RoboCup2004
Rescue Robot League Competition
Lisbon, Portugal
June 27 – July 5, 2004
www.robocup2004.pt

RoboCupRescue - Robot League Team Resquake, Iran

Ehsan Aboosaeedan[1], Ali Jazayeri[2] ,Arash Kalantari[3] , Siamak Kooshayee[4] , Ehsan Mihankhah[5] , Hesam Semsarilar[6]

[1] K.N.Toosi University Of Technology
Seyed Khandan Bridge, Shariati St.,
Tehran , Iran +98 21 8600075
ehsanaboo@hotmail.com

[2] K.N.Toosi University Of Technology
Seyed Khandan Bridge, Shariati St.,
Tehran , Iran +98 21 8600075
sajazayerim@yahoo.com

[3] K.N.Toosi University Of Technology
Daneshgah Bulv., Forth Square Of TehranPars,
Tehran , Iran +98 21 8600075
Arash1362@yahoo.com

[4] K.N.Toosi University Of Technology
Daneshgah Bulv., Forth Square Of TehranPars,
Tehran , Iran +98 21 8600075
s_weasel_k@yahoo.com

[5] K.N.Toosi University Of Technology
Seyed Khandan Bridge, Shariati St.,
Tehran , Iran +98 21 8600075
ehsanmihankhah@yahoo.com

[6] K.N.Toosi University Of Technology
Daneshgah Bulv., Forth Square Of TehranPars,
Tehran , Iran +98 21 8600075
hesam2k@yahoo.com

Abstract.

Resquake is the representative of K.N.Toosi University of Technology in International RoboCup Rescue 2004. The set consists of three Robots and three small moving stations carrying the positioning system. This paper explains different parts of a sample robot which is not any of the above robots but we have implemented the main ideas we found critical for a rescue robot to have, in this robot. The final set contains a big robot which can raise a camera about two meters and has differential steering locomotion system which enables the robot to climb stairs and ramps. The second robot is a very tiny robot with camera lifter and two speed gearbox and differential steering locomotion system. Both robots have camera, gas sensor, light sensor, heat sensor, motion detection system, sound detection system, wireless LAN and transceivers and other necessary equipments described in details later. Third robot is a flying robot which will be only carrying a wireless camera. In this paper we will go through the details of the sample robot and as you will see, almost all the capabilities of the main robots are partially implemented in this sample robot.

Team Members and Their Contributions

• Ehsan Aboosaeedan	Motor Drivers and Power Supply and Electrical Design
• Ali Jazayeri	Microcontroller And Embedded System And Electrical Design
• Arash Kalantari	Mechanical Design And Simulation
• Siamak Kooshayee	Mechanical Design And Implementation
• Ehsan Mihankhah	Software Development And Operator
• Hesam Semsarilar	Mechanical Design And Implementation
• Dr. HemidReza Tghirad	Advisor
• Nima Mokhtarzade	Sponsor

1. Operator Station Set-up and Break-Down (10 minutes)

Everything is packed in a toolbox (the Operator's Laptop and printer and other accessories such as joystick and mouse and a bunch of papers and writing tools and …), and each robot has its own package (Three robots and a set of Positioning system). So we need five people each to carry on box, Operator will not do anything, so we have all the 6 members of the team in setup and Break-Down Time. Each person will setup the parts in the box he is carrying. We estimate the setup time would take 3 minutes and the Break-Down time will take 5 minutes.

Hesam will carry the toolbox. He will plug the Laptop accessories (joystick, mouse, printer, transceivers, USB wireless LAN card, speakers, the mobile phone (for uploading files to the Resquake website), and he will put the papers and writing tools in a places where the operator has gotten used to while practicing before the competition.

Ali is carrying the positioning system. He will setup small robots which will carry the stations to the area. (There are three robots carrying three transceivers and make the triangulation system).

Ehsan (Aboosaeedan) and siamak and Arash will setup the three main robots.

Ehsan (Mihankhah) is the operator and will not do anything.

Break-Down procedure is similar to setup procedure.

2. Communications

Communication part consists of four different parts and four different frequency ranges (including alternative communication system).

First part is wireless LAN cards and access point of IEEE 802.11a class which are working in 8 different switchable channels in 5.4 GHz frequency range. Wireless LAN transmits data and video stream.

Second is the back up system for data transmission, which will be a pair of transceivers operating in 1 GHz frequency range with four switchable channels.

Third one is the video backup system, the video will be sent with a video sender which is working in 1 GHz frequency range with four switchable channels.

Last one is the positioning system. Three transceivers are working in 1GHz frequency range with four switchable channels.

Data and Video Transmission, Software Part:

Here is the point where LAN programming may seem to be the most important and most difficult part. Data and video transmission can be done using VB and VC++.

First let us talk about data transmission codes in VB and VC++

Data transmission using VB:

In VB everything can be done using WinSock control. This control lets us act both as server and client. If we are the server we must specify the port which the control is going to listen to. Let us name the server Winsock Control "ServerSock".

In Form_Load subroutine the following code should be written:

```
Private Sub Form_Load()
     .
     .
     .

   Rem  Port Number is a Long Variable which the Port
   Rem  Number has been assigned to it
     ServerSock.LocalPort = PortNumber
 Rem  Now The Control Should Wait For a Request
     ServerSock.Listen
     .
     .
     .
End Sub
```

Then the control should accept the request from the client. When the client tries to connect to this port the Connection Request event of WinSock control occurs. In this Subroutine we should accept the coming request.

```
Private Sub ServerSock_ConnectionRequest(ByVal requestID As Long)
```

```
    If ServerSock.State <> sckClosed Then ServerSock.Close
    ServerSock.Accept (requestID)

End Sub
```

Now the connection is established and the object can send and receive data. If we want to send data the following code is needed:

```
     .
     .
     .
    Rem StringToBeSent is a String Variable which the data to be
    Rem sent has been assigned to it
     ServerSock.SendData StringToBeSent
     .
     .
     .
```

When data comes to the port which this control is listening to, The Data Arrival event occurs.

```
Private Sub ServerSock_DataArrival(ByVal bytesTotal As Long)
       Rem DataComing Is String Variable whcich the coming
       Rem data will be assigned to it
       ServerSock.GetData DataComing
End Sub
```

Nothing else should be done o the server. Client Part is just as simple as Server part. Let us name the Client WinSock Control "ClientSock" .In Form_Load or subroutine the following code should be written:

```
Private Sub Form_Load()
     .
     .
     .

   Rem  PortNumber is a Long Variable which the Port Number has
   Rem been assigned to it
     ClientSock.RemotePort = PortNumber
   Rem  ServerName is a String Variable which the Server Name
   Rem has been assigned to it
     ClientSock.RemotHost = ServerName
   Rem  Now The Control Should Ask For COnnection
     ClientSock.Connect
     .
     .
     .
End Sub
```

If the server accepts the connection, this control can send and receive data. The code is exactly the same as the code which has been described above for the ServerSock control.

The Operator Interface can be either the client or the server and it makes no difference to choose any computer (the operator's computer or the one on the robot) as server.

Now let's see the code needed for doing the same thing in VC++.

Data transmission using VC++:

If we want to create a data transmitting program in VC++ first we have to activate the "Use Windows Sockets" option in the application wizard. Let's create a MFC Application (EXE) Dialog Based type project.

First we should drive a class from CAsyncSocket Class. Lets Call it "CSocketDriven" .

Then we should identify the window which is going to use Windows Sockets Service. So we add the following function to this Class

```
//  pMyWnd is a pointer to the window which is using Win
//  dows Socket Service
void CSocketDriven::SetParent(CDialog *pWnd)
{
       pMyWnd = pWnd;
}
```

The following Private Virtual Void functions should be added to this Class to respond to the class events

```
//CMultipleSocketsDlg is the name of the Dialog Control which is
using
//the Socket Service
void CSocketDriven::OnAccept(int nErrorCode)
{
       ((CMultipleSocketsDlg*)pMyWnd)->OnAccept();
}

void CSocketDriven::OnConnect(int nErrorCode)
{
       ((CMultipleSocketsDlg*)pMyWnd)->OnConnect();
}

void CSocketDriven::OnSend(int nErrorCode)
{
       ((CMultipleSocketsDlg*)pMyWnd)->OnSend();
}

void CSocketDriven::OnClose(int nErrorCode)
{
       ((CMultipleSocketsDlg*)pMyWnd)->OnClose();
}

void CSocketDriven::OnReceive(int nErrorCode)
{
       ((CMultipleSocketsDlg*)pMyWnd)->OnReceive();

}
```

Now, the Class is ready. We need to define Server Socket and Client Socket of this Class type. Let's name the Client object "Connect1" and the Server object "Listen1" .In the InIt function of the Dialog we should do the initializations. The following function should be added to the dialog class.

```
void CMultipleSocketsDlg::SetInitialValues()
{
  // m_Name1 is a CString Variable which keeps the IP address of
the
  //Server And m_Port1 is a long Variable which
  //keeps the number of communication Port
       m_Name1 = "169.254.145.141";
       m_Port1 = 3000;
       Listen1.SetParent(this);
       Connect1.SetParent(this);
}
```

The following functions trigger the events of the CDrivenSocket objects:

```
void CMultipleSocketsDlg::OnAccept()
{
               Listen1.Accept(Connect1);
}

void CMultipleSocketsDlg::OnConnect()
{
}

void CMultipleSocketsDlg::OnClose()
{
}
void CMultipleSocketsDlg::OnSend()
{
}

void CMultipleSocketsDlg::OnReceive()
{
       char *pBuf = new char[1025];
       int iBufSize = 1024;
       int SocketErr;
       CString MyData;
       SocketErr = Connect1.Receive(pBuf,iBufSize);
       if (SocketErr == SOCKET_ERROR)
              AfxMessageBox("Error Receiving");
       pBuf[SocketErr] = NULL;
       MyData = pBuf;
}
```

Some of the above functions are not doing anything but we have to define them because in the CSocketDriven Class we have called them.

Now, we can choose between being the Client and the Server. The following function prepares the program for being either a client or a server.

```
void CMultipleSocketsDlg::ClientOrServer()
{
       // m_Client is a int variable which selects the
       // program to be either a Client or a Server
       if (m_Client1 == 0)
       {
              Connect1.Create();
```

```
                Connect1.Connect(m_Name1,m_Port1);
        }
        else
        {
                Listen1.Create(m_Port1);
                Listen1.Listen();
}
}
```

Now, Regardless of being Client or Server we can send data using the following function"

```
void CMultipleSocketsDlg::SendData()
{
        CString Dummy;
        // m_Message is a CString Variable which contains the
        //data which is to be sent
        Dummy.Format("%s Port: %d",m_Message,m_Port1);
        if(Connect1.Send(LPCTSTR(Dummy),Dummy.GetLength()) ==
                SOCKET_ERROR)
                AfxMessageBox(Dummy);
}
```

Receive method is clarified in the Socket events.

After making an interface we can run two copies of this program and set one of them Client and the other one Server to send and receive data.

After Talking about data transmission we take a look at Video transmission. Because of using an ActiveX in programs to send video stream and receive it with the same ActiveX control, there will not be much difference between VB and VC++ programs, as we have the same functions available in both environments. So Lets see the VB program keeping in mind that the VC++ program is almost the same. For sending video we need a camera with AV output and a Capture Card (AV signal should be changed to digital ,so we can work with it) or we can use a Camera with USB output(like a Web Cam).The method will actually make no difference in programming, so we tried both systems and result was excellent in both cases. We also need an ActiveX control .We have used VideoCapX Control to send stream and receive frames on the receiver computer.

Video Transmission in VB :

The following Code sends Video on the LAN:

```
Private Sub Form_Load()
    Rem VideoCapX1 is the name of the VideoCapX Control
    Rem which is being used on the form
       VideoCapX1.Connected = True
       VideoCapX1.SetVideoFormat 640,480
       VideoCapX1.Preview = True
       VideoCapX1.ServerMode = True
End Sub
```

On the receiver we need to get frames from the LAN in short intervals using a timer and assign the received value to the picture property of a PictureBox. The following code shows the solution:

```
Private Sub Timer1_Timer()
   Rem HostName is a String Variable which keeps the IP
   Rem Address of the Host and ClientVideoCapX is the name
   Rem of the VideoCapX Control which is used to receive the
     Rem frames sent by the Host
         Picture1.Picture = ClientVideo_
                  CapX.ReceiveFrame(HostName)
End Sub
```

We may also need to take a picture of victim or take picture from an important part of the arenas, so we should be able to capture a picture from the video. To capture a picture from video stream the following code is needed:

```
Picture1.Picture = VideoCapX.GrabFrame
```

Then we will be able to save the picture on hard disk (and as you will see later, we can right it on a CD as a part of report) using SavePicture function :

```
SavePicture  Picture1.Picture,PhysicalMemoryAddress$
```

There is nothing left to do with video stream. The above codes can be similarly written in VC++ and the result will be significantly faster.

There is a subtle point in sending video streams on the LAN. When we increase the resolution we loose more frames in video transmission. To avoid this, we can send frames with lower resolution but higher quality and stretch the received frame in the receiver, doing this we will loose less frames in video transmission.

Resquake Operator Interface now has everything it needs to send data and receive data and video, as far as we are using the default communication system (LAN), but if any problem occurs in the way that we would not be able to get data from robot through LAN or can not send data through LAN, we should be able to switch to backup system. Video backup system will not make any change to the program; the PictureBox should change its input from LAN to Capture Card and do exactly the same thing it was doing before (AV signal comes to the Capture Card with video waves). But data input is a completely different signal coming from the transceiver. Transceiver has serial output and we can get data from serial port using RS232 standard. Data transmission through serial port can be similarly implemented in VB and VC++ as we are taking advantage of Microsoft Comm. Control. To make a serial port ready to transmit data we need to specify the serial port through which the program is going to transmit, then set the baud rate and parity and number of bits and other settings needed for initializing a serial port. Then we open the port and we will be able to send or receive data from port. So, in the Form_Load subroutine we write the following code:

```
Private Sub Form_Load()
```

```
        .
        .
        .
  Rem : PortNumber is an integer variable
    MSComm1.CommPort =  PortNumber
  Rem: Set Baud rate to 1200 , No parity , 8 bit data , 1
   Rem : stop bit
    MSComm1.Settings = "1200,N,8,1"
    MSComm1.PortOpen = True
        .
        .
        .
End Sub
```

Then, by the time we want to send a bit (or a string) the following code can be used:

```
MSComm1.Output = MyChar$
```

And if we want to receive the data coming, the following code solves the problem:

```
Rem : Setting InputLen property to 0 means we want to get
Rem :the whole string
 MSComm1.InputLen = 0
   If MSComm1.InBufferCount Then
        MyString$ = MSComm1.Input
   End If
```

The backup system can now be handled as well as the main system.

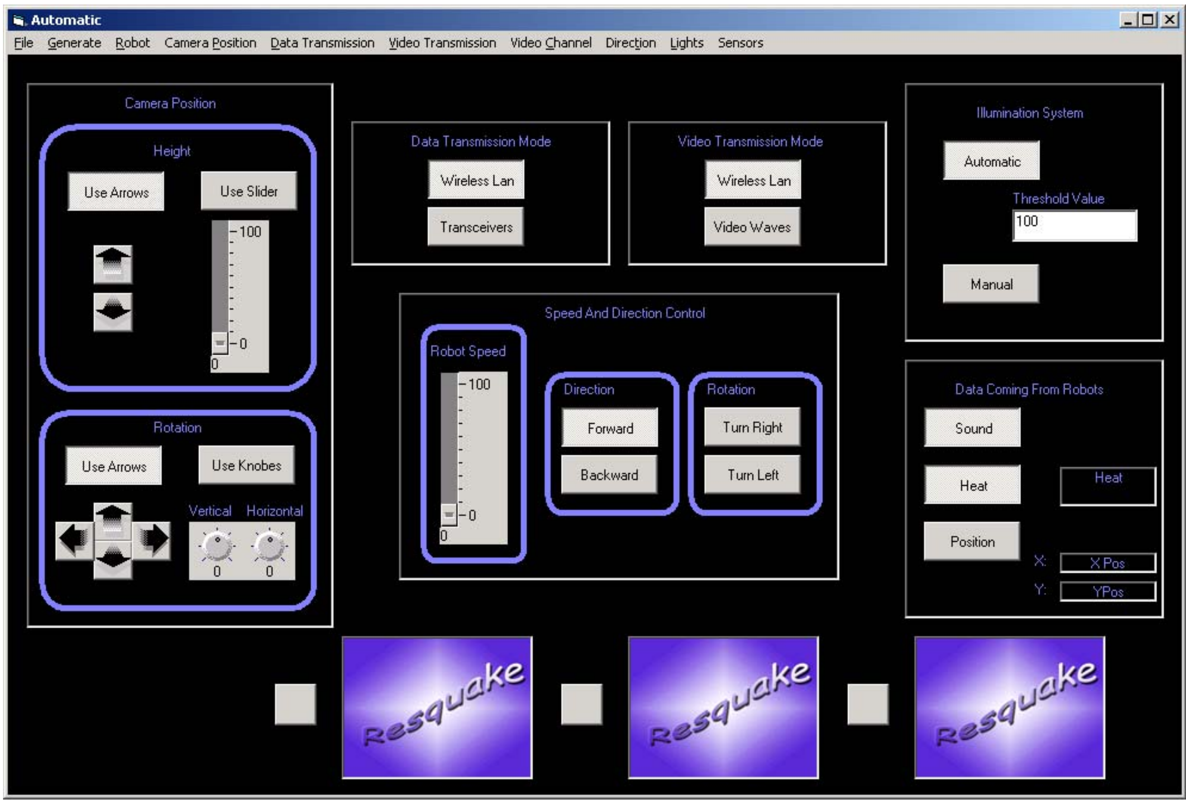
3. Control Method and Human-Robot Interface

We have tried to reduce the operator's responsibilities taking advantage of our positioning system and sonar (Ultrasonic sensors). We have one operator who will control the robots with the joystick and keyboard. Robot is sending data gathered from the sensors to the operator in short intervals and whenever there is a suspicious condition like motion detection (VideoCapX control has a powerful motion detection system which will announce it and the operator finds the sign of life) .Heat and gas sensors are good sensors which we can rely on them when the amount of CO2 increases or a suspicious change in heat is detected. Sonar makes the robot control much easier as it will not let the robot get closer to the surroundings. Taking advantage of sonar let us have the distance from the objects around in the software and gives the operator a better feeling the place where the robot is located.

The operator starts the mission with moving the small positioning station to three different places around the arena. Then the largest robot will go and stay in a place with a widest view and the camera lifter system raises the camera about two meters and gives the operator a general view of the arena then the smaller robots start exploring the arena using the right hand method not to explore a place twice and whenever there is a need up climbing stairs or climb a ramp the big robot lowers the camera and goes to the scene to complete the mission. If there would be the possibility of taking a picture of the arena from top with a flying robot we will do it and the operator will have a picture with low opacity having a grid on it to correct and make needed changes instead of drawing on a plane paper with no reference. This helps a lot in the final map generation.

One part of Resquake software is called Resquake User Interface which is a simple handy environment for the operator to control all the robots (their motors and actuators) and send commands to the robots and of course gather the data coming from sensors and preview video stream. Below the Graphical interface is presented.

Resquake User Interface at this phase is working with keyboard and mouse, but the final program will also have a joystick.



Automatic
File Generate Robot Camera Position Data Transmission Video Transmission Video Channel Direction Lights Sensors
Camera Position
Height
Use Arrows
Use Slider
100
0
Rotation
Use Arrows
Use Knobes
Vertical Horizontal
Data Transmission Mode
Wireless Lan
Transceivers
Video Transmission Mode
Wireless Lan
Video Waves
Speed And Direction Control
Robot Speed
100
0
Direction
Forward
Backward
Rotation
Turn Right
Turn Left
Illumination System
Automatic
Threshold Value
100
Manual
Data Coming From Robots
Sound
Heat
Heat
Position
X: X Pos
Y: YPos
Resquake
Resquake
Resquake

Figure 1

Camera position can be controlled by the following tools:

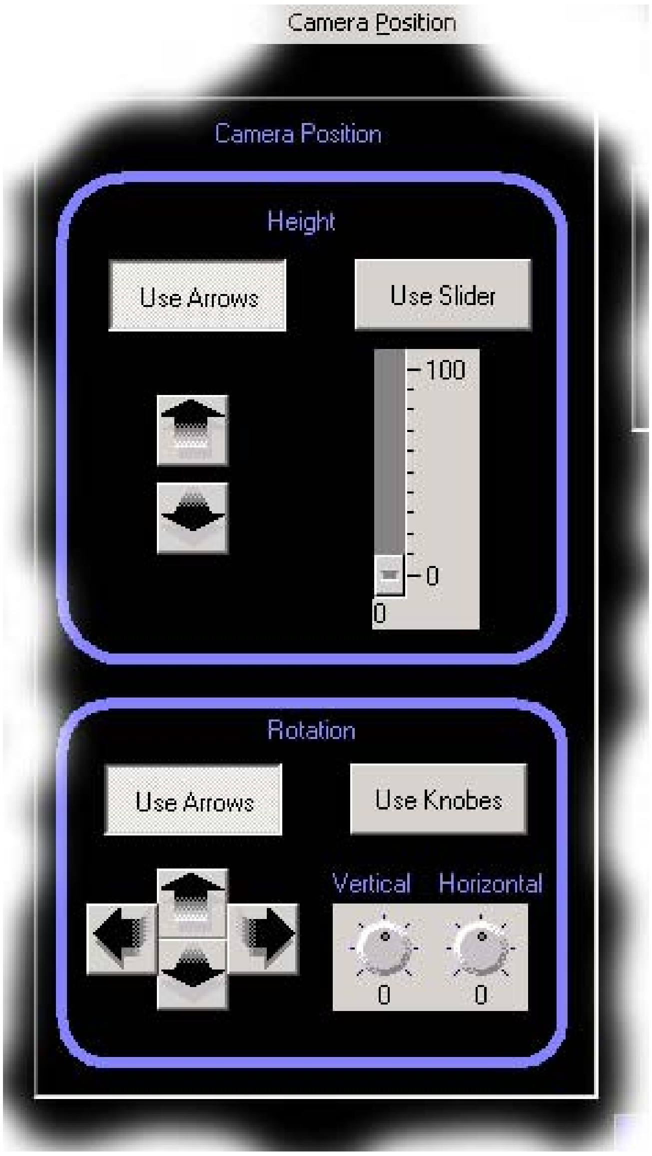


Figure 2

Shortcut keys enable the operator to switch between arrows and sliders and knobs.

Data transmission mode and Video transmission mode can be switched between by the following tools:

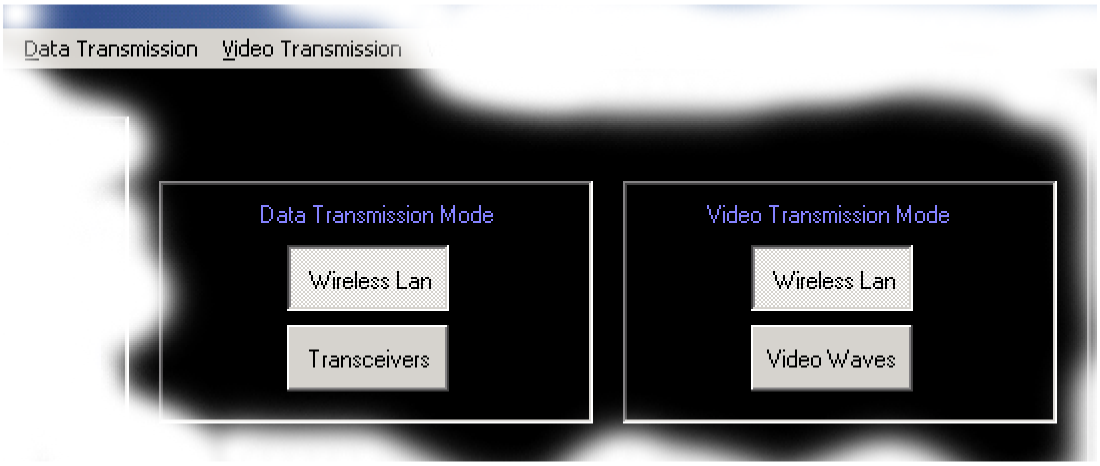


Figure 3

Illumination system can be switched between Automatic and Manual. In Automatic mode the lights will turn on whenever the environment light is less than the threshold value.

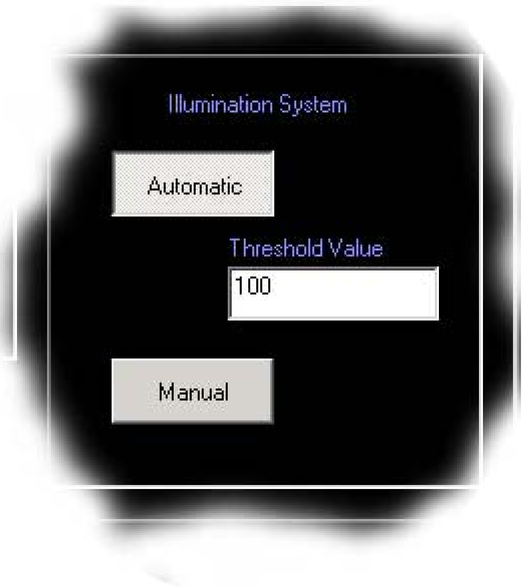


Figure 4

Speed and Direction of the robot can be controlled by the following tools:

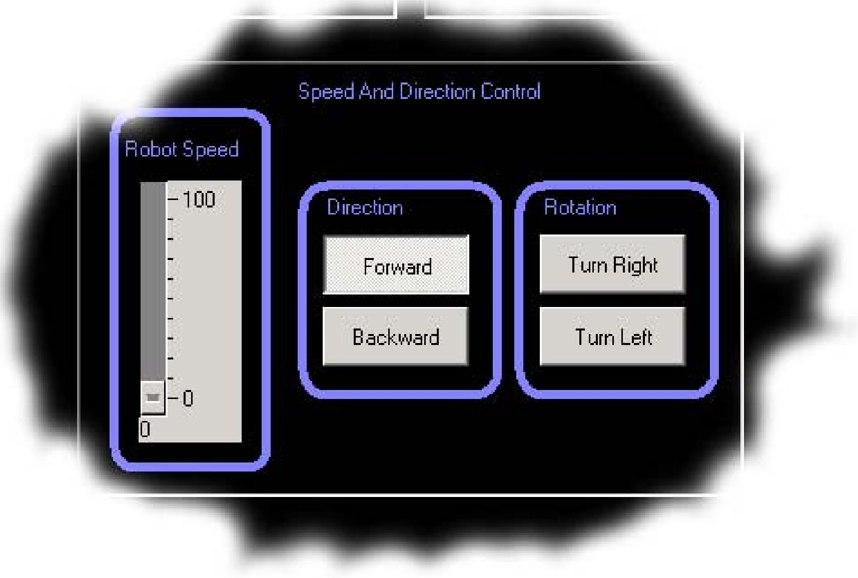


Figure 5

Operator can choose what information he needs to receive from robot by the following tools:

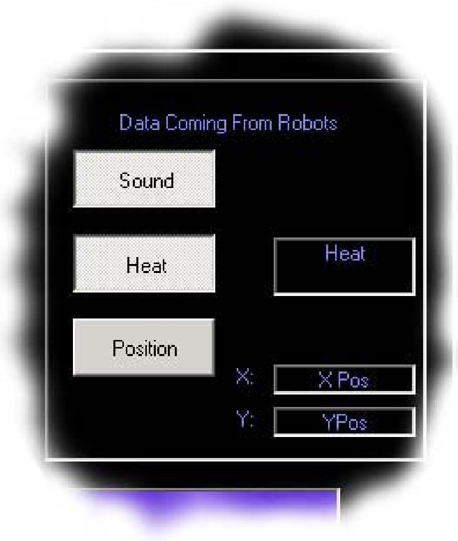


Figure 6

Operator can see the video coming from all robots and can see the large video screen by double clicking on each of small videos.

Figure 7

Operator can switch between robots by pressing F1 or F2 or F3 and can see the large video coming from each robot by pressing Ctrl+F1 or Ctrl+F2 or Ctrl+F3.

The code of the main form interface and the relation of the objects and their properties are given below:

```
  VERSION 6.00
  Object = "{912FB004-DD9A-11D3-BD8D-DAAFCB8D9378}#1.0#0"; "video-
capx.ocx"
  Object = "{648A5603-2C6E-101B-82B6-000000000014}#1.1#0"; "MSCOMM32.OCX
"
  Object = "{248DD890-BB45-11CF-9ABC-0080C7E7B78D}#1.0#0"; "MSWINSCK.OCX
"
  Object = "{F07C23E0-0CF6-11D0-80E9-444553540000}#1.0#0"; "Joy-
sticks.ocx"
  Object = "{831FDD16-0C5C-11D2-A9FC-0000F8754DA1}#2.0#0"; "MSCOMCTL.OCX
"
  Object = "{66D3CBC4-D446-4BAA-B8B2-AF97BC09A7D2}#1.0#0"; "AudioCon-
trol.ocx"
  Begin VB.Form Main
     BackColor       =   &H00000000&
     Caption         =   "Automatic"
     ClientHeight    =   9315
     ClientLeft      =   60
     ClientTop       =   630
     ClientWidth     =   14760
     LinkTopic       =   "Form1"
     ScaleHeight     =   621
     ScaleMode       =   3  'Pixel
     ScaleWidth      =   984
     StartUpPosition =   2  'CenterScreen
     Begin MSComctlLib.ImageList ImageList2
        Left            =   4200
        Top             =   6600
  _      ExtentX         =   1005
  _      ExtentY         =   1005
        BackColor       =   -2147483643
```

```
        ImageWidth      =   160
        ImageHeight     =   120
        MaskColor       =   12632256
  _      Version        =   393216
        BeginProperty Images {2C247F25-8591-11D1-B16A-00C0F0283628 {
           NumListImages   =   1
           BeginProperty ListImage1 {2C247F27-8591-11D1-B16A-
00C0F0283628 {
              Picture         =   "Main.frx":0000
              Key""   =
           EndProperty
        EndProperty
     End
     Begin MSComctlLib.ImageList ImageList1
        Left            =   4200
        Top             =   6000
  _      ExtentX        =   1005
  _      ExtentY        =   1005
        BackColor       =   -2147483643
        ImageWidth      =   32
        ImageHeight     =   32
        MaskColor       =   12632256
  _      Version        =   393216
        BeginProperty Images {2C247F25-8591-11D1-B16A-00C0F0283628 {
           NumListImages   =   6
           BeginProperty ListImage1 {2C247F27-8591-11D1-B16A-
00C0F0283628 {
              Picture         =   "Main.frx":5A7B
              Key""   =
           EndProperty
           BeginProperty ListImage2 {2C247F27-8591-11D1-B16A-
00C0F0283628 {
              Picture         =   "Main.frx":5ECD
              Key""   =
           EndProperty
           BeginProperty ListImage3 {2C247F27-8591-11D1-B16A-
00C0F0283628 {
              Picture         =   "Main.frx":631F
              Key""   =
           EndProperty
           BeginProperty ListImage4 {2C247F27-8591-11D1-B16A-
00C0F0283628 {
              Picture         =   "Main.frx":6771
              Key""   =
           EndProperty
           BeginProperty ListImage5 {2C247F27-8591-11D1-B16A-
00C0F0283628 {
              Picture         =   "Main.frx":6BC3
              Key""   =
           EndProperty
           BeginProperty ListImage6 {2C247F27-8591-11D1-B16A-
00C0F0283628 {
              Picture         =   "Main.frx":7015
              Key""   =
           EndProperty
        EndProperty
     End
     Begin JOYSTICKLib.Joystick Joystick1
        Height          =   420
        Left            =   4320
        TabIndex        =   57
        Top             =   3120
        Visible         =   0   'False
```

```
      Width           =   420
_      Version        =   65536
_      ExtentX        =   741
_      ExtentY        =   741
_      StockProps     =   1
      m_button        =   1243687
   End
   Begin VB.Frame Frame6
      BorderStyle     =   0  'None
      Caption         =   "Frame1"
      Height          =   6855
      Left            =   240
      TabIndex        =   26
      Top             =   360
      Width           =   3855
      Begin VB.Frame Frame8
         BackColor       =   &H00000000&
         BorderStyle     =   0  'None
         Caption         =   "Frame7"
         Height          =   615
         Left            =   480
         TabIndex        =   65
         Top             =   4560
         Width           =   3015
         Begin VB.OptionButton CameraRotationMode
            Caption         =   "Use Knobes"
            Height          =   495
            Index           =   1
            Left            =   1680
            Style           =   1  'Graphical
            TabIndex        =   67
            Top             =   0
            Width           =   1215
         End
         Begin VB.OptionButton CameraRotationMode
            Caption         =   "Use Arrows"
            Height          =   495
            Index           =   0
            Left            =   0
            Style           =   1  'Graphical
            TabIndex        =   66
            Top             =   0
            Value           =   -1  'True
            Width           =   1215
         End
      End
      Begin VB.Frame CameraRotationKnobesFrame
         BackColor       =   &H00000000&
         BorderStyle     =   0  'None
         Enabled         =   0   'False
         Height          =   1215
         Left            =   2040
         TabIndex        =   59
         Top             =   5160
         Width           =   1455
         Begin AUDIOCONTROLLib.Knob CameraRotationKnobeVertical
            Height          =   735
            Left            =   0
            TabIndex        =   68
            Top             =   360
            Width           =   735
_            Version        =   65536
_            ExtentX        =   1296
```

```
_             ExtentY         =   1296
_             StockProps      =   0
            Min             =   -180
            Max             =   180
         End
         Begin AUDIOCONTROLLib.Knob CameraRotationKnobeHorizontal
            Height          =   735
            Left            =   720
            TabIndex        =   69
            Top             =   360
            Width           =   615
_             Version         =   65536
_             ExtentX         =   1085
_             ExtentY         =   1296
_             StockProps      =   0
            Min             =   -180
            Max             =   180
         End
         Begin VB.Label Label18
            AutoSize        =   -1  'True
            BackStyle       =   0  'Transparent
            Caption         =   "Vertical"
            ForeColor       =   &H00FF8080&
            Height          =   195
            Left            =   0
            TabIndex        =   61
            Top             =   120
            Width           =   525
         End
         Begin VB.Label Label19
            AutoSize        =   -1  'True
            BackStyle       =   0  'Transparent
            Caption         =   "Horizontal"
            ForeColor       =   &H00FF8080&
            Height          =   195
            Left            =   720
            TabIndex        =   60
            Top             =   120
            Width           =   705
         End
      End
      Begin VB.CommandButton CameraRotationRight
         Height          =   495
         Left            =   1320
         Picture         =   "Main.frx":732F
         Style           =   1  'Graphical
         TabIndex        =   32
         Top             =   5520
         Width           =   495
      End
      Begin VB.CommandButton CameraRotationLeft
         Height          =   495
         Left            =   360
         Picture         =   "Main.frx":7771
         Style           =   1  'Graphical
         TabIndex        =   31
         Top             =   5520
         Width           =   495
      End
      Begin VB.CommandButton CameraRotationDownward
         Height          =   495
         Left            =   840
         Picture         =   "Main.frx":7BB3
```

```
      Style           =   1  'Graphical
      TabIndex        =   30
      Top             =   5760
      Width           =   495
   End
   Begin VB.CommandButton CameraRotationUpward
      Height          =   495
      Left            =   840
      Picture         =   "Main.frx":7FF5
      Style           =   1  'Graphical
      TabIndex        =   29
      Top             =   5280
      Width           =   495
   End
   Begin VB.CommandButton CameraHeightDecrease
      Height          =   495
      Left            =   840
      Picture         =   "Main.frx":8437
      Style           =   1  'Graphical
      TabIndex        =   28
      Top             =   2640
      Width           =   495
   End
   Begin VB.CommandButton CameraHeightIncrease
      Height          =   495
      Left            =   840
      Picture         =   "Main.frx":8879
      Style           =   1  'Graphical
      TabIndex        =   27
      Top             =   2040
      Width           =   495
   End
   Begin VB.PictureBox Picture5
      BackColor       =   &H80000007&
      Height          =   6855
      Left            =   0
      ScaleHeight     =   6795
      ScaleWidth      =   3795
      TabIndex        =   33
      Top             =   0
      Width           =   3855
      Begin VB.Frame Frame7
         BackColor       =   &H00000000&
         BorderStyle     =   0  'None
         Caption         =   "Frame7"
         Height          =   735
         Left            =   360
         TabIndex        =   62
         Top             =   960
         Width           =   3135
         Begin VB.OptionButton CameraHeightMode
            Caption         =   "Use Slider"
            Height          =   495
            Index           =   1
            Left            =   1800
            Style           =   1  'Graphical
            TabIndex        =   64
            Top             =   120
            Width           =   1215
         End
         Begin VB.OptionButton CameraHeightMode
            Caption         =   "Use Arrows"
            Height          =   495
```

```
         Index           =   0
         Left            =   120
         Style           =   1  'Graphical
         TabIndex        =   63
         Top             =   120
         Value           =   -1  'True
         Width           =   1215
      End
   End
   Begin VB.Frame CameraHeightSliderFrame
      BorderStyle     =   0  'None
      Enabled         =   0   'False
      Height          =   1935
      Left            =   2280
      TabIndex        =   58
      Top             =   1680
      Width           =   735
      Begin AUDIOCONTROLLib.LevelSlider CameraHeightSlider
         Height          =   1935
         Left            =   0
         TabIndex        =   70
         Top             =   0
         Width           =   735
_         Version         =   65536
_         ExtentX         =   1296
_         ExtentY         =   3413
_         StockProps      =   0
         Max             =   100
      End
   End
   Begin VB.Label Label15
      AutoSize        =   -1  'True
      BackStyle       =   0  'Transparent
      Caption         =   "Height"
      ForeColor       =   &H00FF8080&
      Height          =   195
      Left            =   1560
      TabIndex        =   41
      Top             =   720
      Width           =   465
   End
   Begin VB.Shape Shape5
      BorderColor     =   &H00FF8080&
      BorderStyle     =   6  'Inside Solid
      BorderWidth     =   5
      Height          =   2535
      Left            =   120
      Shape           =   4  'Rounded Rectangle
      Top             =   4080
      Width           =   3495
   End
   Begin VB.Label Label17
      AutoSize        =   -1  'True
      BackStyle       =   0  'Transparent
      Caption         =   "Rotation"
      ForeColor       =   &H00FF8080&
      Height          =   195
      Left            =   1560
      TabIndex        =   43
      Top             =   4200
      Width           =   600
   End
   Begin VB.Label Label16
```

```
         Alignment       =   2  'Center
         AutoSize        =   -1  'True
         BackStyle       =   0  'Transparent
         Caption         =   "Camera Position"
         ForeColor       =   &H00FF8080&
         Height          =   195
         Left            =   1200
         TabIndex        =   42
         Top             =   120
         Width           =   1155
      End
      Begin VB.Shape Shape4
         BorderColor     =   &H00FF8080&
         BorderStyle     =   6  'Inside Solid
         BorderWidth     =   5
         Height          =   3375
         Left            =   120
         Shape           =   4  'Rounded Rectangle
         Top             =   480
         Width           =   3495
      End
   End
End
Begin VB.Frame Frame5
   BorderStyle     =   0  'None
   Caption         =   "Frame1"
   Height          =   3615
   Left            =   4920
   TabIndex        =   19
   Top             =   3000
   Width           =   5535
   Begin VB.CommandButton Command2
      Caption         =   "Turn Right"
      Height          =   495
      Left            =   3840
      Style           =   1  'Graphical
      TabIndex        =   25
      Top             =   1440
      Width           =   1215
   End
   Begin VB.CommandButton Command1
      Caption         =   "Turn Left"
      Height          =   495
      Left            =   3840
      Style           =   1  'Graphical
      TabIndex        =   24
      Top             =   2040
      Width           =   1215
   End
   Begin VB.OptionButton DirectionMode
      Caption         =   "Backward"
      Height          =   495
      Index           =   1
      Left            =   2040
      Style           =   1  'Graphical
      TabIndex        =   23
      Top             =   2040
      Width           =   1215
   End
   Begin VB.OptionButton DirectionMode
      Caption         =   "Forward"
      Height          =   495
      Index           =   0
```

```
      Left            =   2040
      Style           =   1  'Graphical
      TabIndex        =   22
      Top             =   1440
      Value           =   -1  'True
      Width           =   1215
   End
   Begin VB.PictureBox Picture8
      BackColor       =   &H80000007&
      Height          =   3615
      Left            =   0
      ScaleHeight     =   3555
      ScaleWidth      =   5475
      TabIndex        =   36
      Top             =   0
      Width           =   5535
      Begin AUDIOCONTROLLib.LevelSlider LevelSlider2
         Height          =   2175
         Left            =   480
         TabIndex        =   71
         Top             =   960
         Width           =   855
         _Version        =   65536
         _ExtentX        =   1508
         _ExtentY        =   3836
         _StockProps     =   0
         Max             =   100
      End
      Begin VB.Label Label11
         AutoSize        =   -1  'True
         BackStyle       =   0  'Transparent
         Caption         =   "Robot Speed"
         ForeColor       =   &H00FF8080&
         Height          =   195
         Left            =   360
         TabIndex        =   47
         Top             =   600
         Width           =   945
      End
      Begin VB.Label Label12
         Alignment       =   2  'Center
         AutoSize        =   -1  'True
         BackStyle       =   0  'Transparent
         Caption         =   "Speed And Direction Control"
         ForeColor       =   &H00FF8080&
         Height          =   195
         Left            =   1800
         TabIndex        =   46
         Top             =   120
         Width           =   2025
      End
      Begin VB.Shape Shape1
         BorderColor     =   &H00FF8080&
         BorderStyle     =   6  'Inside Solid
         BorderWidth     =   5
         Height          =   1815
         Left            =   3600
         Shape           =   4  'Rounded Rectangle
         Top             =   960
         Width           =   1695
      End
      Begin VB.Label Label13
         AutoSize        =   -1  'True
```

```
         BackStyle       =   0  'Transparent
         Caption         =   "Rotation"
         ForeColor       =   &H00FF8080&
         Height          =   195
         Left            =   3840
         TabIndex        =   45
         Top             =   1080
         Width           =   600
      End
      Begin VB.Shape Shape2
         BorderColor     =   &H00FF8080&
         BorderStyle     =   6  'Inside Solid
         BorderWidth     =   5
         Height          =   1815
         Left            =   1800
         Shape           =   4  'Rounded Rectangle
         Top             =   960
         Width           =   1695
      End
      Begin VB.Label Label14
         AutoSize        =   -1  'True
         BackStyle       =   0  'Transparent

         Caption         =   "Direction"
         ForeColor       =   &H00FF8080&
         Height          =   195
         Left            =   2040
         TabIndex        =   44
         Top             =   1080
         Width           =   630
      End
      Begin VB.Shape Shape3
         BorderColor     =   &H00FF8080&
         BorderStyle     =   6  'Inside Solid
         BorderWidth     =   5
         Height          =   3015
         Left            =   240
         Shape           =   4  'Rounded Rectangle
         Top             =   360
         Width           =   1335
      End
   End
End
Begin VB.Frame Frame4
   BorderStyle     =   0  'None
   Caption         =   "Frame4"
   Height          =   3255
   Left            =   11280
   TabIndex        =   15
   Top             =   3840
   Width           =   3255
   Begin VB.CheckBox Sound
      Caption         =   "Sound"
      Height          =   495
      Left            =   240
      Style           =   1  'Graphical
      TabIndex        =   18
      Top             =   600
      Value           =   1  'Checked
      Width           =   1215
   End
   Begin VB.CheckBox Heat
      Caption         =   "Heat"
```

```
      Height          =   495
      Left            =   240
      Style           =   1  'Graphical
      TabIndex        =   17
      Top             =   1320
      Value           =   1  'Checked
      Width           =   1215
   End
   Begin VB.CheckBox Position
      Caption         =   "Position"
      Height          =   495
      Left            =   240
      Style           =   1  'Graphical
      TabIndex        =   16
      Top             =   2040
      Width           =   1215
   End
   Begin VB.PictureBox Picture10
      BackColor       =   &H80000007&
      Height          =   3255
      Left            =   0
      ScaleHeight     =   3195
      ScaleWidth      =   3195
      TabIndex        =   38
      Top             =   0
      Width           =   3255
      Begin VB.Label Label8
         Alignment       =   2  'Center
         BackStyle       =   0  'Transparent
         BorderStyle     =   1  'Fixed Single
         Caption         =   "Heat"
         ForeColor       =   &H00FF8080&
         Height          =   495
         Left            =   1920
         TabIndex        =   53
         Top             =   1320
         Width           =   1215
      End
      Begin VB.Label Label7
         Alignment       =   2  'Center
         BackStyle       =   0  'Transparent
         Caption         =   "Y":
         ForeColor       =   &H00FF8080&
         Height          =   255
         Left            =   1080
         TabIndex        =   52
         Top             =   2760
         Width           =   1215
      End
      Begin VB.Label Label6
         Alignment       =   2  'Center
         BackStyle       =   0  'Transparent
         BorderStyle     =   1  'Fixed Single
         Caption         =   "X Pos"
         ForeColor       =   &H00FF8080&
         Height          =   255
         Left            =   1920
         TabIndex        =   51
         Top             =   2400
         Width           =   1215
      End
      Begin VB.Label Label5
         Alignment       =   2  'Center
```

```
            BackStyle       =   0  'Transparent
            BorderStyle     =   1  'Fixed Single
            Caption         =   "YPos"
            ForeColor       =   &H00FF8080&
            Height          =   255
            Left            =   1920
            TabIndex        =   50
            Top             =   2760
            Width           =   1215
         End
         Begin VB.Label Label4
            Alignment       =   2  'Center
            BackStyle       =   0  'Transparent
            Caption         =   "X":
            ForeColor       =   &H00FF8080&
            Height          =   255
            Left            =   1080
            TabIndex        =   49
            Top             =   2400
            Width           =   1215
         End
         Begin VB.Label Label3
            AutoSize        =   -1  'True
            BackStyle       =   0  'Transparent
            Caption         =   "Data Coming From Robots"
            ForeColor       =   &H00FF8080&
            Height          =   195
            Left            =   600
            TabIndex        =   48
            Top             =   240
            Width           =   1860
         End
      End
      Begin VB.Image Image3
         BorderStyle     =   1  'Fixed Single
         Height          =   3225
         Left            =   0
         Stretch         =   -1  'True
         Top             =   0
         Width           =   3225
      End
   End
   Begin VB.Frame Frame3
      BorderStyle     =   0  'None
      Caption         =   "Frame1"
      Height          =   3255
      Left            =   11280
      TabIndex        =   12
      Top             =   360
      Width           =   3255
      Begin VB.OptionButton LightMode
         Caption         =   "Manual"
         Height          =   495
         Index           =   1
         Left            =   480
         Style           =   1  'Graphical
         TabIndex        =   14
         Top             =   2280
         Width           =   1215
      End
      Begin VB.OptionButton LightMode
         Caption         =   "Automatic"
         Height          =   495
```

```
         Index           =   0
         Left            =   480
         Style           =   1  'Graphical
         TabIndex        =   13
         Top             =   720
         Value           =   -1  'True
         Width           =   1215
      End
      Begin VB.PictureBox Picture9
         BackColor       =   &H80000007&
         Height          =   3255
         Left            =   0
         ScaleHeight     =   3195
         ScaleWidth      =   3195
         TabIndex        =   37
         Top             =   0
         Width           =   3255
         Begin VB.TextBox TheresholdValue
            Height          =   405
            Left            =   1320
            TabIndex        =   54
            Text            =   "100"
            Top             =   1560
            Width           =   1575
         End
         Begin VB.Label Label9
            Alignment       =   2  'Center
            AutoSize        =   -1  'True
            BackStyle       =   0  'Transparent
            Caption         =   "Illumination System"
            ForeColor       =   &H00FF8080&
            Height          =   195
            Left            =   855
            TabIndex        =   56
            Top             =   240
            Width           =   1365
         End
         Begin VB.Label Label10
            AutoSize        =   -1  'True
            BackStyle       =   0  'Transparent
            Caption         =   "Threshold Value"
            ForeColor       =   &H00FF8080&
            Height          =   195
            Left            =   1320
            TabIndex        =   55
            Top             =   1320
            Width           =   1155
         End
      End
      Begin VB.Image Image2
         BorderStyle     =   1  'Fixed Single
         Height          =   3240
         Left            =   0
         Stretch         =   -1  'True
         Top             =   0
         Width           =   3240
      End
   End
   Begin VB.Frame Frame2
      BorderStyle     =   0  'None
      Caption         =   "Frame2"
      Height          =   1815
      Left            =   4320
```

```
      TabIndex        =   9
      Top             =   840
      Width           =   3255
      Begin VB.OptionButton DataTransmissionMode
         Caption         =   "Wireless Lan"
         Height          =   495
         Index           =   0
         Left            =   960
         Style           =   1  'Graphical
         TabIndex        =   11
         Top             =   480
         Value           =   -1  'True
         Width           =   1215
      End
      Begin VB.OptionButton DataTransmissionMode
         Caption         =   "Transceivers"
         Height          =   495
         Index           =   1
         Left            =   960
         Style           =   1  'Graphical
         TabIndex        =   10
         Top             =   1080
         Width           =   1215
      End
      Begin VB.PictureBox Picture6
         BackColor       =   &H80000007&
         Height          =   1815
         Left            =   0
         ScaleHeight     =   1755
         ScaleWidth      =   3195
         TabIndex        =   34
         Top             =   0
         Width           =   3255
         Begin VB.Label Label2
            AutoSize        =   -1  'True
            BackStyle       =   0  'Transparent
            Caption         =   "Data Transmission Mode"
            ForeColor       =   &H00FF8080&
            Height          =   195
            Left            =   720
            TabIndex        =   39
            Top             =   120
            Width           =   1755
         End
      End
   End
   Begin MSCommLib.MSComm MSComm1
      Left            =   4200
      Top             =   5400
_      ExtentX         =   1005
_      ExtentY         =   1005
_      Version         =   393216
      DTREnable       =   -1  'True
   End
   Begin MSWinsockLib.Winsock Winsock1
      Left            =   4320
      Top             =   3600
_      ExtentX         =   741
_      ExtentY         =   741
_      Version         =   393216
   End
   Begin VB.CheckBox VideoPreview
      Height          =   525
```

```
         Index           =   1
         Left            =   6960
         Style           =   1  'Graphical
         TabIndex        =   7
         Top             =   7920
         Width           =   525
      End
      Begin VB.CheckBox VideoPreview
         Height          =   525
         Index           =   2
         Left            =   10560
         Style           =   1  'Graphical
         TabIndex        =   6
         Top             =   7920
         Width           =   525
      End
      Begin VB.CheckBox VideoPreview
         Height          =   525
         Index           =   0
         Left            =   3360
         Style           =   1  'Graphical
         TabIndex        =   5
         Top             =   7920
         Width           =   525
      End
      Begin VIDEOCAPXLib.VideoCapX VideoCapX2
         Height          =   615
         Left            =   4200
         TabIndex        =   4
         Top             =   4680
         Visible         =   0   'False
         Width           =   615
   _         Version         =   131072
   _         ExtentX         =   1085
   _         ExtentY         =   1085
   _         StockProps      =   0
      End
      Begin VB.PictureBox Videos
         Height          =   1800
         Index           =   1
         Left            =   7800
         Picture         =   "Main.frx":8CBB
         ScaleHeight     =   1740
         ScaleWidth      =   2340
         TabIndex        =   3
         Top             =   7320
         Width           =   2400
      End
      Begin VB.PictureBox Videos
         Height          =   1800
         Index           =   2
         Left            =   11400
         Picture         =   "Main.frx":90FD
         ScaleHeight     =   1740
         ScaleWidth      =   2340
         TabIndex        =   2
         Top             =   7320
         Width           =   2400
      End
      Begin VB.Timer Timer1
         Interval        =   1
         Left            =   4320
         Top             =   4080
```

```
   End
   Begin VB.PictureBox Videos
      Height          =   1800
      Index           =   0
      Left            =   4200
      Picture         =   "Main.frx":953F
      ScaleHeight     =   1740
      ScaleWidth      =   2340
      TabIndex        =   1
      Top             =   7320
      Width           =   2400
   End
   Begin VIDEOCAPXLib.VideoCapX VideoCapX1
      Height          =   1800
      Left            =   600
      TabIndex        =   0
      Top             =   7320
      Width           =   2400
_      Version         =   131072
_      ExtentX         =   4233
_      ExtentY         =   3175
_      StockProps      =   0
   End
   Begin VB.Frame Frame1
      BorderStyle     =   0  'None
      Caption         =   "Frame1"
      Height          =   1815
      Left            =   7800
      TabIndex        =   8
      Top             =   840
      Width           =   3255
      Begin VB.OptionButton VideoTransmissionMode
         Caption         =   "Video Waves"
         Height          =   495
         Index           =   1
         Left            =   960
         Style           =   1  'Graphical
         TabIndex        =   21
         Top             =   1080
         Width           =   1215
      End
      Begin VB.OptionButton VideoTransmissionMode
         Caption         =   "Wireless Lan"
         Height          =   495
         Index           =   0
         Left            =   960
         Style           =   1  'Graphical
         TabIndex        =   20
         Top             =   480
         Value           =   -1  'True
         Width           =   1215
      End
      Begin VB.PictureBox Picture7
         BackColor       =   &H80000007&
         Height          =   1815
         Left            =   0
         ScaleHeight     =   1755
         ScaleWidth      =   3195
         TabIndex        =   35
         Top             =   0
         Width           =   3255
         Begin VB.Label Label1
            AutoSize        =   -1  'True
```

```
         BackStyle       =   0  'Transparent
         Caption         =   "Video Transmission Mode"
         ForeColor       =   &H00FF8080&
         Height          =   195
         Left            =   600
         TabIndex        =   40
         Top             =   120
         Width           =   1815
      End
   End
   Begin VB.Image Image1
      BorderStyle     =   1  'Fixed Single
      Height          =   1800
      Left            =   0
      Stretch         =   -1  'True
      Top             =   0
      Width           =   3240
   End
End
Begin VB.Menu FileMenu
   Caption         =   "&File"
   Begin VB.Menu ExitMenu
      Caption         =   "E&xit"
      Shortcut        =   ^X
   End
End
Begin VB.Menu GenerateMenu
   Caption         =   "&Generate"
   Begin VB.Menu BurnCDMenu
      Caption         =   "&Burn CD"
      Shortcut        =   ^D
   End
   Begin VB.Menu PrintableReportMenu
      Caption         =   "Printable &Report"
      Shortcut        =   ^E
   End
   Begin VB.Menu WebPublishMenu
      Caption         =   "&Publish On The Web"
      Shortcut        =   ^W
   End
   Begin VB.Menu Seperator2
      Caption"-"   =
   End
   Begin VB.Menu GenerateAllMenu
      Caption         =   "&Generate All"
      Shortcut        =   ^G
   End
End
Begin VB.Menu RobotMenu
   Caption         =   "&Robot"
   Begin VB.Menu RobotChooserMenu
      Caption         =   "Robot &1"
      Checked         =   -1  'True
      Index           =   0
      Shortcut        =   {F1{
   End
   Begin VB.Menu RobotChooserMenu
      Caption         =   "Robot &2"
      Enabled         =   0   'False
      Index           =   1
      Shortcut        =   {F2{
   End
   Begin VB.Menu RobotChooserMenu
```

```
      Caption         =   "Robot &3"
      Enabled         =   0   'False
      Index           =   2
      Shortcut        =   {F3{
   End
End
Begin VB.Menu CamreraPositionMenu
   Caption         =   "Camera &Position"
   Begin VB.Menu HeightMenu
      Caption         =   "Height"
      Begin VB.Menu CameraHeightModeMenu
         Caption         =   "Use &Arrows"
         Checked         =   -1  'True
         Index           =   0
         Shortcut        =   ^A
      End
      Begin VB.Menu CameraHeightModeMenu
         Caption         =   "Use &Slider"
         Index           =   1
         Shortcut        =   ^S
      End
   End
   Begin VB.Menu Seperator1
      Caption"-"   =
   End
   Begin VB.Menu RotationMenu
      Caption         =   "Rotation"
      Begin VB.Menu CameraRotationModeMenu
         Caption         =   "Use A&rrows"
         Checked         =   -1  'True
         Index           =   0
         Shortcut        =   ^R
      End
      Begin VB.Menu CameraRotationModeMenu
         Caption         =   "Use &Knobes"
         Index           =   1
         Shortcut        =   ^K
      End
   End
End
Begin VB.Menu DataTransmissionModeMenu
   Caption         =   "&Data Transmission"
   Begin VB.Menu DataMode
      Caption         =   "Wireless &Lan"
      Checked         =   -1  'True
      Index           =   0
      Shortcut        =   ^L
   End
   Begin VB.Menu DataMode
      Caption         =   "&Transceivers"
      Index           =   1
      Shortcut        =   ^T
   End
End
Begin VB.Menu VideoTransmissionModeMenu
   Caption         =   "&Video Transmission"
   Begin VB.Menu VideoModeMenu
      Caption         =   "Wireless &Lan"
      Checked         =   -1  'True
      Index           =   0
      Shortcut        =   ^N
   End
   Begin VB.Menu VideoModeMenu
```

```
      Caption         =   "Video &Waves"
      Index           =   1
      Shortcut        =   ^V
   End
End
Begin VB.Menu VideoChannelMenu
   Caption         =   "Video &Channel"
   Begin VB.Menu VideoChannelsMenu
      Caption         =   "Channel &1"
      Index           =   0
      Shortcut        =   ^{F1{
   End
   Begin VB.Menu VideoChannelsMenu
      Caption         =   "Channel &2"
      Index           =   1
      Shortcut        =   ^{F2{
   End
   Begin VB.Menu VideoChannelsMenu
      Caption         =   "Channel &3"
      Index           =   2
      Shortcut        =   ^{F3{
   End
End
Begin VB.Menu DirectionMenu
   Caption         =   "Direc&tion"
   Begin VB.Menu DirectionModeMenu
      Caption         =   "&Forward"
      Checked         =   -1  'True
      Index           =   0
      Shortcut        =   ^F
   End
   Begin VB.Menu DirectionModeMenu
      Caption         =   "&Backward"
      Index           =   1
      Shortcut        =   ^B
   End
End
Begin VB.Menu LightennigSystemMenu
   Caption         =   "&Lights"
   Begin VB.Menu LightenningModeMenu
      Caption         =   "&Automatic"
      Checked         =   -1  'True
      Index           =   0
      Shortcut        =   ^U
   End
   Begin VB.Menu LightenningModeMenu
      Caption         =   "&Manual"
      Index           =   1
      Shortcut        =   ^M
   End
End
Begin VB.Menu SensorsMenu
   Caption         =   "Sensors"
   Begin VB.Menu SoundMenu
      Caption         =   "&Sound"
      Checked         =   -1  'True
      Shortcut        =   ^O
   End
   Begin VB.Menu HeatMenu
      Caption         =   "&Heat"
      Checked         =   -1  'True
      Shortcut        =   ^H
   End
```

```
      Begin VB.Menu PositionMenu
         Caption         =   "&Position"
         Shortcut        =   ^P
      End
   End
End
Attribute VB_Name = "Main"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False

Private Sub CameraHeightMode_Click(Index As Integer(
    If Index = 0 Then
        CameraHeightDecrease.Enabled = True
        CameraHeightIncrease.Enabled = True
        CameraHeightSliderFrame.Enabled = False
        CameraHeightModeMenu(0).Checked = True
        CameraHeightMode(0).Value = True
        CameraHeightModeMenu(1).Checked = False
        CameraHeightMode(1).Value = False
    Else
        CameraHeightDecrease.Enabled = False
        CameraHeightIncrease.Enabled = False
        CameraHeightSliderFrame.Enabled = True
        CameraHeightModeMenu(1).Checked = True
        CameraHeightMode(1).Value = True
        CameraHeightModeMenu(0).Checked = False
        CameraHeightMode(0).Value = False
    End If
End Sub

Private Sub CameraHeightModeMenu_Click(Index As Integer(
    CameraHeightMode_Click (Index(
End Sub

Private Sub CameraRotationMode_Click(Index As Integer(
    If Index = 0 Then
        CameraRotationDownward.Enabled = True
        CameraRotationLeft.Enabled = True
        CameraRotationRight.Enabled = True
        CameraRotationUpward.Enabled = True
        CameraRotationKnobesFrame.Enabled = False
        CameraRotationModeMenu(0).Checked = True
        CameraRotationMode(0).Value = True
        CameraRotationModeMenu(1).Checked = False
        CameraRotationMode(1).Value = False
    Else
        CameraRotationDownward.Enabled = False
        CameraRotationLeft.Enabled = False
        CameraRotationRight.Enabled = False
        CameraRotationUpward.Enabled = False
        CameraRotationKnobesFrame.Enabled = True
        CameraRotationModeMenu(1).Checked = True
        CameraRotationMode(1).Value = True
        CameraRotationModeMenu(0).Checked = False
        CameraRotationMode(0).Value = False
    End If
End Sub

Private Sub CameraRotationModeMenu_Click(Index As Integer(
    CameraRotationMode_Click (Index(
End Sub
```

```
Private Sub Check5_Click()

End Sub

Private Sub DataMode_Click(Index As Integer(
    DataTransmissionMode_Click (Index(
End Sub

Private Sub DataTransmissionMode_Click(Index As Integer(
    If Index = 0 Then
        DataMode(0).Checked = True
        DataTransmissionMode(0).Value = True
        DataMode(1).Checked = False
        DataTransmissionMode(1).Value = False
    Else
        DataMode(1).Checked = True
        DataTransmissionMode(1).Value = True
        DataMode(0).Checked = False
        DataTransmissionMode(0).Value = False
    End If
End Sub

Private Sub DirectionMode_Click(Index As Integer(
    If Index = 0 Then
        DirectionModeMenu(0).Checked = True
        DirectionMode(0).Value = True
        DirectionModeMenu(1).Checked = False
        DirectionMode(1).Value = False
    Else
        DirectionModeMenu(1).Checked = True

        DirectionMode(1).Value = True
        DirectionModeMenu(0).Checked = False
        DirectionMode(0).Value = False
    End If
End Sub

Private Sub DirectionModeMenu_Click(Index As Integer(
    DirectionMode_Click (Index(
End Sub

Private Sub ExitMenu_Click()
    End
End Sub

Private Sub Form_Load()
   VideoCapX1.Connected = True
   VideoCapX1.SetVideoFormat 160, 120
   VideoCapX1.Preview = True
   VideoCapX1.ServerMode = True
   Videos(0).Picture = ImageList2.ListImages(1).Picture
   Videos(1).Picture = ImageList2.ListImages(1).Picture
   Videos(2).Picture = ImageList2.ListImages(1).Picture
End Sub

Private Sub Heat_Click()
    If Heat.Value Then
        HeatMenu.Checked = True

        Heat.Value = 1
    Else
        HeatMenu.Checked = False
```

```
        Heat.Value = 0
    End If
End Sub

Private Sub HeatMenu_Click()
    If Heat.Value Then
        HeatMenu.Checked = False
        Heat.Value = 0
    Else
        HeatMenu.Checked = True
        Heat.Value = 1
    End If
End Sub

Private Sub LightenningModeMenu_Click(Index As Integer(
    LightMode_Click (Index(
End Sub

Private Sub LightMode_Click(Index As Integer(
    If Index = 0 Then
        TheresholdValue.Enabled = True
        LightenningModeMenu(0).Checked = True
        LightMode(0).Value = True
        LightenningModeMenu(1).Checked = False
        LightMode(1).Value = False
    Else
        TheresholdValue.Enabled = False
        LightenningModeMenu(1).Checked = True
        LightMode(1).Value = True
        LightenningModeMenu(0).Checked = False
        LightMode(0).Value = False
    End If
End Sub

Private Sub Position_Click()
    If Position.Value Then
        PositionMenu.Checked = True
        Position.Value = 1
    Else
        PositionMenu.Checked = False
        Position.Value = 0
    End If
End Sub

Private Sub PositionMenu_Click()
    If Position.Value Then
        PositionMenu.Checked = False
        Position.Value = 0

    Else
        PositionMenu.Checked = True
        Position.Value = 1
    End If
End Sub

Private Sub Sound_Click()
    If Sound.Value Then
        SoundMenu.Checked = True
        Sound.Value = 1
    Else
        SoundMenu.Checked = False
        Sound.Value = 0
    End If
```

```
  End Sub

  Private Sub SoundMenu_Click()
      If Sound.Value Then
          SoundMenu.Checked = False
          Sound.Value = 0
      Else
          SoundMenu.Checked = True
          Sound.Value = 1
      End If
  End Sub

  Private Sub Timer1_Timer()
  If Check2.Value Then Picture1.Picture = Video-
CapX2.ReceiveFrame("localhost("
  If Check3.Value Then Picture2.Picture = Video-
CapX2.ReceiveFrame("localhost("
  If Check4.Value Then Picture3.Picture = Video-
CapX2.ReceiveFrame("localhost("
  End Sub

  Private Sub VideoChannelsMenu_Click(Index As Integer(
      Videos_DblClick (Index(
  End Sub

  Private Sub VideoModeMenu_Click(Index As Integer(
      VideoTransmissionMode_Click (Index(
  End Sub

  Private Sub VideoPreview_Click(Index As Integer(
      If VideoPreview(Index) Then
       VideoPreview(Index).Picture = ImageList1.ListImages(1).Picture
      Else
       VideoPreview(Index).Picture = ImageList1.ListImages(6).Picture
       Videos(Index).Picture = ImageList2.ListImages(1).Picture
      End If
  End Sub

  Private Sub Videos_DblClick(Index As Integer(
      Dim counter As Byte
      For counter = 0 To 2
          VideoChannelsMenu(counter).Checked = False
      Next counter
          VideoChannelsMenu(Index).Checked = True
      VideoChoice = Index
      VideoProjector.Show
      VideoProjector.Timer1.Enabled = True
  End Sub

  Private Sub VideoTransmissionMode_Click(Index As Integer(
      If Index = 0 Then
          VideoModeMenu(0).Checked = True
          VideoTransmissionMode(0).Value = True
          VideoModeMenu(1).Checked = False
          VideoTransmissionMode(1).Value = False
      Else
          VideoModeMenu(1).Checked = True
          VideoTransmissionMode(1).Value = True
          VideoModeMenu(0).Checked = False
          VideoTransmissionMode(0).Value = False
      End If

  End Sub
```

The operator will command the robots and whenever a victim is detected a form related to a victim will be in the screen with some of the fields pre-filled having the data coming from sensors and the related pictures (and other attachments such as a short video and a short music file for CD Report and Web Published report), the operator fills the rest of the fields and checks the other automatically filled fields for being correct. He also helps the software to generate the final map (As we are taking advantage of positioning system, the software can generate some parts of the map automatically). The program will generate the report (all three reports) automatically after the mission by pressing "Generate All" menu. The reports will be described later.

4. Map generation/printing

Immediately after each mission we must give a map containing the location of the victims and the obstacles which may help finding the victims or increasing the accuracy of the victim location. As we have seen the films of previous competitions in the previous years, it always has been the operator's responsibility to see a video stream and draw something on a plane sheet of paper, but Resquake claims to have a better way of map generating and reduce the pressure on the operator during the mission. If it will be possible, a flying robot may take a picture of the whole arena and the map will be generated on a paper having grids and this taken picture with low opacity as the background. This way the map looks more accurate. If taking the picture would not be possible, we do the rest of the work on the paper with grids on it.

The other measurement being taken is using the data coming from the positioning system. The positioning system is telling us the exact position of each robot with reasonable accuracy -always and real time- so the location of the victim can be estimated with more accuracy, also the path in which the robot is exploring the arena, can be sketched having known the place of the robot in each second, so this part of map will be generated automatically. Thus, we are awaiting a very clean and accurate map counting on our reliable positioning system.

Sonar can also help us having the distance of robot from the objects around and can help generating a better map. We may put as many sonars as possible around the robot or put one sonar turning around with a stepper motor. We try to have a map without any line being drawn on it by the operator and operator should only help the software to generate a better map, the more accurate and successful positioning system, the more accurate and clean map generation. We will talk more about our positioning system later.

5. Sensors for Navigation and Localization

We are applying two techniques for localization, first the positioning system and second the sonar sensors. At the beginning of competition three stations will be setup in three corners of the arena. Three small robots will carry transceivers and the first job of operator is directing these three robots to the best place. Our positioning technique is triangulation. If we know the distance of robot from three constant stations, we can determine the exact position of robot.

Position of stations does not affect positioning system, because after placing three stations in their places, we can find position of each robot at the beginning of the competition. And positioning will be done considering the first position of robots.

The system can find the position of the robot in less than a millisecond. So, we can claim that we have the position of robot in all the moments of area exploring continuously. We measure the distance using FPGA technology which means to increase the accuracy of measurement due to the high frequency clock pulse we can apply to the FPGA chips.

We can find distance between a station and robot with high accuracy by measuring data transmission time. Robot must get a signal and send a reply. Circuits have constant delay that is measurable and can not make any trouble for us.

If we want to find the position of a robot with triangulation system we have to send an electromagnetic wave and receive the reply. By measuring the time (we know the electromagnetic waves travel with the light speed), we can have the distance from one station, all we know is a distance from a single reference which is the geometrical position of the points on a circle around a reference. By getting the distance from the second station we have another circle and the robot will be somewhere on the intersection of these two circles which are two points. The distance from the third station will give us a single point which is the position of the robot.

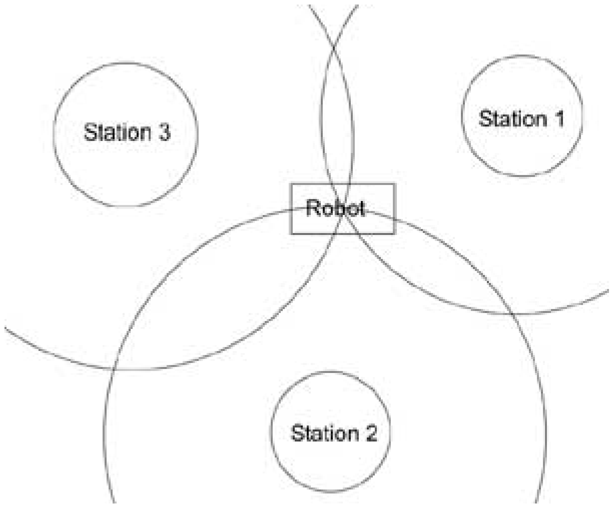


Figure 8

The following picture is the FPGA circuit which measures the distance. When the signal in sent, a counter starts counting and by the time the replication signal comes, the counter stops counting. The distance can be measured using this measured time.

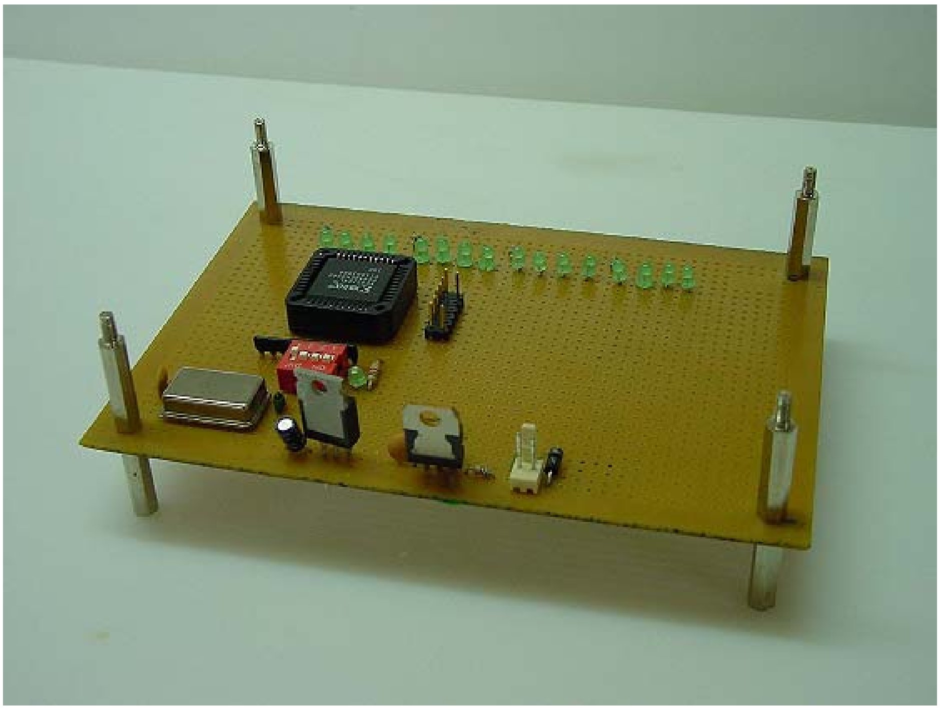


Figure 9

The IC datasheet can be found at Xilinx website. (IC name is XC9572XL which is a high performance CPLD).
http://direct.xilinx.com/bvdocs/publications/ds052.pdf
Sonar let us have the distance from the objects around. We can have number of sonars around the robot or use single sonar and turn it around with a stepper motor.

Distance measurement with ultrasonic waves is just the same as the above technique. We should send an ultra sonic wave (of about 40 KHz frequency) and wait to receive the reflection. A counter counts from the starting point of sending the wave until getting the reflection. Ultra sonic waves travel with sound speed and we can easily calculate the distance having speed and time.

6. Sensors for Victim Identification

1. Thermal Sensor Circuit

In this section we describe our thermal sensor which can determine temperature of victim's body. The temperature of live victim's skin is about 30˚C.

Our sensor can determine temperature using infra red radiation and without contact. It can also measure its own surface temperature and we can apply this temperature to find the measurement error. The temperature range of the sensor-element is between -40 to 100˚C. Output of the sensor-element is a voltage proportional to radiation of IR and generated by thermoelectric effect.

Complete datasheet of the sensor element can be found in the manufacturer website: http://www.smartec.nl/infrared_sensor.htm

We have designed a circuit that can amplify the signal and prepare it for entering analog to digital converter stage. Figure below shows how we have used Op-Amp in our circuit:

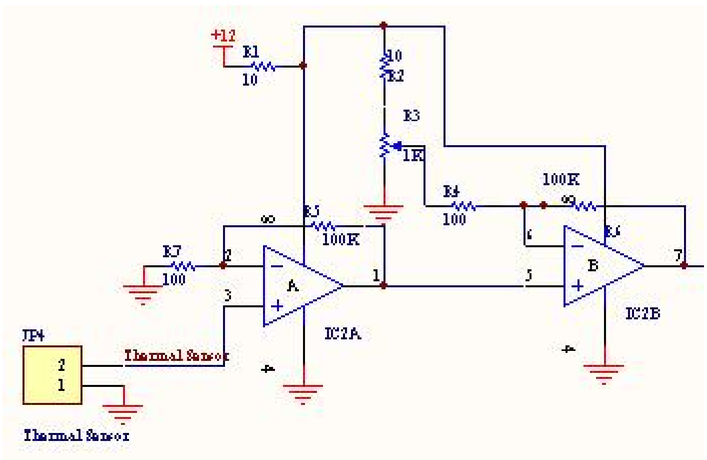


Figure 10 – Schematic diagram of Thermal Sensor

In the first stage, output voltage of sensor-element are multiplied my 1000 and in the second stage differences between reference voltage and output of first stage are multiplied by 1000 again. Output voltage is proportional to temperature of skin of the nearest object and can be directly connected to A/D input pin of AVR Microcontroller (PORTA.0 pin 40).

2. Light Sensor

The robot can measure light intensity of the environment and send a feed back for operator. Operator can visit the light status near the robot and turn off or turn on the lights (Illumination system). Light sensor-element is only a photo-resistor and our circuit is very simple as can be seen in below figure:

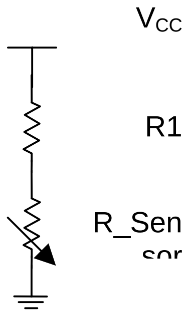


Figure 11

Our Sensor Resistance is $1^{K\Omega}$ in regular light and $30^{K\Omega}$ in dark environments. Placing a $10^{K\Omega}$ resistor as R1 leaves $\frac{1}{11}Vcc$ or $\frac{3}{4}Vcc$ for output voltage. The output of this circuit can be directly connected to A/D input pin of AVR Microcontroller (PORTA.1 pin 39). Notice that this voltage is not proportional to resistance of the sensor. It is corrected by the software. In other words, in microcontroller we can assign a new value to each A/D value by regression or taking a table of data bytes in ROM and refer to it.

3 .Gas Sensor:

We take advantage of a gas sensor as well. The sensor has a probe and takes in an amount of air and gives the amount of CO2 as an electrical signal. The signal will be analyzed with an ADC and the operator can see the amount of CO2 in Resquake Operator Interface.

If the amount of CO2 exceeds a threshold value, the operator will be prompted to be aware of the surroundings because their might be a victim around which may not be easily seen in the first look.

4.Motion Detection:

The video sent to the operator will be analyzed using VideoCapX control and if the any motion would be detected the operator will be prompted.

5.Sound Detection:

Video and sound will be sent at the same time on the LAN and are separately detectable in the software using VideoCapX control. Because of having a lot of noise in the area , there is no threshold value for the sound power and if there would be a sound which could be recognized as the victim's sound, the operator will understand.

7. Robot Locomotion

Robot locomotion is a very important point in exploring the arenas as we should be ready to tackle any possible obstacle. Resquake is working on three robots with different locomotion methods, but until the March 1st, a mixture of what we are thinking to get to is ready and successfully implemented. Here we go through the details of the system:

Differential steering is the type of locomotion chosen for the robot. In each side of the robot, wheels are connected with a timing belt. In this type of locomotion, spinning on the robot's axis is accomplished by moving one wheel in one direction and the other in the opposite direction. A sharp turn is accomplished by stopping one wheel while moving the other and a shallower turn is also accessible by moving one wheel at a lower speed making the robot to turn in the direction of the slower wheel.

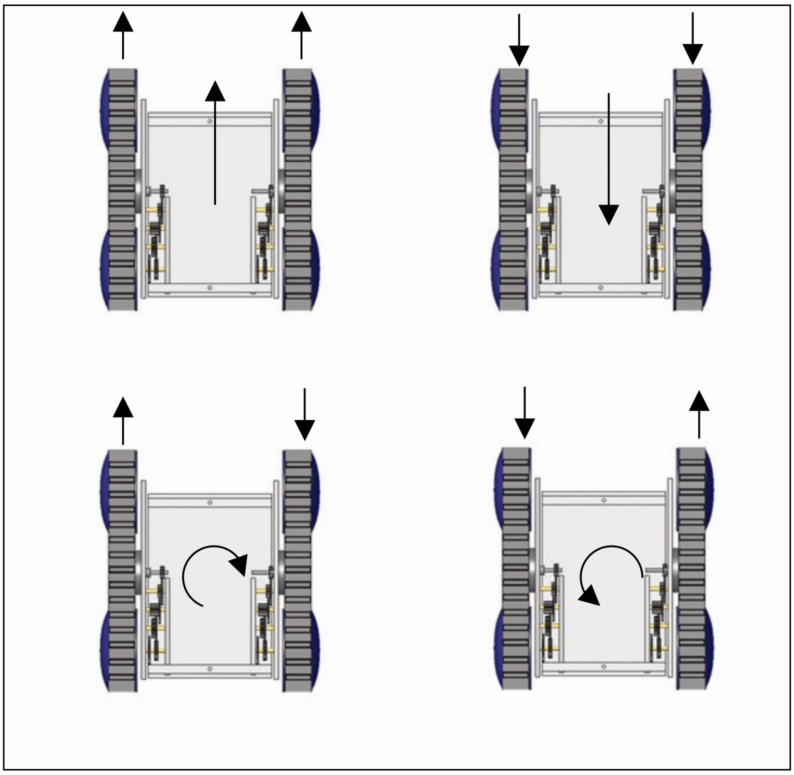


Figure 12 - How the differential steering acts

The timing belts, in addition to be a part of differential steering, help the robot climb blocks easie

Power Transmission: Two speed gearbox

As described above, the type of locomotion is differential steering. Two motors mounted on two sides of robot and between the wheels, apply the power to the wheels.

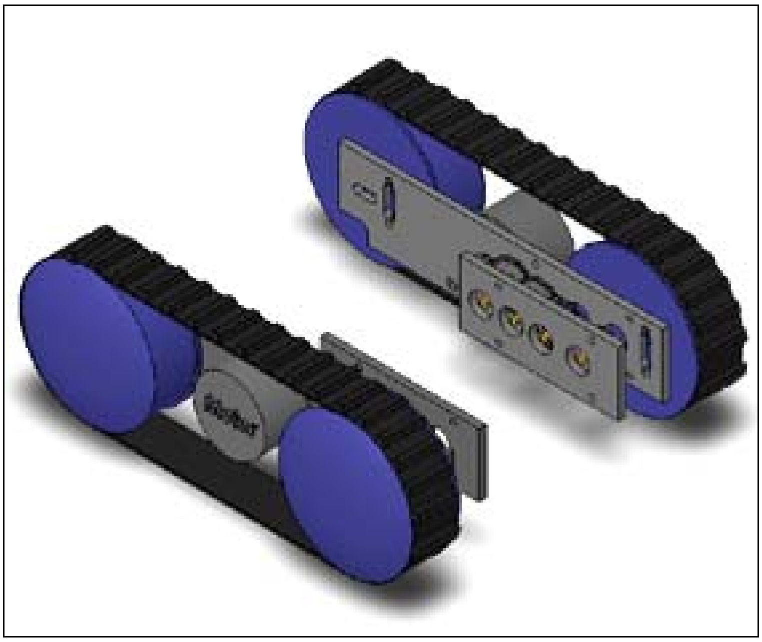


Figure 13. Motor is connected to the back wheel with a two speed gearbox

Kinetic design of gearbox:

Motor specifications:

Speed = 2000 rpm in 20 volts
Diameter of wheels = 94 mm
Required output speeds= 800 mm/s & 400 mm/s

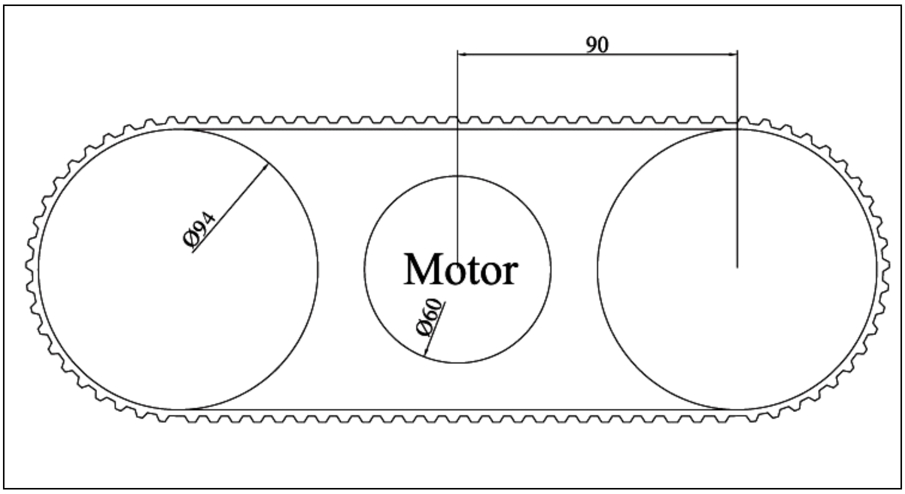


Figure 14. Limits for length of the gearbox

What limits the length of the gears train is the distance between the shaft of the motor and the shaft of the back wheel. This distance should have the minimum length of 85 mm.

The speed of the robot is a function of the rotational speed and the diameter of the wheels. Equation 1 shows this relationship, where v is the velocity of the robot, d is the diameter of the driven wheels and N is the rotational speed of the wheels:

$$v = \pi dN \tag{1}$$

So, to determine the required rotational speed of the wheel, equation 1 is solved for N, which is shown in equation 2.

$$N = \frac{v}{\pi D} \tag{2}$$

$$N = \frac{0.8}{0.094\pi} = 2.7 \ \ rps \equiv 162\, rpm$$

So, the gear ratio of the gearbox train should be:

$$m_G = \frac{N_1}{N_2} \tag{3}$$

$$m_G = \frac{2000}{162} = 12.35$$

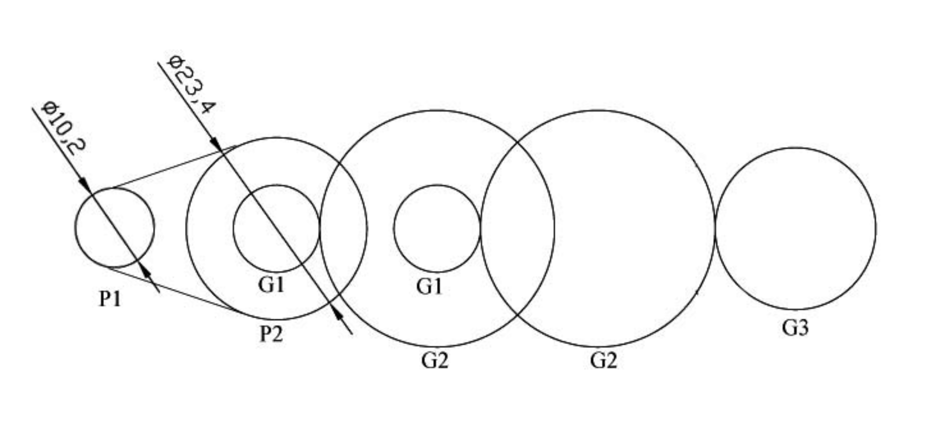


Figure 15. High speed Gear train

Module selected = 0.8

The number of teeth of gears is as follow:

G1 = 14 T
G2 = 38 T
G3 = 26 T

It should be noted that, as gears are going to be constructed by milling method, it's better that number of teeth be an even number.

As it is shown, the first step of transmission is done with two pulleys. The next steps are carried out by gears.

So, the gear ratio of this train of gears will be:

$$m_G = \frac{23.4}{10.2} \times \frac{38}{14} \times \frac{38}{14} \times \frac{26}{38} = 11.56$$

According to Equ.1 and Equ.3by applying this ratio the speed of robot will be:

$$m_G = \frac{N_1}{N_2}$$

$$N_2 = \frac{2000}{11.56} = 173\,rpm \equiv 2.88\,rps$$

v = πDN

v= π × 0.094 × 2.88 = 0.85 m/s

Thus, by applying this set, the speed of robot will be 0.85 m/s and it's close to the required speed.

To reach to the other required speed, the following set of gears will act:

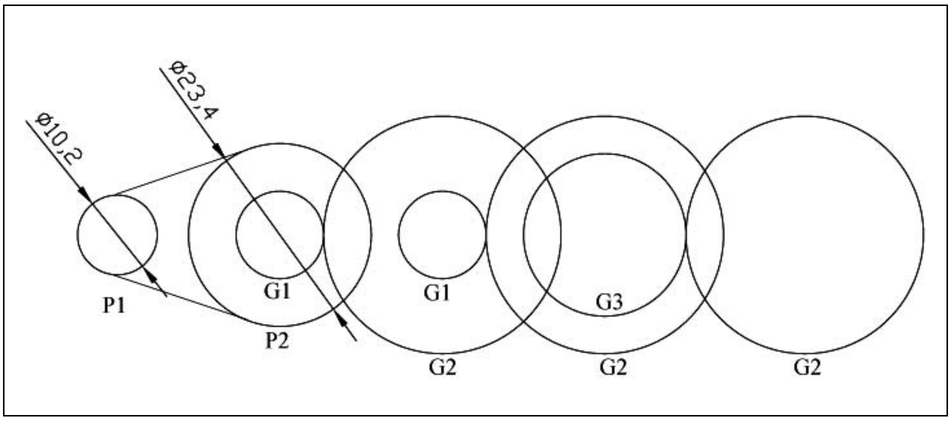


Figure 16. Low speed Gear train

The gear ratio of this train of gears will be:

$$m_G = \frac{23.4}{10.2} \times \frac{38}{14} \times \frac{38}{14} \times \frac{38}{26} = 24.7$$

According to Equ.1 and Equ.3by applying this ratio the speed of robot will be:

$$m_G = \frac{N_1}{N_2}$$

$$N_2 = \frac{2000}{24.7} = 80.97\,\text{rpm} \equiv 1.35\,\text{rps}$$

$v = \pi DN$

$v = \pi \times 0.094 \times 1.35 = 0.4$ m/s

By applying this set, the speed of robot will be 0.4 m/s and this speed is as required.

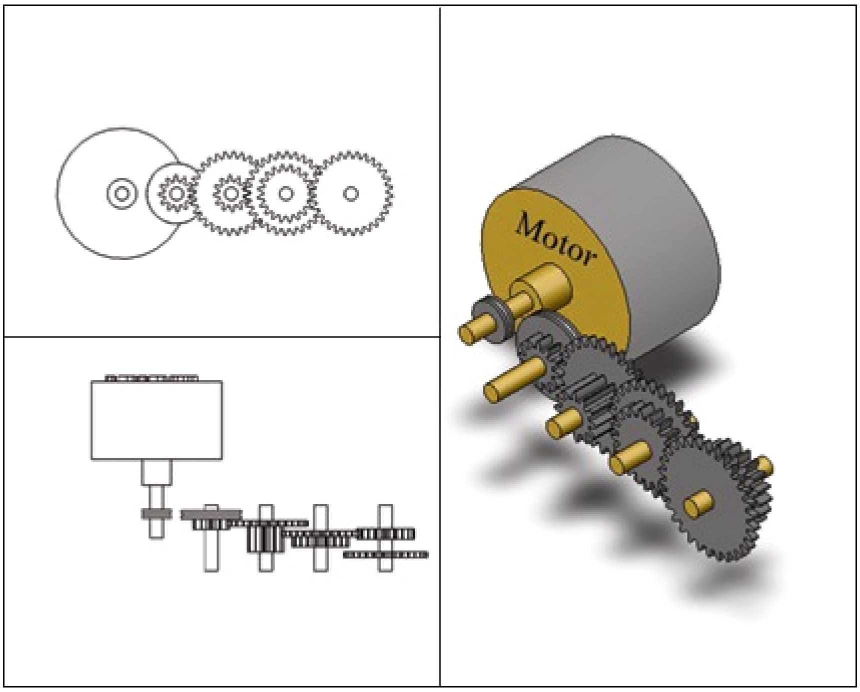


Figure 17. Sketches showing the two speed gearbox

This power is transmitted to the wheels by two gearboxes.

In order to change the speed of the robot it is needed to have a linear actuator, which pulls shaft of G2 and G3 out and in. This actuator can act on both sides of robot.

Stress designation of gears

The most critic gear is the one on the shaft of the wheel with 26 teeth. Failure can be checked by applying the Lewis bending equation.

According to Lewis bending equation:

$$\sigma = \frac{K'_V W_t}{FmY} \tag{4}$$

$$\Rightarrow \; W_t = \frac{\sigma FmY}{K'_V}$$

Where the face width F and the module are both in millimeters, W_t in Newtons and then the resultant stress will be in MegaPascals.

Gears are made of AISI 1030 Steel:

σ_y = 648 Mpa

Y is the Lewis factor which for a gear with 26 teeth is equal to 0.346:

Y = 0.347

The velocity factor, K'_v is:

$$K'_V = \frac{6.1 + V}{6.1} \tag{5}$$

$$V = \frac{\pi dN}{60} = 0.2 \; \frac{m}{s}$$

$$K'_V = \frac{6.1 + 0.2}{6.1} = 1.03$$

Thus:

$$W_t = \frac{648 \times 3 \times 0.8 \times 0.346}{1.03} = 522 \quad N$$

The max power that can be transferred with this gear will be:

$$P = W_t \times V \tag{6}$$

$$= 522 \times 0.2 = 104.48 \, \text{watts}$$

The max power of motor in his system can be 40 watts, so:

$$F.S. = \frac{104.48}{40} = 2.6$$

Shafts Designation

The most critic shaft in gearbox is the last one, to which the back wheel is connected.

The max torque of motor:

$$\tau = \frac{P}{\omega} \tag{7}$$

$$= \frac{40}{2000 \times \frac{2\pi}{60}} = 0.2\,\text{N.m}$$

When gearbox is working with the low speed, the torque acting on the 38 teeth gear will be:

$$\tau = 0.2 * 24.7 = 4.7 \text{ N.m}$$

$$\tau = rW_t$$

$$W_t = \frac{4.7 \times 2}{38 \times 0.4} = 310\,\text{N}$$

$$W_n = 0.31 \times \tan 20 = 110\,\text{N}$$

According to the ASME code for rotating shafts:

$$d = \sqrt[3]{\frac{16\,F.S.}{\pi\tau_p}\sqrt{(C_m.M)^2 + (C_T.T)^2}} \tag{7}$$

Where $C_m = 1.5$ and $C_T = 1$.

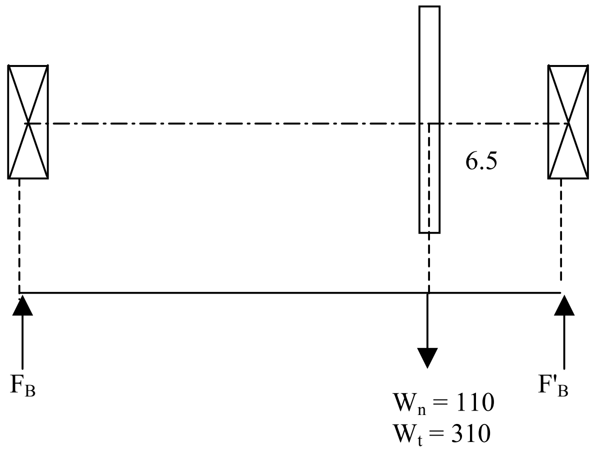
$$\tau_P = \text{Min}(0.3\ S_y\ \&\ 0.18\ S_u)$$

Shafts are made of CK45 steel with :

$S_u = 650$ Mpa $\quad S_y = 430$ Mpa

So:

τ_P = 117 Mpa



$$M = \sqrt{(110 \times 0.0065)^2 + (310 \times 0.0065)^2} = 2.13\,\text{N.m} \qquad (7)$$

T = 4.7 N.m

$$d = \sqrt[3]{\frac{16 \times 4}{\pi \times 117}\sqrt{(1.5 \times 2.13)^2 + (4.7)^2}} = 1\text{ mm}$$

It is very hard to work with a 1 mm diameter shaft. So, we decided to apply shafts with 5mm diameter in whole parts of gearbox and wheels.

Frame of Gearbox

All the gearbox parts, wheels, and motor are connected to a plate (P1). This plate is built from an aluminum sheet with 5 mm diameter. The drawing of this plate is added to the report. Another plate (P2) also fixes the parts on this place.

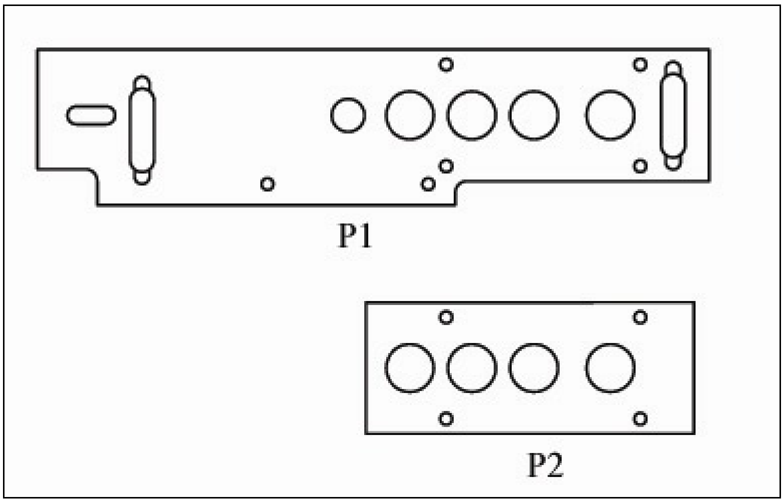


Figure 18. Plates holding part of gearbox and wheels

It should be noted that all the shafts are connected to the plate with ball bearing.

So, the assembly of one side of robot will look like below:

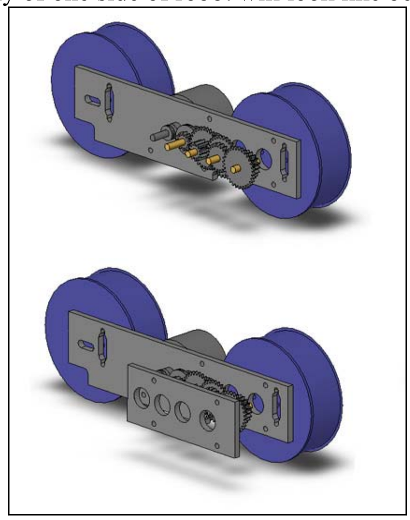


Figure 19. **The assembly of gearbox, motor, and the wheels of one side of robot**

Suspension System

In order to give the robot more flexibility and reliability to work in scattered areas, we got to the point that the robot needs a suspension system. So, we decided to place four springs in the plates and connect the whole body of robot to these springs.

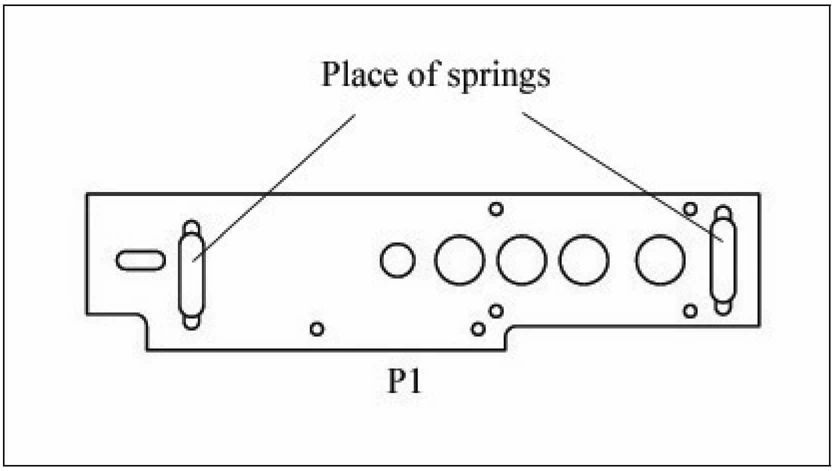


Figure 20. **Place of springs on one side of robot**

Shafts with 4 mm diameter will be placed in these places and a pin, designed to hold the body will be placed on the spring.

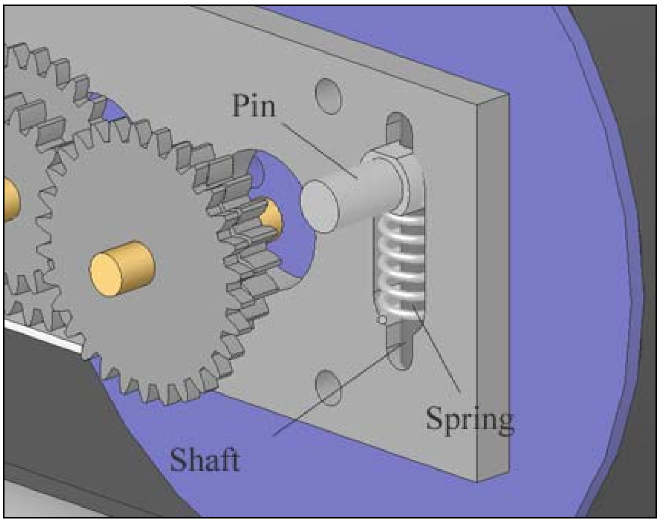


Figure **21.** Pin loaded on spring passing through 4mm shaft

There are springs on four sides and the body is screwed to the pins.
Shafts are also fixed to the plate and are placed in its slot with screws on top and bottom of them.

Construction Techniques

After talking about the design, we describe the techniques used in construction of the robot.

It should be noted that all Mechanical parts (including gears, plates, wheels, body, and cover and …) are constructed and machined by the members of the group.

Gears:

All Gears are constructed with milling method. In this method gear teeth are cut with a form milling cutter shaped to conform to the tooth space. With this method it is theoretically necessary to use a different cutter for each gear. Because a gear having 25 teeth, for example, will have a different-shaped tooth space from one having, say, 24 teeth. Actually the change in space is not too great, and it has been found that eight cutters may be used to cut with reasonable accuracy any gear in the range of 12 teeth to a rack. A separate set of cutters is, of course, required for each pitch.

In order to mill the gears, first it is needed to machine disks with diameter of outside of gear and with the same face width of gear.

Plates:

Plates are cut from 5 mm width aluminum sheet as mentioned before. All the holes and slots are then made.

Wheels:

Wheels are constructed by machining a cylinder and shaped as required. The drawings of wheels are attached to the end of the report.

Body:

Body is also constructed by cutting and shaping 3mm width aluminum sheet.

Cover:

Cover is made of 5 mm width Plexiglas sheet. This sheet is first cut out and then heated and formed as required.

8. Other Mechanisms

a) Mechanical Part

Geometrical Design

After reviewing the configuration of the arenas, team members started laying out the conceptual design of the robot and this was done by sketching out what the robot will look like. In this step we tried to keep the center of gravity of robot as low as possible to prohibit the possible turn over of the body.

What if the robot fall upside-down with all this? This was one of the questions we tried to find an answer for. After thinking about Mechanical mechanisms that could return the robot to the operating state, a physical solution was found to help the robot in this situation.

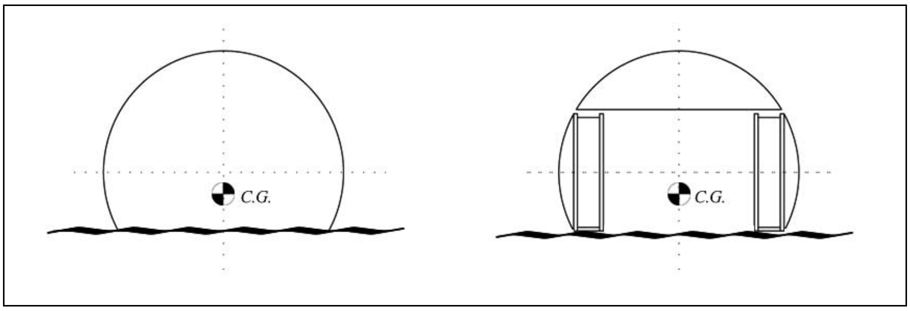


Figure 22. **The first approaches in geometrical design**

As it is shown in Fig1, the cover of robot and the wheels, along with each other, form a cylinder. So when the robot turns over, this cylindrical shape and low Center of gravity make it easy for robot to roll and fall on wheels again.

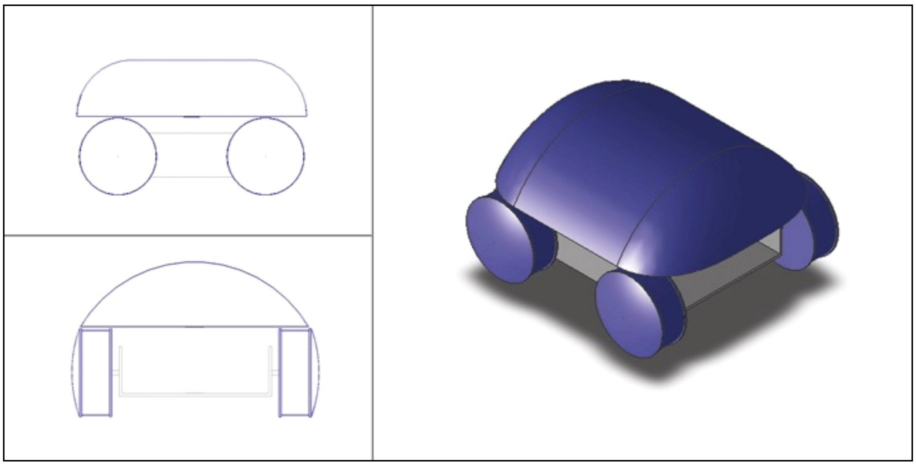


Figure 23. Sketches showing what the robot will look like

After clarifying what the robot will look like, the size of the wheels was determined. As the robot is going to work in a scattered place, with blocks of unknown size, the bigger the size of the wheels, the more passable the blocks will be. In this robot the wheels are designed to have 9 cm diameter.

Camera lifter and turner

In order to capture better scenes from the field, the robot is able to raise the camera up and also spin it in two directions. This capability enhances the view of the arena.

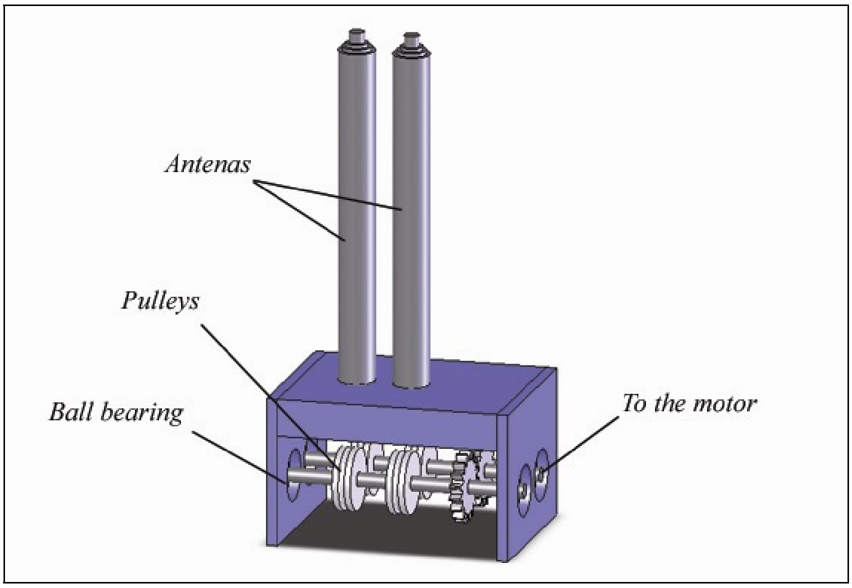


Figure 24. **Parts of camera raiser**

Camera is loaded on top of antennas. Two flexible wires pass through pulleys and go inside antennas. When the motor (and consequently shafts and pulleys) rotate, the wire will be sent into the antennas and will raise them. The mechanism for taking the camera down is just the same.

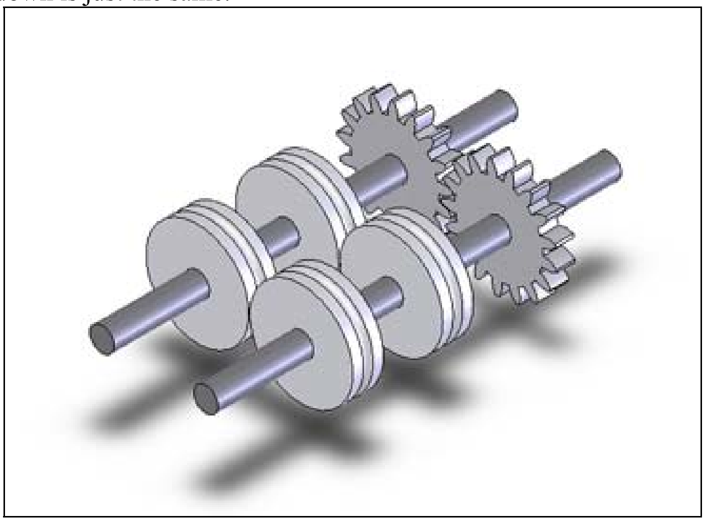


Figure 25. Shafts are connected with gears and one of them are actuated with motor

Construction Techniques

After talking about the design, we describe the techniques used in construction of the robot.

It should be noted that all Mechanical parts (including gears, plates, wheels, body, and cover and …) are constructed and machined by the members of the group.

Gears:

All Gears are constructed with milling method. In this method gear teeth are cut with a form milling cutter shaped to conform to the tooth space. With this method it is theoretically necessary to use a different cutter for each gear. Because a gear having 25 teeth, for example, will have a different-shaped tooth space from one having, say, 24 teeth. Actually the change in space is not too great, and it has been found that eight cutters may be used to cut with reasonable accuracy any gear in the range of 12 teeth to a rack. A separate set of cutters is, of course, required for each pitch.

In order to mill the gears, first it is needed to machine disks with diameter of outside of gear and with the same face width of gear.

Plates:

Plates are cut from 5 mm width aluminum sheet as mentioned before. All the holes and slots are then made.

Wheels:

Wheels are constructed by machining a cylinder and shaped as required. The drawings of wheels are attached to the end of the report.

Body:

Body is also constructed by cutting and shaping 3mm width aluminum sheet.

Cover:

Cover is made of 5 mm width Plexiglas sheet. This sheet is first cut out and then heated and formed as required.

b) Electrical Part

Microcontroller duties
Motor Drivers
Illumination
Serial Port Interface Circuit
Battery checking System
Microcontroller Software

Microcontroller duties:

Microcontroller can translate computer commands for motors, generate PWM for them, get analog data from sensors, convert it to digital signals and send it to computer using serial port. Omitting Microcontroller imposes numerous extra ICs on the board which make it larger and more expensive.

In our last robot (Championship of Iranian Intelligent Mice competitions of November 2003 in which our team achieved the first place) we used 89C51 and 89C52, but in this robot we are using AVR Microcontrollers because of numerous advantages over 89C51:

1. AVR Microcontrollers are In-System Programmable. This relieves us from taking the chip out of the main board each time we want to program it. These microcontrollers do not need a programmer set.

2. AVR Microcontrollers have internal components that make the board smaller. Using these peripheral features we can cancel external ICs. For example if we had used 8051 family ICs, we would have to use an ADC chip for Analog to Digital Conversion and a separate PWM chip for generating PWM pulses for all motors.

3. AVR Microcontrollers can drive up to 40mA in both Source and Sink modes that can drive a LED without any external buffers. This can be an advantage over 8051 family Microcontrollers that need to be buffered before connecting to LED.

4. AVR Microcontrollers are much faster than other Microcontrollers. In AVR one instruction can be executed in a clock cycle, but in 8051 family Microcontrollers one instruction needs at least twelve clock cycles to be executed.

5. AVR Microcontroller has six sleep modes. Chip can enter one of these sleep modes. In the sleep modes power consumption of Microcontroller are very low, but timers, interrupts and other important jobs of CPU are functioning. Using sleep mode we can save power. When our robot is not working for a long time chip enters the sleep mode and wakes up when operator sends it a command. So the robot does not have an ON-OFF key.

6. Very fast three wire communication between up to 127 microcontrollers or other devices. This technology called Serial Peripheral Interface (SPI). AVR microcontrollers can transfer data in Full-Duplex Synchronous mode. One microcontroller could be Master and the others are Slave.

7. AVR Microcontrollers can automatically restart the chip when CPU does not work (Watchdog Timer).

8. These Microcontrollers are very cheap. ATMega8535L costs only about three times an 89C51.

So, we have chosen ATMega8535L; its datasheet can be found in Atmel Website: http://www.atmel.com/dyn/resources/prod_documents/doc2502.pdf

Serial Port Interface Circuit

Microcontroller must send sensor data to computer on the robot and must receive commands and send them to motors. So it must communicate with computer. There are several ways of connecting a Microcontroller to a computer, for example we can use parallel port or USB port communication. But serial port is very reliable and efficient. Parallel port wastes lots of the port pins of microcontroller. For connecting a computer to a Microcontroller using serial port, we need an interface to convert RS232 standard to standard TTL. Some circuits and ICs do this, and in our robot we are using ADM232. Complete datasheet can be found in Analog Device Corporation web site.

In our robot we have omitted hand shaking. Schematic diagram of our two wire serial communication, TxD and RxD, is shown in the following figure:

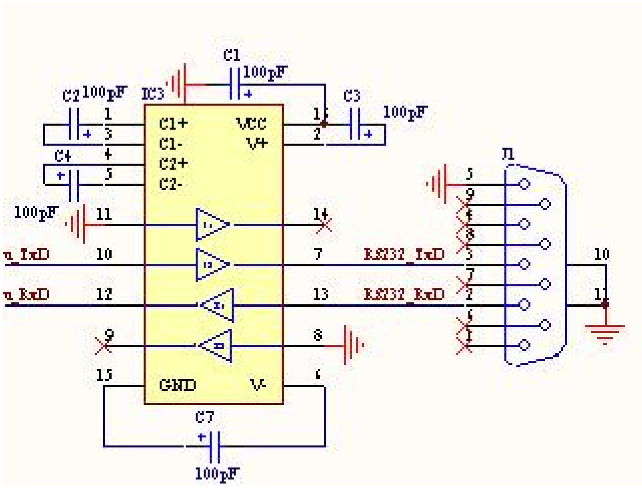


Figure 26

Illumination

The robot has some lights which can automatically or manually be turned on and off. Lights are some white LED's with significant brightness and very low power consumption. These LEDs are biased in 3.7 volts and sinks 20mA. For illumination system of our robot we need only 24 LEDs. An interesting point is their very low price. The figure below shows the LED effect in dark room:

Figure 27

Figure 28

AVR microcontroller could drive up to 40mA, so, it can drive only two LEDs when LEDs are connected directly. We have to buffer them and we have chosen 74HC541 as buffer. The buffer circuit of illumination system is shown below:

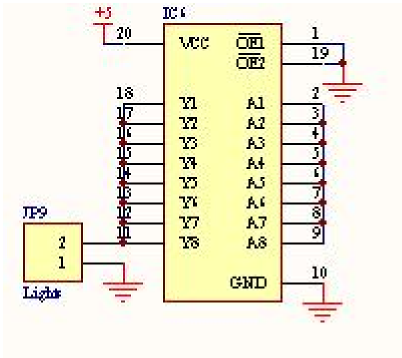


Figure 29

Battery checking System

Our robot has 24 Lithium-Ion Batteries in 6*4 packages. Each battery supplies 3.7 volts and 1.8 A, and weights only 60 grams. If a battery voltage drops under 3.5 volts, it may be harmful. Thus, a feedback from the battery voltage is sent to the microcontroller. Operator can check voltage of each battery and when it drops down the critical threshold a warning will be shown. At this situation the batteries are charged simply by plugging to 110 or 220 volt.

Motor Drivers

Our robot has five motors. Two of them are used for driving the wheels and the others are for the camera lifter,

In below table we have summarized the electrical characteristics of motors:

Table 1

Number	Name	Voltage	Free Running Current	Rotor Locked Current
1	Left wheel	20^{V}	0.3^{A}	2^{A}
2	Right wheel	20^{V}	0.3^{A}	2^{A}
3	Lifter	12^{V}	0.2^{A}	1^{A}
4	H Turner	5^{V}	0.1^{A}	0.3^{A}

5	V Turner	5^{V}	0.1^{A}	0.3^{A}

It is important to control the current of driving motors so a separate IC is used to generate PWM with limited current. In our design we have used LMD14245.

This IC has several advantages:

First, it generates a PWM using four logic input signal and allows us produce 16 different speeds for motors. The second advantage is the ability to determine the motor current and telling us whether it is higher or lower than the input reference current. And finally it is cheap. Figure shown below is the schematic circuit of these two motor drivers:

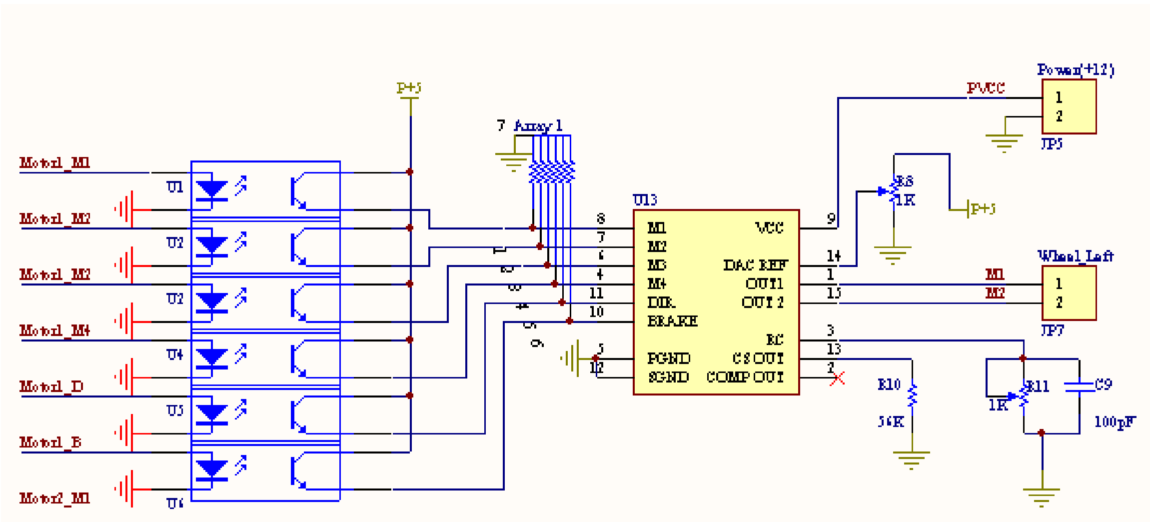


Figure 30

In this circuit, we have used 6 opto-isolators to ensure that current of motors does not harm the microcontroller. So, we have taken separate power supplies. Opto-isolation is one of the best ways to reduce noises.

For turner motors we have used L293 as shown below:

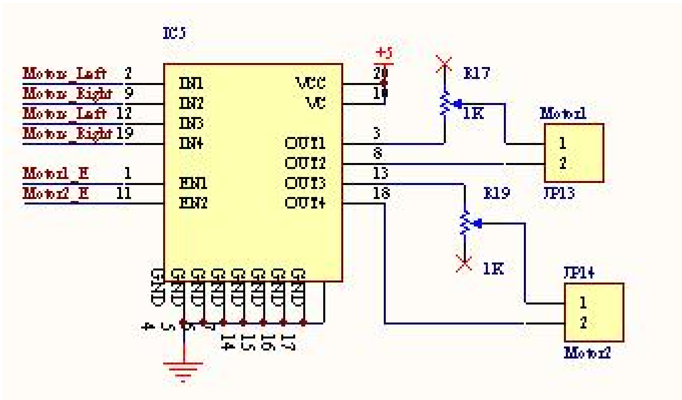


Figure 31

Microcontroller Software

In the previous sections we described hardware parts and circuits dealing with microcontroller. Now we explain the software of microcontroller. Let's take a brief look at all the responsibilities of microcontroller:

Table 2

	Responsibility	pin
1	Receiving Command and Data from Computer	RxD
2	Sending Data to Computer	TxD
3	Scanning A/D for Thermal Sensor	A/D bit0
4	Scanning A/D for Light Sensor	A/D bit1
5	Scanning A/D for Battery Situation	A/D bit 2
6	Generating PWM for three motors	3 I/O pins
7	Applying Motor Command to Motor of wheels	2*6 I/O pins
8	Monitoring Data on a LCD	7 I/O pins
9	Receive distance from second microcontroller and send it to computer	8 I/O pins

As you can see in the table, we do not have any interrupts. Computer of our robot is Master and microcontroller is Slave. When computer wants to change the speed of motor, it sends a message to microcontroller and ask microcontroller to do it. When computer asks for sensor situation, it sends a request message and then the microcontroller sends back sensor status. This configuration is quite ideal because when the microcontroller chip resets, it can start its work without any problem. The general flowchart of the software of microcontroller is shown here:

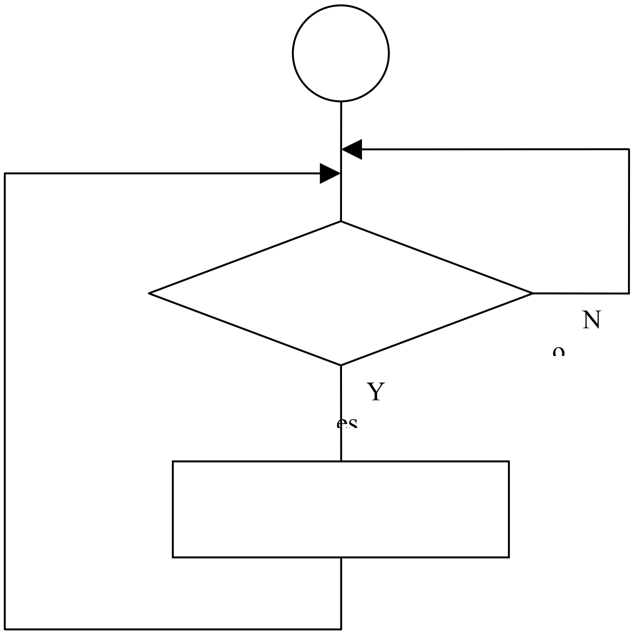


Figure 32

The computer on robot receives commands from the operator’s system and sends them to its serial port. Operator’s interface software and microcontroller must have same commands with same concepts:

Table 3 – 'R means Receive and T means Transmit

Com-mand Byte	Description	Argument
1	Turn on lifter motor	None
2	Turn off lifter motor	None
3	Direction of lifter motor: Clock Wise	None
4	Direction of lifter motor: Counter Clock Wise	None
5	Speed of lift motor	R(amount of speed)
6	Change left wheel motor status	R(new command)
7	Change right wheel motor status	R(new command)
8	Turn on lights	None
9	Turn off lights	None
10	Change display of LCD	R(position,Data)
11	Send temperature	T (Temp.)

12	Send light sensor value	T (Value)
13	Send battery status	R (Battery Number) T(Status)
14	Send distance from walls	R(Angle) T(distance)

We wrote our program in C language and compiled it with CodeVisionAVR:
http://www.hpinfotech.ro/
Now see the code:

```
/*********************************************
© Copyright 1998-2003 HP InfoTech s.r.l.
http://www.hpinfotech.ro
e-mail:office@hpinfotech.ro

Project :
Version :
Date    : 2/26/2004
Author  : Freeware, for non-commercial use only
Company :
Comments:


Chip type           : ATmega8535L
Program type        : Application
Clock frequency     : 4.433600 MHz
Memory model        : Small
External SRAM size  : 0
Data Stack size     : 128
*********************************************/

#include <mega8535.h>

// Alphanumeric LCD Module functions
#asm
   .equ __lcd_port=0x15
#endasm
#include <lcd.h>

// Standard Input/Output functions
#include <stdio.h>

// Declare your global variables here
char k;
void TurnOnMotor3(void);
void TurnOffMotor3(void);
void CWMotor3 (void);
void CCWMotor3 (void);
void ChangeMotor1(void);
void ChangeMotor2(void);
void TurnOnLED (void);
void TurnOffLED (void);
void ChangeLCD(void);
void Temperature (void);
void Light_Intensity(void);

void main(void)
{
// Declare your local variables here
```

```
  // Input/Output Ports initialization
  // Port A initialization
  // Func0=In Func1=In Func2=In Func3=In Func4=In Func5=In
Func6=In Func7=In
  // State0=T State1=T State2=T State3=T State4=T State5=T
State6=T State7=T
  PORTA=0x00;
  DDRA=0x00;

  // Port B initialization
  // Func0=In Func1=In Func2=In Func3=In Func4=In Func5=In
Func6=In Func7=In
  // State0=T State1=T State2=T State3=T State4=T State5=T
State6=T State7=T
  PORTB=0x00;
  DDRB=0x00;

  // Port C initialization
  // Func0=In Func1=In Func2=In Func3=In Func4=In Func5=In
Func6=In Func7=In
  // State0=T State1=T State2=T State3=T State4=T State5=T
State6=T State7=T
  PORTC=0x00;
  DDRC=0x00;

  // Port D initialization
  // Func0=In Func1=In Func2=In Func3=In Func4=In Func5=In
Func6=In Func7=In
  // State0=T State1=T State2=T State3=T State4=T State5=T
State6=T State7=T
  PORTD=0x00;
  DDRD=0x00;

  // Timer/Counter 0 initialization
  // Clock source: System Clock
  // Clock value: Timer 0 Stopped
  // Mode: Normal top=FFh
  // OC0 output: Disconnected
  TCCR0=0x00;
  TCNT0=0x00;
  OCR0=0x00;

  // Timer/Counter 1 initialization
  // Clock source: System Clock
  // Clock value: Timer 1 Stopped
  // Mode: Normal top=FFFFh
  // OC1A output: Discon.
  // OC1B output: Discon.
  // Noise Canceler: Off
  // Input Capture on Falling Edge
```

```
  TCCR1A=0x00;
  TCCR1B=0x00;
  TCNT1H=0x00;
  TCNT1L=0x00;
  OCR1AH=0x00;
  OCR1AL=0x00;
  OCR1BH=0x00;
  OCR1BL=0x00;

  // Timer/Counter 2 initialization
  // Clock source: System Clock
  // Clock value: Timer 2 Stopped
  // Mode: Normal top=FFh
  // OC2 output: Disconnected
  ASSR=0x00;
  TCCR2=0x00;
  TCNT2=0x00;
  OCR2=0x00;

  // External Interrupt(s) initialization
  // INT0: Off
  // INT1: Off
  // INT2: Off
  MCUCR=0x00;
  MCUCSR=0x00;

  // Timer(s)/Counter(s) Interrupt(s) initialization
  TIMSK=0x00;

  // USART initialization
  // Communication Parameters: 8 Data, 1 Stop, No Parity
  // USART Receiver: On
  // USART Transmitter: On
  // USART Mode: Asynchronous
  // USART Baud rate: 1200 (Double Speed Mode)
  UCSRA=0x02;
  UCSRB=0x18;
  UCSRC=0x86;
  UBRRH=0x01;
  UBRRL=0xCD;

  // Analog Comparator initialization
  // Analog Comparator: Off
  // Analog Comparator Input Capture by Timer/Counter 1:
Off
  // Analog Comparator Output: Off
  ACSR=0x80;
  SFIOR=0x00;

  // LCD module initialization
```

```
lcd_init(16);
lcd_clear();
lcd_putsf("Initializing...");
while (1)
      {
      // Place your code here
    k=getchar();
    putchar(k);

    switch (k)
    {
    case 1: TurnOnMotor3(); break;
    case 2: TurnOffMotor3(); break;
    case 3: CWMotor3(); break;
    case 4: CCWMotor3(); break;
//  case 5: ChangePWMMotor3(); break;
    case 6: ChangeMotor1();
    case 7: ChangeMotor2();
    case 8: TurnOnLED();
    case 9: TurnOffLED();
    case 10: ChangeLCD();
    case 11: Temperature();
    case 12: Light_Intensity();
    }

      };
}

void TurnOnMotor3(void)
{

lcd_gotoxy(0,1);
lcd_putsf("Motor3: ON ");
putsf("M3: On");
PORTB.4 = 1; // Pin 5
}

void TurnOffMotor3(void)
{
lcd_gotoxy(0,1);
lcd_putsf("Motor3: OFF");
putsf("M3: Off");
PORTB.4 = 0; // Pin 5
}

void CWMotor3 (void)
{
lcd_gotoxy(0,1);
lcd_putsf("Motor3: CW ");
putsf("M3: Clock Wise ");
```

```
PORTC.3 = 0; // pin 25
PORTA.2 = 1; // pin 38
}

void CCWMotor3 (void)
{
lcd_gotoxy(0,1);
lcd_putsf("Motor3: CCW ");
putsf("M3: Counter Clock Wise");
PORTC.3 = 1; // pin 25
PORTA.2 = 0; // pin 38

}

/*
void ChangePWMMotor3(void)
(
char data_in1;
lcd_gotoxy(0,1);
lcd_putsf("Motor3: PWM ");
putsf("M3: PWM");
data_in1=getchar();
)
*/

void ChangeMotor1(void)
{
char data_in1;
lcd_gotoxy(0,1);
lcd_putsf("Motor1:     ");
putsf("M1");
data_in1=getchar();
lcd_putsf("");
}

void ChangeMotor2(void)
{
char data_in1;
lcd_gotoxy(0,1);
lcd_putsf("Motor2:    ");
putsf("M2");
data_in1=getchar();
data_in1=data_in1 & 0b00011111 ;
PORTA.4 = ((data_in1 & (1<<0) )== (1<<0)); // pin 36
PORTA.5 = ((data_in1 & (1<<1) )== (1<<1)); // pin 35
PORTA.6 = ((data_in1 & (1<<2) )== (1<<2)); // pin 34
PORTA.7 = ((data_in1 & (1<<3) )== (1<<3)); // pin 33
PORTD.7 = ((data_in1 & (1<<4) )== (1<<4)); // pin 21
}
```

```
void TurnOnLED (void)
{
lcd_gotoxy(0,1);
lcd_putsf("LED: ON     ");
putsf("LED: ON");
PORTA.3 = 1; // pin 37
}

void TurnOffLED (void)
{
lcd_gotoxy(0,1);
lcd_putsf("LED: OFF    ");
putsf("LED: OFF");
PORTA.3 = 0; // pin 37
}

void ChangeLCD(void)
{
char position,data;
lcd_gotoxy(0,1);
lcd_putsf("LCD:Wait...");
putsf("LCD:Wait..  ");
position = getchar();
data = getchar();
lcd_gotoxy(position,0);
lcd_putchar(data);
}

void Temperature (void)
{
lcd_gotoxy(0,1);
lcd_putsf("Sensor:    ");
putchar('a');

}

void Light_Intensity(void)
{

lcd_gotoxy(0,1);
lcd_putsf("Light Int.: ");
putchar('a');

}
```

The code has three major parts. First, we set register value. Second we write main loop and finally we define some small functions that must be executed when related command is received.

We can not go through register description thoroughly as it is a vast subject. Further discussion is available in the IC datasheet at:

http://www.atmel.com/dyn/resources/prod_documents/doc2502.pdf

Our main loop is running without any interrupt. At the beginning of the loop, program waits to receive commands from operator's computer. Then according to the received command, related function is done and then it comes back to the first command line of loop and it goes on.

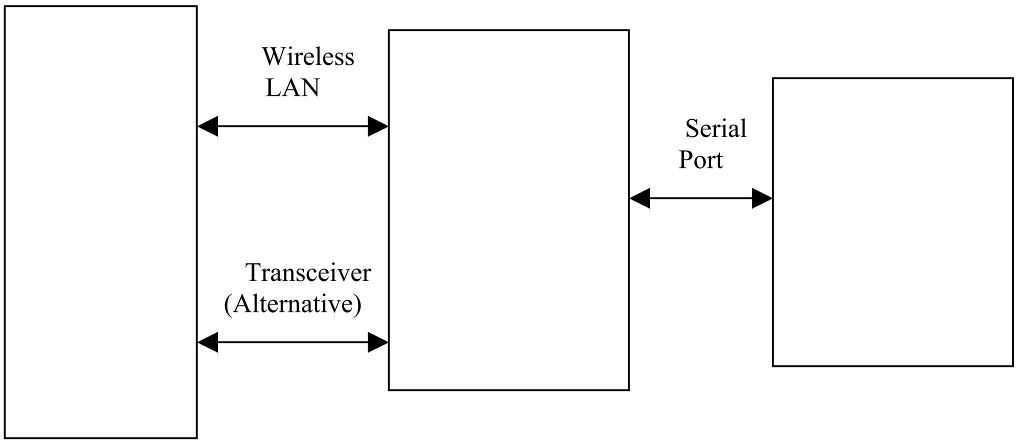


Figure 33

c. Software part

Resquake Robot Station:

Resquake Robot Station will be setup on each robot and receives commands from operator and sends video and data to the operator through wireless LAN (or with the backup system). It uses the same subroutines described in Resquake Operator Interface as it is working with sockets and serial port. Here we have serial communication with transceivers and microcontroller. This program does not need a graphical interface and is preferably designed in VC++. This part is still under construction.

Resquake Report Reader:

Right after the end of each mission, Resquake Operator Interface generates three outputs automatically.

First one is a complete report of what has happened during the mission and what has been found in each arena with lots of pictures and all the available data coming from sensors during the mission and also the map is partially generated by the software (as we are taking advantage of positioning system the software can learn many things using the data coming from the positioning system. Of course the operator helps the software to generate a better map at last).

Second report will be burnt on CD and will be handed to the judge. This CD contains all the data in the printed report in addition to some mp3 and mpg files which are recorded whenever an important event occurs in each arena. The CD contains a program which is able to open our file type. This program is called Resquake Report Reader and most part of it is already done using database (mdb files and Microsoft Jet Engine), but it is still under construction as we are not completely done with the other robots. This software will be designed using VB.

We try to publish a third report on Internet so the report would be available for a committee or organization who are making a complete rescue team and possibly are not in the disaster scene in the real case of earthquake.

9. Team Training for Operation (Human Factors)

Resquake has tried to build a very simple set using whatever the team members could learn and implement. The devices are going to be plug and play and there should be no complicated training needed for the user. Resquake Operator Interface has the least possible objects in the best and easily accessible places and may need something about 10 minutes to describe what an operator should do to control the system. It is also very easy to setup Hardware parts, and we do our best to finalize the system in the way they only would need power supply and no other setup.

For team member training, as we were all involved with the implementation procedure no training will be needed. But of course we need to practice several times before the main competition. We try to make a similar field in the university and practice.

For getting to this point and building this robot all the members has learnt many things and has read many books. What we really learnt during these months is not comparable to any part of life of any one of us.

10. Possibility for Practical Application to Real Disaster Site

Resquake has done his best to make a robust system practical for the real disasters. There are some new ideas for working in real disaster, but also there are some limitations at least until the end of the competition.

New idea is publishing the report on Internet which makes the data available for the ones who are not present at the scene yet, but should do something for the victims or send commands to the rescuers.

The limitation is the wireless LAN which solves the problem in small areas but not in the real site. It means we are to improve the communication system for operation in longer distances.

11.System Cost

These are the costs of each robot (all in US Dollars)

Table 4

	Part Name	Qty.	Total Price
Mechanical Part	Motors	5	60
	Gear box	5	80
	Structure	1	100
	Other Mechanical costs	1	100
Electronic Part	Wireless Camera	1	50
	Data Transceiver	2	80
	Positioning System	1	100
	Other Electronic boards	1	85
Computer Part	Mother board	1	500
	Wireless LAN and Access point	2	200
	AV to USB adaptor	1	60
	Laptop	1	1000
Sum			2415

References

1. Shigley, Joseph Edward: Mechanical Engineering Design, 6th edn. , McGraw-Hill
2. Martin George Henry: Kinematics and Dynamics of Machines
3. McComb, Gordon: The Robot Builders Bonanza, McGraw-Hill, 2000
4. Oberg, Erik: Machinery Handbook, 26th edn. , Industrial Press, 2000
5. MSDN Library, Microsoft Corporations

Appendix 1: Mechanical Drawings

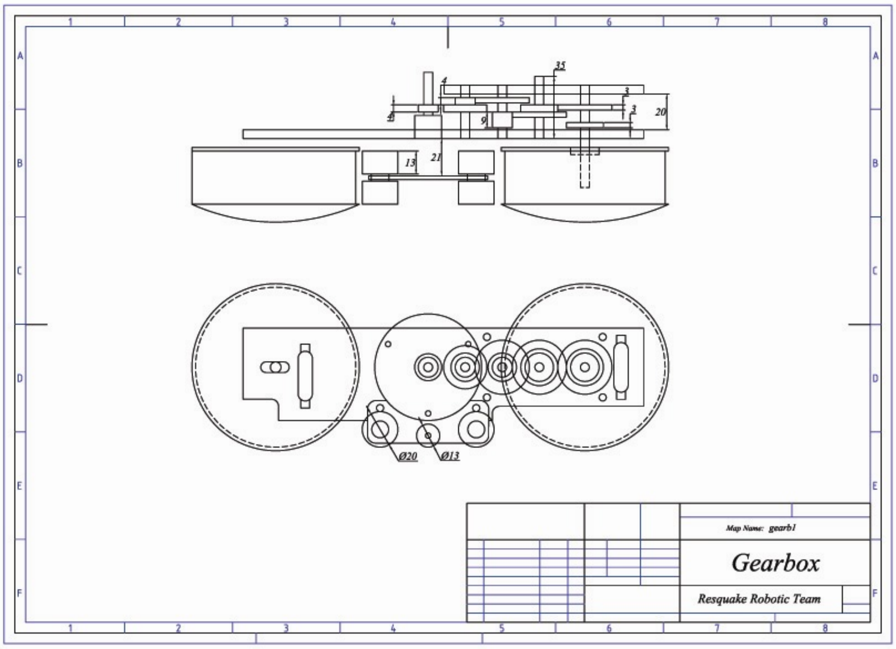


Figure 34

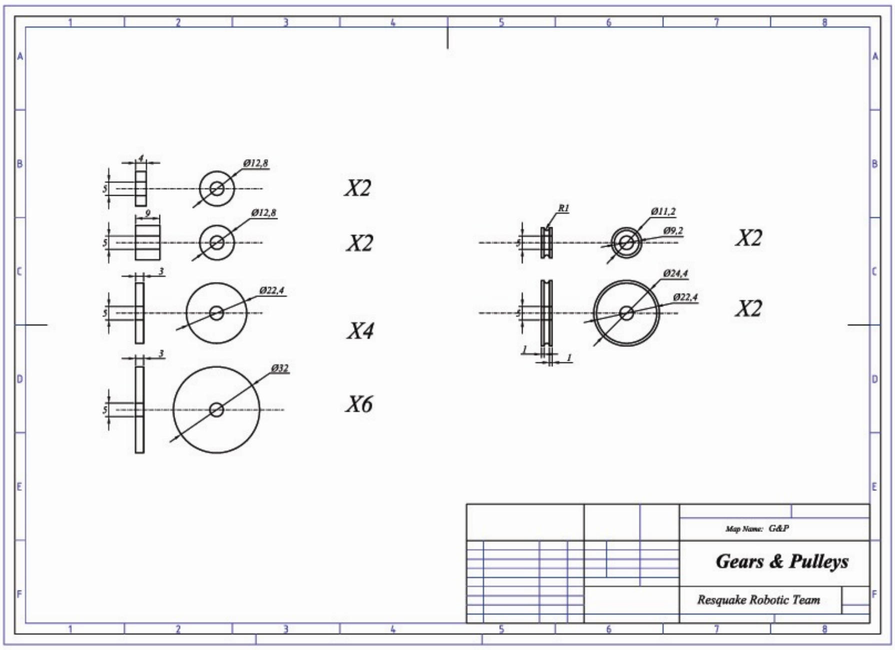


Figure 35

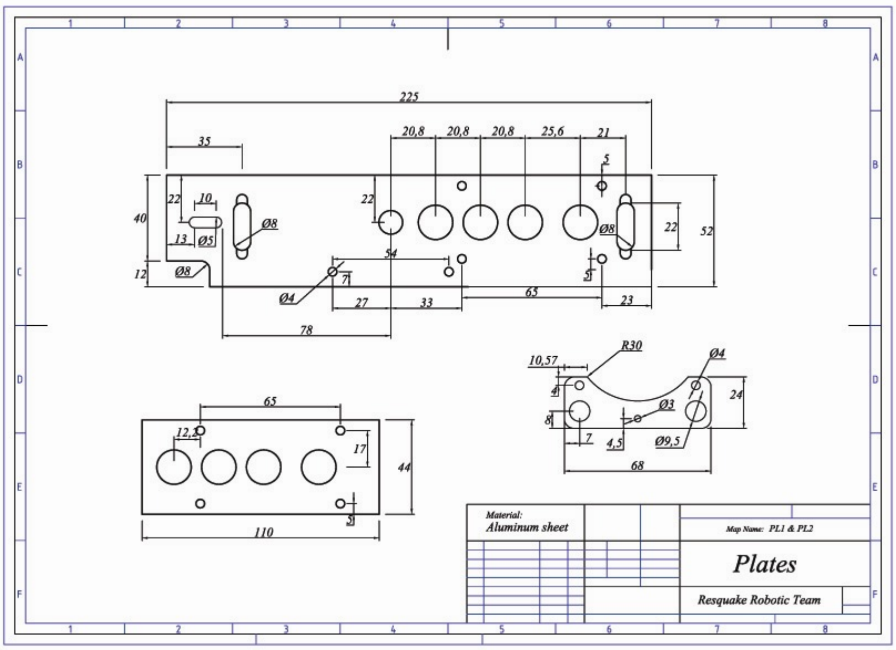
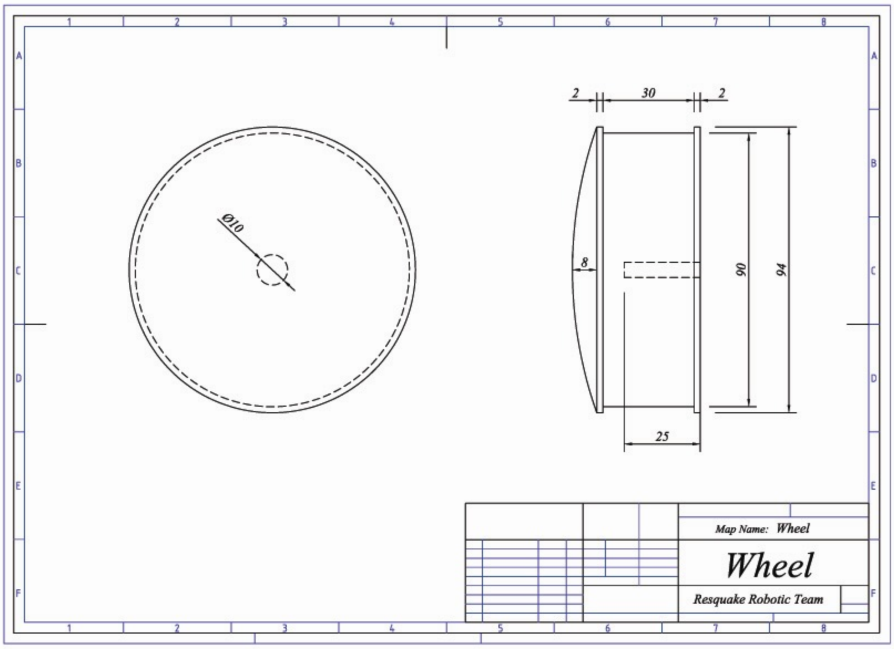


Figure 36



Ø10
2
30
2
8
90
94
25
Map Name: Wheel
Wheel
Resquake Robotic Team

Figure 37

Appendix 2: Photo Gallery

Figure 38

Figure 39

Figure 40

Figure 41

Figure 42

Figure 43

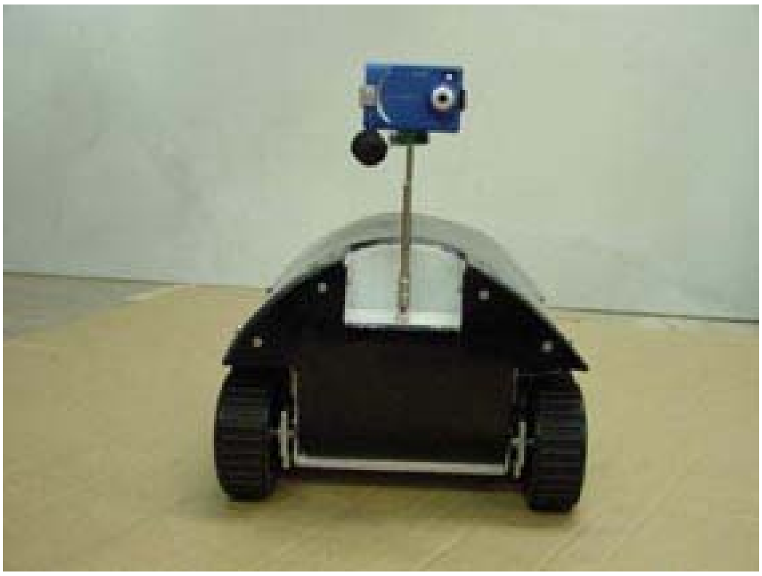


Figure 44

Figure 45

Figure 46

Figure 47

Figure 48

Figure 49

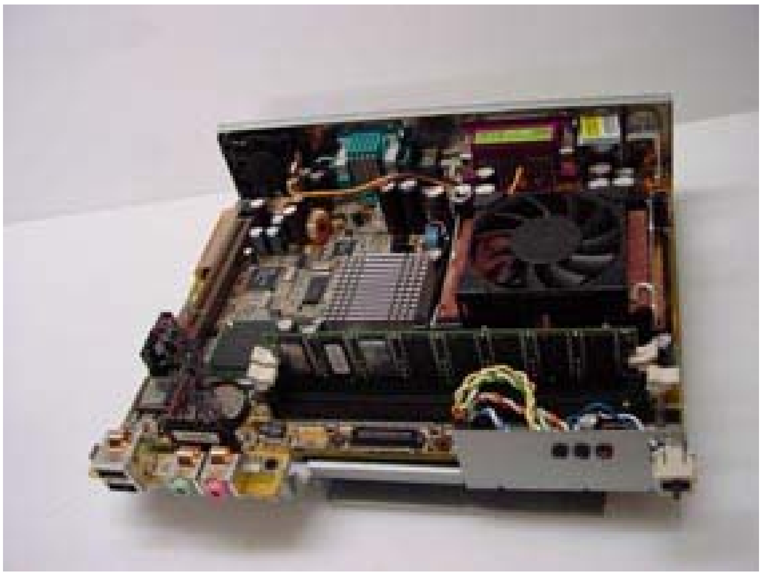


Figure 50

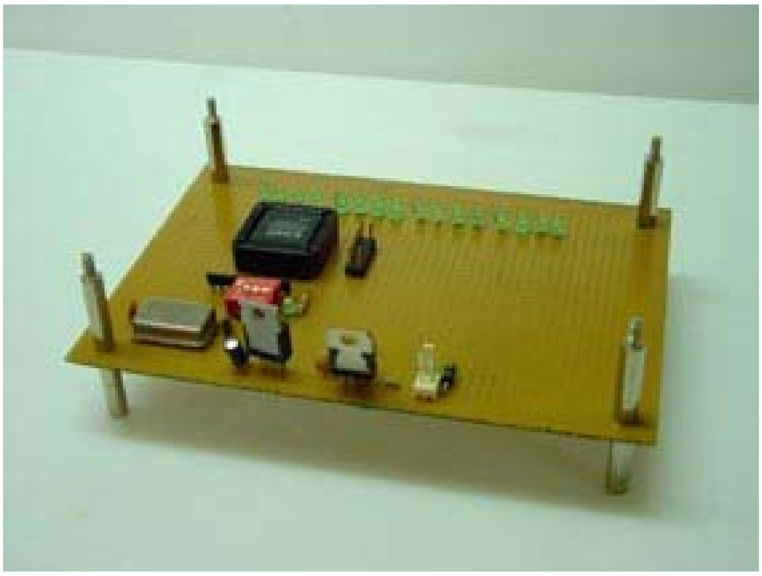


Figure 51

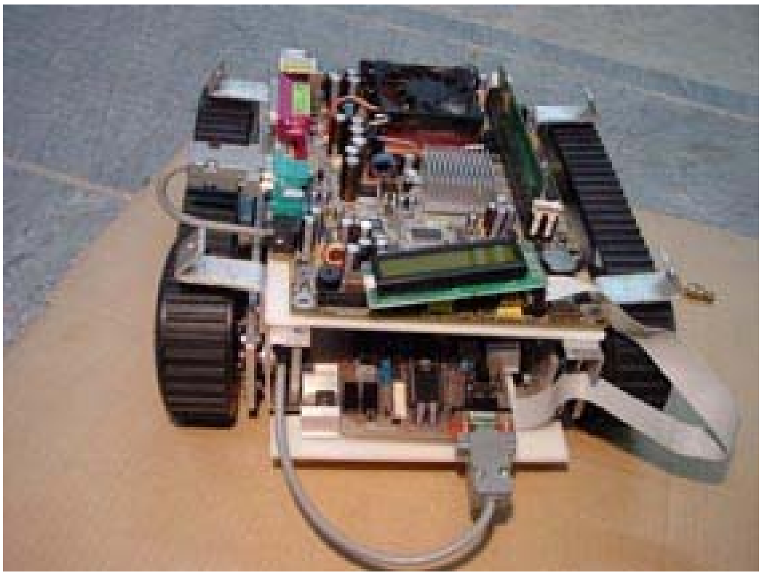


Figure 52

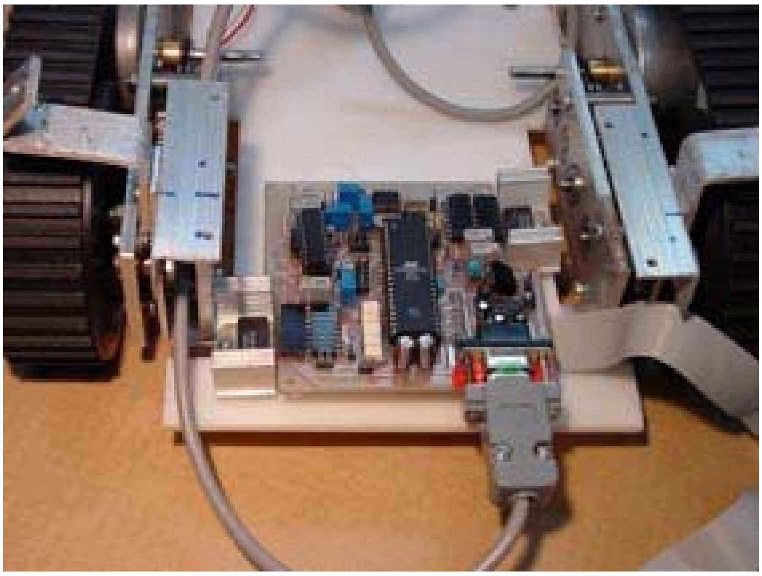


Figure 53

Figure 54

Figure 55

Figure 56

Figure 57

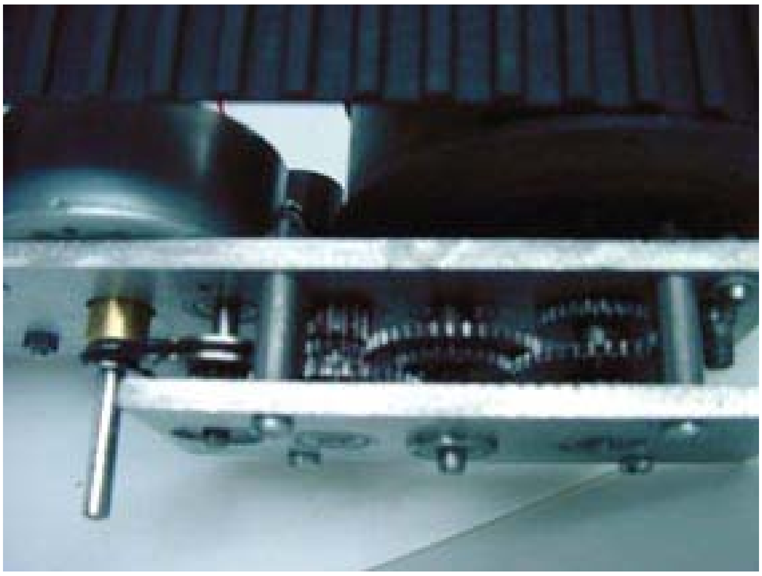


Figure 58

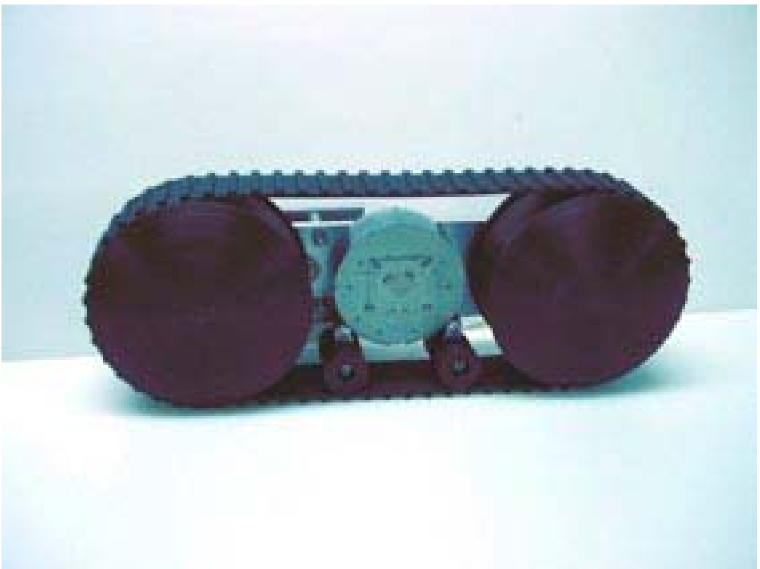


Figure 59

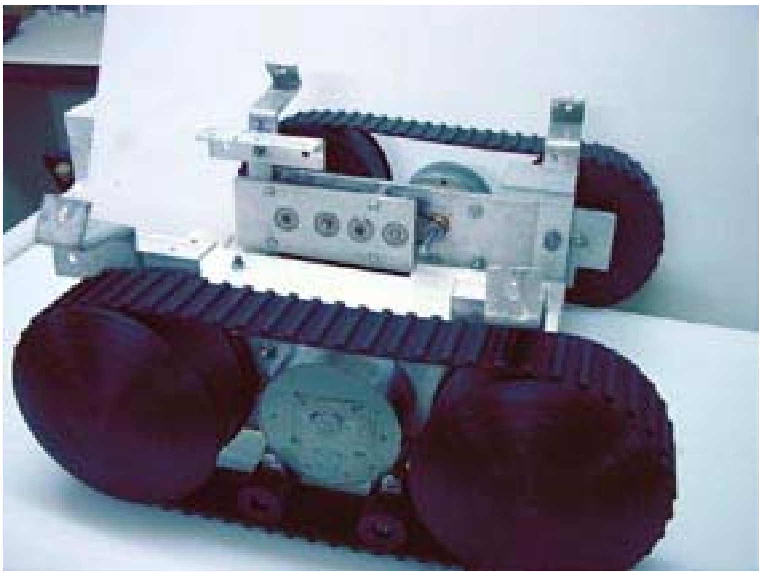


Figure 60

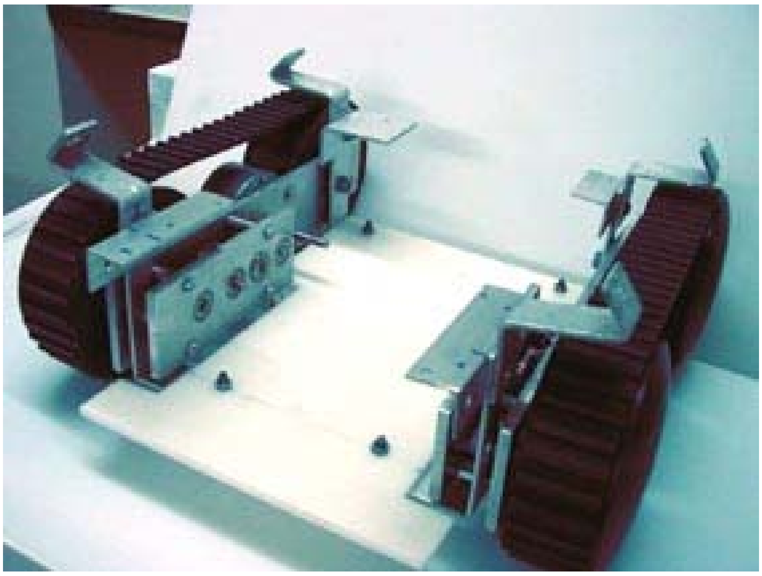


Figure 61

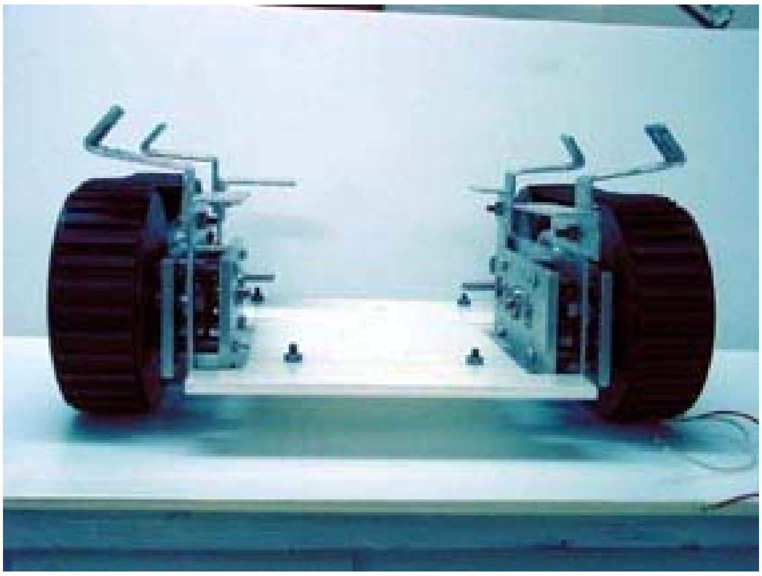


Figure 62

Figure 63

Figure 64

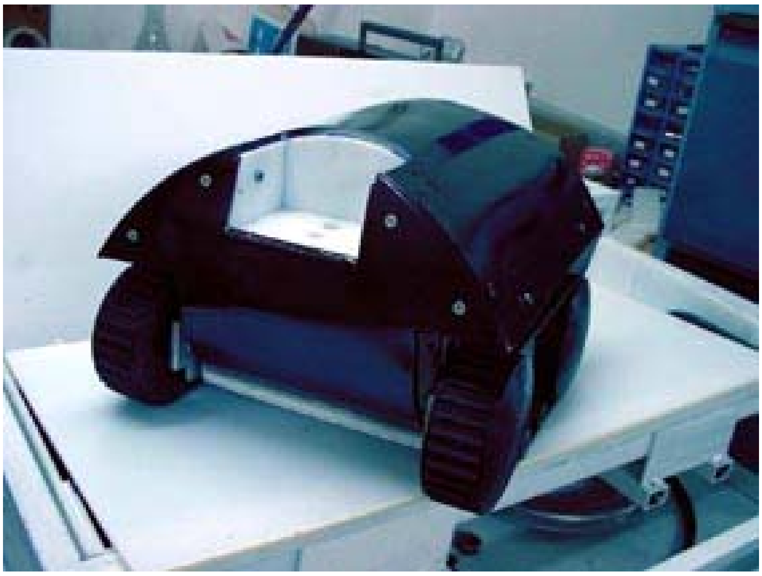


Figure 65

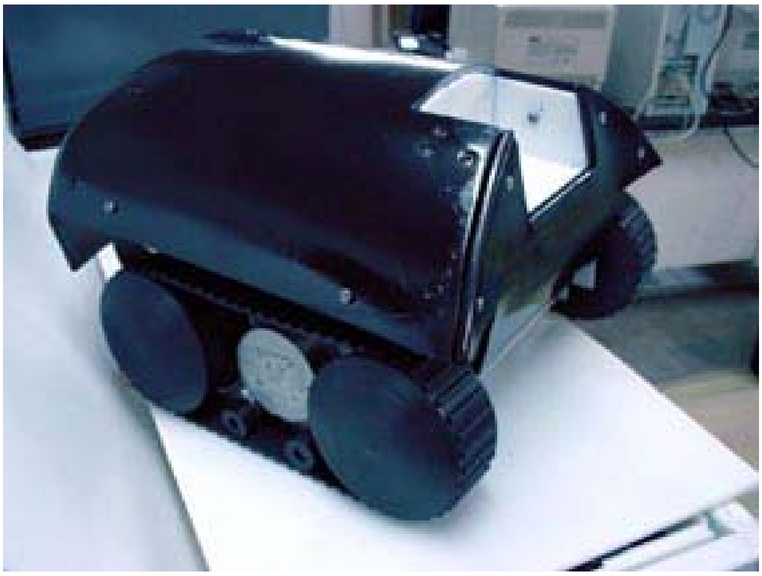


Figure 66

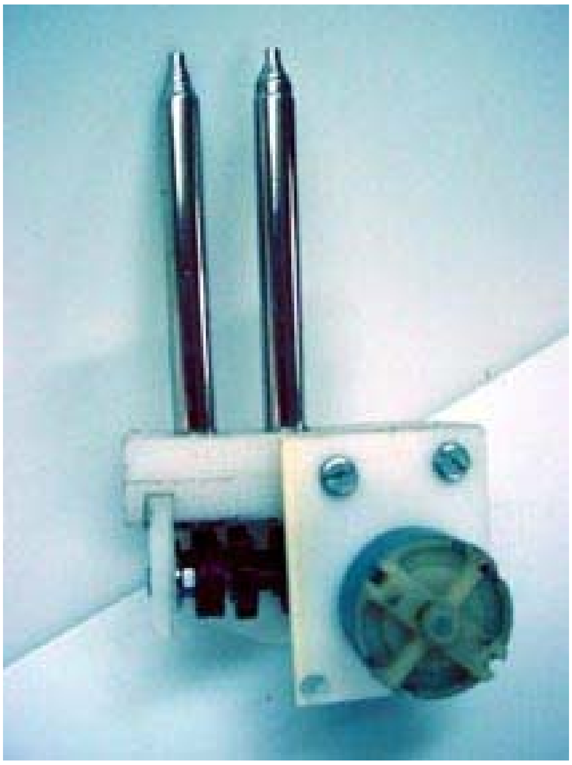


Figure 67

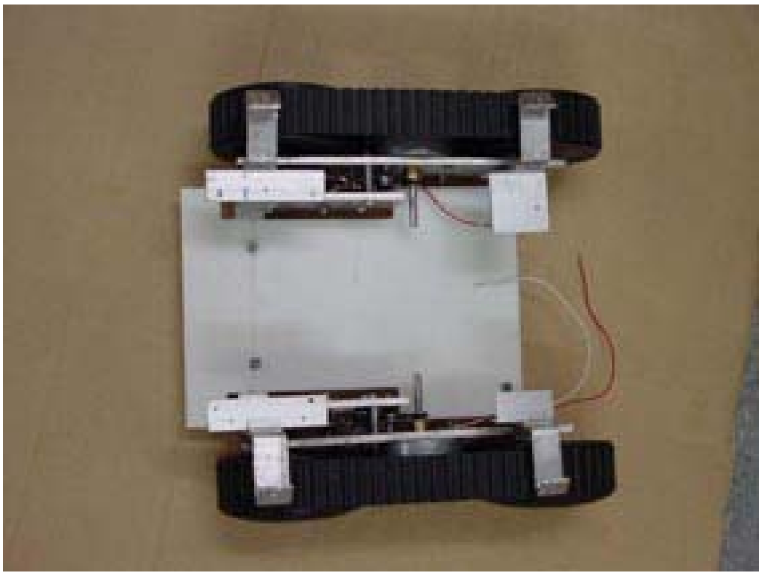


Figure 68

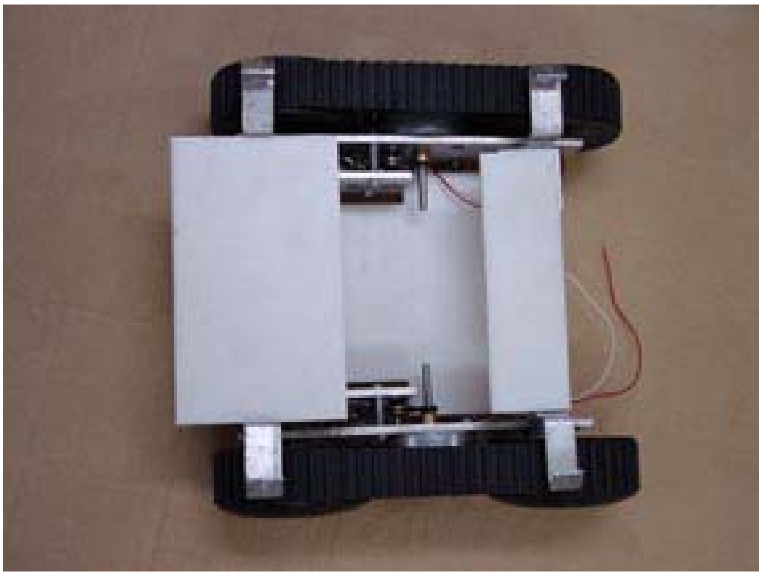


Figure 69

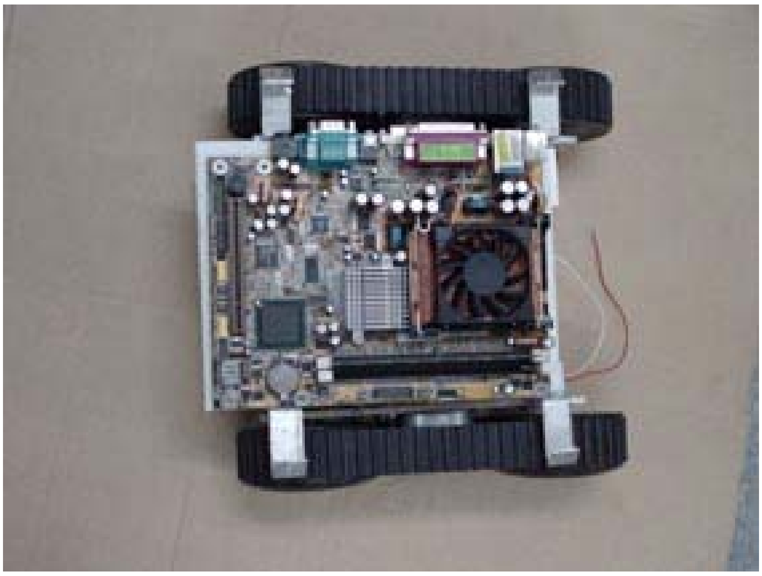


Figure 70