

## 5 Sistema Experimental

Este capítulo apresenta o sistema experimental utilizado e é composto das seguintes seções:

- 5.1 – Robô ER1: Descreve o robô utilizado. É dividida nas seguintes sub-seções:
  - 5.1.1 – Hardware;
  - 5.1.2 – Software;
  - 5.1.3 – Interface com o robô;

### 5.1. Robô ER1

O robô utilizado neste projeto foi construído com o kit comercial “*ERI Personal Robot System*” da empresa *Evolution Robotics*<sup>TM</sup>. O robô pode ser visto na Figura 5-1 em diferentes vistas.



Figura 5-1: ER1 – Personal Robot System

O ER1 difere da maioria das plataformas robóticas por não possuir processamento embarcado. Seu conceito é de ser uma plataforma móvel na qual pode-se colocar um *notebook* que através de programa proprietário pode controlar o robô. Tanto seus motores, quanto câmera e sensores são conectados ao computador através de interface USB. Como o processamento do robô é feito através de um *notebook*, consegue-se poder muito superior a sistemas micro-controlados, sendo o processamento limitado ao computador utilizado.

Também faz parte do conceito deste kit ser modular, pois através de acessórios é possível construir diferentes configurações do robô, além de poderem ser adicionados alguns periféricos, tais como: garra; sensores infra-vermelhos; câmeras; e outros através de entradas analógicas e digitais.

O ER1 tem sido usado em alguns centros de pesquisa e também em universidades como ferramenta de ensino. [45] apresenta este kit como uma plataforma comercialmente acessível que por ser composta de uma série de periféricos, ressalta a natureza modular do robô, e reforça a idéia de que um robô é um conjunto composto por computador, atuadores e sensores, sendo recomendado como ferramenta para experimentação e aprendizado. Além disso, também apresenta o uso do ER1 em sala de aula sendo usado para implementação de algoritmos de localização. Outros também apresentam este robô sendo utilizado em pesquisas como pode ser verificado em [45-47].

A próxima seção detalha o hardware do robô ER1.

### 5.1.1. Hardware

O kit do ER1 é composto de:

- Chassis: Vigas de perfil x de alumínio, conectores de plástico, porcas e placas laterais em alumínio;
- Sistema de acionamento (*Drive System*): Mecanismo pré-montado sobre placas de alumínio com dois motores de passo, rodas de 4 polegadas de diâmetro, correias e polias. O kit também contém um rodízio de 360°;
- Módulo de força: Possui bateria recarregável de 12V e 5,4A-h com fusível interno para proteção. Tem durabilidade de 2,5

horas ao ser utilizada com o robô. Tem duas saídas de 12V DC não reguladas e uma entrada para o carregador de bateria;

- Módulo de controle: Eletrônica que faz a interface dos motores e I/O com o computador. Contém: Conector USB *Slave*; Entrada DC para conexão com módulo de força; Sub-conector de 25 pinos D para I/O digital com 8 entradas e 8 saídas; Sub-conector de 25 pinos D para 15 entradas analógicas;
- Carregador de bateria, conector USB e conector de força;
- *Web* Câmera com resolução de 80x60 até 640x800;
- Programa “*ERI Robot Control Center*”;

As peças das quais o kit é composto podem ser vistas em Figura 5-2, Figura 5-3, Figura 5-4 e Figura 5-5.



Figura 5-2: Vigas em x, placas laterais e saco com conectores e porcas



Figura 5-3: Rodízio e conjunto de motor com roda



Figura 5-4: Carregador de bateria, cabo de força e conector USB

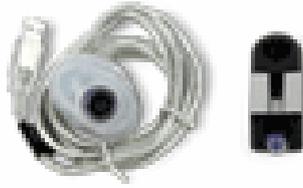


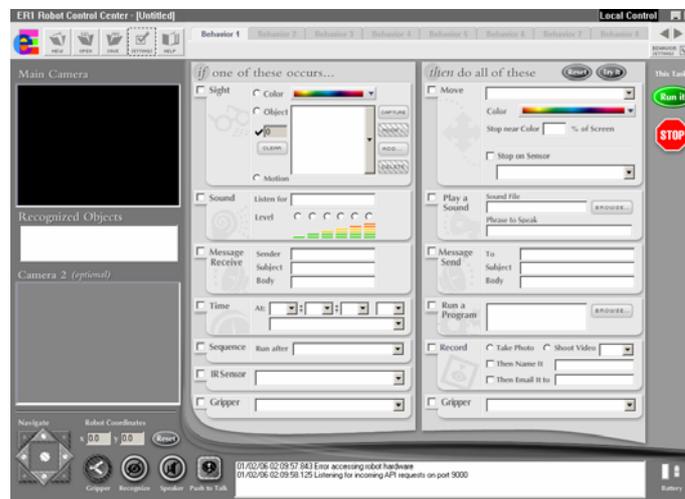
Figura 5-5: Web Câmera utilizada

Além do robô, também foi utilizado o acessório “*ER1 IR Sensor Pack*” que traz três sensores infra-vermelhos com conexão USB.

A próxima seção detalha o programa que é utilizado para controlar o robô.

### 5.1.2. Software

O programa que acompanha o ER1 é chamado de “*ER1 Robot Control Center*”, ou simplesmente RCC, e pode ser visto na Figura 5-6. É através deste programa que é feita a comunicação entre o computador e o robô.

Figura 5-6: *ER1 Robot Control Center*

O programa RCC é usado para controlar e receber *feedback* do robô. Sua interface permite configurar comportamentos através de conjuntos de ordens do tipo “SE condição ENTÃO ação”. Esta interface é limitada, não sendo possível utilizar o programa RCC de modo isolado para criar os algoritmos desejados e

trabalhar com o ER1. Porém, o programa pode trabalhar com três configurações distintas:

- ER1 controlado remotamente utilizando-se outro RCC em uma máquina remota conectada ao computador que controla o robô por rede;
- ER1 controlado através de comportamentos configurados no próprio programa;
- ER1 controlado remotamente através do protocolo TCP/IP;

A terceira configuração apresentada é uma solução para as limitações do programa. Podendo-se comunicar com o RCC através de TCP/IP torna-se possível controlar o robô através de um programa externo.

A seção seguinte explicará como foi feita a interface entre as aplicações desenvolvidas no presente projeto e o robô.

### **5.1.3. Interface com o Robô**

Como visto na seção 5.1.2, é possível se comunicar com o RCC através de TCP/IP. Essa comunicação é feita através de uma API (“*Application Programming Interface*”) por linha de comando que permite a criação de programas que enviem comandos diretamente para o robô. Também se pode usar esta API para coletar informação sensorial do robô.

Os comandos e o feedback do ER1 são enviados através de *sockets* TCP. Desta maneira, é possível comandar o ER1 através de qualquer linguagem de programação ou plataforma que suporte TCP/IP tais como *Visual Basic*, C, C++, Java, Python e Matlab.

Esta API também suporta acesso às 15 portas analógicas e 16 portas digitais do ER1, possibilitando a adição de hardware adaptado para ser controlado pelo ER1.

Há, porém o inconveniente de ser obrigatório que o programa RCC esteja rodando para que a API funcione, pois na verdade a comunicação TCP/IP é feita entre o programa externo e o programa RCC e não diretamente com o robô. A

interface entre o robô e o computador é feita pelo RCC. Toda esta comunicação está exemplificada no diagrama apresentado na Figura 5-7.



Figura 5-7: Interface com ER1

Para poder controlar o ER1 via TCP/IP é necessário configurar o programa RCC. Isto é feito através da aba “*Remote Control*” dentro da caixa de diálogo “*Settings*”. A opção de controle remoto é ativada através da opção “*Allow API Control of this instance*”. Para melhor entendimento do procedimento veja a Figura 5-8.

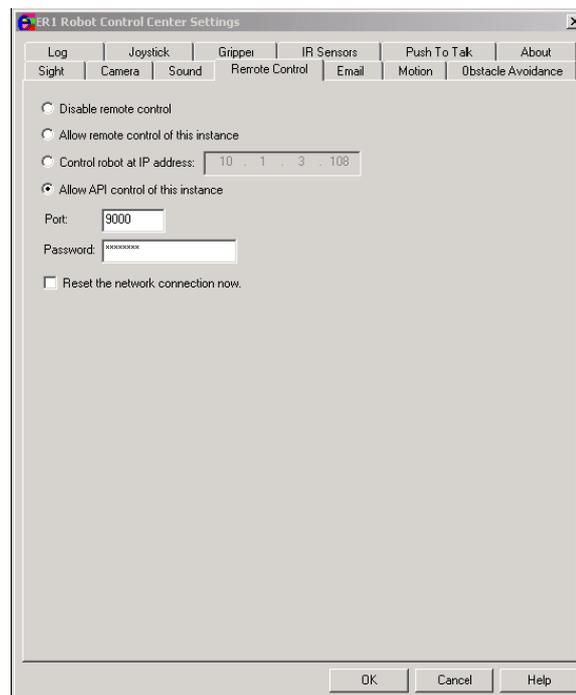


Figura 5-8: ER1 Robot Control Center Settings

Através da caixa “*Settings*” e aba “*Remote Control*” também é possível configurar a porta pela qual é estabelecida a comunicação com o programa. Não é possível manter controle através da API e da GUI do RCC ao mesmo tempo.

Também é necessário desligar o uso da câmera pelo RCC para permitir que esta seja acessada exteriormente. Isto é feito na caixa “*Settings*” e aba “*Camera*” desmarcando-se a opção “*enable camera usage*”. O acesso à imagem da câmera é feito através dos *drivers* do *Windows*.

A API apresenta, dentre outros, os seguintes comandos que interessam para o projeto aqui descrito e que podem ser enviados em modo texto através de TCP/IP para a porta configurada:

- “*move*”: Faz com que o robô se movimente. Só é possível fazer com que o robô ande em linha reta ou rode sobre seu eixo central. A API responde a este comando após este ter sido verificado, mas antes de que um movimento tenha sido concluído. Isto quer dizer que caso outro comando “*move*” seja dado, o comando anterior será interrompido e o novo executado. Há porém como se determinar se o robô terminou um movimento através do comando “*events*”;
- “*stop*”: Manda o robô parar;
- “*position*”: Retorna à posição  $(x,y)$  atual do robô relativa à posição em que o robô estava quando o programa RCC iniciou. Esta função também deveria retornar o ângulo que indica a direção do robô, porém este recurso não funciona devido a um problema no programa;
- “*sense*” e “*ir*”: Estas funções permitem configurar os sensores IR e ativar o feedback dos mesmos. Porém, este feedback é feito através de outra função, a função “*events*”;
- “*events*”: Esta função faz com que qualquer feedback do robô seja enviado como uma mensagem de retorno na comunicação TCP/IP. O tratamento de sensores e do estado de movimentação do robô é feito com este comando. Para se tratar um feedback é necessário portanto chamar o comando “*events*” e ficar “escutando” a comunicação TCP/IP até se obter alguma resposta;
- “*exit*”: Faz com que o programa RCC seja terminado. O *socket* pelo qual a conexão a API é feita também é fechado;

Como foi apresentado, todo feedback do robô implica no processo de se executar o comando “*events*” e ficar esperando alguma resposta, para então, fazer o tratamento desta resposta. Infelizmente, este processo não é simples e muitas

vezes é extremamente lento. Para se saber o estado dos sensores IR por exemplo, é necessário ficar chamando e esperando alguma resposta dos sensores, sendo que esta resposta pode não acontecer.

O controle do robô através da API dada é limitado. Algumas de suas limitações e problemas são:

- Movimento restrito a rotações e trajetórias em linha reta. Não é possível acessar os motores do robô diretamente para se controlar as velocidades individuais de cada roda. Portanto, é impossível descrever trajetórias curvilíneas.
- Não há acesso direto às contagens referentes aos *encoders* de cada roda. Só é possível saber a posição do robô através das coordenadas  $(x,y)$ . Deveria ser possível adquirir também a coordenada  $\theta$  relativa à direção do robô, porém descobriu-se que a API apresenta um problema que não foi nem será resolvido que impossibilita se adquirir esta coordenada (informação dada pelo suporte da empresa *Evolution Robotics*<sup>TM</sup>).
- A resposta ao comando “*move*” apresenta discrepâncias em relação às distâncias percorridas ou giradas em relação às distâncias pedidas. Isto significa que é necessário estar checando a posição do robô para se corrigir possíveis erros. Porém, isto não é possível de ser feito quando o robô gira porque não é possível se obter a coordenada de giro do robô como já discutido. Estes problemas não deveriam ser esperados já que o robô trabalha com motores de passo e *encoders*, mas infelizmente acontecem.
- O acesso ao feedback do robô é lento e complicado como já descrito anteriormente;
- A resposta aos comandos não é imediata fazendo com que a comunicação com o robô possa ser lenta em vários momentos;

O capítulo seguinte passará a utilizar as teorias apresentadas nos capítulo 2, 3 e 4 para construir o algoritmo de navegação e exploração proposto, tal como suas componentes.