Desarrollo de Software para la Programación y Operación del Manipulador Robótico CRS A255

Douglas Casa M., Danilo Coque C.

***Resumen*-- El presente Proyecto optimiza la operación que se realiza sobre el manipulador para su correcto funcionamiento, mediante la construcción de tarjetas electrónicas que se conectan al controlador actual (CAD) mediante comunicación I2C, adicionalmente se desarrolla una interfaz gráfica, que incorpora un compilador, la misma que permite al usuario controlar al manipulador robótico mediante comandos, dicho software tiene similares características al RobComm tanto en programación online, offline y guiada/textual.**

***Palabras-Clave*— Manipulador Robótico CRS A255, Comunicación I2C, Comunicación Serial, Compilador e intérprete de comandos, Algoritmo de movimiento simultáneo.**

I. INTRODUCCIÓN

Hoy en día los procesos industrializados a gran escala incorporan manipuladores robóticos con mayores prestaciones, razón por la cual, se han desarrollado aplicaciones en hardware y software para cumplir con todas las exigencias, permitiendo optimizar recursos, reduciendo tiempo de trabajo y mejorando la calidad del producto final. Por tal motivo, es importante el desarrollo de aplicaciones con tecnología abierta que puedan complementarse e innovarse con el transcurso del tiempo.

El presente proyecto propone que los manipuladores robóticos CRS A255 vuelvan a estar operativos y no depender de proveedores que utilizan tecnología cerrada y software propietario. Una vez realizado un estudio y análisis previo, el proyecto se divide en dos etapas: la primera etapa es la optimización de las tarjetas de control y potencia que permiten la operación de las cinco articulaciones del manipulador robótico simultáneamente.

Esto se realizó mediante la implementación de algoritmos de movimientos a localizaciones en su área de trabajo; y una segunda etapa que comprende el diseño y desarrollo de un software de código abierto, seleccionando comandos que no impliquen matemática de robots.

Douglas Casa M., Danilo Coque C.
Universidad de las Fuerzas Armadas – ESPE, Ingeniería Electrónica en Automatización
E-mails: doug2208@hotmail.com, daniloxc_28@live.com.

Además se incorporó al software un compilador-interprete que permite la transmisión de datos desde el controlador al computador y viceversa; dotando de varias funcionalidades al manipulador y así mejorar el aprendizaje de los estudiantes en la cátedra de robótica industrial.

II. MANIPULADOR ROBÓTICO CRS A255

El manipulador robótico CRS A255 posee una configuración antropomórfica es decir anatómicamente similar a la extremidad superior del cuerpo humano, está constituido por cinco grados de libertad o articulaciones que son cadera, hombro, codo, muñeca y su rotador.

Adicionalmente cuenta con un efector final o gripper, el cual le permite manipular diferentes tipos de objetos.

Fig. 1. Manipulador Robótico CRS A255.

PRINCIPALES CARACTERÍSTICAS	
Estructura	Cinco grados de libertad
Sistema de Accionamiento	Motores DC electromecánicos
Peso total	19 Kilogramos
Frenos	Un freno por articulación con excepción de la primera articulación.
Conexión efector final	Accionamiento eléctrico-neumático
Carga útil nominal	1 kilogramos

Tabla 1. Características del Manipulador CRS A255.

El controlador que actualmente gobierna al manipulador robótico CRS A255 no es el de fábrica, sino que es la primera etapa de actualización del controlador. Se diseñó e implementó un nuevo controlador denominado CAD (Controlador Andrea Daniel). Su estudio y análisis está constituido por tres etapas en su diseño:

Etapa de Alimentación

Encargada de suministrar el voltaje y amperaje necesarios para el correcto funcionamiento del sistema, tomando la alimentación proveniente de la red eléctrica y acoplándola a los requerimientos del sistema. El sistema implementado cuenta con dos fuentes independientes.

Etapa de Potencia

Encargada de acondicionar las señales para poder accionar los motores y el solenoide desde el controlador. También de proveer la potencia necesaria para manejar un gripper neumático. Los requerimientos eléctricos de cada motor que son 35 VDC a 2 A.

Etapa de Control

El sistema de control tiene una configuración "maestro-esclavo". El maestro realiza una comunicación bidireccional con el computador, además gestiona la comunicación con los esclavos. Los esclavos son los encargados de proveer las señales necesarias para el circuito de accionamiento de cada motor en la tarjeta de potencia. El maestro y los esclavos son PIC's 16f877A

III. DISEÑO E IMPLEMENTACIÓN DE HARDWARE

Para cumplir con todas las expectativas propuestas; es necesario realizar algunas modificaciones en la etapa de potencia y en la etapa de control, ya que el sistema requiere:

• El funcionamiento de las cinco articulaciones del manipulador robótico simultáneamente.
• Disponer con la capacidad necesaria para soportar el software.
• Dotar de puertos de entrada y salida para la interacción del manipulador con su entorno.

Diseño de Tarjeta de Potencia.

Para que el manipulador robótico realice un correcto desempeño, se debe dejar en funcionamiento los dos circuitos de activación restantes de los motores que no se encontraban operativos. Tomando en cuenta que los cinco motores que controlan las articulaciones ahora deben funcionar simultáneamente

Se debe proteger a cada motor con fusibles independientes, contra posibles variaciones en la corriente o daños que se presente durante la manipulación de los mismos. Adicionalmente se colocará porta-fusibles en la parte frontal exterior del chasis, ya que al ser un sistema de entrenamiento para estudiantes, el reemplazo de fusibles defectuosos no sea demoroso.

En el siguiente diagrama de bloques se describe los cambios que se realizará a la tarjeta de potencia

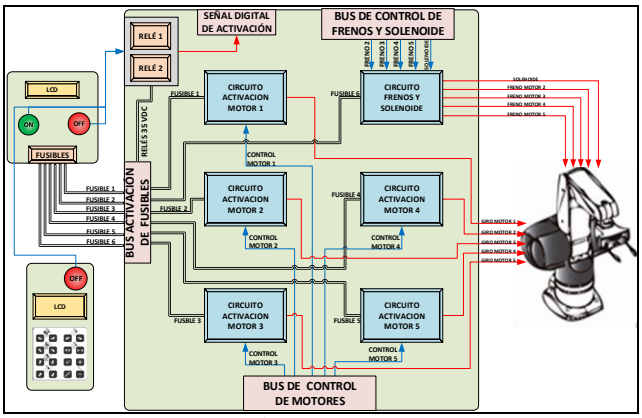

Fig. 2. Diagrama de funcionamiento de tarjeta de control.

Se puede observar que se adaptará un bus que permite la conexión entre los porta-fusibles ubicados en el panel exterior del case con la tarjeta de potencia.

El acabado final de la tarjeta montada en el case es el siguiente

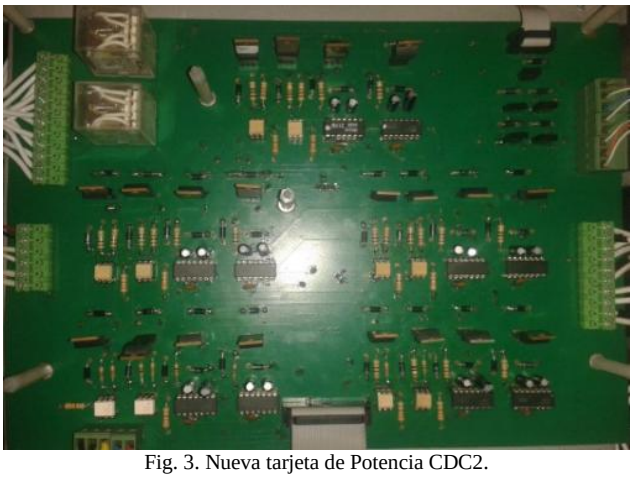
Fig. 3. Nueva tarjeta de Potencia CDC2.

Diseño de Tarjeta de Control

Es necesario que los tipos de operación del sistema (modo manual y modo online) trabajen en conjunto, es decir que el sistema reciba órdenes del teach pendant y por comandos.

El microcontrolador "maestro" actual será reemplazado por un PIC 18F452.

La razón principal por la cual se decidió trabajar con este componente fue que su capacidad de memoria cuadruplica la capacidad del microcontrolador actual, permitiendo que a futuro se pueda implementar nuevas opciones y aplicar técnicas de control avanzadas.

Se trabajará con cinco microcontroladores “esclavos” PIC 16F88.

Al trabajar con un microcontrolador por cada articulación, se puede aprovechar al máximo los recursos y capacidades que ofrece este, permitiendo que se pueda implementar nuevas alternativas en el control del motor.

En el siguiente diagrama de bloques se muestra el diseño de la nueva tarjeta de control.

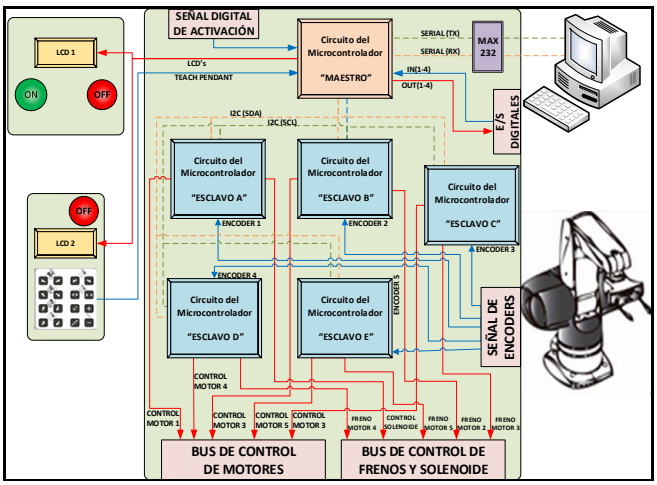

Fig. 4. Diagrama de tarjeta de control.

Al microcontrolador llegan tres señales de entrada de enconder, cabe recalcar que el circuito integrado 74LS14 se descartó para no tener pérdida de pulsos como sucedía en la antigua tarjeta.

Las tres señales de encoder (A, B, Z) se configuran de la siguiente forma:

• El canal A está conectado al pin RA4, este pin cuenta con el módulo contador Timer0, permitiendo que el conteo de pulsos se lleve a cabo de forma independiente al programa principal del microcontrolador.

• El canal B está conectado al pin RB0, este pin cuenta con la interrupción INT0, la cual es útil para conocer el sentido de desplazamiento de una articulación.

• El canal Z está conectado del al pin RA3, este pin se utiliza cuando se desee conocer cuántas revoluciones ha dado el motor.

Para el control de movimiento del motor no fue necesario añadir cambio por lo que se conserva la misma configuración de la tarjeta anterior.

El acabado final de la tarjeta montada en el case es el siguiente

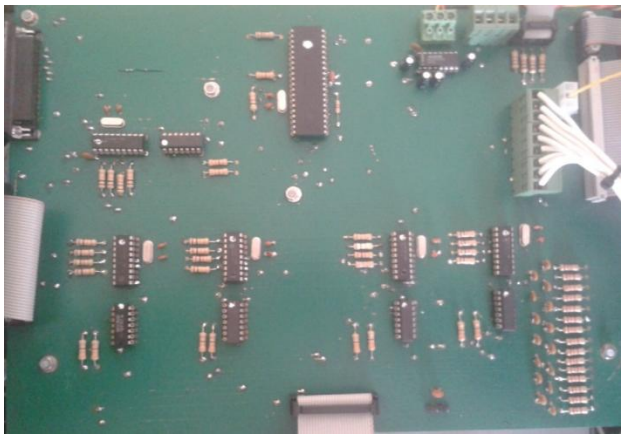
Fig. 5. Nueva tarjeta de control CDC2.

Los cambios realizados al case del controlador, con los nuevos componentes que se implementaron son los siguientes


Fig. 6 Case del Manipulador Robótico.

IV. DISEÑO E IMPLEMENTACIÓN DE SOFTWARE

Se pretende desarrollar un software (interfaz gráfica de usuario), que emule el entorno de trabajo y ciertas funcionalidades del software RobComm. Para lo cual se requiere seleccionar los comandos que no impliquen cinemática o dinámica de robots, estructurar la sintaxis de los mismos para el envío y recepción de datos. Además, al ser un software de programación se requiere un compilador para que los comandos sean ejecutados correctamente por el manipulador robótico.

En el siguiente diagrama se presenta los casos de uso general del sistema.

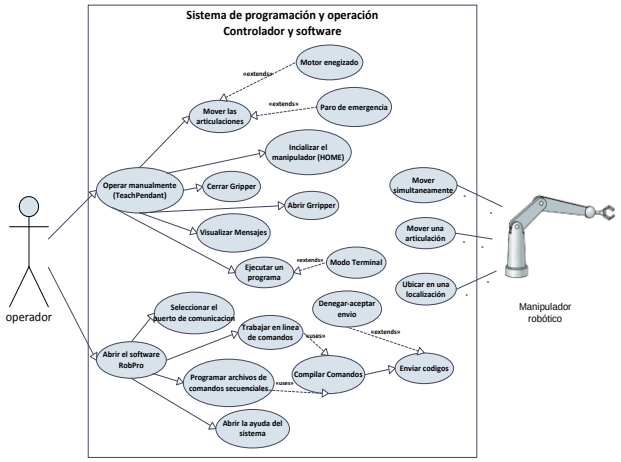

Fig. 7. Diagrama de uso de casos general.

Para establecer los comandos que se van a implementar en el software, se consideraron aquellos que no requieren de un modelo matemático.

COMANDO DE ROBCOMM			
Control de Programa	**Localización**	**Movimiento**	**Operadores**
IF	HERE	MOTOR	OPEN
WHILE	DLOCN	MOVE	CLOSE
INT		JOINT	LIMP
VAR		READY	LOCK
RUN		HOME	NOLIMP
FINISH		PITCH	UNLOCK
PAUSE		ROLL	
DELAY		CPATH	
RUN			
WAIT			
INPUT			
OUTPUT			

Tabla 2. Comandos del manipulador robótico.

Diseño del Algoritmo de Posicionamiento de Motores

Se pretende establecer un algoritmo general para el desplazamiento simultáneo de los motores de un punto de localización (coordenadas angulares de todas las articulaciones) a otro, para lo cual se enumera los pasos a seguir:

Leer posición deseada: PosD
Leer posición actual: PosA
Calcular pulsos y sentido a moverse:
SI sentidodeseado==sentidoactual ENTONCES
SI PosD>PosA ENTONCES
Calcular: PosF=PosD-PosA
SI sentidodeactual==1 ENTONCES
Colocar : SenM=1
SI NO
Colocar: SenM=0
FINSI
SI NO
SI PosA>PosD ENTONCES
Calcular: PosF=PosA-PosD
SI sentidoactual==1 ENTONCES
Colocar: SenM=0
SI NO
Colocar: SenM=1
FINSI
SI NO
PosF=0
Desactivar PWM
Habilitar freno
FINSI
SINO
PosF=PosD+PosA
SI sentidodeseado==1 ENTONCES
SenM=1
SI NO
SenM=0
FINSI
FINSI
Realizar: el movimiento de una articulación.

Para mejor entendimiento del diagrama de flujo del Comando Move, a continuación describe el funcionamiento del algoritmo de posicionamiento del motor de un esclavo, para lo cual se analizan los siguientes casos:

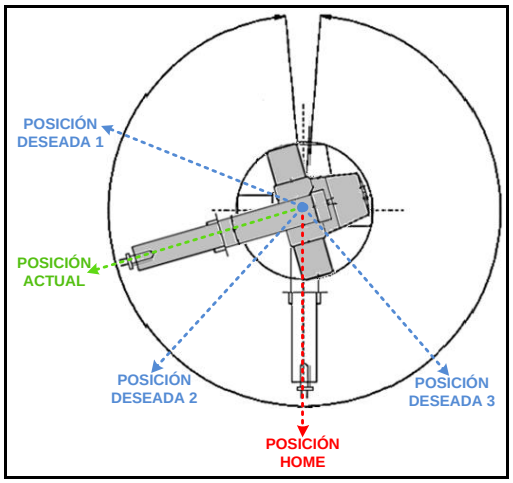

Fig. 8. Comando de MOVE entre dos puntos.

En la figura 8 se conoce cuantos pulsos se ha movido desde POSICIÓN HOME hasta POSICIÓN ACTUAL (PosF) y su sentido (Sen). En (PosD) se almacena la POSICIÓN DESEADA y en (dato [0x01]) el sentido al que se desea desplazar.

En el primer caso: La POSICIÓN DESEADA 1 se encuentra en el mismo cuadrante que POSICIÓN ACTUAL, por lo tanto, el sentido es el mismo. El desplazamiento de la POSICIÓN DESEADA es mayor a POSICIÓN ACTUAL, entonces el número de pulsos a mover (PosF2) será el resultado de la resta de: **PosD-PosF** y el sentido se mantiene.

En el segundo caso: La POSICIÓN DESEADA 2 se encuentra en el mismo cuadrante que POSICIÓN ACTUAL, por lo tanto, el sentido es el mismo. El desplazamiento de la POSICIÓN DESEADA es menor a POSICIÓN ACTUAL, entonces el número de pulsos a mover (PosF2) será el resultado de la resta de: **PosD-PosF** y en sentido será contrario al actual.

En el tercer caso: La POSICIÓN DESEADA 3 se encuentra en cuadrante opuesto al de POSICIÓN ACTUAL, entonces el sentido de desplazamiento cambia, entonces el número de pulsos a mover (PosF2) será el resultado de la suma de **PosD+PosF** y el sentido será el de POSICIÓN DESEADA.

Implementación de Software

El algoritmo para realizar el compilador-interprete del software es siguiente.

```
Seleccionar las 3 primeras letras del comando
SI serror==1 ENTONCES
SI las 3 primeros caracteres == HOM ENTONCES
  SI home==1 ENTONCES
    envío serial "O\r"
    Home-1
  SI NO
    envío serial "K\r"
  FINSI
FINSI
FinSi
SI serror==1 Y home==1 ENTONCES
  SEGÚN Las 3 primeros caracteres HACER
    CASO Comando directo
      envío código
    CASO Comando con parámetros
      ang=vacío
      SEGÚN art HACER
        CASO 1
          Determinar sentido
          Ang=seleccionar dato hasta ";"
          Envío código
        CASO 2
          Similar Caso 1
        CASO 3
          Similar Caso 1
        CASO 4
          Similar Caso 1
        CASO 5
          Similar Caso 1
        FINSEGUN
      FINSEGUN
    CASO Comando con retorno y parámetros
      Crear un espacio de memoria para la recepción
      Envío código
    CASO Comando interno
      Seleccionar la operación
      Evaluar si existe el elemento
      Realizar acción
    FIN SEGÚN
SI NO
  SI serror==0 Y paroE==falso ENTONCES
    Presentar "Error de compilación"
  FINSI
  SI home==0 Y serror ==1 ENTONCES
    Presentar "Indique posición HOME"
  FINSI
FINSI
```

El siguiente diagrama de bloques se esquematiza el funcionamiento interno del software.

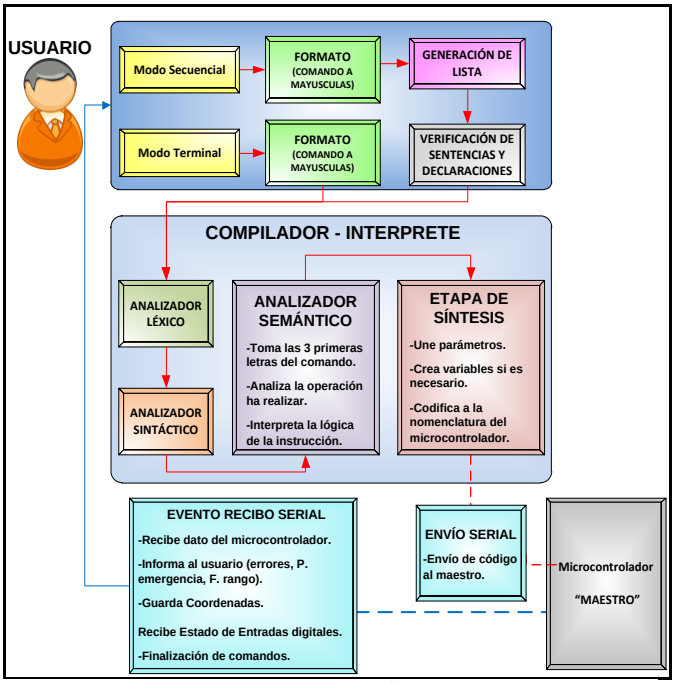

Fig. 9. Diagrama de bloques del software implementado.

Cuando el usuario interactúa sobre la interfaz y realiza acciones como: abrir, crear, editar, borrar, los comandos ingresados pasan a una etapa de formato donde se los transforma a mayúsculas, con el objetivo de normalizar la información y evitar errores.

Si se encuentra en el modo de trabajo terminal la información pasa directo a una etapa de tratamiento del mismo; caso contrario, si se encuentra en modo secuencial, antes de ir a la etapa de tratamiento, los comandos son ingresados a una lista, donde se asigna un índice a cada uno de ellos.

Una vez en la lista pasan al reconocimiento de sentencias de control de programa como: IF, WHILE, VAR, INT, PAUSE; y luego cada comando pasa a la etapa de tratamiento.

Siguiendo el algoritmo de diseño de un traductor, se ha decidido no diseñar un compilador compacto; sino, la unión de un compilar-interprete, el mismo que analizará la

estructura de cada comando, verificando que haya concordancia en el mismo y generar instrucciones codificadas para que el microcontrolador "maestro" realice acciones sobre el mismo.

Para lo cual el compilador-interprete tiene la siguiente estructura:

- Analizador léxico.
- Analizador sintáctico.
- Analizador semántico.
- Etapa de síntesis.

Jlex genera el analizador léxico de nuestro compilador. Java Cup genera el analizador sintáctico del compilador.

Esta etapa permite conocer la operación que el usuario quiere realizar sobre el manipulador. Además determina si se debe enviar parámetros o reservar espacios de memoria para los datos de retornos por parte del microcontrolador. Y finalmente codifica el comando y lo envía.

El proyecto de habilitación del manipulador robótico CRS A255, cuenta con los siguientes componentes:

- Manipulador robótico
- Controlador CAD.
- Teach Pendant.
- Software RobPro.

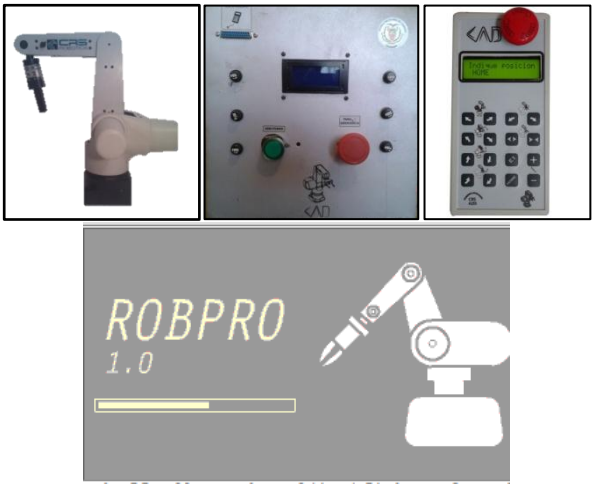

Fig.10 Componentes actuales del sistema.

V. PRUEBAS Y RESULTADOS

Las razones por la cual se decidió rediseñar la tarjeta de control, se debió a pérdidas de pulsos en el contador del microcontrolador, por motivos del inversor por disparo (LN7414). Las pruebas para determinar estos resultados se realizaron con el motor de la primera articulación a una frecuencia constante, variando el porcentaje de duty, en sentido horario y anti-horario.

Freq: 20.2 KHz	Teórico	Experimental CW (subida)	Experimental CCW (bajada)
Duty 25%	150 pulsos/10ms	144 pulsos/10ms	166 pulsos/10ms
Duty 50%	300 pulsos/10ms	207 Pulsos/10ms	222 Pulsos/10ms
Duty 75%	450 pulsos/10ms	327 pulsos/10 ms	339 pulsos/10 ms
Duty 100%	600 pulsos/10ms	413 pulsos/10ms	440 pulsos/10ms

Tabla 3 Conteo de pulsos con tarjeta CAD.

Se puede observar que a un mayor porcentaje del duty, la pérdida de pulsos se vuelve considerable, evitando que se pueda determinar la posición correctamente del manipulador.

Freq: 20.2 KHz	Teórico	Experimental CW (subida)	Experimental CCW (bajada)
Duty 25%	150 pulsos/10ms	148 pulsos/10ms	147 pulsos/10ms
Duty 50%	300 pulsos/10ms	298 Pulsos/10ms	297 Pulsos/10ms
Duty 75%	450 pulsos/10ms	445 pulsos/10 ms	443 pulsos/10 ms
Duty 100%	600 pulsos/10ms	592 pulsos/10ms	589 pulsos/10ms

Tabla 4. Conteo de pulsos tarjeta CDC2.

La pérdida de pulsos a porcentajes de duty alto no son tan considerables como ocurría con las anteriores tarjetas, lo que nos permite llevar un correcto conteo de pulsos cuando se analice velocidad angular

Pruebas de Conteo de Pulsos en cada Articulación

Para realizar pruebas de presión se utilizarán los comandos JOINT, MOTOR, PITCH Y ROLL. Los cual nos proporcionará conocer el porcentaje de error en el movimiento realizado por cada motor.

Movimiento Articulación 1

	Conteo de pulsos	% error	Angulo desplazado	% error
Teóricos	18000	-	90º	-
Sentido CW	18001	0.005	91º	1.11
Sentido CCW	18001	0.005	91º	1.11

Tabla 5. Conteo de pulsos y ángulo desplazado de Art.1.

Movimiento Articulación 2

	Conteo de pulsos	% error	Angulo desplazado	% error
Teóricos	9000	-	45º	-
Sentido CW	9001	0.005	46º	2.22
Sentido CCW	9001	0.005	40º	11.11

Tabla 6. Conteo de pulsos y ángulo desplazado de Art.2.

Movimiento Articulación 3				
	Conteo de pulsos	**% error**	**Angulo desplazado**	**% error**
Teóricos	18000	-	90º	-
Sentido CW	18001	0.005	93º	3.33
Sentido CCW	18001	0.005	91º	1.11

Tabla 7. Conteo de pulsos y ángulo desplazado de Art.3.

Movimiento Articulación 4				
	Conteo de pulsos	**% error**	**Angulo desplazado**	**% error**
Teóricos	2430	-	45º	-
Sentido CW	2431	0.005	46º	2.22
Sentido CCW	2431	0.005	40º	11.11

Tabla 8. Conteo de pulsos y ángulo desplazado de Art.4.

Movimiento Articulación 5				
	Conteo de pulsos	**% error**	**Angulo desplazado**	**% error**
Teóricos	18000	-	90º	-
Sentido CW	18001	0.005	91º	1.11
Sentido CCW	18001	0.005	91º	1.11

Tabla 9. Conteo de pulsos y ángulo desplazado de Art.5.

Se comprueba que la primera y la quinta articulación realizan movimientos precisos, con un error relativamente pequeño, la tercera articulación tiene un porcentaje no mayor al 5% pero es considerable al momento de aplicar el concepto de repetibilidad.

La segunda y cuarta articulaciones tienen un porcentaje de error muy alto, dichos errores se hacen presentes ya que se encuentran perturbados por factores como: el peso que ejercen las articulaciones sobre una específica, la gravedad presente en movimientos de bajada, inercia por el frenado instantáneo que se realiza sobre ellos.

Pruebas de Repetibilidad

Para realizar estas pruebas se utilizaran los comandos MOVE, READY y CPATH, ya que estos permiten volver a localizaciones programadas las veces que sean necesarias desde un punto en cualquier localización y trasladarse otro punto indistintamente.

Se realizaron pruebas donde se todas las articulaciones se desplazaron con un ángulo específico, y que regresen a la posición inicial. Se detectó que al realizar estos movimientos repetitivos simultáneamente, el manipulador robótico cada vez pierde su orientación de posición inicial Home. Esto se ve afectado por las perturbaciones que se añaden a las articulaciones dos, tres y cuatro.

Como se ven los resultados el robot cuenta con la capacidad de repetibilidad, pero al tener errores significativos, en ciertas articulaciones hace que cada vez que regresa al punto programado aumenta el porcentaje de error, ya que el movimiento simultaneo es decir un error de un motor afecta a la otra por acción de las cadenas.

Pruebas de Transmisión de Datos desde Pc a Controlador y de Controlador a Pc

Al realizar el envío de datos para el comando Move hacia el controlador, se detectó que al transmitir datos consecutivamente sin tiempo de retardo, estos se perdían lo que conlleva a que el controlador realice operaciones erróneas.

Cuando se intentaba enviar datos del comando Here desde el controlador hacia la PC sucedía el mismo efecto anterior se perdían localizaciones de algunos motores.

Para dar solución a dicho problema se determinó enviar los datos con un retardo adicional tanto desde el controlador a la PC y viceversa. Por lo cual el envío secuencial de datos desde la PC se debe considerar un tiempo no menor a 10 milisegundos, adicionalmente el tiempo mínimo de envío desde el controlador debe ser no menor a 250 milisegundos.

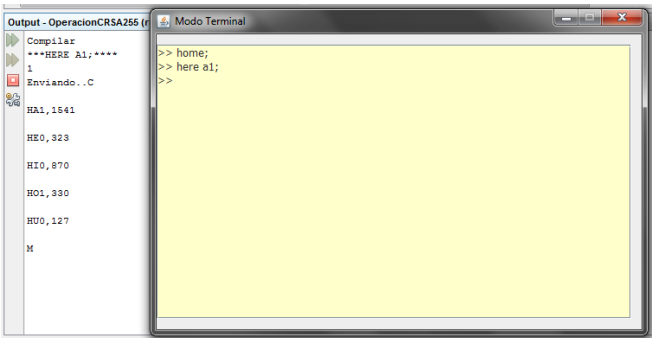

Fig. 11. Recepción de datos con el comando Here.

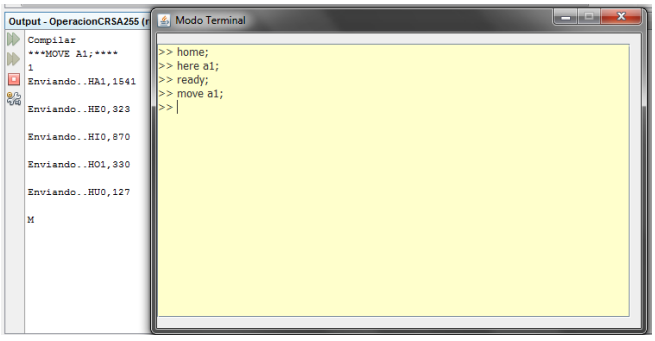

Fig. 12. Envío de datos con el comando Move.

Pruebas de Seguridades del Sistema

Se realizó pruebas de respuestas del sistema cuando se

acciona el paro de emergencia y posteriormente se hizo pruebas de seguridad de movimientos dentro del rango.

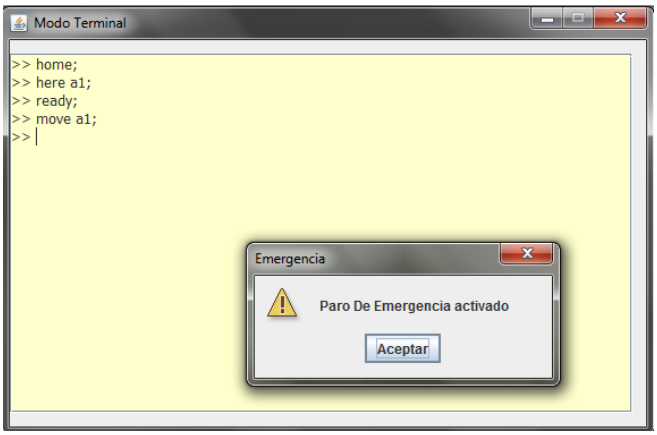

Fig. 13. Activación de paro de emergencia en modo terminal.

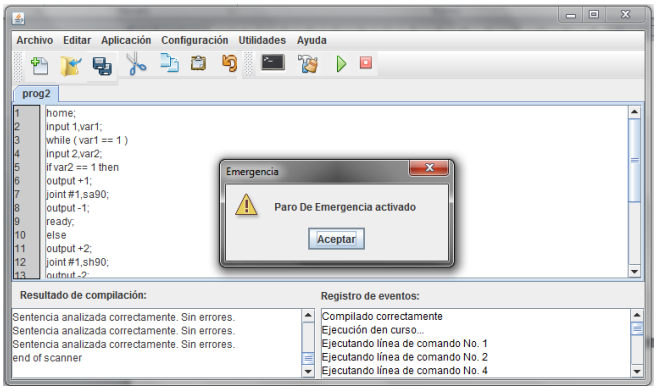

Fig. 14. Activación de paro de emergencia en modo secuencial.

VI. Conclusiones

– Al comandar con un solo microcontrolador el movimiento de un motor, este dota de mayores recursos y capacidades para realizar un mejor desempeño del sistema, ya que no se comparte: memoria, módulos temporizadores, módulos contadores y puertos con otro motor.

– Para el movimiento simultáneo de las cinco articulaciones del manipulador robótico, se establece que cada microcontrolador esclavo tome las decisiones para el correcto funcionamiento de un motor, realizando cálculos para el movimiento y solo depender del maestro para recibir y enviar datos. Sin importar las acciones que estén realizando los demás esclavos.

– Para realizar el movimiento del manipulador a localizaciones previamente guardadas, se implementó un algoritmo de movimiento simultáneo de las cinco articulaciones, sin que involucre un desarrollo matemático complejo, dotando al sistema de la capacidad de trasladarse de una posición a otra sin tener que pasar por la posición de HOME, teniendo en cuenta que no genera una trayectoria especifica.

– El uso de herramientas y librerías que ofrece el lenguaje java, como JLex y Java Cup, han permitido adaptar la estructura de los comandos seleccionados a un compilador básico, que permite de manera independiente del programa fuente, conocer si existe un error en la gramática del comando, así poder prevenir de posibles errores, antes que se envíe información al controlador y provoque un funcionamiento erróneo.

– Para la implementación del analizador semántico, se optó por desarrollar un algoritmo que seleccione las tres primeras letras del comando escrito por el usuario, esto permite que se optimice los recursos de procesamiento de información por parte de la PC, logrando que la codificación del mismo sea realice en un menor tiempo.

– El uso de clases tipo hilo en la programación del software, es de gran importancia ya que permiten realizar subprocesos que no tienen mayor relevancia en la ejecución del programa principal, es decir esto permite que mientras el usuario este interactuando con la interfaz, otra operación puede estar ejecutándose, sin importar que este haya finalizado la operación para comenzar una nueva acción.

– Al implementar la segunda etapa del proyecto de rehabilitación de los controladores del manipulador robótico CRS A255, se ha logrado que la cátedra de robótica cuente con prototipos funcionales para realizar prácticas de laboratorio y así contribuir para un mejor aprendizaje de los estudiantes.

VII. Recomendaciones

– Antes de utilizar el software implementado es recomendable, primero revisar la sintaxis de las instrucciones, y como se deben escribir las mismas, además de conocer la operación que realiza cada comando sobre el manipulador robótico.

– Es recomendable familiarizarse con el manipulador en modo manual con el Teach Pendant, antes de trabajar con el sistema en modo online, ya que se puede conocer las características de movimiento de cada articulación.

– Hay que tomar precauciones en la programación de microcontroladores cuando se utiliza diferentes familias como los PIC's 18f y 16f, ya que no mantienen las mismas estructura dentro de un compilador, esto provoca

incorporar librerías adicionales para el correcto funcionamiento.

– Se recomienda para la realización de trabajos futuros dotar al sistema de una técnica de control avanzada, lo cual permitirá que el mismo pueda trabajar a velocidades diferentes, dependiendo de las tareas a realizar.

– Un aporte adicional para el mejoramiento del software, es la optimización del compilador implementado, por uno más avanzado que permita adicionar mayores prestaciones, mejorando la estructura de las sentencias, auto competición de instrucciones, opciones de corrección, entre otros.

– Es recomendable el estudio de técnicas y normas para el diseño de interfaces graficas ya que así el usuario puede interactuar amigablemente con el software, además el mismo debe contar con herramientas que faciliten el trabajo como ayudas, manual de usuario, botones de acceso rápido y avisos emergentes.

– Se recomienda el desarrollo de la técnica de control sobre el modulo temporizador TIMER1 ya que se dejó este módulo activado para no dañar la programación ya realizada.

– La realización de prácticas de laboratorio, simulando que el manipulador robótico se encuentra en un ambiente industrial. Como son: procesos de selección, control de calidad, desarrollo de producción, entre otros.

– Dotar al software de comunicación DDE, esto permite que el mismo pueda comunicarse con otros programas, y así poder complementarlo con un sistema SCADA.

REFERENCIAS

- Barrientos, A., Peñín, L., Balaguer, C., & Aracil, R. (1997). *Fundamentos de Robótica* (Primera ed.). Madrid, España: McGRAW-HILL.
- Craig, J. (2006). *Robótica* (Tercera ed.). México Df: Pearson educación.
- Gadgets, H. (13 de 12 de 2007). *Hacked Gadgets*. Obtenido de Toyota robotic musical: http://hackedgadgets.com/2007/12/13/toyota-robotics-musical-robots/#section3
- Galvez, S., & Mora, M. (2005). *Compiladores.* Málaga: Universidad de Málaga.
- Kuka. (2013). *Kuka-robotics*. Obtenido de Products: http://pdf.directindustry.es/pdf/kuka-roboter/nuevos-rapidos-precisos-kuka-small-robots/17587-435871.html
- Maldonado, D., & Sánchez, A. (2013). *Estudio, Diseño e implementación de un controlador de tres grados de libertad para el manipulador robótico crs a255*. Sangolquí: Escuela Politécnica del Ejército.
- Mocencahua Mora, D. (2009). *Facultad de ciencias electrónicas*. Obtenido de Ftp Descargas: http://www.ece.buap.mx/ini/des.html
- Ollero Baturone, A. (2001). *Robótica Manipuladores y Robots Móviles.* Barcelona, España: Marcombo.
- Platea, & Gonzalez F, V. (2013). *Control y Robótica*. Obtenido de Clase de robots: http://platea.pntic.mec.es/vgonzale/cyr_0708/archivos/_15/Tema_5.2.htm
- Reyes Cortés, F. (2011). *Robótica Control de Robots Manipuladores.* México Df: Alfaomega.
- Sass, L. (25 de 07 de 2013). *Introducción a la Robótica.* Obtenido de Universidad de San Francisco de Quito: http://profesores.usfq.edu.ec/laurents/IME440/IME440_RobotManip.pdf.
- Williams, K. (2006). *Thinkbotics*. Obtenido de Products: http://www.thinkbotics.com/products.html