




## Robô de telepresença com a estrutura toda feita em PVC.

### Componentes Elétricos:

Quantidade	Descrição	Foto
1 und	Raspberry PI 2	
1 und	<u>Arduíno NANO</u>	
2 und	Ponte H	
2 und	Motor de vidro elétricos universal	

### Ponte H com mosfet:

Projeto da ponte H desenvolvido pelo, [Daniel O. Basconcello Filho](#), do site [robotizando](#).

Link do site: ([http://www.robotizando.com.br/artigo\\_ponte\\_h\\_pg1.php](http://www.robotizando.com.br/artigo_ponte_h_pg1.php));








### Componentes da ponte H:

- 2 x capacitores 10uF / 50V
- 1 x capacitor 100nF / 100V
- 1 x capacitor 1000uF / 25V
- 1 capacitor 10nF / 100V
- 2 x diodos 1N4007
- 2 x transistores BC548
- 2 x transistores BC558





<b>Quantidade</b>	<b>Descrição</b>	<b>Foto</b>
1 m	Cano de esgoto 40mm	
2 und	T de cano de esgoto 40mm	
2 und	Terminal de PVC 40mm	
1 und	Joelho de 45° PVC	
2 und	Abraçadeira de PVC	
1 und	Caixa de passagem de PVC;	

1 und	Cola para PVC	
0,5 m	Cano de esgoto 150mm (pode ser de largura maior);	
2 und	Terminal de PVC 150mm (rodas)	
0,5 m	EVA preto	
15 und	Parafusos com rosca e porca;	
2 und	Conectores reto sem rosca 1/2" (pode ser soldado um parafuso direto no eixo dos motores)	
1 und	Massa Epoxi (para solda os conectores reto sem rosca 1/2" no eixo do motores)	

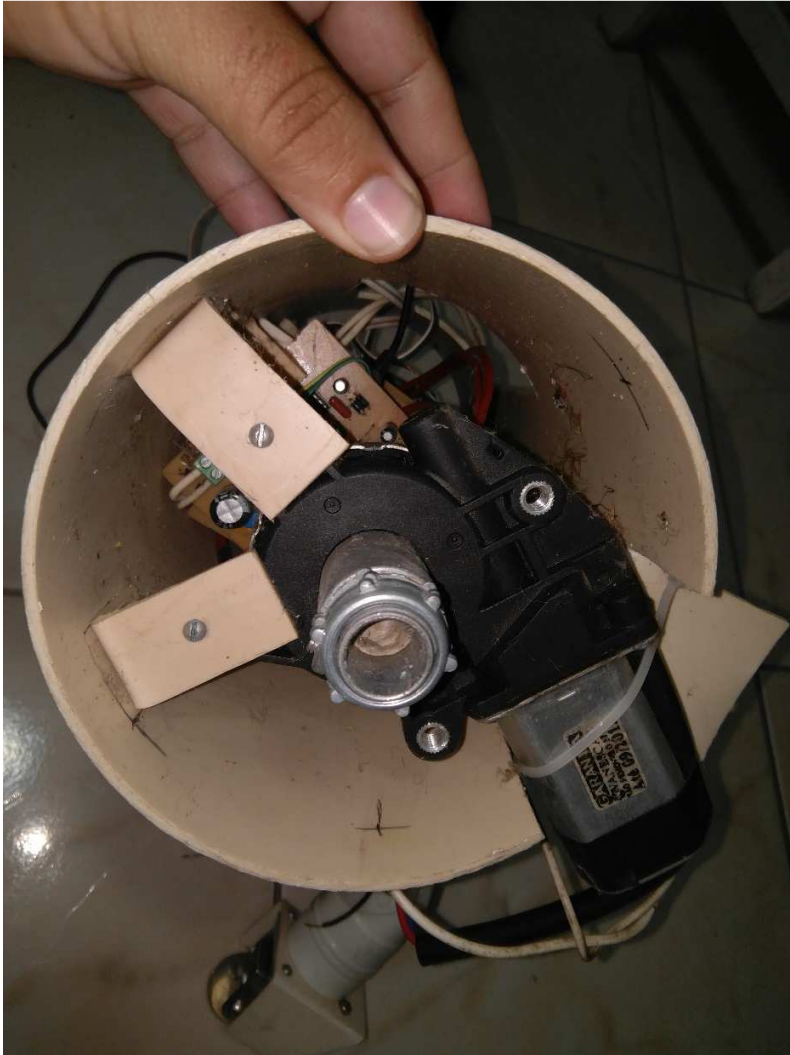
**Base do robô:**



**Conexão estrutura vertical**



## Detalhe da conexão do moto









**Programação:**

**Código de programação do arduíno:**

```
// Pinos do motor 1
// Pinos: 10 direito - 11 esquerda
#define motor1D 10
#define motor1E 11

// Pinos do motor 2
// Pinos: 9 direito - 3 esquerda
#define motor2D 9
#define motor2E 3

//Não foi utilizado as portas PWM 5 e 6 para evitar
// diferença de rotação do motor pois as mesmas utilização
// 1KHz ao contrario da portas 3,9,10,11 que utilizam 500Hz

//Pinos de entrada
#define v12 0 //Medição da bateria de 12 volts (Motores)
#define v5 1 //Medição da bateria de 5 volts (Arduino e
Raspberry)
#define temp1 2 //Temperatura dos motores do motor 1
#define temp2 3 //Temperatura dos motores do motor 2

int temp = 0;

//Sensores analogicos
float sv5 = 0;
float sv12 = 0;
float t1 = 0;
float t2 = 0;
```

```

String inputString;
boolean stringComplete = false;

int parar = 0;
int motor = 0;
int angulo = 0;
String ang;

//=====
// ----- SETUP -----
//=====
void setup() {

    //Configuração dos pinos
    pinMode(motor1D,OUTPUT);
    pinMode(motor1E,OUTPUT);
    pinMode(motor2D,OUTPUT);
    pinMode(motor2E,OUTPUT);

    //Inicialização da portas
    analogWrite(motor1D,0);
    analogWrite(motor1E,0);
    analogWrite(motor2D,0);
    analogWrite(motor2E,0);

    pinMode(13, OUTPUT);

    Serial.begin(9600);    // Debugging only
    digitalWrite(13, LOW);

    // MSG inicial da comunicação serial

    Serial.println("*****");
    Serial.println("*   E.N.E. - v 1.0 - Comunicação com os motores   *");
    Serial.println("*****");

```

```

Serial.println(" ");

}

//=====
//----- SETUP -----
//=====

//*****

//=====
//----- LOOP -----
// =====

void loop() {

// --- Comunicação Serial ---
  if(stringComplete){

    ang = inputString.substring(1);

    Serial.print("Comando recebido com sucesso! (");
    Serial.print(inputString.charAt(0));
    Serial.print(") - (");
    Serial.print(ang.toInt());
    Serial.println(")");

    int chtemp = inputString.charAt(0) - 48;

    comandos(chtemp,ang.toInt());

    stringComplete = false;
    inputString = "";
  }
// --- Comunicação Serial ---

```

```

    if(!parar){
        analogWrite(motor1D,0);
        analogWrite(motor1E,0);
        analogWrite(motor2D,0);
        analogWrite(motor2E,0);
    }else{
        parar--;
    }
}

//=====
//----- LOOP -----
//=====

//----- SERIAL -----
void serialEvent(){
    while(Serial.available()){
        char inChar = (char)Serial.read();
        inputString += inChar;

        if(inChar == '\n'){
            stringComplete = true;
        }
    }
}

void comandos(int motor, int vel){
    /*
    Formato do comando XYYY
    X- motor (1 ou 2)
    YYY- velocidade
    001-499 - frente
    501-999 - traz
    */
}

```

```
if(motor == 1){
  // 1000 - motor parado
  // 1001 - 1255 - motor para frente
  // 1501 - 1755 - motor para tras
  if(vel == 0){
    analogWrite(motor1D,0);
    analogWrite(motor1E,0);
  }else if(vel > 0 && vel < 256){
    analogWrite(motor1E, 0);
    analogWrite(motor1D, vel);
  }else if(vel > 500 && vel < 756){
    analogWrite(motor1D, 0);
    analogWrite(motor1E, (vel-500));
  }else
    Serial.println(" Comando invalido - velocidade nao especificado
ou especificada foram do parametros!");
  }else if(motor == 2){
    // 2000 - motor parado
    // 2001 - 2255 - motor para frente
    // 2501 - 2755 - motor para tras
    if(vel == 0){
      analogWrite(motor2D,0);
      analogWrite(motor2E,0);
    }else if(vel > 0 && vel < 256){
      analogWrite(motor2E, 0);
      analogWrite(motor2D, vel);
    }else if(vel > 500 && vel < 756){
      analogWrite(motor2D, 0);
      analogWrite(motor2E, (vel-500));
    }else
      Serial.println(" Comando invalido - velocidade nao especificado
ou especificada foram do parametros!");
    }else if(motor == 3){
      // acionar os dois motores para frente ou para traz
      // 3000 - motores parados
```

```
// 3001 - 3255 - motores para frente
// 3501 - 3755 - motores para tras
if(vel == 0){
    analogWrite(motor1D,0);
    analogWrite(motor1E,0);
    analogWrite(motor2D,0);
    analogWrite(motor2E,0);
}else if(vel > 0 && vel < 256){
    analogWrite(motor1E, 0);
    analogWrite(motor2E, 0);
    analogWrite(motor1D, vel);
    analogWrite(motor2D, vel);
}else if(vel > 500 && vel < 756){
    analogWrite(motor1D, 0);
    analogWrite(motor2D, 0);
    analogWrite(motor1E, (vel-500));
    analogWrite(motor2E, (vel-500));
}else
    Serial.println(" Comando invalido - velocidade nao especificado
ou especificada foram do parametros!");
}else if(motor == 4){
    // gira para direita ou esquerda
    // 4000 - motor parado
    // 4001 - 4255 - gira para direita por 10 segundos
    // 4501 - 4755 - gira para esquerda por 10 segundos
    if(vel == 0){
        analogWrite(motor1D,0);
        analogWrite(motor1E,0);
        analogWrite(motor2D,0);
        analogWrite(motor2E,0);
    }else if(vel > 0 && vel < 256){
        analogWrite(motor1E, 0);
        analogWrite(motor2D, 0);
        analogWrite(motor1D, vel);
        analogWrite(motor2E, vel);
    }
```

```
}else if(vel > 500 && vel < 756){
    analogWrite(motor1D, 0);
    analogWrite(motor2E, 0);
    analogWrite(motor1E, (vel-500));
    analogWrite(motor2D, (vel-500));
}else
    Serial.println(" Comando invalido - velocidade nao especificado
ou especificada foram do parametros!");

//gira durante 500ms
delay(1000);
analogWrite(motor1D,0);
analogWrite(motor1E,0);
analogWrite(motor2D,0);
analogWrite(motor2E,0);

}else if(motor == 5){
    // 5000 - motor parado
    // 5001 - 5255 - motor para frente
    // 5501 - 5755 - motor para tras
    if(vel == 0){
        analogWrite(motor1D,0);
        analogWrite(motor1E,0);
    }else if(vel > 0 && vel < 256){
        analogWrite(motor1E, 0);
        analogWrite(motor1D, vel);
        parar = 10;
    }else if(vel > 500 && vel < 756){
        analogWrite(motor1D, 0);
        analogWrite(motor1E, (vel-500));
        parar = 10;
    }else
        Serial.println(" Comando invalido - velocidade nao especificado
ou especificada foram do parametros!");
```



```

}else if(motor == 6){
    // 6000 - motor parado
    // 6001 - 6255 - motor para frente
    // 6501 - 6755 - motor para tras
    if(vel == 0){
        analogWrite(motor2D,0);
        analogWrite(motor2E,0);
    }else if(vel > 0 && vel < 256){
        analogWrite(motor2E, 0);
        analogWrite(motor2D, vel);
        parar = 10;
    }else if(vel > 500 && vel < 756){
        analogWrite(motor2D, 0);
        analogWrite(motor2E, (vel-500));
        parar = 10;
    }else
        Serial.println(" Comando invalido - velocidade nao especificado
ou especificada foram do parametros!");

}

}else if(motor == 7){
    // acionar os dois motores para frente ou para traz
    // 7000 - motores parados
    // 7001 - 7255 - motores para frente
    // 7501 - 7755 - motores para tras
    if(vel == 0){
        analogWrite(motor1D,0);
        analogWrite(motor1E,0);
        analogWrite(motor2D,0);
        analogWrite(motor2E,0);
    }else if(vel > 0 && vel < 256){
        analogWrite(motor1E, 0);
        analogWrite(motor2E, 0);
        analogWrite(motor1D, vel);
        analogWrite(motor2D, vel);
        parar = 10;
    }
}
}

```

```

}else if(vel > 500 && vel < 756){
    analogWrite(motor1D, 0);
    analogWrite(motor2D, 0);
    analogWrite(motor1E, (vel-500));
    analogWrite(motor2E, (vel-500));
    parar = 10;
}else
    Serial.println(" Comando invalido - velocidade nao especificado
ou especificada foram do parametros!");

}else if(motor == 9){
    //leitura das portas analogicas
    // 9100 - 9199 - retorna voltagem da bateria do motores
    // 9200 - 9299 - retorna voltagem da bateria do arduino\raspberry
    // 9300 - 9399 - retorna temperatura dos mosfet do motor 1
    // 9400 - 9499 - retorna temperatura dos mosfet do motor 2
    if(vel > 0 && vel < 200){
        Serial.print(" Voltagem da bateria dos motores: ");
        sv12 = 5.0 * analogRead(v12) / 1024.0;
        Serial.print(map(sv12, 0.0, 5.0, 0.0, 12.0)); //valor errado precisa
ser corrigido
        Serial.println(" v");
    }else if(vel > 199 && vel < 300){
        Serial.print(" Voltagem da bateria do Arduino\\Raspberry: ");
        sv5 = 5.0 * analogRead(v5) / 1024.0;
        Serial.print(sv5*2);
        Serial.println(" v");
    }else if(vel > 299 && vel < 400){
        Serial.print(" Temperatura dos mosfets do motor 1: ");
        t1 = 5.0 * analogRead(temp1) / 1024.0;
        Serial.print((1.8663 - t1) / 0.01169 );
        Serial.println(" C");
    }else if(vel > 399 && vel < 500){
        Serial.print(" Temperatura dos mosfets do motor 2: ");
        t2 = 5.0 * analogRead(temp2) / 1024.0;

```

```
Serial.print((1.8663 - t2) / 0.01169 );
Serial.println(" C");
}else
    Serial.println(" Comando invalido - velocidade nao especificado
ou especificada foram do parametros!");

}else
    Serial.println(" Comando invalido - motor nao especificado!");
}
```

## Código do Raspberry PI 2:

### Python 2.7

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

"""
Interface básica para controle dos motores
Desenvolvedor: Ewerton L. de Sousa
"""

import serial,sys, glob

from ttk import Frame, Label, Combobox
from Tkinter import *

class Frame1(Frame):

    def __init__(self, parent):
        Frame.__init__(self, parent, background="white")
```

```

self.parent = parent

self.initUI()

def initUI(self):

    self.lbox = Listbox(self, height = 10, width = 55)
    sbar = Scrollbar(self, command=self.lbox.yview)
    sbar.place(x=360, y=240)
    self.lbox.config(yscrollcommand=sbar.set)
    self.lbox.place(x=10, y=240)

    self.lbox.insert('end', "Interface Básica de Controle dos Motores
- v1.0")
    self.lbox.insert('end', "S.O. (%s)" % (sys.platform))

    self.parent.title("Interface básica para controle dos motores -
v1.0")
    self.pack(fill=BOTH, expand=1)

    self.opts = ""
    self.cbox = Combobox(self, textvariable=self.opts,
state='readonly')
    for n,s in scan():
        self.opts += "%s " % (s)

    self.cbox['values'] = self.opts
    if(self.opts != ""):
        self.cbox.current(0)
    self.cbox.place(x=10, y=10)
    "self.cbox.bind('<<ComboboxSelected>>', self.conectar)"

    btConectar = Button(self, text="Conectar", width=10)
    btConectar.bind("<Button-1>", self.conectar)

```

```
btConectar.place(x=200, y=10)
```

```
btFrente = Button(self, text="/\\", width=5)
btFrente.bind("<Button-1>", self.comandoF)
btFrente.place(x=160, y=100)
btTraz = Button(self, text="\\", width=5)
btTraz.bind("<Button-1>", self.comandoT)
btTraz.place(x=160, y=130)
```

```
btEsqFrente = Button(self, text="/\\", width=5)
btEsqFrente.place(x=50, y=70)
btEsqFrente.bind("<Button-1>", self.comandoEF)
btEsqTraz = Button(self, text="\\", width=5)
btEsqTraz.place(x=50, y=150)
btEsqTraz.bind("<Button-1>", self.comandoET)
```

```
btDirFrente = Button(self, text="/\\", width=5)
btDirFrente.place(x=260, y=70)
btDirFrente.bind("<Button-1>", self.comandoDF)
btDirTraz = Button(self, text="\\", width=5)
btDirTraz.place(x=260, y=150)
btDirTraz.bind("<Button-1>", self.comandoDT)
```

```
btGiraEsq = Button(self, text=">>", width=5)
btGiraEsq.place(x=90, y=200)
btGiraEsq.bind("<Button-1>", self.comandoGE)
btParar = Button(self, text="-x-", width=5)
btParar.place(x=160, y=200)
btParar.bind("<Button-1>", self.comandoP)
btGiraDir = Button(self, text="<<", width=5)
btGiraDir.place(x=230, y=200)
btGiraDir.bind("<Button-1>", self.comandoGD)
```

```
def conectar(self, event):
```

```
self.lbox.insert('end', "conectando...")
self.lbox.insert('end', "Porta:", self.cbox.get())
self.lbox.insert('end', "Baund: 9600")

self.arduino = None

try:
    self.arduino = serial.Serial(self.cbox.get(), 9600);
    self.lbox.insert('end', "Conectado! \n")

    try:
        self.lbox.insert('end', self.arduino.readline() )
        self.lbox.insert('end', self.arduino.readline())
        self.lbox.insert('end', self.arduino.readline())
        self.lbox.insert('end', self.arduino.readline())

    except serial.serialutil.SerialException:
        pass
except:
    pass
finally:
    if self.arduino:
        "self.arduino.close()"
        pass

def comandoP(self, event):
    self.lbox.insert('0', "Comando 3000")
    try:
        self.arduino.write("3000\n")
        self.lbox.insert('0', self.arduino.readline() )
    except:
        pass

def comandoF(self, event):
    self.lbox.insert('0', "Comando 3300")
```

```
try:
    self.arduino.write("3300\n")
    self.lbox.insert('0', self.arduino.readline() )
except:
    pass

def comandoT(self, event):
    self.lbox.insert('0', "Comando 3700")
    try:
        self.arduino.write("3700\n")
        self.lbox.insert('0', self.arduino.readline() )
    except:
        pass

def comandoEF(self, event):
    self.lbox.insert('0', "Comando 2300")
    try:
        self.arduino.write("2300\n")
        self.lbox.insert('0', self.arduino.readline())
    except:
        pass

def comandoET(self, event):
    self.lbox.insert('0', "Comando 2700")
    try:
        self.arduino.write("2700\n")
        self.lbox.insert('0', self.arduino.readline())
    except:
        pass

def comandoDF(self, event):
    self.lbox.insert('0', "Comando 1300")
    try:
        self.arduino.write("1300\n")
        self.lbox.insert('0', self.arduino.readline())
```



```
except:  
    pass
```

```
def comandoDT(self, event):  
    self.lbox.insert('0', "Comando 1700")  
    try:  
        self.arduino.write("1700\n")  
        self.lbox.insert('0', self.arduino.readline())  
    except:  
        pass
```

```
def comandoGE(self, event):  
    self.lbox.insert('0', "Comando 4300")  
    try:  
        self.arduino.write("4300\n")  
        self.lbox.insert('0', self.arduino.readline() )  
    except:  
        pass
```

```
def comandoGD(self, event):  
    self.lbox.insert('0', "Comando 4700")  
    try:  
        self.arduino.write("4700\n")  
        self.lbox.insert('0', self.arduino.readline() )  
    except:  
        pass
```

```
def scan():  
    """scan for available ports. return a list of tuples (num, name)"""  
    available = []
```

```
if sys.platform.startswith('linux'):  
    available.append((0, glob.glob('/dev/ttyS*')[0] ))
```

```
available.append((1, glob.glob('/dev/ttyUSB*')[0]))
""available.append((2, glob.glob('/dev/ttyACM*')[0]))""
available.append((3, glob.glob('/dev/serial/by-id/*')[0]))
return available
```

```
elif sys.platform.startswith('win'):
```

```
for i in range(256):
```

```
    try:
```

```
        s = serial.Serial(i)
```

```
        available.append((i, s.portstr))
```

```
        s.close() # explicit close 'cause of delayed GC in java
```

```
    except serial.SerialException:
```

```
        pass
```

```
    return available
```

```
else: self.lbox.insert('end', "S.O. não suportado!")
```

```
return None
```

```
def main():
```

```
def teclado(event):
```

```
    app.lbox.insert('end', "Tecla: %d" % event.keycode)
```

```
    if event.keycode == 39:
```

```
        app.lbox.insert('0', "Direita")
```

```
        app.arduino.write("4700\n")
```

```
        app.lbox.insert('0', app.arduino.readline())
```

```
    elif event.keycode == 40:
```

```
        app.lbox.insert('0', "Tras")
```

```
        app.arduino.write("3700\n")
```

```
        app.lbox.insert('0', app.arduino.readline() )
```

```
    elif event.keycode == 38:
```

```
        app.lbox.insert('0', "Frente")
```

```
        app.arduino.write("3300\n")
        app.lbox.insert('0', app.arduino.readline() )
    elif event.keycode == 37:
        app.lbox.insert('0', "Esquerda")
        app.arduino.write("4300\n")
        app.lbox.insert('0', app.arduino.readline())
    elif event.keycode == 32:
        app.lbox.insert('0', "Parado")
        app.arduino.write("3000\n")
        app.lbox.insert('0', app.arduino.readline())

root = Tk()
root.geometry("400x420")
app = Frame1(root)
app.bind_all("<KeyPress>", teclado)
root.mainloop()

if __name__ == '__main__':
    main()
```