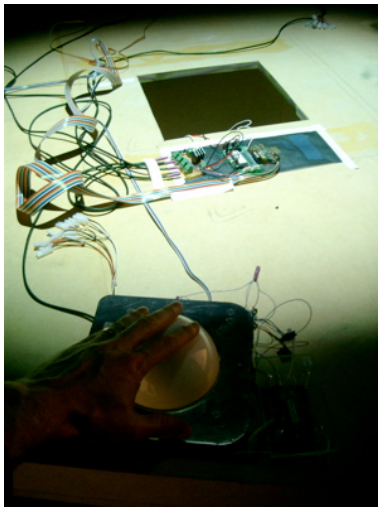
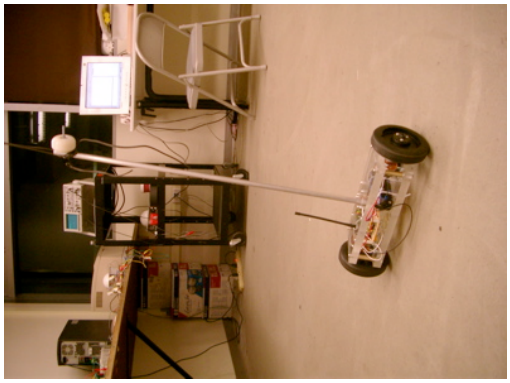
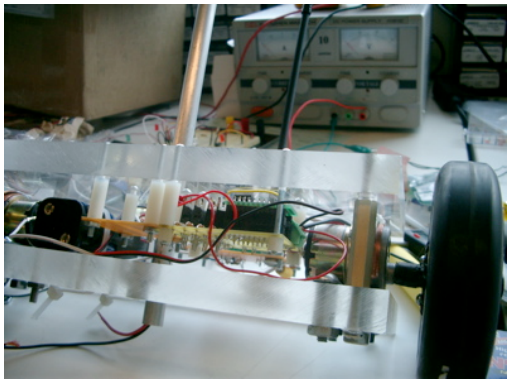
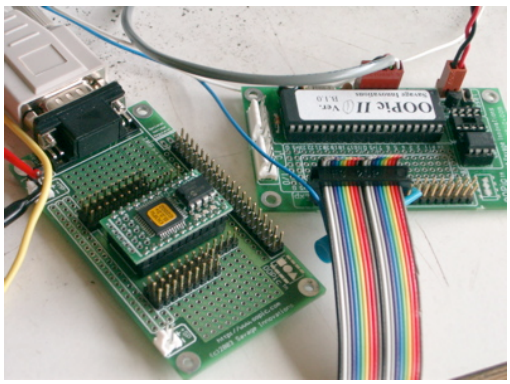
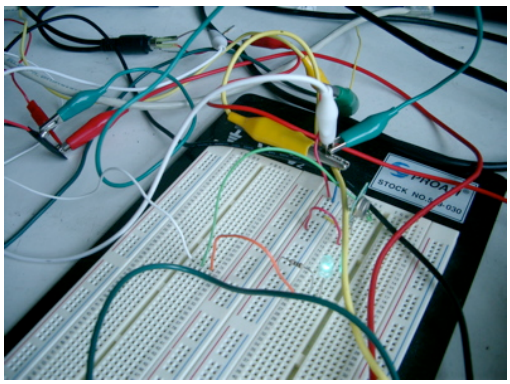
ROBOTIC SPHERES - Technical rider

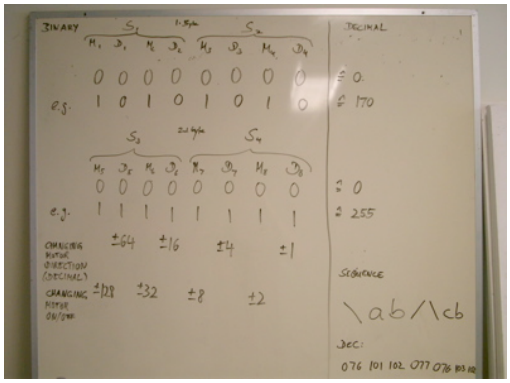

left: Kids 'interacting' with the robotic spheres; middle: Atomic Manipulation Table setup; right: nanobot prototype.

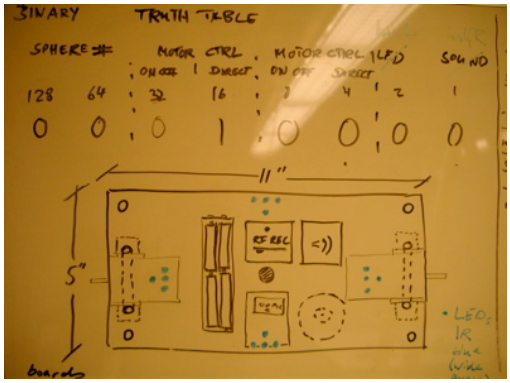



left: testing battery life, force, and durability. Rod is used in the sphere to keep the robotic device in an upright position as well as connector to recharge the device without opening the sphere (modified mini-audio-jack)
right: all components (RF receiver, driver-board, motors, batteries) are sandwiched between 1" acrylic sheets which provide weight and durability. Also visible: the weakest part of the robotic device, a steal connector between motor axle and wheel. Kids kicking the sphere made the several pounds heavy device tumble inside the sphere which made the steal connector and motor gears break. The challenge is in the right balance between overall weight of the whole device (including Polycarbonate sphere), the force created by the robotic device (which needs a certain weight to be able to move the sphere), battery life and mechanical parts.

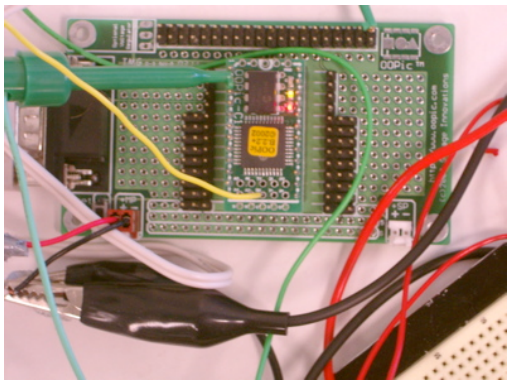



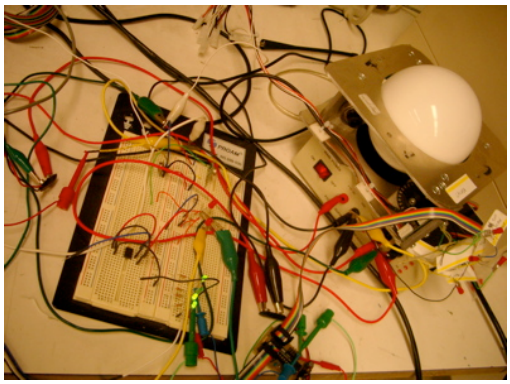
left: OOPic B II (transmitter micro-controller, takes in all 4 trackballs) and OOPic C (receiver micro-controller); right: the Laipac RF Module transmitter (www.laipac.com, Model TLP and RLP A 434, for better receiver quality a WLP 434 Antenna has been used)



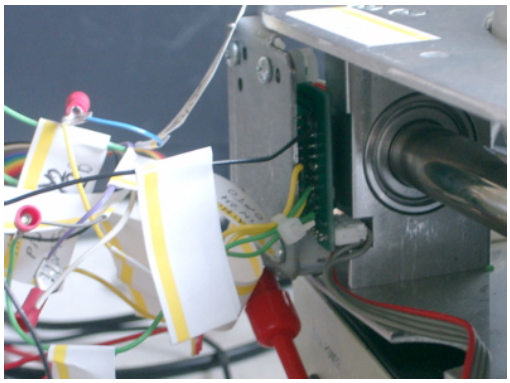


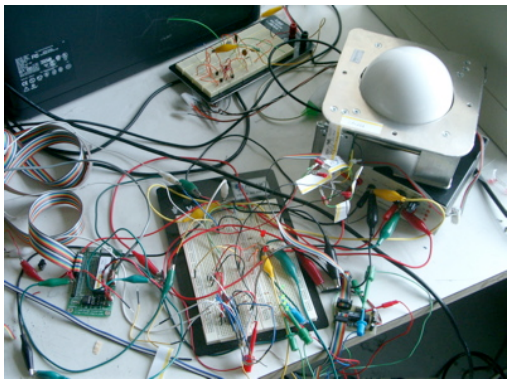
left: design of the 8 bit (1 WORD) command structure. The first 2 bits determine which sphere to talk to (1-4), bit 3 turns left motor on and off, bit 4 determine the left motor's direction, bit 5 and 6 do the same thing for the right motor, bit 7 and 8 determine if the motors are pulsed (speed control)
right: command scheme and layout of the robotic device



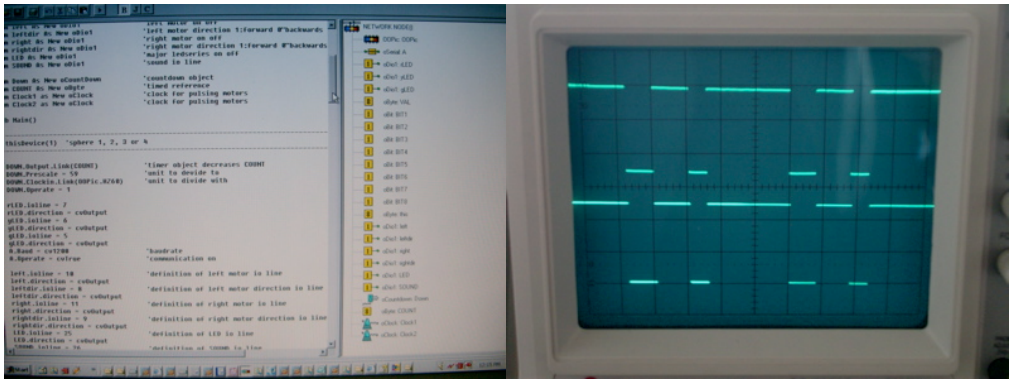


left: OOPic C micro-controller, receives commands via RF; right: reading our the trackballs' optical converters and transforming into directional information (front, front-right, right, back-right, back, back-left, left, front-left.





left: wire attached directly attached to the optical converter of the Happ Controls trackballs (http://www.happcontrols.com/index.html?http://www.happcontrols.com/trackballs/55022600.htm!). Due to the high pulse speed speed of the optical converter, a customized ATMEL chip is used to determine speed and directional information. The result is then processed into an OOPic B micro-controller which sends control commands to the individual spheres (see OOPic B Transmitter code)



left: OOpic Programming Environment; right Test of the 8 bit RF signal (upper section sender, lower section receiver signal)

Serial Communication:

OOPic B Transmitter Code:

```
' (c) Daniel Sauter, 2003. contact@daniel-sauter.com
' trackball transmitter prgram for serial communication (RLP/TLP 434
www.laipac.com)
' processed input from trackball values by atmel microchip

Dim A As New oSerialX                   'io pin for TX communication

Dim X1_1 As new oDio1                   'front Trackball 1
Dim X2_1 As new oDio1                   'back Trackball 1
Dim Y1_1 As new oDio1                   'right Trackball 1
Dim Y2_1 As new oDio1                   'left Trackball 1

Dim X1_2 As new oDio1                   'front Trackball 2
Dim X2_2 As new oDio1                   'back Trackball 2
Dim Y1_2 As new oDio1                   'right Trackball 2
Dim Y2_2 As new oDio1                   'left Trackball 2

Dim X1_3 As new oDio1                   'front Trackball 3
Dim X2_3 As new oDio1                   'back Trackball 3
Dim Y1_3 As new oDio1                   'right Trackball 3
Dim Y2_3 As new oDio1                   'left Trackball 3

Dim X1_4 As new oDio1                   'front Trackball 4
Dim X2_4 As new oDio1                   'back Trackball 4
Dim Y1_4 As new oDio1                   'right Trackball 4
Dim Y2_4 As new oDio1                   'left Trackball 4

Dim Val1 as new oByte                   'output value for TX communication Sphere 1
Dim Val2 as new oByte                   'output value for TX communication Sphere 2
Dim Val3 as new oByte                   'output value for TX communication Sphere 3
Dim Val4 as new oByte                   'output value for TX communication Sphere 4

Const S1 = 0                        'sets bit 1 and 2 to 0      —◡ Sphere 1
Const S2 = 64                       'sets bit 1 to 0 and bit 2 to 1  —◡ Sphere 2
Const S3 = 128                          'sets bit 1 to 1 and bit 2 to 0 —◡ Sphere 3
Const S4 = 192                          'sets bit 1 and 2 to 1      —◡ Sphere 4

Sub Main()

 A.IOLineS = 10                         'io pin for TX communication
 A.Baud = cv1200                        'Baud rate

 X1_1 = cvInput
 X1_1.ioLine = 23                   'T1 front
 X2_1 = cvInput
 X2_1.ioLine = 24                   'T1 back
 Y1_1 = cvInput
 Y1_1.ioLine = 22                   'T1 right
```

```
 Y2_1 = cvInput
 Y2_1.ioLine = 25                    'T1 left

 X1_2 = cvInput
 X1_2.ioLine = 21                    'T2 front
 X2_2 = cvInput
 X2_2.ioLine = 26                    'T2 back
 Y1_2 = cvInput
 Y1_2.ioLine = 20                    'T2 right
 Y2_2 = cvInput
 Y2_2.ioLine = 27                    'T2 left

 X1_3 = cvInput
 X1_3.ioLine = 19                    'T3 front
 X2_3 = cvInput
 X2_3.ioLine = 28                    'T3 back
 Y1_3 = cvInput
 Y1_3.ioLine = 18                    'T3 right
 Y2_3 = cvInput
 Y2_3.ioLine = 29                    'T3 left

 X1_4 = cvInput
 X1_4.ioLine = 17                    'T4 front
 X2_4 = cvInput
 X2_4.ioLine = 30                    'T4 back
 Y1_4 = cvInput
 Y1_4.ioLine = 16                    'T4 right
 Y2_4 = cvInput
 Y2_4.ioLine = 31                    'T4 left

 call init

 do
     'A.value = 190
     call move
 loop
end sub



Sub move()


'Trackball 1
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''


    if X1_1.value = 1 then                   'if front
        if Y1_1.value = 1 then               'and right
            Val1 = 62
        end if
        if Y2_1.value = 1 then               'and left
            Val1 = 61
        end if
        if Y1_1.value = 0 then               'and nothing (only front)
            if Y2_1.value = 0 then
                Val1 = 63
            end if
        end if
    else                     'if neither front nor back (only right-left)
        if X2_1.value = 0 then
            if Y1_1.value = 1 then
                Val1 = 54
            else
                if Y2_1.value = 0 then
                    Val1 = 0
                end if
            end if
            if Y2_1.value = 1 then
                Val1 = 29
            end if
        end if
    end if
    if X2_1.value = 1 then                   'if back
        if Y1_1.value = 1 then               'and right
            Val1 = 42
        end if
```

```
		if Y2_1.value = 1 then				'and left
			Val1 = 41
		end if
		if Y1_1.value = 0 then				'and nothing (only front-back)
			if Y2_1.value = 0 then
				Val1 = 43
			end if
		end if
	end if


	'A.Value = Val1


'Trackball 2
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

	if X1_2.value = 1 then				'if front
		if Y1_2.value = 1 then				'and right
			Val2 = 126
		end if
		if Y2_2.value = 1 then				'and left
			Val2 = 125
		end if
		if Y1_2.value = 0 then				'and nothing (only front)
			if Y2_2.value = 0 then
				Val2 = 127
			end if
		end if
	else
					'if neither front nor back (only right-left)
		if X2_2.value = 0 then
			if Y1_2.value = 1 then
				Val2 = 136
			else
				if Y2_2.value = 0 then
					Val2 = 64
				end if
			end if
			if Y2_2.value = 1 then
				Val2 = 93
			end if
		end if
	end if
	if X2_2.value = 1 then				'if back
		if Y1_2.value = 1 then				'and right
			Val2 = 106
		end if
		if Y2_2.value = 1 then				'and left
			Val2 = 105
		end if
		if Y1_2.value = 0 then				'and nothing (only front-back)
			if Y2_2.value = 0 then
				Val2 = 107
			end if
		end if
	end if


	'A.Value = Val2


'Trackball 3
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

	if X1_3.value = 1 then				'if front
		if Y1_3.value = 1 then				'and right
			Val3 = 190
		end if
		if Y2_3.value = 1 then				'and left
			Val3 = 189
		end if
		if Y1_3.value = 0 then				'and nothing (only front)
			if Y2_3.value = 0 then
				Val3 = 191
			end if
```

```
			end if
		else
								'if neither front nor back (only right-left)
			if X2_3.value = 0 then
				if Y1_3.value = 1 then
					Val3 = 182
				else
					if Y2_3.value = 0 then
						Val3 = 128
					end if
				end if
				if Y2_3.value = 1 then
					Val3 = 157
				end if
			end if
		end if
		if X2_3.value = 1 then				'if back
			if Y1_3.value = 1 then			'and right
				Val3 = 170
			end if
			if Y2_3.value = 1 then			'and left
				Val3 = 169
			end if
			if Y1_3.value = 0 then			'and nothing (only front-back)
				if Y2_3.value = 0 then
					Val3 = 171
				end if
			end if
		end if


		A.Value = Val3
		A.Value = Val3
		A.Value = Val3


	'Trackball 4
	''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

		if X1_4.value = 1 then				'if front
			if Y1_4.value = 1 then			'and right
				Val4 = 254
			end if
			if Y2_4.value = 1 then			'and left
				Val4 = 253
			end if
			if Y1_4.value = 0 then			'and nothing (only front)
				if Y2_4.value = 0 then
					Val4 = 255
				end if
			end if
		else
						'if neither front nor back (only right-left)
			if X2_4.value = 0 then
				if Y1_4.value = 1 then
					Val4 = 246
				else
					if Y2_4.value = 0 then
						Val4 = 192
					end if
				end if
				if Y2_4.value = 1 then
					Val4 = 250
				end if
			end if
		end if
		if X2_4.value = 1 then				'if back
			if Y1_4.value = 1 then			'and right
				Val4 = 234
			end if
			if Y2_4.value = 1 then			'and left
				Val4 = 233
			end if
			if Y1_4.value = 0 then			'and nothing (only front-back)
				if Y2_4.value = 0 then
					Val4 = 235
```

```
                end if
            end if
        end if


        'A.Value = Val4


Sub init()
 Val1 = 0
 Val2 = 0
 Val3 = 0
 Val4 = 0
End Sub
```

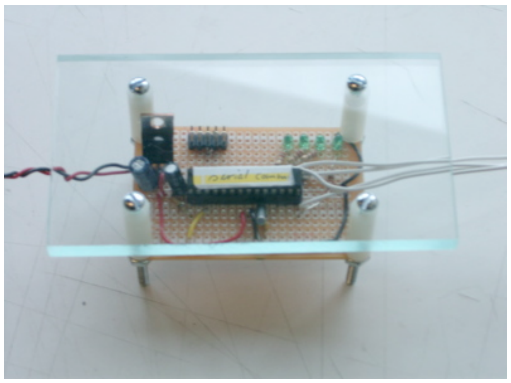
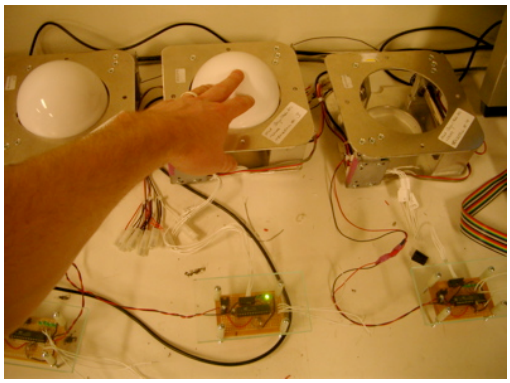
Receiver code, Sphere 1

```
'(c) Daniel Sauter, 2003. contact@daniel-sauter.com
'contact@daniel-sauter.com
'This program takes any serial
'data that a oSerial Object
'has received and sends it
'back out the oSerial Object.

Dim rLED As New oDio1               'onboard red LED
Dim yLED As New oDio1               'onboard yellow LED
Dim gLED As New oDio1               'onboard green LED

Dim VAL As New oSerial              'incoming value

Dim left As New oDio1               'left motor on off
Dim leftdir As New oDio1            'left motor direction 1:forward 0"backwards
Dim right As New oDio1              'right motor on off
Dim rightdir As New oDio1           'right motor direction 1:forward 0"backwards

Dim frontLED As New oDio1           'IR LEDs
Dim backLED As New oDio1
Dim leftLED As New oDio1
Dim rightLED As New oDio1

Dim Down As New oCountDown          'countdown object
Dim COUNT As New oByte              'timed reference

Sub Main()

 DOWN.Output.Link(COUNT)          'timer object decreases COUNT
 DOWN.Prescale = 59               'unit to devide to
 DOWN.Clockin.Link(OOPic.HZ60)    'unit to divide with
 DOWN.Operate = 1

 rLED.ioline = 7
 rLED.direction = cvOutput
 yLED.ioline = 6
 yLED.direction = cvOutput
 gLED.ioline = 5
 gLED.direction = cvOutput
 VAL.Baud = cv1200                    'baudrate
 VAL.Operate = cvTrue             'communication on

 left.ioline = 10                 'definition of left motor io line
 left.direction = cvOutput
 leftdir.ioline = 8               'definition of left motor direction io line
 leftdir.direction = cvOutput
 right.ioline = 11                'definition of right motor io line
 right.direction = cvOutput
 rightdir.ioline = 9              'definition of right motor direction io line
 rightdir.direction = cvOutput

 frontLED.ioline = 29
 frontLED.direction = cvOutput
```

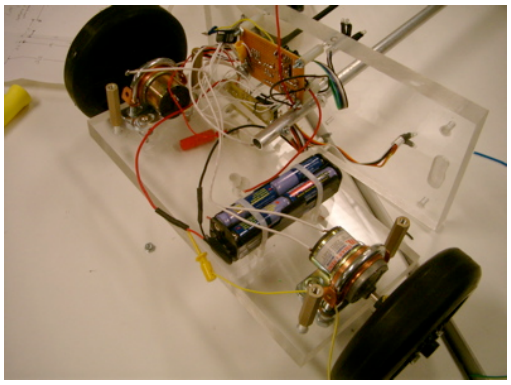
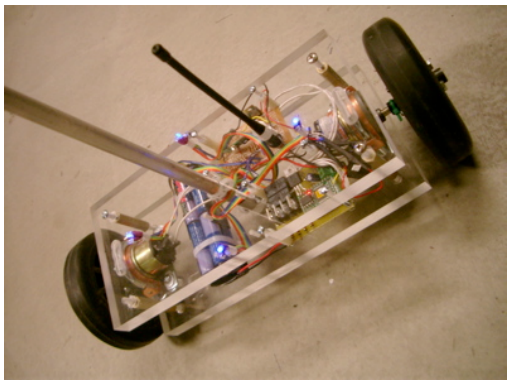
```
backLED.ioline = 26
backLED.direction = cvOutput
leftLED.ioline = 27
leftLED.direction = cvOutput
rightLED.ioline = 28
rightLED.direction = cvOutput

call init

Do
If COUNT.nonzero = 0 then          'if countdown comes true, switch everything
off
        call init
end if

If VAL.Received = cvTrue then
    count = 4

    if VAL.value >= 0 then
    if VAL.value <= 64 then

        rLED.invert                    'if data received red LED changes value


        Select Case VAL.value
            Case 63             'forward
                call LED(1,0,0,0)
                leftMotor(1,1)
                leftDirection(1)
                rightMotor(1,1)
                rightDirection(1)
            Case 62             'forward right
                call LED(1,0,0,1)
                leftMotor(1,1)
                leftDirection(1)
                rightMotor(1,0)
                rightDirection(1)
            Case 61             'forward left
                call LED(1,0,1,0)
                leftMotor(1,0)
                leftDirection(1)
                rightMotor(1,1)
                rightDirection(1)
            Case 43             'backward
                call LED(0,1,0,0)
                leftMotor(1,1)
                leftDirection(0)
                rightMotor(1,1)
                rightDirection(0)
            Case 42             'backward right
                call LED(0,1,0,1)
                leftMotor(1,1)
                leftDirection(0)
                rightMotor(1,0)
                rightDirection(0)
            Case 41             'backward left
                call LED(0,1,1,0)
                leftMotor(1,0)
                leftDirection(0)
                rightMotor(1,1)
                rightDirection(0)
            Case 54             'right
                call LED(0,0,0,1)
                leftMotor(1,1)
                rightMotor(0,0)
                if leftdir = 1 then
                    leftDirection(1)
                else
                    leftDirection(0)
                end if
            Case 29             'left
                call LED(0,0,1,0)
                leftMotor(0,0)
                rightMotor(1,1)
                if rightdir = 1 then
                    rightDirection(1)
```

```
                    else
                        rightDirection(0)
                    end if
                Case 0
                    call init
            End Select
        end if
        end if
 end if

 yLED.value = left.value                    'yellow status LED for left motor
 gLED.value = right.value                   'green status LED for right motor




Sub leftMotor(lonoff as Byte, dirl as Byte)
        if lonoff = 1 then
            if dirl = 1 then
                left = 1
            end if
            if dirl = 0 then
                left = oopic.Hz1
            end if
        elseif lonoff = 0 then
            if dirl = 1 then
                left = oopic.Hz1
            end if
            if dirl = 0 then
                left = 0
            end if
        end if
End Sub

Sub rightMotor(ronoff as Byte, dirr as Byte)
        if ronoff = 1 then
            if dirr = 1 then
                right = 1
            end if
            if dirr= 0 then
                right = oopic.Hz1
            end if
        elseif ronoff = 0 then
            if dirr= 1 then
                right = oopic.Hz1
            end if
            if dirr= 0 then
                right = 0
            end if
        end if
End Sub

Sub leftDirection(ldir as Byte)
        if ldir = 1 then
            leftdir.value = 1
        elseif ldir = 0 then
            leftdir.value = 0
        end if
End Sub

Sub rightDirection(rdir as Byte)
        if rdir = 1 then
            rightdir.value = 1
        elseif rdir = 0 then
            rightdir.value = 0
        end if
End Sub

Sub init()                          ' initializing 8 bits which hold binary information of
incoming value
 right.value = 0
 left.value = 0
 LED(0,0,0,0)
End Sub
```

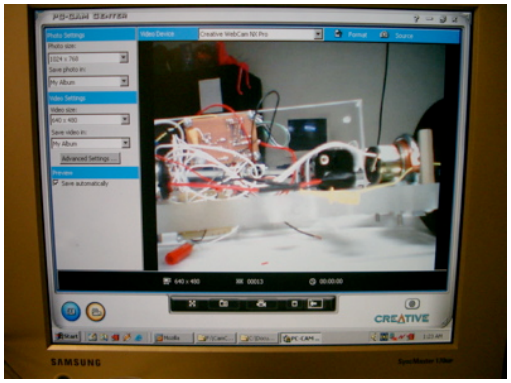
```
Sub LED(one as boolean, two as boolean, three as boolean, four as boolean)
 frontLED.value = one
 backLED.value = two
 leftLED.value = three
 rightLED.value = four
End Sub
```

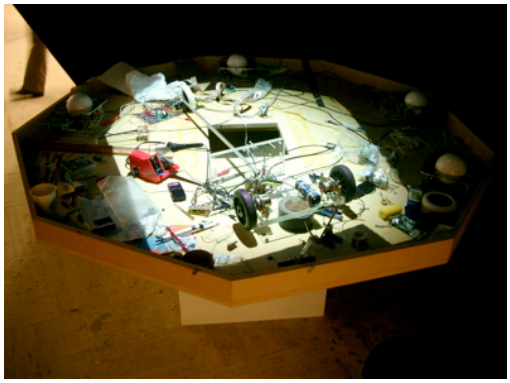
left and right: ATMEL chip reading out the optical converter of the trackballs, transforming them into directional information

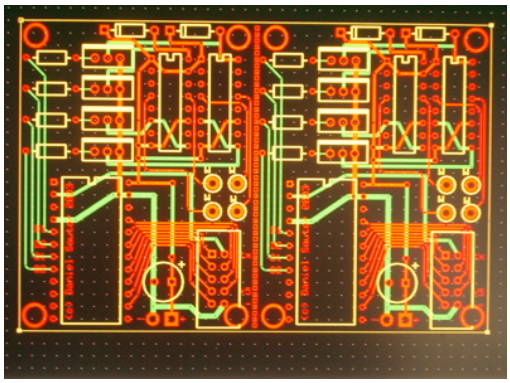
left: robotic device assembly and the finished device. The 4 super-bright LEDs make the opaque sphere slightly glow blue.



left: testing the infrared LEDs with a infrared-sensitive webcam. The infrared LEDs, pointing in all 4 directions give a visual response to where the sphere is going. The 4 birds-eye view tracking cameras have use an infrared gel to block all "visible" light. A processed picture of this birds-eye vies is projected as a visual feedback of the interaction onto the Atomic Manipulation Table where visitors control the spheres with the trackballs. Moving the trackballs causes the infrared LEDs in the spheres to glow, highlighting the section of the sphere where it is going to.




left: installing the four trackballs in the atomic manipulation table. right: birds-eye view projected onto the table showing people interacting with the spheres which are controlled by the trackballs.



left: design of the driver board for two motors and the OOPic C receiver micro-controller.

Power: powered by 8 rechargeable AA batteries (1000 mA/hours each). 5 - 9 hours (full/normal load). 1" hole in the spheres allows to recharge the batteries without opening the sphere through a modified mini-audio-jack connector
Motors: Geared motors 20-30 rpm
Electronic parts: Jameco, DigiKey, Radioshack
Polycarbonate spheres: Gavrilei Plastics (http://www.gavrieli.com/).
The spheres have been made of two 1/4" white polycarbonate domes (36" diameter), joint by a customized Polycarbonate flange.
Trackballs: Happ Controls (http://www.happcontrols.com/)
RF Transmitter/Receiver: Laipac (http://laipac.com)

Designed by Daniel Sauter (www.daniel-sauter.com), Steve Boyer, Adam Stieg, Ted Chung