***UNIVERSIDAD TECNOLÓGICA NACIONAL***



**“ ROBOTICA ”**

**Trabajo Práctico Nº 1**

***Modelado cinematico del robot.***

***Generacion de trayectorias mediante dsp .***

*Profesor*: **Ing. Hernan Giannetta**

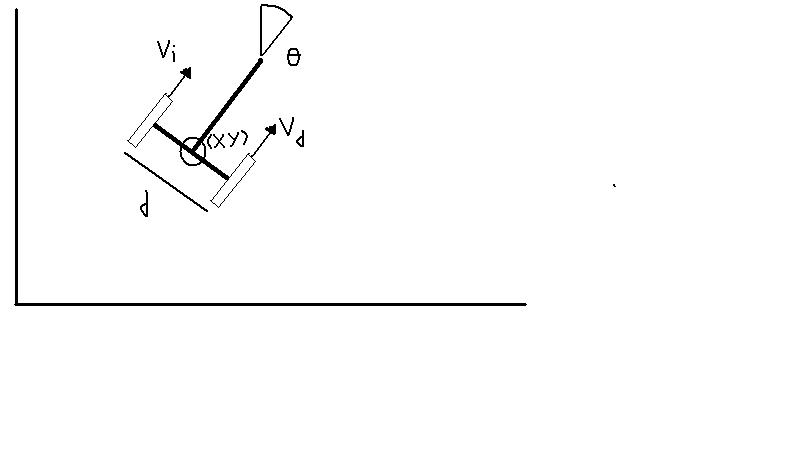
*Jefe de Trabajos Prácticos:* **Ing. Damian Granzella**

*Alumno*: **Giachino José**

*Legajo:* **147465-0**

**Ciclo lectivo 2012**

Guiado diferencial \_Cinematica Directa

  
De la observación de la configuración se obtiene :

= ∙ + ∙

= ∙

Estimacion de la posición y la orientación

= +

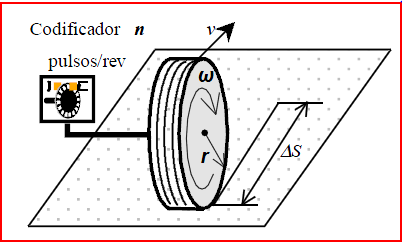
Odometria :

Es una técnica para conocer la posición de un móvil a través del desplazamiento angular de sus ruedas , suministrados por encoders.

Es el sistema mas básico para controlar el robot , y posee la desventaja de acumular error a medida que crece la distancia recorrida.

Suele combinarse con otros tipos de sensado de posición , para ir corrigiendo el error.

Modelo de la rueda :



n : cantidad de pulsos que entrega el encoder en una vuelta.

Vt = w ∙ r

∆S= Vr ∙ ∆t

∆S= w∙r ∙ ∆t w∙∆t = ∆φ ; Angulo que llogra la rueda en un intervalo ∆t

∆S= r ∙ ∆φ

El codificador/encoder genera ∆N pulsos para un desplazamiento lineal de longitud ∆S.

∆φ = 2π ∙

∆S= ∙ ∆N

Se define como constante de sensibilidad odometrica al cociente :

C =

Siendo :

∆S= C ∙ ∆N

Considerando las dos ruedas se tiene :

∆Sd = Cd ∙ ∆Nd

∆Si = Ci ∙ ∆Ni

El desplazamiento infinitesimal del centro de giro del robot es :

∆Sk  =

Y el desplazamiento angular del mismo será :

∆θk  = ;

siendo “d” lo que se llama “ via del vehiculo “ , la distancia entre las ruedas.

Las integrales del modelo cinematico se reducen a ecuaciones en diferencia si el intervalo de muestreo de la posición se hace lo suficientemente pequeño y constante.

=

DSP

Codigo :

/\*\* ###################################################################

\*\* Filename : TPN1.C

\*\* Project : TPN1

\*\* Processor : 56F8367

\*\* Version : Driver 01.12

\*\* Compiler : Metrowerks DSP C Compiler

\*\* Date/Time : 10/05/2009, 19.19

\*\* Abstract :

\*\* Main module.

\*\* Here is to be placed user's code.

\*\* Settings :

\*\* Contents :

\*\* No public methods

\*\*

\*\* (c) Copyright UNIS, spol. s r.o. 1997-2006

\*\* UNIS, spol. s r.o.

\*\* Jundrovska 33

\*\* 624 00 Brno

\*\* Czech Republic

\*\* http : www.processorexpert.com

\*\* mail : info@processorexpert.com

\*\* ###################################################################\*/

/\* MODULE TPN1 \*/

/\* Including used modules for compiling procedure \*/

#include "Cpu.h"

#include "Events.h"

#include "MFR1.h"

#include "MEM1.h"

#include "TFR1.h"

/\* Include shared modules, which are used for whole project \*/

#include "PE\_Types.h"

#include "PE\_Error.h"

#include "PE\_Const.h"

#include "IO\_Map.h"

#include "stdio.h"

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// DEFINE

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#define MXRAD 360 //100

#define PULSE2RAD 32767/MXRAD // 32767/100 impulsos // #define PULSE2RAD 450

#define MITAD 16384

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// GLOBAL VARIABLE

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Frac16 x[1000];

Frac16 y[1000];

Frac16 theta[1000];

Frac16 Delta\_Sn;

Frac16 Delta\_theta;

Frac16 Media\_Delta\_Theta;

Frac16 Pre\_Delta\_Sn;

Frac16 Pre\_Delta\_Theta;

Frac16 Delta\_Sd;

Frac16 Delta\_Si;

Frac16 Prueba;

Frac16 K;

Frac16 Delta\_THETA\_FRAC ;

Frac16 argu\_x;

Frac16 cos\_x;

Frac16 sum\_x;

Frac16 argu\_y;

Frac16 sen\_y;

Frac16 sum\_y;

int j, k ,nMax ,d;

void main(void)

{

/\* Write your local variable definition here \*/

//pruebo imprimir a archivo

FILE \*fichero;

char nombre[10] = "datos.txt";

//unsigned int p;

/\* x[0] =0;// Posicion x,y inicial. NORMALIZAR!!!!

y[0] =0;

theta[0] =0;// ORIENTACION INICIAL...NORMALIZAR EN ANGULO!!!\*/

/\*\*\* Processor Expert internal initialization. DON'T REMOVE THIS CODE!!! \*\*\*/

PE\_low\_level\_init();

/\*\*\* End of Processor Expert internal initialization. \*\*\*/

/\* Write your code here \*/

// prueba imp. a archivo

fichero = fopen( nombre, "w" );

printf( "Fichero: %s (para escritura) -> \t\n\n", nombre );

if( fichero )

printf( "creado (ABIERTO) \t\n\n" );

else

{

printf( "Error (NO ABIERTO)\n" );

}

// Generador trayectorias para dinamica :Linea inicial

Delta\_Sd=50;

Delta\_Si=50;

d= 1966 ;

printf ("Velocidad de la rueda izq..=%d \t\n\n ",Delta\_Si );

printf ("Velocidad de la rueda der..=%d \t\n\n ",Delta\_Sd );

Pre\_Delta\_Sn =add(Delta\_Sd,Delta\_Si);

Delta\_Sn =mult\_r(Pre\_Delta\_Sn,MITAD );

Pre\_Delta\_Theta = sub(Delta\_Sd,Delta\_Si);

Delta\_theta = div\_s(Pre\_Delta\_Theta, d) ;

K = div\_s(182,572) ;

Delta\_THETA\_FRAC =mult\_r(Delta\_theta,K );

Media\_Delta\_Theta = mult\_r(Delta\_THETA\_FRAC, MITAD);

nMax=40 ; // solo 40 puntos para probar

for(k = 1; k < nMax; k++)// cada k es un incremento infinitesimal...

{

argu\_x = add(theta[k-1], Media\_Delta\_Theta);

cos\_x = tfr16CosPIx( argu\_x );

sum\_x = mult\_r(Delta\_Sn, cos\_x );

x[k] = add (x[k - 1], sum\_x );

argu\_y = add(theta[k-1], Media\_Delta\_Theta);

sen\_y = tfr16SinPIx( argu\_y );

sum\_y = mult\_r(Delta\_Sn, sen\_y );

y[k] = add (y[k - 1], sum\_y );

theta[k] = add (theta[k - 1],Delta\_theta);

}

for(j=0; j<nMax; j++)

{

fprintf (fichero,"%d \t %d \t %d \t \n", x[j],y[j],theta[j]);

}

// Fin del generador de trayectorias : Linea inicial

// Generador trayectorias para dinamica : Curva cerrada

Delta\_Sd=61;

Delta\_Si=39;

d= 1966 ;

printf ("Velocidad de la rueda izq..=%d \t\n\n ",Delta\_Si );

printf ("Velocidad de la rueda der..=%d \t\n\n ",Delta\_Sd );

Pre\_Delta\_Sn =add(Delta\_Sd,Delta\_Si);

Delta\_Sn =mult\_r(Pre\_Delta\_Sn,MITAD );

Pre\_Delta\_Theta = sub(Delta\_Sd,Delta\_Si);

Delta\_theta = div\_s(Pre\_Delta\_Theta, d) ;

K = div\_s(182,572) ;

Delta\_THETA\_FRAC =mult\_r(Delta\_theta,K );

Media\_Delta\_Theta = mult\_r(Delta\_THETA\_FRAC, MITAD);

nMax=91 ; // solo 100 puntos para probar , dio 91 para diametro 180

for(k = 40; k < 130; k++)// cada k es un incremento infinitesimal...

{

argu\_x = add(theta[k-1], Media\_Delta\_Theta);

cos\_x = tfr16CosPIx( argu\_x );

sum\_x = mult\_r(Delta\_Sn, cos\_x );

x[k] = add (x[k - 1], sum\_x );

argu\_y = add(theta[k-1], Media\_Delta\_Theta);

sen\_y = tfr16SinPIx( argu\_y );

sum\_y = mult\_r(Delta\_Sn, sen\_y );

y[k] = add (y[k - 1], sum\_y );

theta[k] = add (theta[k - 1],Delta\_theta);

}

for(j=40; j<130; j++)

{

fprintf (fichero,"%d \t %d \t %d \t \n", x[j],y[j],theta[j]);

}

// Fin del generador de trayectorias : Curva cerrada

// Generador trayectorias para dinamica :Linea final

Delta\_Sd=50;

Delta\_Si=50;

d= 1966 ;

printf ("Velocidad de la rueda izq..=%d \t\n\n ",Delta\_Si );

printf ("Velocidad de la rueda der..=%d \t\n\n ",Delta\_Sd );

Pre\_Delta\_Sn =add(Delta\_Sd,Delta\_Si);

Delta\_Sn =mult\_r(Pre\_Delta\_Sn,MITAD );

Pre\_Delta\_Theta = sub(Delta\_Sd,Delta\_Si);

Delta\_theta = div\_s(Pre\_Delta\_Theta, d) ;

K = div\_s(182,572) ;

Delta\_THETA\_FRAC =mult\_r(Delta\_theta,K );

Media\_Delta\_Theta = mult\_r(Delta\_THETA\_FRAC, MITAD);

nMax=40 ; // solo 40 puntos para probar

for(k = 130; k < 169; k++)// cada k es un incremento infinitesimal...

{

argu\_x = add(theta[k-1], Media\_Delta\_Theta);

cos\_x = tfr16CosPIx( argu\_x );

sum\_x = mult\_r(Delta\_Sn, cos\_x );

x[k] = add (x[k - 1], sum\_x );

argu\_y = add(theta[k-1], Media\_Delta\_Theta);

sen\_y = tfr16SinPIx( argu\_y );

sum\_y = mult\_r(Delta\_Sn, sen\_y );

y[k] = add (y[k - 1], sum\_y );

theta[k] = add (theta[k - 1],Delta\_theta);

}

for(j=130; j<169; j++)

{

fprintf (fichero,"%d \t %d \t %d \t \n", x[j],y[j],theta[j]);

}

// Fin del generador de trayectorias : Linea final

// esta porción siguiente describe círculos al aumentar el desplazamiento de una rueda

//for del Si

/\*

for(Delta\_Si = 40 ; Delta\_Si < 51; Delta\_Si++)// cada k es un incremento infinitesimal...

{

Delta\_Sd=50;

d= 1966 ;

printf ("Velocidad de la rueda izq..=%d \t\n\n ",Delta\_Si );

printf ("Velocidad de la rueda der..=%d \t\n\n ",Delta\_Sd );

Pre\_Delta\_Sn =add(Delta\_Sd,Delta\_Si);

Delta\_Sn =mult\_r(Pre\_Delta\_Sn,MITAD );

Pre\_Delta\_Theta = sub(Delta\_Sd,Delta\_Si);

Delta\_theta = div\_s(Pre\_Delta\_Theta, d) ;

K = div\_s(182,572) ;

Delta\_THETA\_FRAC =mult\_r(Delta\_theta,K );

Media\_Delta\_Theta = mult\_r(Delta\_THETA\_FRAC, MITAD);

nMax=800 ;

for(k = 1; k < nMax; k++)// cada k es un incremento infinitesimal...

{

argu\_x = add(theta[k-1], Media\_Delta\_Theta);

cos\_x = tfr16CosPIx( argu\_x );

sum\_x = mult\_r(Delta\_Sn, cos\_x );

x[k] = add (x[k - 1], sum\_x );

argu\_y = add(theta[k-1], Media\_Delta\_Theta);

sen\_y = tfr16SinPIx( argu\_y );

sum\_y = mult\_r(Delta\_Sn, sen\_y );

y[k] = add (y[k - 1], sum\_y );

theta[k] = add (theta[k - 1],Delta\_theta);

}

for(j=0; j<nMax; j++)

{

fprintf (fichero,"%d \t %d \t %d \t \n", x[j],y[j],theta[j]);

}

} // cierro el for del Si \*/

if( !fclose(fichero) )

printf( "Fichero cerrado\n" );

else

{

printf( "Error: fichero NO CERRADO\n" );

}

}

/\* END TPN1 \*/

/\*

\*\* ###################################################################

\*\*

\*\* This file was created by UNIS Processor Expert 2.98.02 [03.79]

\*\* for the Freescale 56800 series of microcontrollers.

\*\*

\*\* ###################################################################

\*/

Representacion en matlab

Codigo :

g=load('C:\Users\Jose\Documents\TP1\_JOSE\_ROBOT\datos.txt');

ang = 0.0054931\*g(:,3);

h=2000/32768;

x=g(:,1)\*h;

y=g(:,2)\*h;

plot(x,y,'ob');

hold on;

axis([-500 2500 -20 2000]);

grid on;

hold on;

krad=3.1415/32768;

ang\_rad=krad\*g(:,3);

dx=10\*cos(ang\_rad);

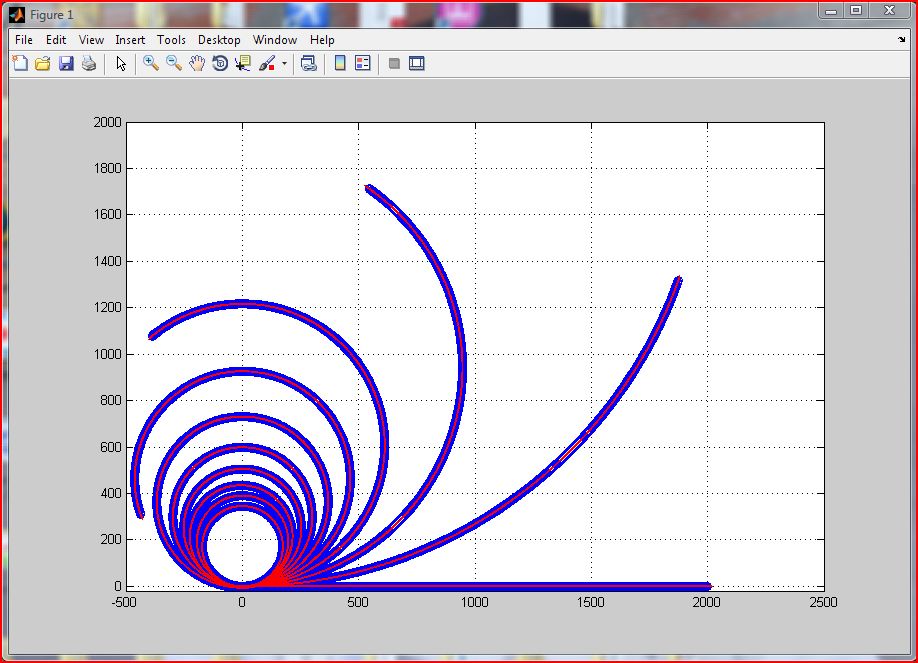
dy=10\*sin(ang\_rad);

quiver(x,y,dx,dy,0.5,'r');

El código adquiere el archivo de texto que provee el dsp y lee las coordenadas de cada punto y su orientación.

Por seguridad el archivo se guarda en una carpeta , y matlab lo lee de otra, en la cual es copiado en forma manual.

Trayectorias:



En este acercamiento se pueden apreciar los vectores de la orientación

