MR2006B.1 Automatización industrial.

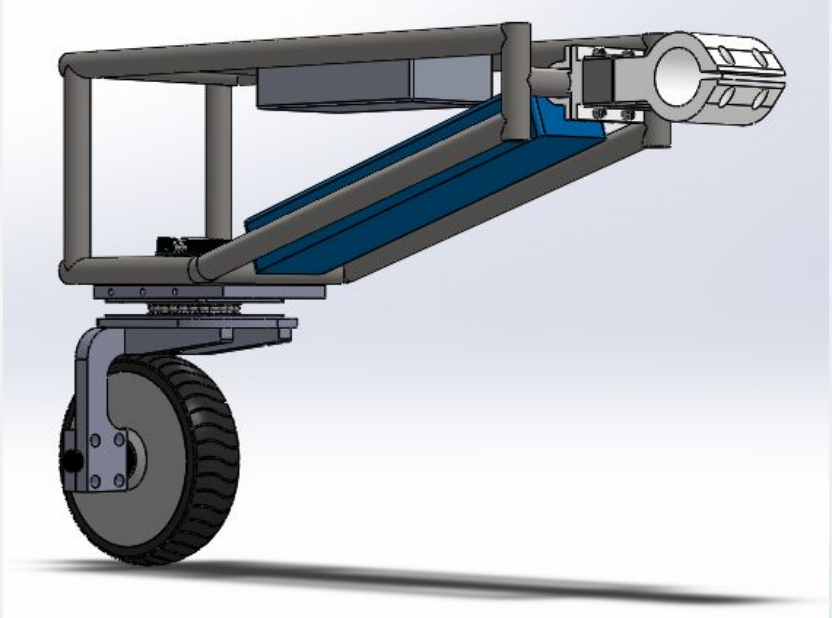
Sistema de quinta rueda motorizada.

Instituto Tecnológico de Estudios Superiores de Monterrey.

Griselda Soriano Rosas A01638852
Oscar Alfredo Mercado Rico A01638228
Seth Gaed Plancarte Silva A01638066
Julio David Morales Valtierra A01067648
Francisco Javier López Sánchez A01637518

Propuesta de solución.

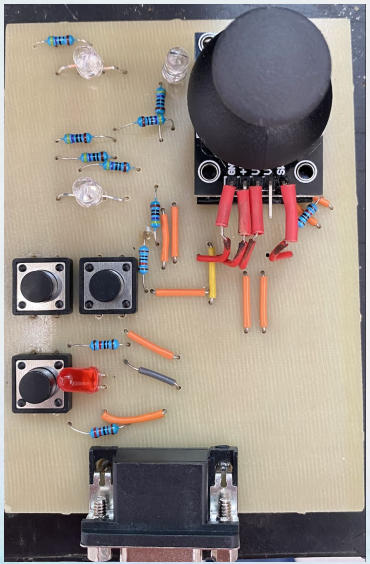
- Un sistema de 5ta rueda motorizado y de fácil ajuste el cual complementa el uso de una silla de rueda tradicional.
- Donde el manejo sea por medio de dos opciones (manual y asistido).
- El usuario pueda visualizar el nivel de batería.
- Se busca que el sistema electrónico y batería sea encapsulado dentro del chasis, por medio de placas de acero alrededor.



Electrónica y programación

Funcionalidades:

Control de sistema por medio de joystick (manual)



Control de velocidad conforme a una referencia

Electrónica y programación

Funcionalidades:

- 2 modos de operación:

  - Control por joystick

  - Control asistido (referencia dada por el usuario)

- Sistema de alerta ante descarga de batería, aumento de temperatura del sistema o una inclinación inadecuada.

Monitoreo de variables:

- Temperatura
- Carga de batería
- Inclinación

Electrónica y programación

Botones GPIO

```
void PORTA_IRQHandler()
{
    if (((PORTA_PCR1 & (1<<24)) != 0))//Interrumpe, boton modo 1
    {
        PORTA_PCR1 |= (1<<24);//Se apaga la bandera
        if(btm1 == 0)
        {
            GPIOC_PCOR = (1<<0);//Se apaga el led verde
            GPIOC_PSOR = (1<<3) + (1<<4);//Se enciende el led rojo y azul
            btm1=1;//Se activa el modo de operación1
            btm2=0;//Se desactiva el modo de operación2
            ninter = 0;
            PORTA_PCR12 |= (1<<19) + (1<<17);//Se habilita la interrupción en flancos de baj
            voltajeDAC = 1160;
            frecuenciarj = 0;//Se limpia la variable
            frecuenciarb = 0;//Se limpia la variable
            frecuenciar = 0;//Se limpia la variable
            frecuencian = 0;//Se limpia la variable
        }
        else
        {
            GPIOC_PCOR = (1<<4) + (1<<3) + (1<<0);//Se apaga led rgb
            btm1 = 0;//Se desactiva el modo de operación 1
            PORTA_PCR12 &=~(1<<19) + (1<<17);//Se desactiva la interrupción de botón de velo
            //Se desactiva entrada del joystick en ADC
            voltajeDAC = 1200;
            frecuenciarj = 0;//Se limpia la variable
            frecuenciarb = 0;//Se limpia la variable
            frecuenciar = 0;//Se limpia la variable
            frecuencian = 0;//Se limpia la variable
        }
```

```
else if (((PORTA_PCR2 & (1<<24)) != 0))//interrumpe boton modo 2
{
    PORTA_PCR2 |= (1<<24);//Se apaga la bandera
    if(btm2 == 0)
    {
        GPIOC_PCOR = (1<<3);//Se apaga el led rojo
        GPIOC_PSOR = (1<<4) + (1<<0);//Se enciende el led azul y verde
        btm2=1;//Se activa el modo de operación 2
        btm1=0;//Se desactiva el modo de operación 1//desactiva la entrada de joystick en ADC
        PORTA_PCR12 &=~(1<<19) + (1<<17);//Se desactiva la interrupción de botón de velocidad constante
        frecuenciarj = 0;//Se limpia la variable
        frecuenciarb = 0;//Se limpia la variable
        frecuenciar = 0;//Se limpia la variable
        frecuencian = 0;//Se limpia la variable
        voltajeDAC = 1160;
    }
    else
    {
        btm2 = 0;//Se desactiva el modo de operación 2
        ninter = 0;
        GPIOC_PCOR = (1<<4) + (1<<3) + (1<<0);//Se apaga led rgb
        voltajeDAC = 1200;
        frecuenciarj = 0;//Se limpia la variable
        frecuenciarb = 0;//Se limpia la variable
        frecuenciar = 0;//Se limpia la variable
        frecuencian = 0;//Se limpia la variable
    }
}
```

```
    else if (((PORTA_PCR12 & (1<<24)) != 0))//interrumpe botón de velocidad constante
    {
        PORTA_PCR12 |= (1<<24);//Se apaga la bandera
        frecuenciarb = frecuencia;//Se mantiene la frecuencia de joystick
    }
    else if (((PORTA_PCR5 & (1<<24)) != 0))//interrumpe botón de paro
    {
        PORTA_PCR5 |= (1<<24);//Se apaga la bandera
        if (paropresionado == 0)paropresionado = 1;
        else paropresionado = 0;
    }
}
```

Electrónica y programación

Monitoreo de variables

- Inclinación

```
void LPTimer_IRQHandler()//ISR de LPT
{
    LPTMR0_CSR |= (1<<7);//se apaga la bandera de LPT , se deja activada la interrupción de LPT y se deja activado el contador
    //Se reinicia el contador
    ADC0_SC1A =  entradas[entradah] ;//Se habilita la interrupción local de ADC y la entrada analogica i
    //Se inicia la conversión


    //Se lee la salida del acelerometro cada 100ms

    unsigned char i=0;
    unsigned short msb=0;
    writeI2C = I2C0read(SLAVE_ADDR, 0x01, 6, datos, &count);//Se leen los datos de los registos de la aceleración en los ejes
     //Se almacenan en la lista de datos tanto la parte alta como la baja

     //se juntan los 14 bits de los registros para cada salida
    salidasejes[0]=((datos[0]<<8)|(datos[1]))>>2;//salida x
    salidasejes[1]=((datos[2]<<8)|(datos[3]))>>2;//salida y
    salidasejes[2]=((datos[4]<<8)|(datos[5]))>>2;//salida z
    i=0;
    while (i<3)
    {
        msb=salidasejes[i]>>13;
        if ((msb==1))
        {
            magnitudesejes[i]=-4096+(salidasejes[i] &= ~1<<13);
        }
        else
        {
            magnitudesejes[i]=salidasejes[i];
        }
        i++;
    }
    inclinacion=90-(atan2(magnitudesejes[2],magnitudesejes[0])*18000/314);
```

- Carga de batería

```
if (entradah == 0)//Entrada analogica 8 habilitada (PTB0) // voltaje bateria
{
    voltajebateria = (ADC0_RA * 42000)/255;//voltaje de bateria en mV
    if(voltajebateria >= 40000)//Voltaje de bateria alto
    {
        GPIOE_PCOR = (1<<4) + (1<<3);//Apagan led de carga media y baja
        //PIT_TCTRL0 &=~(1<<1);//Se desactiva timer y la interrupción local
        //GPIOE_PCOR = (1<<2);//Se desactiva el buzzer
        GPIOE_PSOR = (1<<5);//enciende el led verde de carga alta
        parobateria = 0;
    }
    if((voltajebateria >= 38000 ) && (voltajebateria < 40000))//Voltaje de bateria medio
    {
        GPIOE_PCOR = (1<<5) + (1<<3);//Apagan led de carga alta y baja
        //PIT_TCTRL0 &=~(1<<1);//Se desactiva timer y la interrupción local
        //GPIOE_PCOR = (1<<2);//Se desactiva el buzzer
        GPIOE_PSOR = (1<<4);//enciende el led amarillo de carga media
        parobateria = 0;
    }
```

```
else if (voltajebateria < 38000)//voltaje de bateria bajo
{
    GPIOE_PCOR = (1<<5) + (1<<4);//Apagan led de carga alta y media
    GPIOE_PSOR = (1<<3);//enciende el led rojo de carga baja
    if (voltajebateria <= 37000) parobateria = 1;//Se enciende el paro del sistema por carga critica
    //Configura buzzer
    //PIT_LDVAL0 = 4000000/20;//tarda 50ms en llegar a valor
    //el buzzer tendrá una frecuencia de 20HZ
    //PIT_TCTRL0 = (1<<1) + (1<<0);//Se activa timer y la interrupción local
    else parobateria = 0;
}
entradah = 1;//próxima entrada a habilitar
```

- Temperatura

```
temp_grados = ((((ADC0_RA * 3300) / 255) / 2)) * 0.1;//temperatura en grados
if (temp_grados >= 50)//temperatura de riesgo para operación
{
    GPIOB_PSOR = (1<<11);//Se prende led de temperatura elevada
    parotemperatura = 1;
}
else
{
    GPIOB_PCOR = (1<<11);//Se apaga led de temperatura elevada//temperatura segura para operación
    parotemperatura = 0;
}
```

Electrónica y programación

- Inductivos

```
salidainductivo = (ADC0_RA * 3300)/255;
if (salidainductivo >= 3000) paroinductivo = 1; //sensores inductivos no detectan placa
//angulo de posición de rueda riesgoso
else paroinductivo = 0; //sensores inductivos detectan placa
//angulo de posición de rueda seguro
if(btml == 1)entradah = 3;//próxima entrada a habilitar//si esta presionado el botón de modo
else entradah = 0;
```

- Voltaje Joystick

```
voltjoy = (ADC0_RA * 3300)/255;//Voltaje de joystick en mV

if (frecuenciarb == 0)//boton de velocidad constante desactivado
//posición de reposo joystick es 0hz
{
    if ((voltjoy >= 1640) && (voltjoy <= 1900)) frecuenciarj = 0;//el motor esta detenido
    else if (voltjoy > 1900)//el motor avanza
    //Se habilitan 90 grados del joystick en el eje positivo
    {
        frecuenciarj = ((((voltjoy - 1900) * (155-0)) / (3300 - 1900)) + 0);//  Frecuencia
    }
    else frecuenciarj = 0;
}
```

```
else //boton de velocidad constante activado
//posición de reposo joysitick es la es la frecuencia de referencia de la ultima posición del joystick
{
    if ((voltjoy >= 1640) && (voltjoy <= 1900)) frecuenciarj = frecuenciarb;//el motor avanza a la frec
    //la velocidad se mantiene constante
    else if (voltjoy > 1900)//el motor incrementa su velocidad
    //Se habilitan 90 grados del joystick en el eje positivo
    {
        frecuenciarj = ((((voltjoy - 1900) * (155-frecuenciarb)) / (3300 - 1900)) + frecuenciarb);//
    }
    else // el motor disminuye la velocidad
    {
        frecuenciarj = ((((voltjoy - 0) * (frecuenciarb-0)) / (1640 - 0)) + 0);//   Frecuencia de refer
    }

}
```

```
if (frecuenciarj >= 155) frecuenciarj = 155;
else if(frecuenciarj <= 0)frecuenciarj = 0;

if((frecuenciarj == 0))frecuenciar = 0;
else if (((frecuenciarj !=0) && (paropresionado != 1))) frecuenciar = frecuenciarj;
entradah = 0;//Próxima entrada a habilitar
```

```
void FTM1_IRQHandler()
{


    if ((TPM1_SC & (1<<7))!= 0)//Se se reinicia el contador TOF activada
    {
        TPM1_SC|= (1<<7);//Se apaga la bandera TOF
        nvueltasc ++;//número de vueltas actual + 1

        if (nvueltasc >= 10)//La silla esta detenida
        {
            //Se supera el periodo mín de 100ms del periodo más lento del sensor
            periodo4 = 0;
            periodo = 0;
            frecuencia = 0;//Se calcula la frecuencia del periodo hz
            nvueltass = frecuencia/15;//Se calcula en numero de vueltas por segundo
            velocidadm = nvueltass * .5186;//Se calcula velocidad
            rpm = nvueltass * 60;//Se calculan las rpm
        }
    }
```

```
else if ((TPM1_C1SC & (1<<7)) != 0)//Si se interrumpe el canal 1 CHF1 activada
{
    TPM1_C1SC |=  (1<<7);//Se apaga la bandera CHF
    altoac = TPM1_C1V;
    if(altoac > altoant) periodo4 = altoac - altoant + (nvueltasc * 65536);//Se calcula el periodo/4 en caso de que el alt
    else if (altoac < altoant) periodo4 = altoac - altoant + ((nvueltasc-1) * 65536);//Se calcula el periodo/4 en caso de
    periodo = 4 * periodo4;//Se calcula el periodo real
    frecuencia = 1000000/periodo;//Se calcula la frecuencia del periodo hz
    nvueltass = frecuencia/15;//Se calcula en numero de vueltas por segundo
    velocidadm = nvueltass * .5186;//Se calcula velocidad en ms
    rpm = nvueltass * 60;//Se calculan las rpm
    altoant = TPM1_C1V;//Se almacena valor en el que se dió el alto anterior
    nvueltasc = 0;
    if ( (frecuencia >= frecuenciar - 2) && (frecuencia <= frecuenciar + 2) ) estabilidad =1;//se llega a la referencia
    else estabilidad = 0;//no se llega a la referencia
    if (btm2 == 1)
    {
        ninter ++;//Se cuentan las interrupciones
        if(ninter >300) ninter = 0;
        if ( (estabilidad == 0) && ( (ninter >= 10) && (ninter <= 60) ) )//impulso
        {
            if( (periodo > (periodoant - 100)) )//caso de querer frenar
            {
                frecuenciar = frecuencian;//Frecuencia calculada es la de referencia

                if(frecuenciar < 15)
                {
                    frecuenciar = 0;
                    frecuencian = 0;
                }
            }
```

- velocidad de motor

```
                    frecuenciar = frecuencian;//Frecuencia calculada es la de referencia
                    if(frecuenciar >= 80)
                    {
                        frecuenciar = 80;
                        frecuencian = 80;
                    }
                }
                ninter = 0;
            }
        }
    }
    if(frecuencia < (frecuenciar - 8))
        {
            voltajeDAC = voltajeDAC + 10;// aumenta voltaje en un paso 10mV
            if (voltajeDAC >= 3130) voltajeDAC = 3130;
        }
    if((frecuencia > (frecuenciar + 8)) && (frecuenciar != 0))
        {
            voltajeDAC = voltajeDAC - 10;//se disminuye voltaje en un paso 10mV
            if (voltajeDAC <= 1160) voltajeDAC = 1160;
        }
    if ( (frecuenciar == 0) && ((btm1 == 1) || (btm2 == 1)) )
    {
        voltajeDAC = voltajeDAC - 10;//se aumenta voltaje en un paso 10mV
        if (voltajeDAC <= 1160) voltajeDAC = 1160;
    }
```

```
    unsigned short dvalue;//instruccion 12 bits para DAC
    dvalue = (voltajeDAC * 4095) / 3300;//Instrucción de 12 bits a mandar a DAC
    DAC0_DAT0H = (dvalue>>8);//Parte alta delimitada
    DAC0_DAT0L = (dvalue & 0xff);//Parte baja delimitada
    //Se hace conversión de señal a digital
    //Se tiene voltaje deseado en PTE30
    if(btm2 == 1)
    {
        periodoant = periodo;//periodo actual pasa a ser el anterior
        frecuencian = frecuencia;//frecuencia anterior
    }
}
```

- velocidad de motor

Electrónica y programación

- Sistema de seguridad

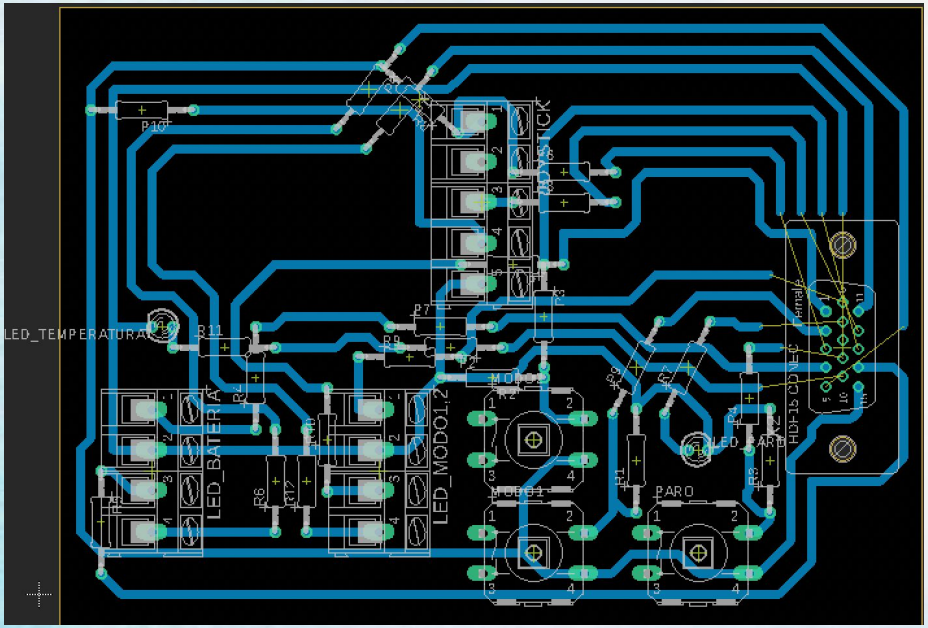
```
//Sistema de seguridad
if((paropresionado == 1) ||(paroinductivo ==1) || (parotemperatura == 1) || (parobateria == 1))
{
    //en caso de tener el boton presionado
    //en caso de estar en una posición de riesgo
    //en caso de tener una temperatura elevada
    //en caso de tener una carga de batería crítica
    //se pone en paro el sistema

    GPIOC_PSOR = (1<<7);//Se enciende led del botón paro
    frecuenciarj = 0;
    frecuenciarb = 0;
    frecuenciar = 0;
    frecuencian = 0;
    voltajeDAC = 1190;
```

- Sistema de emergencia (buzzer)

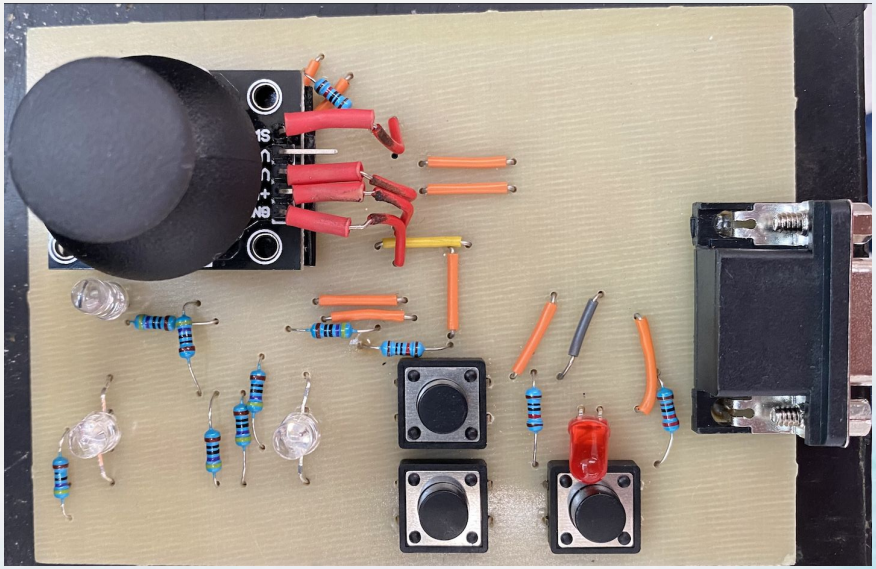
```
//Sistema de emergencia buzzer

if (((temp_grados >= 50) || (voltajebateria <= 37000) || (inclinacion>30)) && (LPTMINT == 10))//Se tiene temperatura alta o carga critica o silla est
//Se activa el buzzer cada s //inclinación menor que 150
//Misma frecuencia para ambos casos
{
    GPIOE_PTOR = (1<<2);//Se prende/apaga buzzer
    LPTMINT = 0;//Se reinicia contador de interrupciones
}
else if((temp_grados < 50) && (voltajebateria > 1300) && (inclinacion<30)) GPIOE_PCOR = (1<<2);// operación segura //Se desactiva buzzer en caso de e
else if (LPTMINT > 10)LPTMINT = 0;//Se reinicia contador de interrupciones en caso de superar 10
LPTMINT ++;//Se cuentan las interrupciones
```

PCB (Centro de control)

Fotos esquemáticos

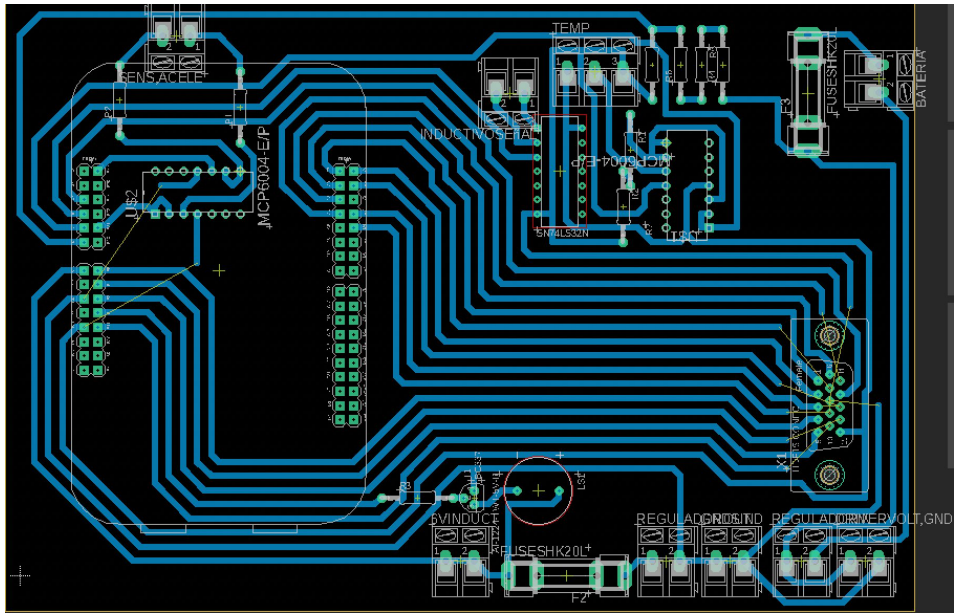


Fotos del sistema completo

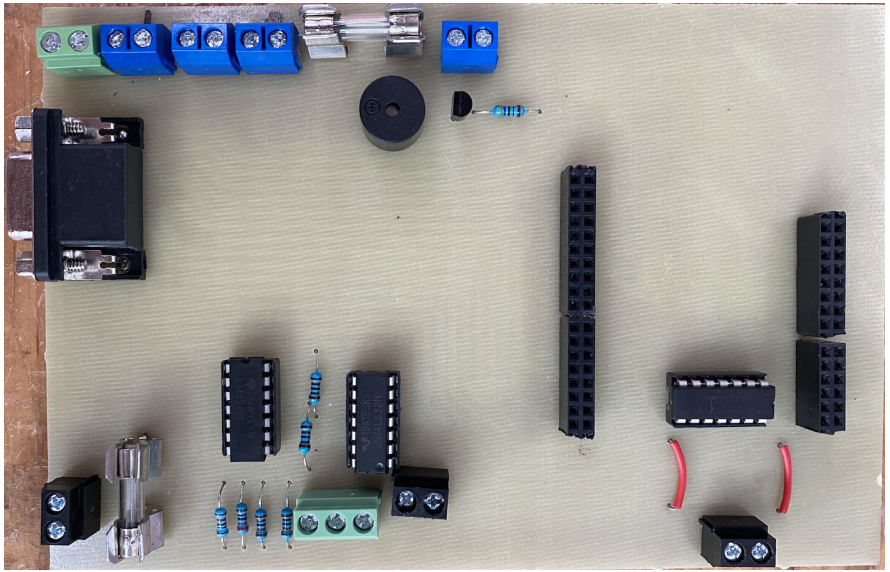


PCB (SENSORES Y MICROCONTROLADOR)

Fotos esquemáticos

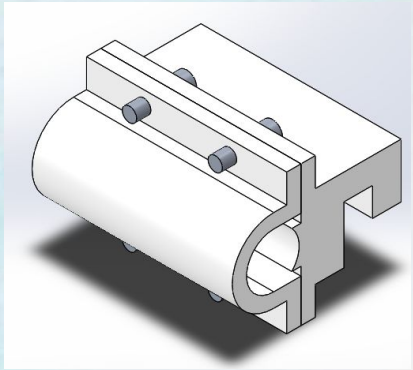


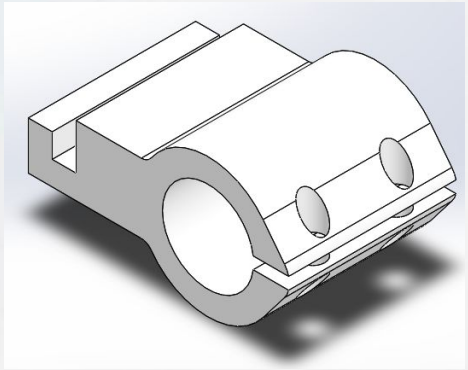
Fotos del sistema completo

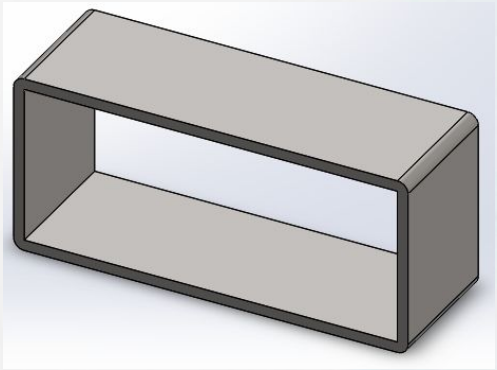


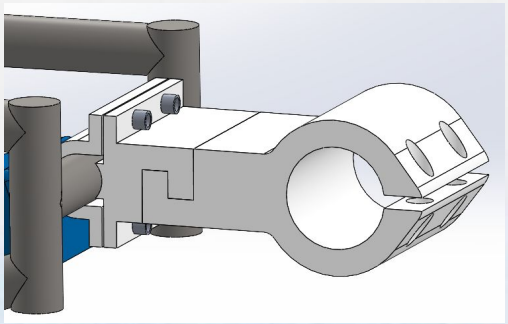
Sistema mecánico

Modelo CAD (Sistema de montaje)



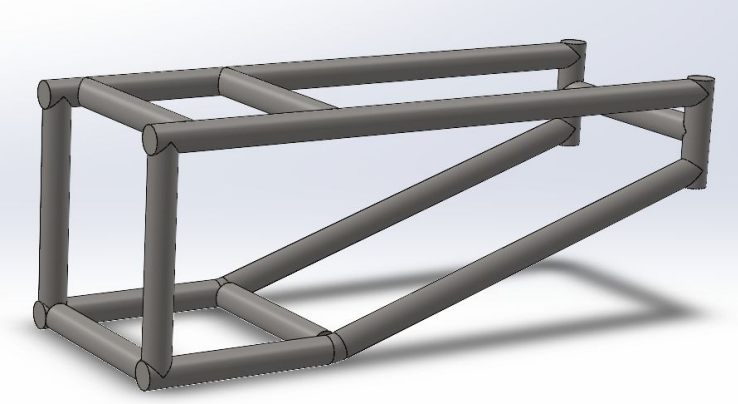






Sistema mecánico

Modelo CAD (Ensamble final)



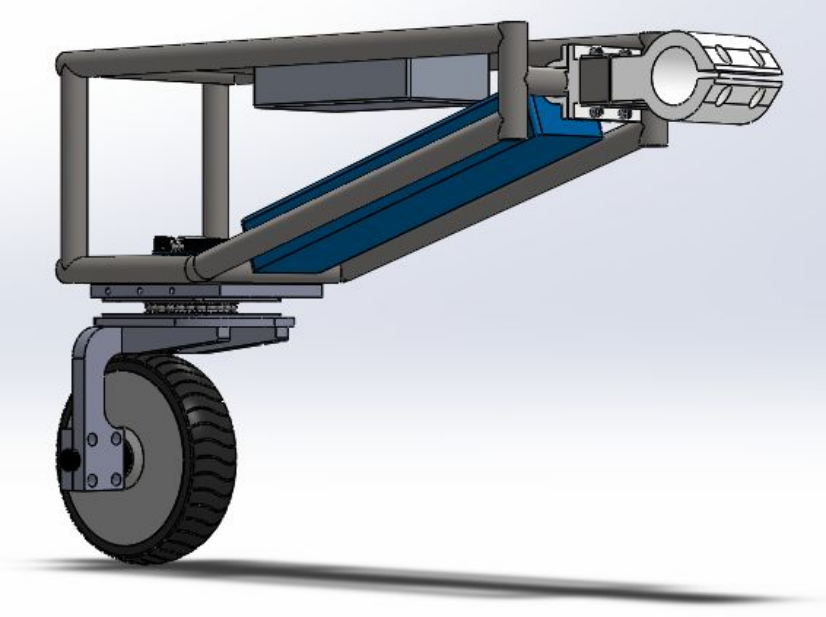


Tabla de costos

Descripción	Costo
Modulo de Voz	$790.00
Sensores Ind.	$263.00
componentes	$250.00
Pintura	$199.00
Tornillería	$84.00
Piezas Aluminio	$400.00
Primer	$189.00
Pasta automotriz	$310.00
Empaque	$118.00
Cable VGA	$295.00
Lijas	$84.00
Piezas ABS	$600.00
Electronica.Zapopan	$50.00
Componentes Electronicos	$600.00
Precio total	***$4,232.00***

Enlace video presentación

https://drive.google.com/file/d/19sHOX5XgHXO24kA1OQd3Z3gIe_t6Hhj7/view?usp=sharing