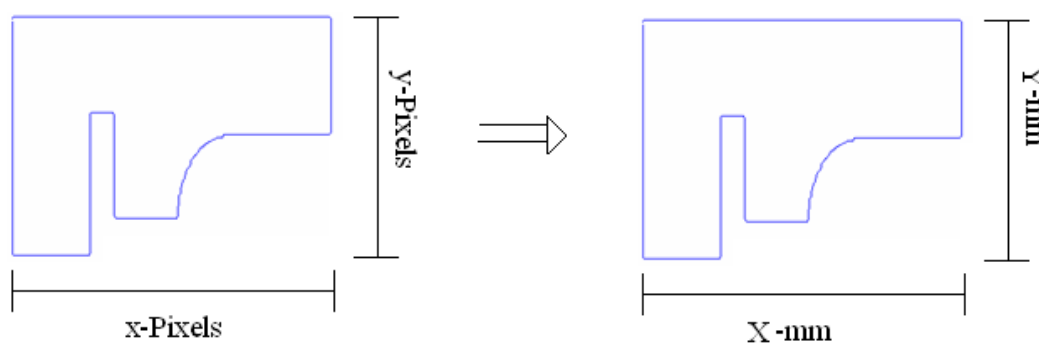


Figura 30 – Relação entre pixels e comprimento

No software criado no MatLab, a imagem inserida corresponde ao caminho a ser seguido sobre um plano fixo, à uma distância estabelecida da base do sistema robótico.

### 3.1 Mapeamento do caminho

Uma das funções do nosso programa é mapear o caminho a ser seguido para que este seja traçado continuamente sem interrupções pelo sistema robótico, similar ao braço humano enquanto desenha algo em um papel. Após mapear o caminho, ele é submetido a uma conversão através das equações da cinemática inversa do respectivo modelo, Articulado ou Manipulador planar, assim a imagem que representa as coordenadas cartesianas, é convertida em uma matriz que representa os ângulos de cada junta para cada ponto da imagem, com o número de pontos igual ao número de pixels que formam um caminho contínuo na imagem.

Para mapear o caminho, utilizamos um algoritmo simples que trabalha utilizando a conectividade em imagens da seguinte maneira; depois de encontrado o primeiro ponto da imagem, o algoritmo busca nas redondezas deste ponto um próximo ponto vizinho, utilizando a conectividade do tipo D8 (onde são levados em conta os pixels em linha reta e em diagonal), encontrado um novo ponto o algoritmo repete a rotina até não encontrar nenhum ponto em sequência. Este algoritmo é representado pela Figura 31 a seguir.



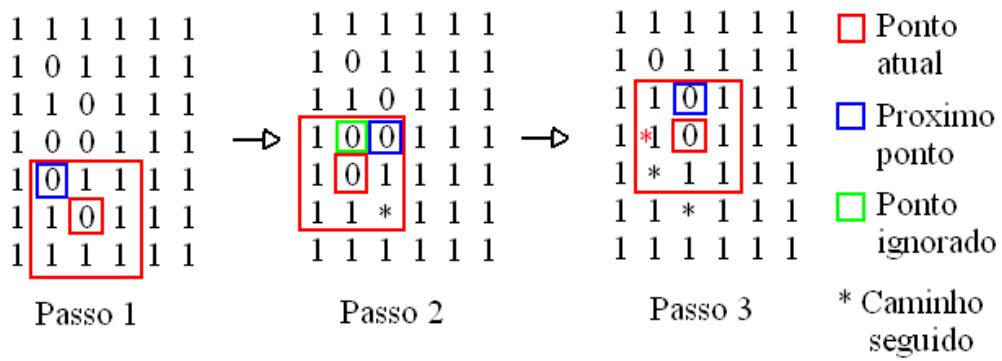


Figura 31 – Representação do algoritmo de mapeamento

Note que quando mais de um ponto é encontrado na redondeza, o algoritmo dá preferência ao caminho em diagonal e ignora os pontos em linha reta, esta foi a escolha, pois apresentou melhores resultados no mapeamento.

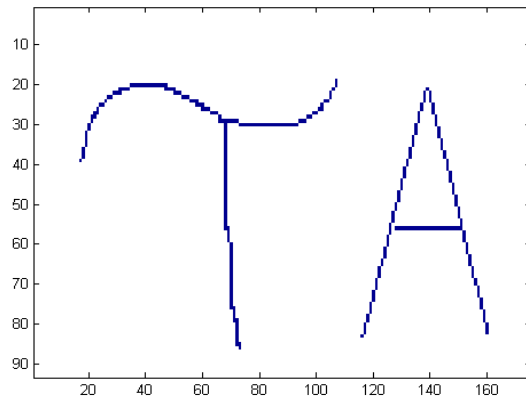


Figura 32- Caminho a ser mapeado

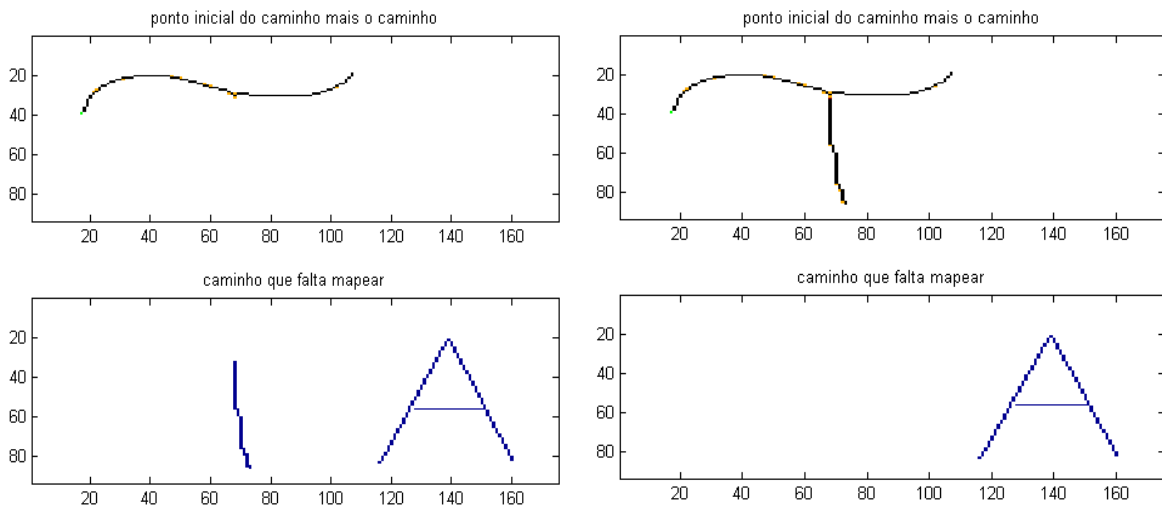


Figura 33 – Primeira parte e segunda parte identificadas pelo código, pontos claros são os pontos ignorados do caminho.

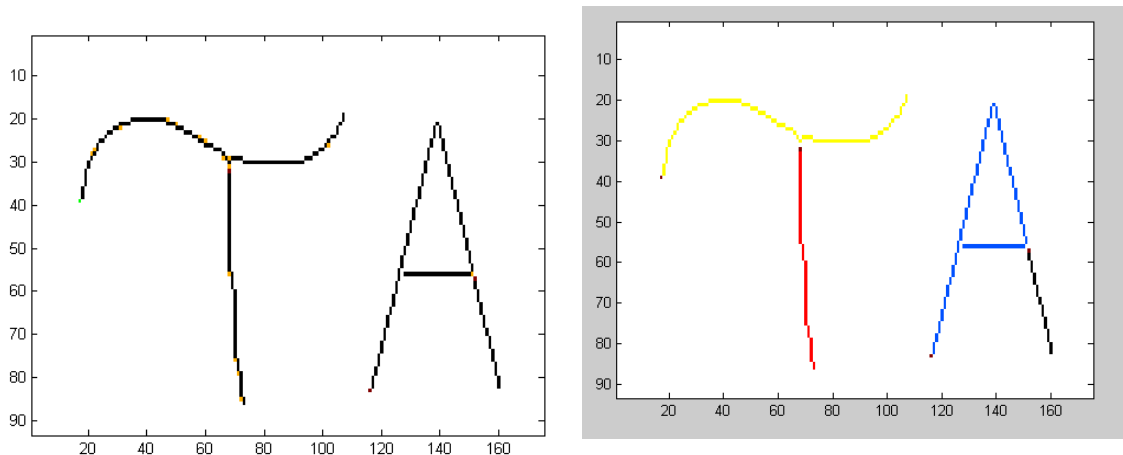


Figura 34-Caminho final mapeado e traços sem interrupções respectivamente, onde cada cor representa um traço sem interrupções

Através das figuras de mapeamento acima, pode-se notar que o algoritmo de mapeamento funciona exatamente como foi descrito segue o caminho ignorando os pontos desnecessários a sua formação.

### 3.2 Cinemática inversa aplicada ao caminho mapeado

Assim que o caminho é mapeado, as coordenadas cartesianas são inseridas nas equações (3) e (4) de cinemática inversa para o modelo robótico em questão (No caso da interface com o MatLab, Manipulador planar) desta forma são geradas as coordenadas de cada junta para cada ponto no plano cartesiano. A seguir na Figura 35 e Figura 37 são mostradas as imagens do caminho desejado e dos ângulos de cada junta associadas a cada ponto do caminho.

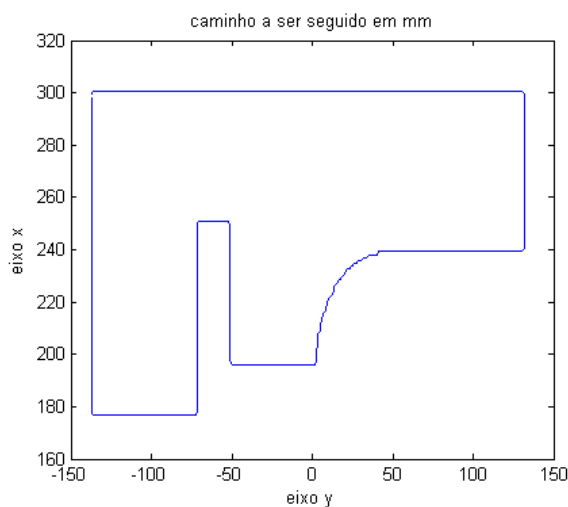


Figura 35-Caminho desejado.

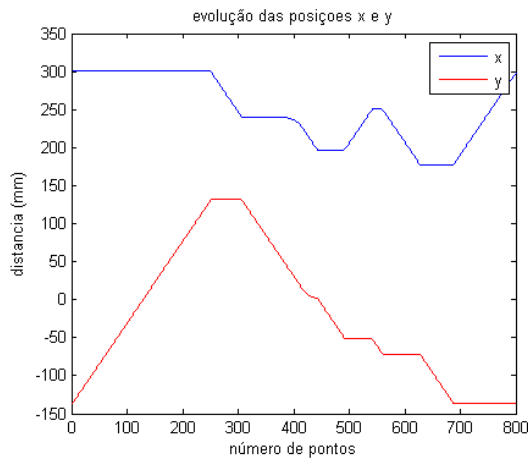


Figura 36 – Evolução das coordenadas X e Y do caminho

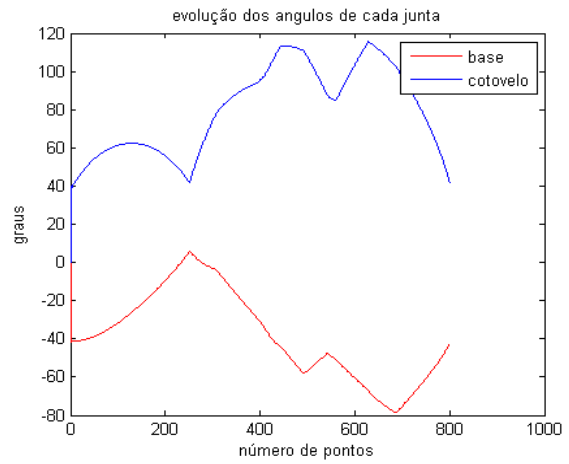


Figura 37- Evolução dos ângulos de cada junta

### 3.3 O braço mecânico

Devido à grande complexidade mecânica de um braço robótico com três graus de liberdade, foi descartada a possibilidade de construir-lo, assim optamos em fazer a montagem de um manipulador planar, pois além de ser mais simples sua construção, este ainda nos permite estudar perfeitamente a cinemática de um robô.

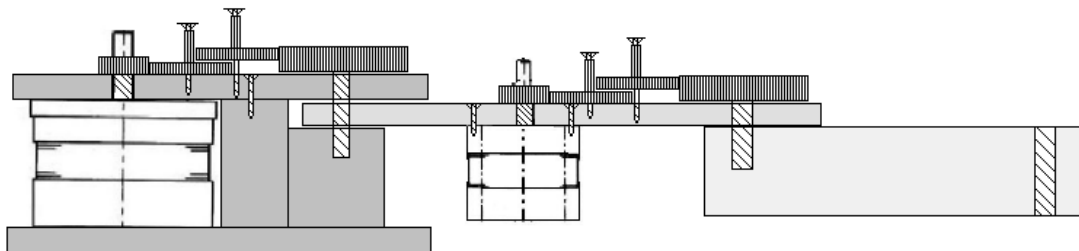


Figura 38- Manipulador planar construído.



Figura 39 – Foto do manipulador planar construído.

Uma redução mecânica de 15 vezes, foi inserida em cada junta devido a dois fatores:

- Aumento do torque de eixo
- Aumento da precisão dos movimentos

Fazendo uma analogia com os sistemas elétricos, podemos explicar a redução mecânica como se fosse um transformador, Com a redução da velocidade temos um aumento do torque e da precisão, Figura 40 e Figura 41, visto que para o motor de passo, o passo é reduzido na mesma proporção que a velocidade, assim foi possível aumentar a precisão em um fator de aproximadamente 15 vezes.

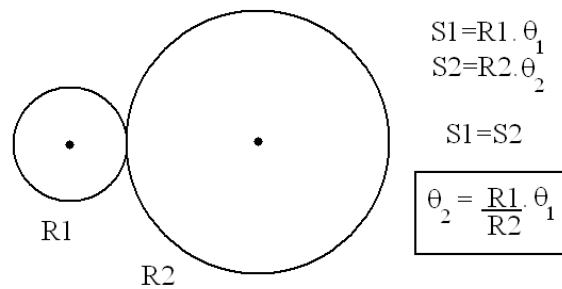


Figura 40- Aumento da precisão.

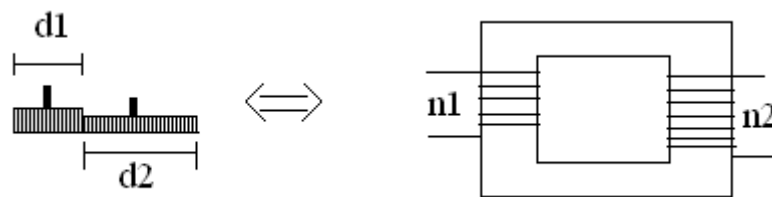


Figura 41- Analogia entre sistema elétrico e mecânico.

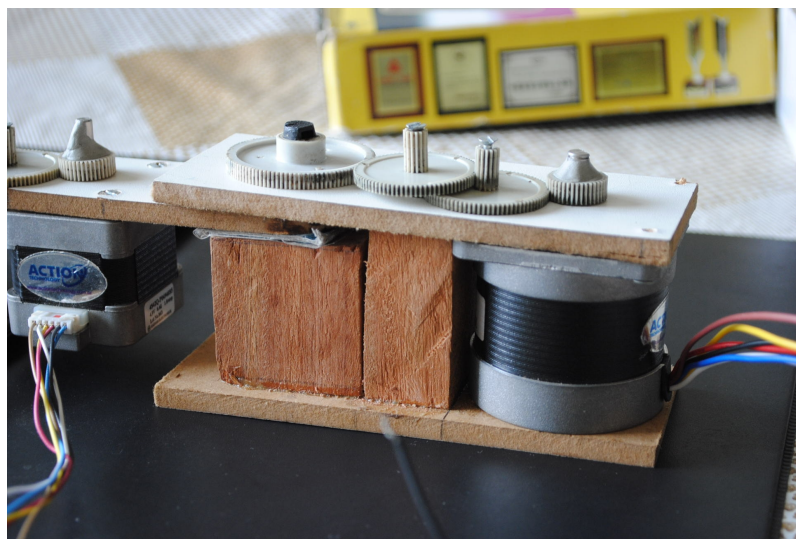


Figura 42 - Sistema de redução criado

## 4.Resultados Alcançados

Após a finalização da placa foram feitos os testes para confirmar sua eficácia, e mostrar sua versatilidade tanto funcionando com o programa Manipulador quanto com o MatLab. Na Figura 43, temos as telas de iniciação da plataforma, para o software Manipulador e para o MatLab.

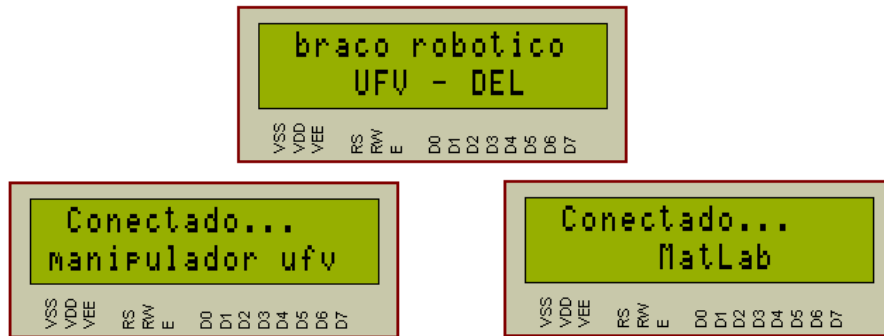


Figura 43 Circuito funcionando com MatLab ou software Manipulador

### 4.1 Resultados com software Manipulador

Para mostrar o desempenho e falhas no sistema ligado ao software Manipulador, vamos submetê-lo a algumas situações e ver suas reações perante a estas determinadas situações.

A Figura 44 mostra que o software manipulador envia somente a parte inteira do valor dos ângulos. Isto em alguns pontos é bom e em outros causa transtornos. O ponto positivo é que independente se o valor enviado é  $51,5^\circ$  ou  $51^\circ$  o programa envia só a parte inteira facilitando a comunicação e tornando o programa do microcontrolador mais rápido por trabalhar somente números inteiros.

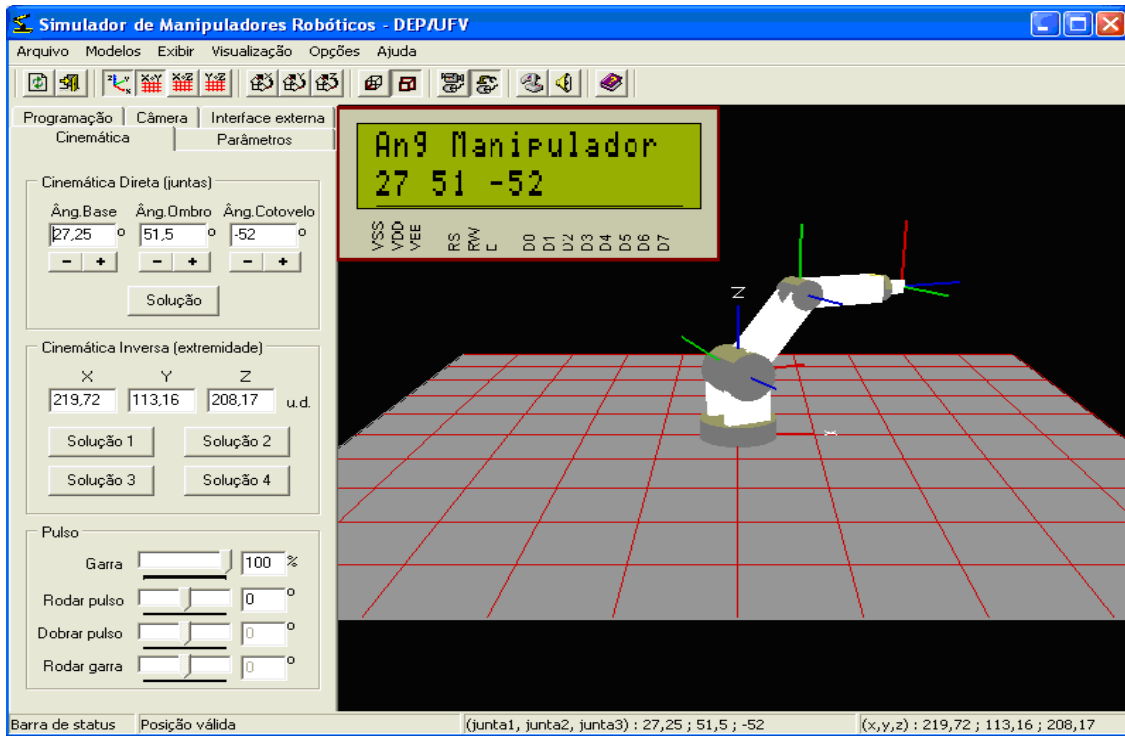


Figura 44 – Programa em funcionamento e posicionamento recebido pela plataforma

OBS: A posição mostrada no LCD da plataforma é a posição recebida pela plataforma e posição instantânea do motor.

Mas há um ponto negativo, como a comunicação do software trabalha apenas com números inteiros de 8 bits, ele pode representar somente 256 números, de 0 a 255, assim a plataforma conectada ao software Manipulador só terá uma amplitude de movimentação de 256 graus, que no caso vai de  $-99^\circ$  a  $156^\circ$ , Figura 45.

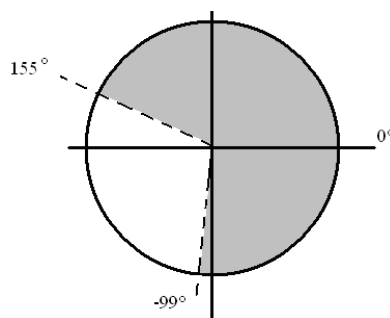


Figura 45 – Região coberta por cada motor utilizando software Manipulador

Pela Figura 46 vemos que para um ângulo enviado de  $173^\circ$  para a base, o valor recebido continua em zero. Pois o software Manipulador foi desenvolvido com limitações na comunicação serial.

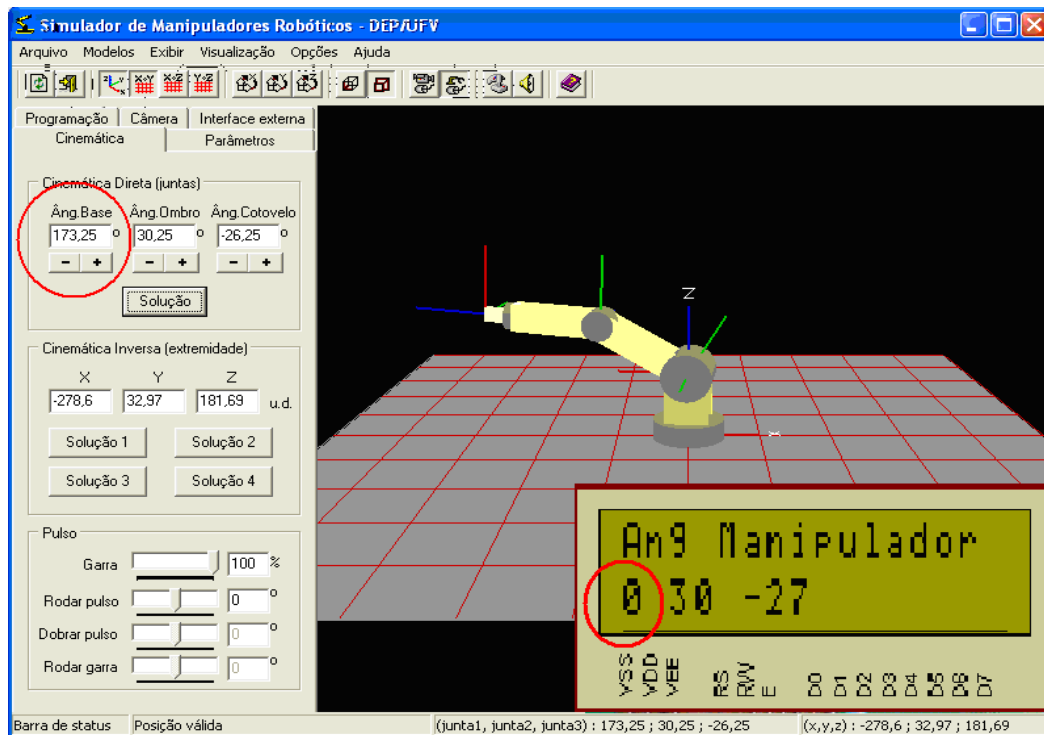


Figura 46 – Erro devido à limitação da plataforma

Apesar de o software Manipulador apresentar este pequeno inconveniente, este ainda é de grande valia, pois alia a simulação gráfica à um controle em tempo real da plataforma física construída, fazendo com que possamos ver o comportamento de um sistema por duas visões distintas, real e virtual. O que não ocorre no sistema ligado ao MatLab, visto que a interface criada no MatLab não apresenta a simulação gráfica.

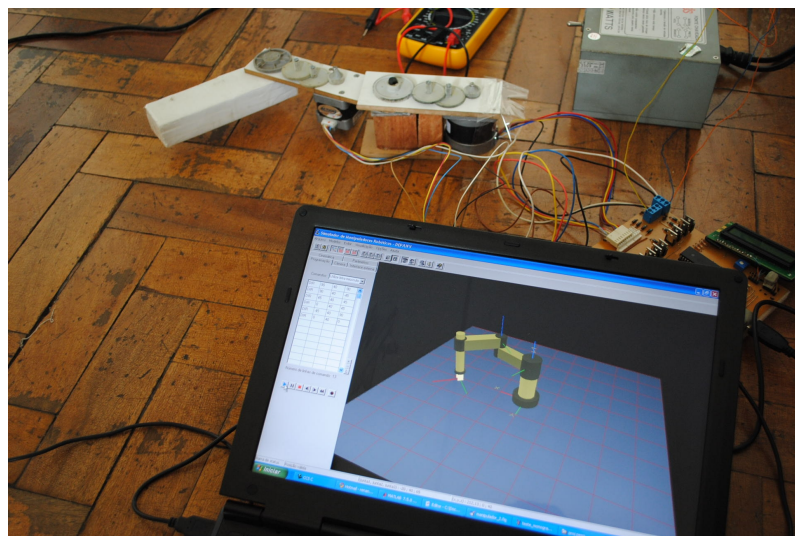


Figura 47 – Sistema em operação, conectado ao software Manipulador

## 4.2 Resultados alcançados com o MatLab

Para demonstrar o desempenho em funcionamento do sistema ligado ao MatLab, gerou-se um caminho, onde este foi executado e comparado visualmente com o resultado esperado. Para comparar o caminho executado com o caminho esperado, realizou-se o experimento da seguinte forma; foi inserida uma caneta na ponta do braço construído. Uma folha de papel branco do tipo A4 foi fixada em um plano perpendicular à caneta, assim o caminho percorrido pelo braço robótico foi marcado sobre o papel.

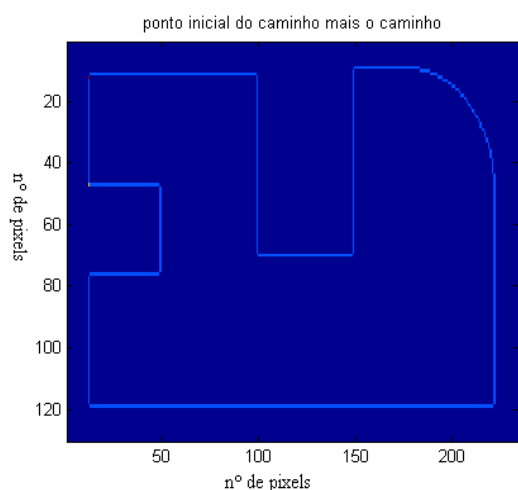


Figura 48- Caminho mapeado a ser seguido.

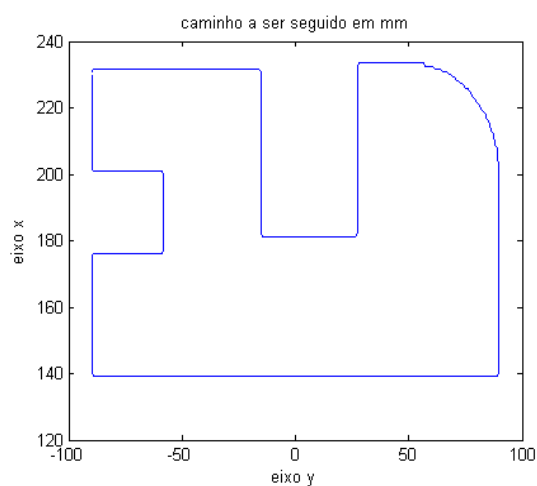


Figura 49 - Caminho a ser seguido em mm.

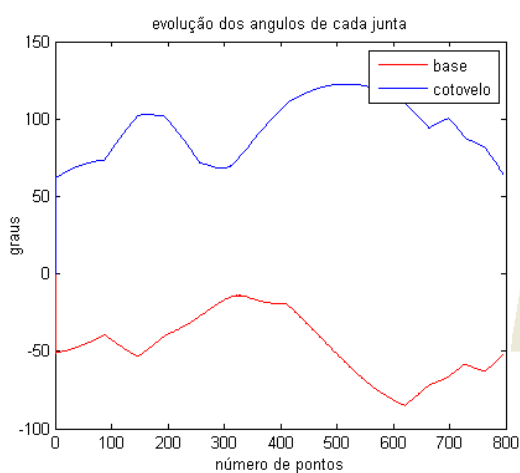


Figura 50 –Evolução dos ângulo das juntas

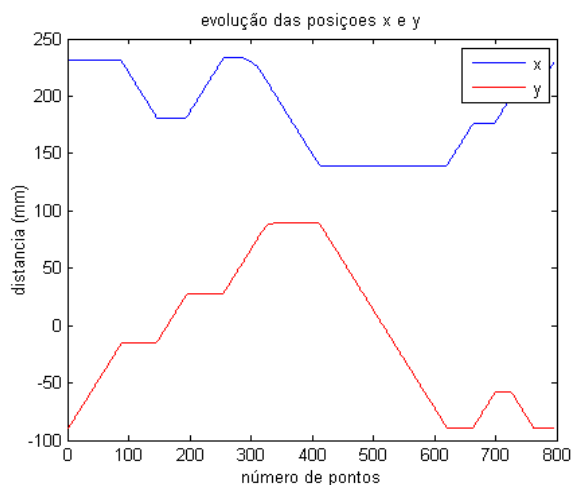


Figura 51 – Evolução das posições X e Y



Pela figura 52 vemos que o braço pode realmente seguir um caminho pré-determinado, dentro das limitações de precisão do nosso modelo. Visto que todo o sistema mecânico é feito a partir de material reutilizado, onde nada foi feito sob medida, pode-se admitir que obtive-se um ótimo resultado em relação ao caminho traçado, onde este está extremamente bem definido mas com algumas distorções na forma, advindas principalmente de folgas mecânicas, erro nas medições dos parâmetros da plataforma (como valor da redução e comprimento dos elos) e aproximações feitas no programa do microcontrolador em relação aos ângulos das juntas.

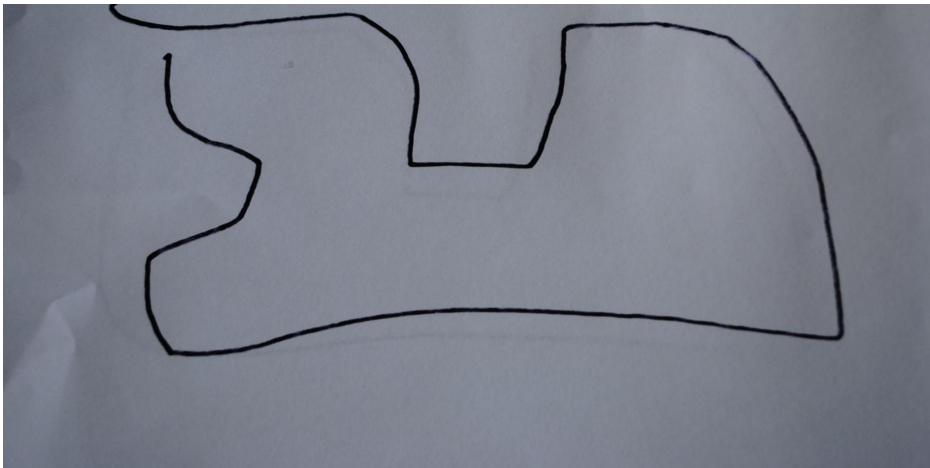


Figura 52 – Caminho percorrido pelo braço robótico

## 5. Conclusões e propostas futuras

Este projeto apresentou como tema a construção de uma plataforma simuladora para manipuladores robóticos, sendo esta plataforma reprogramável. O projeto utilizou somente componentes de baixo custo, e de alta robustez, com um ótimo desempenho. Mesmo utilizando motores de passo de baixa precisão o resultado final foi bom, devido aos métodos de programação utilizados e ao sistema mecânico, que devido à redução mecânica conseguiu contornar as limitações de precisão dos motores pouco precisos.

O sistema apresentou algumas desvantagens quando utilizado juntamente com o software “Manipulador”, mas a plataforma simuladora provou ser versátil não ficando limitada a este software, podendo o sistema ser comandado pelo MatLab com exatidão. O controle de caminho se mostrou de bastante valia e preciso, ficando limitado pela natureza da plataforma física, que foi construída com materiais impróprios para fins de exatidão como este.

A plataforma também provou ser de grande valia, pois traz para a realidade o sistema robótico virtual, tornando a visualização da realidade muito mais agradável e interessante que uma simples simulação de computador.

Como proposta futura seria de grande valia a construção do sistema mecânico para um braço de três graus de liberdade (uma vez que a plataforma já está equipada para três motores), além da reconstrução do manipulador planar com materiais próprios para este fim e feitos especialmente para ele, isso levaria a um grande aumento da precisão e consequentemente da qualidade do resultado final.


## 6. Referências bibliográficas

- [1]. MORAES, C. C. e CASTRUCCI, P. L. *Engenharia de Automação Industrial*. LTC. RJ. 2001.
- [2]. DORF, R. C. e BISHOP, R. H. *Sistemas de Controle Modernos*. LTC. RJ. 2001.
- [3] TOCCI, Ronald J., **Sistemas digitais: princípios e aplicações**. 8.ed.- São Paulo: Prentice Hall, 2003.
- [4] SOUZA, David José de, **Desbravando o PIC: ampliado e atualizado para PIC 16F628A** --11.ed.--São Paulo: Érica 2007.
- [5] Microchip PIC18F2455/2550/4455/4550 Data Sheet. Disponível em :  
[http://www.microchip.com/stellent/idcplg?IdcService=SS\\_GET\\_PAGE&nodeId=1999&ty=&dt=&section=&NextRow=&ssUserText=18f4550](http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1999&ty=&dt=&section=&NextRow=&ssUserText=18f4550)
- [6] AXELSON, J. **USB Complete**. 3ª ed. Madison: Lakeside Research, 2005.
- [7] Universal Serial Bus Specifications 2.0. Disponível em:  
< <http://www.usb.org/developers/docs/> >
- [8] Motor de Passo . Disponível em :  
[http://pt.wikipedia.org/wiki/Motor\\_de\\_passo](http://pt.wikipedia.org/wiki/Motor_de_passo)
- [9] TORO, Vicent Del . **Fundamentos de Máquinas Elétricas** . 1º Edição – LTC -
- [10] Action Motors Datasheet shared\_sm1.8-b disponível em :  
<http://www.actionmotors.com.br>
- [11] BOYLESTAD, Robert L; NASHELKY, Louys. **Dispositivos eletrônicos e teoria de circuitos**. 8ª edição – São Paulo Pretince Hall, 2004.
- [12] SANTOS, Dêdison Moura . **Software Simulador Didático Aplicado a Robótica** . UFV 2008.
- [13] Nota de Aplicação AN956: Migrando Aplicações RS-232 UART para USB com mínimo impacto sobre programa no PC. Disponível em: <  
[http://www.microchip.com/Stellent/idcplg?IdcService=SS\\_GET\\_PAGE&nodeId=1824&appnote=en021631](http://www.microchip.com/Stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1824&appnote=en021631)> Acesso em: 01 out. 2008.
- [14] Drivers USB e Manual de uso Disponível em :  
[http://picmania.garcia-cuervo.com/USB\\_0\\_Desencadenado.php](http://picmania.garcia-cuervo.com/USB_0_Desencadenado.php)

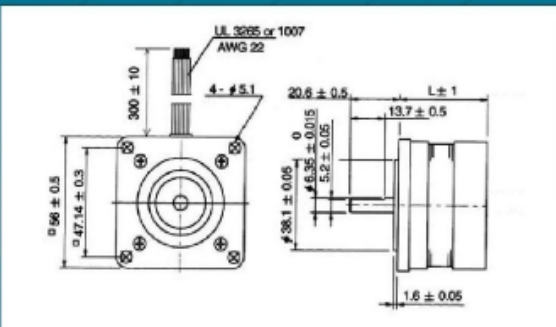
## 7. Anexos

### 7.1 Anexo-1, Datasheets dos motores de passo.

**MOTOR DE PASSO - SM1.8 - NEMA 23**



**DIMENSIONAL**



**ESPECIFICAÇÕES**

MODELO	ESQUEMA DE LIGAÇÃO	TENSÃO NOMINAL (V <sub>dc</sub> )	CORRENTE (A/fase)	RESISTÊNCIA (OHMS/fase)	INDUTÂNCIA (mH/fase)	HOLDING TORQUE (kgf.cm)	DETENT TORQUE (gf.cm)	INÉRCIA DO ROTOR (g.cm <sup>2</sup> )	PESO (g)	L L (mm)
SM1.8B1-SE	UNIPOLAR	5,0	1,0	5,0	9,8	5,0	300	120	420	51
SM1.8B25B-SE <sup>1)</sup>	UNIPOLAR	12,0	0,6	20,0	30,0	5,0	400	120	420	51

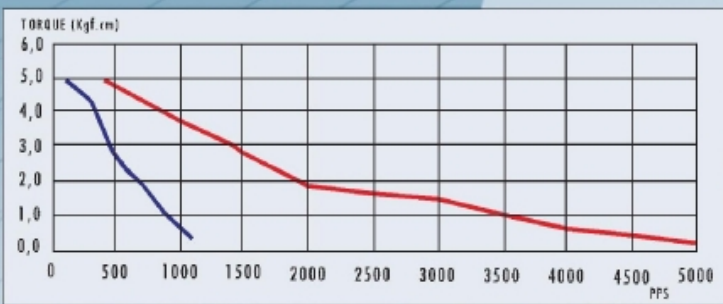
<sup>1)</sup> Sem charrido

Ângulo de Passo	1.8 °
Passos / Volta	200
Temp. Ambiente	-10°C a 50°C
Resistência da Isolação	100 Ohm / DC500V
Rigidez Dielétrica	Ac 500V / 1 min.
Classe de Isolação	B
Folga Radial (máx.)	0,03mm / máx. - Carga=500g
Folga Axial (máx.)	0,08mm / máx. - Carga=700g

**SEQUÊNCIA DE FASES**

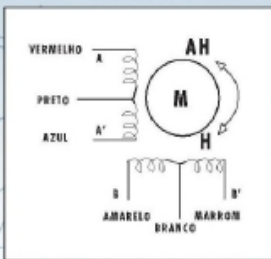
Passo	A	B	A'	B'	Branco	Preto
0	+	+			+	+
1		+	+		+	+
2			+	+	+	+
3	+			+	+	+


Rotação horária - visto lado acionamento



Especificações sujeitas a alteração sem aviso prévio.

— SM1.8-B1-SE  
— SM1.8-B25B-SE

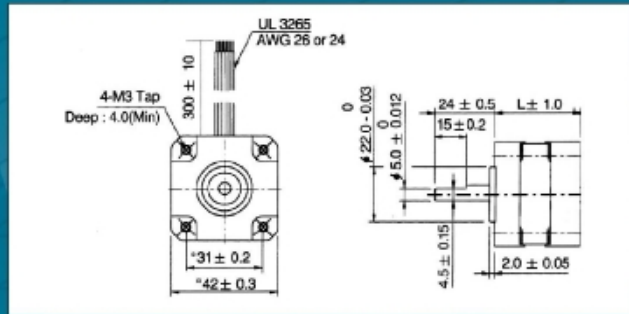
**ESQUEMA DE LIGAÇÃO**




**TEL. 55-19 3936-2133 - FAX 55-19 3936-2137**  
**e-mail: actionmotors@actionmotors.com.br • www.actiontechnology.com.br**

# MOTOR DE PASSO - SM1.8 - NEMA 17

## DIMENSIONAL



## ESPECIFICAÇÕES

MODELO	ESQUEMA DE LIGAÇÃO	TENSÃO NOMINAL (V,dc)	CORRENTE (A/Fase)	RESISTÊNCIA (OHMS/Fase)	INDUTÂNCIA (mH/Fase)	HOLDING TORQUE (Kgf.cm)	DETENT TORQUE (Kgf.cm)	INÉRCIA DO ROTOR (g.cm <sup>2</sup> )	PESO (g)	L (mm)
SM1.8-A1740I-SE <sup>(1)</sup>	UNIPOLAR	4,9	0,8	6,4	5,2	2,4	0,20	52	250	40
SM1.8-A1740CH-SE	BIPOLAR	6	1,2	5	4,7	3,5	0,30	52	250	40
SM1.8-A1734C-MN <sup>(2)</sup>	BIPOLAR	12	0,40	30	27	2,6	0,12	38	200	34
SM1.8-A1736Q-SE	UNIPOLAR	12	0,40	45	28	1,2	0,06	48	220	35,5
SM1.8-A1733-FL <sup>(2)</sup>	UNIPOLAR	7,5	0,75	10	16,5	3	—	35	220	33

<sup>(1)</sup>Eixo sem chaveta <sup>(2)</sup>Eixo 3x13(mm)

Ângulo de Passo	1.8°
Nº de Fases	2
Passos / Volta	200
Temp. Ambiente	-10°C a 50°C
Resistência da Isolação	100 Ohm / DC500V
Rigidez Dielétrica	Ac 500V / 1 min.
Classe de Isolação	B
Folga Radial (máx.)	0,03mm / máx. - Carga=400g
Folga Axial (máx.)	0,08mm / máx. - Carga=500g



Especificações sujeitas a alteração sem aviso prévio.

## SEQUÊNCIA DE FASES

Rotação horária - vista lado acionamento

Modelo SM1.8-A1740I/A1736Q-SE

Passo	A	B	A'	B'	Branco	Preto
0	+	+			+	+
1		+	+		+	+
2			+	+	+	+
3	+			+	+	+

Modelo SM1.8-A1733-FL

Passo	A	B	A'	B'
0	+		+	
1		+	+	
2		+		+
3	+			+

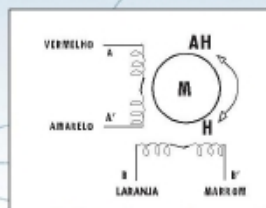
Modelo SM1.8-A1734C-MN

Passo	A	B	A'	B'
0	+	+		
1		+	+	
2			+	+
3	+			+

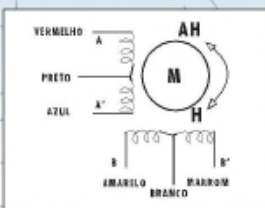
- SM1.8-A1734C-MN
- SM1.8-A1736Q-SE
- SM1.8-A1740I-SE
- SM1.8-A1740CH-SE
- SM1.8-A1733-FL

## ESQUEMA DE LIGAÇÃO

Modelo SM1.8-A1734C-MN



Modelo SM1.8-A1740I/A1736Q-SE



SM1.8-A1740CH-SE e SM1.8-A1733-FL sob consulta



TEL. 55-19 3936-2133 - FAX 55-19 3936-2137  
e-mail: actionmotors@actionmotors.com.br • www.actiontechnology.com.br