Application of WLAN to the control of mobile robots

A.F.T. Winfield, O.E. Holland

Presented by : Priyanka Lingayat
Basic idea

- Use of WLAN technology and IP for communication and control of distributed mobile robots
Hardware design considerations

• Implementation of proprietary microcontroller based on IEEE PC/104 standard.

• System with 386/486 processor, RAM, BIOS, floppy, IDE interfaces, socket for flash EPROM or solid state disk device.

• Support for MS-DOS or Linux OS

• PCMCIA WLAN network interface card
Hardware block diagram

- PC/104 PCMCIA Adapter
- 386/486 PC/104 Processor and solid-state disk
- PC/104 Bus
- Wireless LAN
- Robot Motherboard
  - Motor drivers
  - Power management
  - Sensor drivers
- Motors
- Battery
- Sensors
OS considerations

• MS-DOS

• Linux
  -- Multi-tasking, useful in a robot controller
  -- Linux integrates TCP/IP networking as standard
  -- Open source code, internal operation of Linux is transparent, unlike proprietary operating systems
  -- Real time extensions to the Linux kernel exist which allow time-critical tasks, or interrupts, to be precisely scheduled
Internet Protocols

• ‘Protocol stack', needs to be present at both ends of the communications link; in our case the mobile robot and the desktop controller.

• Any message from the applications layer at one end of the link (an instruction from the control PC for the robot to move, for instance), is transferred down the protocol stack at the originating end of the link, then across the physical network interface (in our case the wireless connection), and finally up the protocol stack at the destination (i.e. the robot).

• Adoption of TCP/IP so that mobile robots could send and receive messages to and from anywhere with Internet connectivity.
System Software design

• Mobile robots with WLAN interfaces offers two distinct opportunities:
  -- enabling communication between robots
  -- enabling communication between robots and a base station.
System Software design continued..

• System: Client – Server paradigm
  -- the desktop base station at the hub of the WLAN acts as the ‘client’ software
  -- the application software in each mobile robot acts as a ‘server’

• Each robot(server) must have its own unique IP address.
Multi Robot wireless local area network
Command Protocol

• The use of the reliable TCP communications protocol is sufficiently robust that a command can be sent to a robot (to start a motor with a given speed, for instance), without the need for the robot to respond with an acknowledgement message.

• For ease of debugging, and to simplify command parsing in the Robot Server software, each command from controller to robot is encoded as a fixed length ASCII numeric string.

• The Robot Server software will be able to determine from the command value whether there will be parameter values and how many.
Controller to robot command values

<table>
<thead>
<tr>
<th>Command</th>
<th>Value</th>
<th>Parameters</th>
<th>Response</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>NULL</td>
<td>00</td>
<td>None</td>
<td>Yes</td>
<td>Keep alive</td>
</tr>
<tr>
<td>STATUS</td>
<td>04</td>
<td>None</td>
<td>Yes</td>
<td>Read status switches</td>
</tr>
<tr>
<td>MOTOR</td>
<td>06</td>
<td>Two (left, right)</td>
<td>No</td>
<td>Set motor speeds</td>
</tr>
<tr>
<td>LEDS</td>
<td>07</td>
<td>One (data)</td>
<td>No</td>
<td>Set LED values</td>
</tr>
</tbody>
</table>

Example commands:
06 00100 00100 – forward at speed of 100
06 00050 00050 – rotate on the same spot
06 00000 00000 – all stop
Robot Server software

• Application code for the robot server written in C

• Has 3 distinct phases:
  -- ‘initialization’ phase in which the sockets and associated data structures are created
  -- ‘listening’ phase, in which the server is waiting for a connection (SVC) to be established by the network controller (client)
  -- quasi-infinite loop that waits for commands from the client, then executes them when they are received.
Pseudo code

// PHASE 1, initialisation.
// Get a TCP stream socket, for listening.
socket (...);
// Get the local (our) IP address.
gethostid (...);
// bind our local IP address & port number to the socket.
bind (...);
// “passively open” the socket
listen (...);
// PHASE 2, listening.
// wait for Client controller to ‘connect’.
accept (...);
// now close listening socket (not accepting any more connections).

close (...);
// set the connected socket to non-blocking.
ioctl (...);
// PHASE 3, Loop checking for received commands from
Client Controller.
while ()
{
  // Execute a non-blocking read.
  if read (...) 
  {
    // We’ve received a command, so go and act on it.
    // first parse the command value
    cmd = strtol (...);
    switch (cmd)
    {
      case NULL: // echo null message
        send (...);
      case STATUS: // read status switches
        case MOTORS: // set motor speeds
        }
  } else // nothing has been received, so perform local control here.
} // end of while loop
Robot Client software

• The Client software, running on the desktop PC at the hub of the WLAN, consists of ANSI compliant C code.

• To implement TCP/IP compliant networking, the Client code makes use of the well-known Windows implementation of the Berkeley sockets API known as Winsock.
Robot Client Human Machine Interface (HMI)

Basic requirements for HMI:

-- the means to configure and manage IP addresses for the fleet of mobile robots

-- the means to initiate connections (virtual circuits) with each mobile robot, and to monitor the connection status of each virtual circuit

-- the means to send commands to any particular mobile robot

-- the means to monitor both outgoing (command) and incoming (response) messages.
Fig. 5. HMI main window.
References


Questions ? ? ?