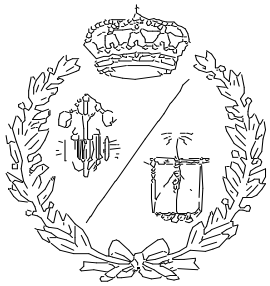
ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

UNIVERSIDAD DE CANTABRIA



Trabajo Fin de Grado

Implementation of a 6-rss Stewart platform as a earthquake wave simulator

Para acceder al Título de

GRADUADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

Autor: Jorge Varona Madrid

9 -2020

Resumen

En este trabajo de fin de grado se desarrollará el diseño de un simulador de seis grados de libertad, para reproducir ondas sísmicas. Para ello se emplearán diversos componentes: el robot paralelo, el autómata programable, el software del autómata programable; así como el software que reproducirá y simulará los movimientos sísmicos, y el ordenador donde se ejecutará este último. Se tratará como se relacionan entre si los componentes para dar lugar al simulador. El componente central de nuestro simulador será la plataforma Stewart, que es un manipulador paralelo.

Índice

Índice de figuras

Índice de esquemas

1. Introducción.

Durante el desarrollo de este trabajo de fin de grado veremos en primer lugar el objetivo que se persigue. Tras lo cual estudiaremos los diferentes aspectos teóricos tanto en el lado del hardware (plataforma Stewart) como del software (Simulink, CoDeSys, OPC-ua). También veremos la implementación de forma práctica del simulador, junto con los resultados obtenidos.

2. Objetivo.

Con este trabajo de fin de grado se pretende desarrollar la implementación de una plataforma Stewart (o robot hexápodo) como simulador de ondas sísmicas, a través del uso de un autómata

programable que realizara el control de los actuadores de la plataforma. Los actuadores serán 6 servomotores que proporcionarán el movimiento a la plataforma móvil del robot. El autómata realizara el control de los servomotores en función de los datos recibidos, los cuales serán gestionados por el programa del autómata. Los datos serán enviados desde un ordenador, para la comunicación entre los programas del ordenador y el autómata se empleará como protocolo de comunicación: OPC-ua.

3. Antecedentes.

3.1. Plataformas Robóticas.

Para comprender con solvencia que es el robot paralelo del simulador de ondas sísmicas, en este apartado nos detendremos en las principales definiciones y características de las plataformas robóticas, poniendo especial interés en la plataforma Stewart (que ha sido la empleada en este simulador). En él veremos la evolución que han sufrido los robots paralelos a lo largo del tiempo, desde la antigüedad hasta el siglo XXI; y realizaremos un análisis cinemático de ellos como mecanismo, su espacio de trabajo y las singularidades que pueden alcanzar.

También observaremos y analizaremos, las ventajas y desventajas que presentan los robots con esta configuración(paralela) frente a los robots en serie.

Finalmente expondremos las principales aplicaciones de los robots paralelos, entre las que se incluyen: simuladores de vuelo y conducción, en el campo de la medicina en aplicaciones como

cirugías o rehabilitaciones ortopédicas entre otras; su uso como manipuladores para desplazar y rotar objetos en la posición deseada; o como giroscopio.

3.1.1. Contexto histórico.

Un robot es una entidad virtual o mecánica artificial. En la práctica, esto es por lo general un sistema electromecánico que, por su apariencia o sus movimientos, ofrece la sensación de tener un propósito propio. La independencia creada en sus movimientos hace que sus acciones sean la razón de un estudio razonable y profundo en el área de la ciencia y tecnología.

El término Robot, proviene de la palabra checa robota que significa “trabajo forzado”, fue introducida por primera vez por el dramaturgo y autor checoslovaco Karel Capek, en su obra de teatro R.U.R (Robots Universales de Rossum) en 1921; que con frecuencia giraba alrededor de sus puntos de vista sobre el posible peligro de estas máquinas, incorporando la idea de que el ser humano hace el robot y el robot mata al ser humano.

Actualmente la Federación Internacional de Robótica (IFR-ISO 8373) ofrece una definición estandarizada para robot industrial, que es la siguiente:

"Por robot industrial de manipulación se entiende una máquina de manipulación automática, reprogramable y multifuncional con tres o más ejes que pueden posicionar y orientar materias, piezas, herramientas o dispositivos especiales para la ejecución de trabajos diversos en las diferentes etapas de la producción industrial, ya sea en una posición fija o en movimiento (IFR,2011)."

Los robots que conocemos hoy en día fueron desarrollados después de la Segunda Guerra Mundial, debido a la creciente demanda de automatización en la industria del automóvil. Cabe señalar, que, antes los robots no eran más que herramientas para la automatización. Estaban teóricamente programados a realizar una tarea específica: transportar, cargar, descargar, soldar, pintar, etc. Actualmente, existen los llamados robots inteligentes, que efectúan funciones tales como la detección de cualquier modificación de su medio ambiente. Estos actúan

en consecuencia considerando las nuevas modificaciones ya sea cambiando de operaciones o descubriendo una nueva.

La robótica en general ha tenido un papel muy importante en la historia, pues gracias a experimentos y proyectos llevados a cabo por inventores y empresas dedicadas al desarrollo de esta tecnología, han brindado la facilidad de resolver problemas, realizar tareas peligrosas o complicadas que un ser humano no podría hacerlas en condiciones normales.

Se puede dividir la historia de la robótica en 3 etapas, claramente distinguidas. La primera, que hace referencia a hechos o inventos, que para su época podían verse como inútiles, pero serían precursores de la robótica actual. La segunda, una etapa donde todo el esfuerzo se concentró en la industria y que puede ser asociada como la revolución industrial de la robótica, donde se definieron las bases sólidas para la robótica existente. La tercera etapa es la actual, donde la robotización ha entrado fuertemente al mercado de consumo, mostrando muchos de los avances que serán comunes en un futuro cercano. A continuación, se muestra brevemente una línea temporal de la robótica, desde sus inicios, en la época de los griegos, hasta la actualidad (véase Tabla 1).

Tabla 1

AÑO	INVENTO	CARACTERÍSTICAS	AUTOR
270A.C	Relojes de agua	Hecho con figuras móviles	Físico griego Ctesibius
150A.C	Mecanismo de Anticera	Computadora analógica	Griegos
1495	Antrobot	Hombre mecánico	Leonardo Da Vinci
1772	L' Ecrivain	Escribía mensajes de hasta 40 caracteres. Cerebro mecánico.	Pierre Y Henry Jaquet-Droz
1801	Telar programable	Operada mediante tarjetas perforadas.	Joseph Jacquard
1890	Vehículo de control remoto	Usa por primera vez la teleoperación inalámbrica.	Nikola Tesla
1892	Grúa motorizada con garra	Para remover lingotes en el horno.	SewardBabbitt
1940	Elsie la Tortuga	Pre-robot	Grey Walters
1946	Computador electrónico (ENIAC)	Construida con tubos de vacío.	J.Presper Eckert y Jonh Mauchly
1951	Brazo articulado teleoperado	Construido para la comisión de la energía atómica de Estados Unidos.	Raymond Goertz

1954	Primer robot programable	Define el termino Automatización Universal	George Devol
1960	"La bestia"	Robot controlado por transistores	Johns Hopkins
1961	UNIMATE	Soldadura y extracción de piezas de fundición	General Motors
1963	Rancho Arm	Brazo robot controlado por computadora	John McCarthy
1965	Sistema DENDRAL	Sistema experto. Ejecuta el conocimiento acumulado de su información agregada.	Edward Feigenbaum,Universidad de Stanford
1968	Brazo-tentáculo	Tipo pulpo	Marvin Minsk
1969	Brazo Stanford	Brazo robot completamente eléctrico, controlado por computadora	Universidad de Stanford
1970	Robot móvil	Controlado por inteligencia artificial	Shakey
1976	Brazos para las sondas espaciales Viking 1 y 2	Incorporan microcomputadoras	Vircam Inc.
1978	PUMA (Programable Universal Machine for Assembly)	Posee un controlador manual y un brazo robótico	Vircam, Unimation, General motors
1979	Stanford Cart	Cruza exitosamente un salón lleno de sillas	James Adams y Hans Moravee de Standford
1986	NavLab I	Pionero en la navegación en exteriores	NavLab
1990	Ambler	Plataforma de pruebas para investigación sobre robots caminantes	Carnegie MellonUniversity
1992	NavLab II	Pionero en visión trinocular, computación WARP y fusión de sensores	NavLab
1994	Dante II	Toma muestras de gases volcánicos	CMU Robotics
1997	Panthinder	Aterriza en Marte	NASA
1997	P3	Robot humanoide	HONDA
1999	Aibo	Perro-robot	SONY
2000	Robodex 2000	Pequeño humanoide	SONY
2003	Qrio	Primer humanoide comercial autónomo	SONY
2004	Humanoide Robosapiens	Con el fin de reducir las bajas militares de USA en Irak	Dr. Mark W Tilden
2005	MurataBoy	Robot capaz de montar en bicicleta	MurataManufacturing

2005	Big Dog	Robot que pude cargar hasta 340 libras en cualquier terreno	Boston Dinamics
2007	iRobotCreate	Robot diseñado para estudiantes y desarolladores de la robótica	iRobot
2007	Walkamaru	Robot con personalidad independiente	Dr.HiroshiIshiguro
2008	Traje robot HAL(HybridAssistedLimb)	Permite caminar grandes distancias y cargar mucho más peso del que se podría normalmente	Cyberdyne
2009	Roomba	Aspiradora robótica	iRobot
2011	Watson	Robot gana concurso de preguntas y respuestas	IBM
2013	Atlas	Robot bípedo de rescate	Boston Dinamics
2014	Eugene	Supera el test de Turing	Vladimir Veselov y Eugene Demchenko
2016	Taxi Autónomo	Vehículo de conducción autónoma	MIT
2018	AlphaZero	Aprende a jugar al ajedrez por si misma	DeepMind

Como se puede apreciar en esta tabla, desde el inicio, la robótica ha estado presentado avances significativos, y en la actualidad su desarrollo se ha visto impulsado a raíz de la miniaturización de los componentes electrónicos, y especialmente en el aumento de la potencia de cálculo de los ordenadores. Tareas que hasta hace algunos años eran impensables, tales como el reconocimiento visual de objetos, ahora resultan de uso cotidiano en los sistemas industriales.

3.1.2. Contexto histórico de los robots paralelos.

Al contrario de los que podríamos pensar, los trabajos teóricos relacionados con estructuras mecánicas paralelas aparecieron ya hace siglos, antes de que se empezara a hablar de robots, cuando los primeros geómetras franceses e ingleses realizaron sus estudios sobre los poliedros y sus aplicaciones.

Ya en la época más actual, el primer mecanismo paralelo fue patentado en el año 1931. Se trataba de una plataforma de movimiento destina a la industria del entretenimiento diseñada por James E. Gwinnett (desafortunadamente este mecanismo no fue nunca construido). En la Fig.1. se muestra diseño de Gwinnett. Merlet [1].

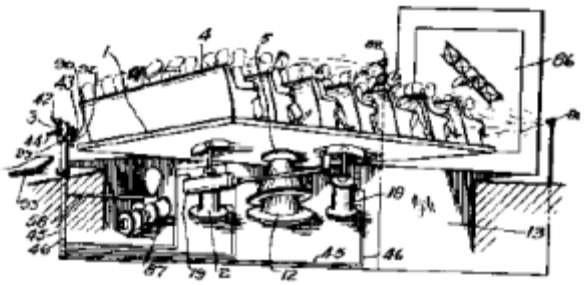


Figura 1 Primer mecanismo espacial paralelo.

No tuvo que pasar mucho tiempo hasta que el primer diseño de un robot industrial paralelo apareciera. En 1940 WillardPollard presento un ingenioso robot de 5 GDL destinado a operaciones de pintura con espray. En la Fig.2., se puede observar que el robot estaba formado por 3 brazos de dos eslabones cada no. Los eslabones estaban unidos mediante juntas universales. Los tres actuadores de la base comandaban la posición de la herramienta, mientras que la orientación era proporcionada por los otros dos actuadores situados en la base y que transmitían el movimiento a la herramienta mediante la rotación proporcionada mediante unos cables flexibles.

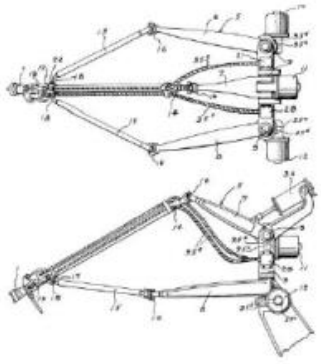


Figura 2 Primer robot industrial paralelo.

En 1947, el DR. Eric Gough diseño un octaedro hexápodo con los lados de longitud variable, como plataforma para la comprobación del comportamiento de los neumáticos de la compañía Dunlop, bajo cargas aplicadas en diferentes ejes. De esta forma intentaba simular el proceso de aterrizaje de un avión, véase Fig.3.

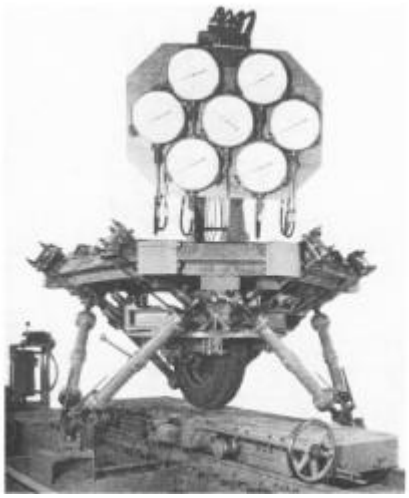


Figura 3Plataforma de Gough.

En 1965 Mr. Stewart presento un artículo en el que describía una plataforma de 6 GDL destinada a trabajar como simulador de vuelo, el diseño de esta se muestra en la Fig.4. Donde la conjunción de las diversas cadenas cinemáticas del mecanismo podía proveer los varios y complejos movimientos de la cabina de un piloto en entrenamiento.

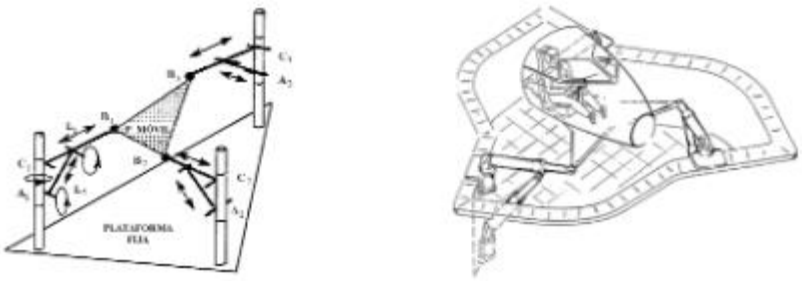


Figura 4 Plataforma Stewart.

El ingeniero Klaus Cappel patento en 1967 un simulador de movimiento basado en un hexápodo y construyo varias de sus invenciones, véase Fig.5.

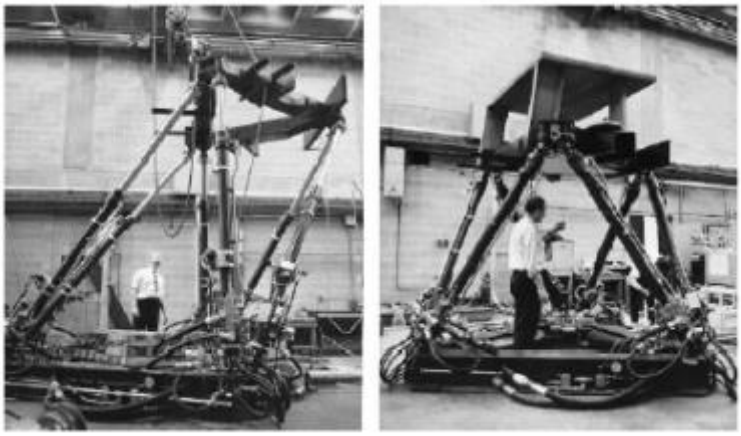


Figura 5 Simulador de movimiento de klauscappel.

En realidad, todos los autores enumerados en este apartado se pueden considerar pioneros ya que desarrollaron sus inventos sin conocimiento previo de los anteriores. Dentro de este grupo también podemos incluir el manipulador paralelo 6-RUS con seis grados de libertad accionado por actuadores giratorios, presentado por K.H. Hunt en 1983, véase Fig.6.

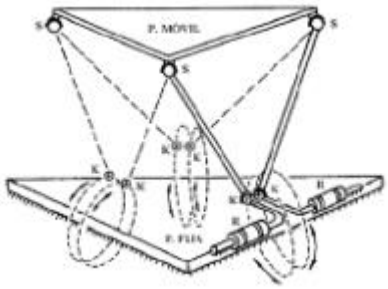


Figura 6 Manipulador paralelo 6-RUS.

Los robots paralelos tienen su propia historia, como se ha visto en este apartado, y en la actualidad se están desarrollando rápidamente. Gracias a la gran capacidad de cómputo de los nuevos procesadores, permitiendo que se interconecten y comuniquen las plataformas robóticas con otros dispositivos externos, por ejemplo, cámaras, ordenadores, sensores táctiles, etc. No obstante, plantean desafíos como la resolución de su cinemática para el control del actuador final.

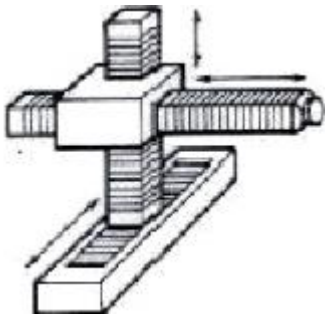
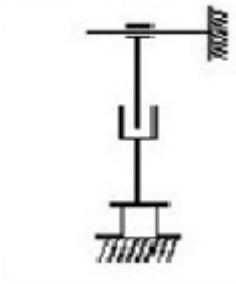
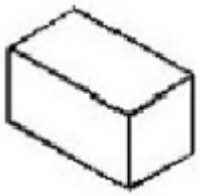
3.2. Clasificación.

En este apartado nos centraremos en las diferentes configuraciones morfológicas y parámetros característicos de los robots industriales. Los parámetros que se consideran es esta clasificación son los siguientes:

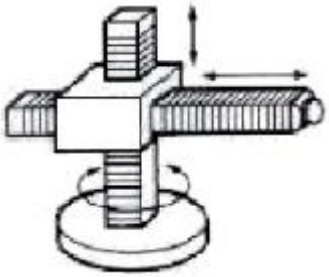
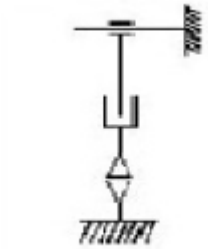
- Numero de grados de libertad.
- Espacio de trabajo.
- Capacidad de posicionamiento del actuador final.
- Capacidad de carga.
- Velocidad.

Por lo tanto, en función de los parámetros anteriormente mencionados y de la estructura mecánica podemos diferencias seis tipos diferentes, que son los que se mencionan a continuación:

- Cartesiano: su posicionamiento en el espacio se lleva a cabo mediante articulaciones lineales.

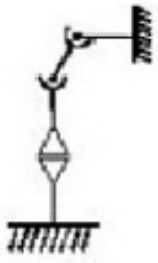

- Cilíndrico: posee una articulación rotacional sobre una base y articulaciones lineales para el movimiento en altura y en radio.

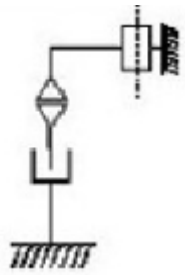
- Polar: cuenta con dos articulaciones rotacionales y una lineal.



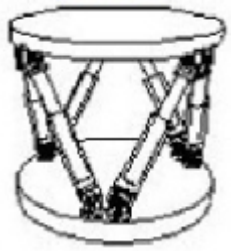
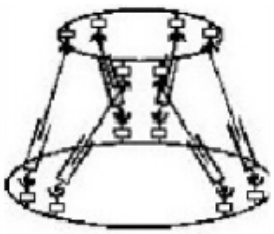

- Esférico (o brazo articulado): cuenta con tres articulaciones rotacionales.

- Mixto: posee varios tipos de articulaciones, combinaciones de los anteriores. Destacando la configuración SCARA.

- Paralelo: posee brazos con articulaciones prismáticas o rotacionales concurrentes.

En las imágenes superiores tenemos cuatro imágenes de cada tipo de robot articulado, siendo la primera su configuración geométrica, la segunda su estructura cinemática, la tercera el espacio de trabajo y en último lugar un ejemplo.

Vemos como el espacio de trabajo del robot paralelo es el mejor adaptado a los requerimientos de un simulador, permitiendo cualquier tipo de trayectoria. Permite trayectorias a través del centro del espacio de trabajo, a diferencia del resto, los cuales se encuentran limitados en este sentido, puesto que en el centro del espacio de trabajo se localiza el brazo robótico. Los robots cartesianos también permiten trayectorias a través del centro del espacio de trabajo, pero estos solamente poseen 3 grados de libertad, lo cual es insuficiente para determinadas aplicaciones. En nuestro caso, para realizar una simulación de una onda sísmica el espacio de trabajo que mejor se adapta a la aplicación es el del robot paralelo, permitiendo cualquier tipo de trayectoria dentro de este espacio y contando con 6 grados de libertad. Como hemos visto todos los robots polimórficos están formados por eslabones, estos están unidos mediante articulaciones, el tipo de articulación que emplea un robot determina en gran medida su movilidad (en la Fig.7. se muestran las articulaciones más comunes clasificadas según los grados de libertad que pueden desarrollar).

Figura 7 Clasificación de articulaciones según sus grados de pertenencia.

Todos los diferentes tipos de robots vistos en este apartado forman el grupo de los robots poliarticulados. Este grupo, los poliarticulados, se diferencia de otros cuatro grupos; robots móviles, androides, zoomórficos y híbridos. Estos cinco grupos engloban cualquier diseño de un robot.

3.2.1. Clasificación de los robos paralelos.

Los robots los robots paralelos pueden ser clasificados en base a varios estándares. Uno de los estándares principales para clasificar a los manipuladores paralelos es su movilidad. De acuerdo con esto podemos diferencias entre robots planares(2D) y articulares(3D). Dentro del grupo de los robots articulares podemos diferenciar dos grupos: los que tienen mecanismos paralelos esféricos y a los transaccionales.

Robot Paralelos Planares.

Los robots planares son aquellos robots paralelos, cuyo efector final tiene el movimiento restringido exclusivamente a un plano en el espacio. El robot paralelo más simple es el mecanismo de 5 barras con 4 eslabones que forman un solo brazo en lazo cerrado, se interconectan entre sí y a la plataforma fija o base mediante cinco articulaciones rotacionales; posee dos grados de libertad (movimiento dentro de un plano, x e y). El diseño de esta estructura lo observamos en la Fig.9.

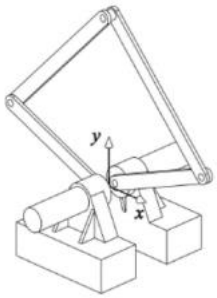


Figura 9 Robot planar.

Figura 8 Robot planar 3RR.

Otro tipo de robot paralelo planar es el 3RR, véase la Fig.8., este está formado por 3 brazos con 3 articulaciones rotacionales, la articulación que esta fija en la base debe ser actuada. Este robot cuenta con 3GDL, dos traslacionales y una rotacional.

Robot Paralelo Espacial

Los robots paralelos espaciales, como su nombre indica, son aquellos cuyo efector final pude realizar movimientos dentro de un espacio tridimensional.

Dentro de los robots paralelos espaciales tenemos la plataforma Stewart (véase la Fig.10.), que cuenta con 6 grados de libertad. En principio fue creada con el fin de realizar simulaciones de vuelo, pero en la actualidad son diversas las aplicaciones que se le dan. En este proyecto, por ejemplo, se tratará su empleo como simulador de ondas sísmicas. Existen múltiples configuraciones para esta plataforma, las cuales se analizan en la sección 3.4(*Configuraciones de la plataforma Stewart.*). El espacio de trabajo que puede alcanzar depende del tipo de configuración, de los actuadores que utilice, y por supuesto, del sistema de control.

Figura 10 Plataforma Stewart.

EL robot DELTA es otro tipo de robot paralelo traslacional. Contiene articulaciones esféricas en los paralelogramos que lo conforman (es un mecanismo muy similar a la plataforma Stewart).

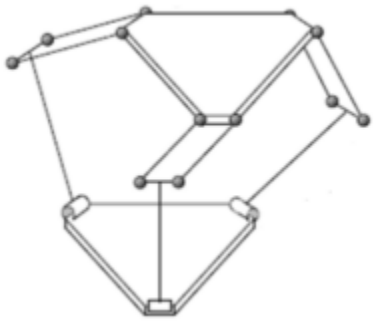


Figura 11 Manipulador DELTA de 3 grados de liberta traslacionales.

El Orthoglide es una variante del robot traslacional DELTA. Este ha optimizado el espacio de trabajo mediante una disposición diferente de los brazos y actuadores. En la figura inferior se pueden observar las semejanzas y diferencias con el robot Delta.

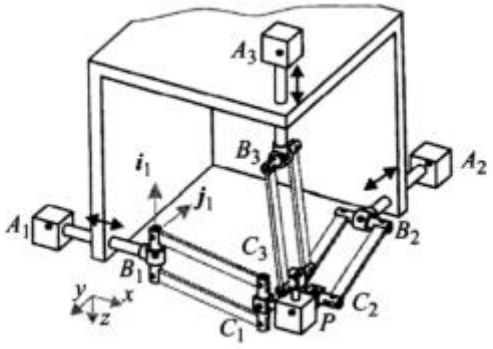


Figura 12 Robot paralelo Orthoglide.

Fuente:(León,2009)

En la Fig.13. Se muestra el prototipo H4 (este es un robot paralelo, su versión industrial es Quattro). Este tipo de robot posee cuatro motores en la base, lo cual le permite realizar 3 movimientos traslacionales y una rotación a la plataforma, que le proporciona una mayor eficiencia que el robot delta.

Figura 13 Robot Quattro

En la Fig.14. se muestra un robot paralelo construido a partir de dos mecanismos de 5 barras unidos a una plataforma por medio de dos articulaciones esféricas.

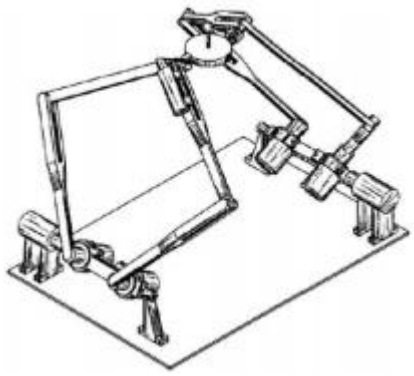


Figura 14 Robot paralelo 6 grados de libertad, basado en 5 barras.

Fuente:(León,2009)

3.3. Plataforma Stewart.

La plataforma Stewart (o Gough-Stewart), también conocida como hexápodo debido a su configuración de 6 actuadores (en nuestro caso servomotores) establecidos entre una base fija y una plataforma móvil, es un mecanismo definido como: posicionador paralelo de seis ejes. Esta configuración permite obtener seis grados de libertad para la plataforma móvil, la cual será utilizada, en este caso, para realizar un simulador de ondas sísmicas. La plataforma será la encargada de replicar el movimiento sísmico, movimiento que será generado en función de los datos aportados. La plataforma Stewart es un sistema cinemático paralelo que tiene importantes diferencias mecánicas sobre las etapas cinemáticas serie convencionales, aportando una serie de ventajas e inconvenientes como veremos en un próximo apartado (como su cadena cinemática cerrada y estructura paralela dan una gran rigidez y una alta relación fuerza-peso). También veremos en uno de los apartados de este capítulo los enfoques analíticos para el modelado y caracterización de la plataforma Stewart, cinemática inversa y directa, que permitirán establecer, predecir y diseñar la plataforma en función de la posición y orientación.

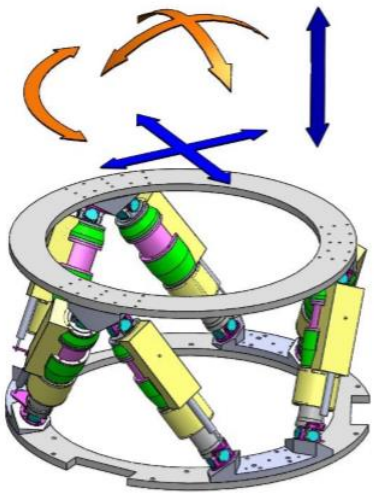


Figura 15 Modelo CAD de la plataforma Stewart.

En la Fig. superior podemos observar un modelo CAD de la plataforma Stewart con seis actuadores lineales, los cuales dotan a la plataforma de los 6 grados de libertad. Esto permite el movimiento en los tres ejes lineales X, Y, Z; así como la rotación en cada uno de los ejes, consiguiendo un desplazamiento angular. El desplazamiento angular en cada uno de los ejes es generalmente conocido como balanceo (rotación entrono al eje x), inclinación (rotación entrono al eje y) y guiñada (rotación entrono al eje z). El rango de movimiento, así como el rango de giro, estará restringido por factores estructurales y constructivos de la plataforma (estos deberán tenerse en cuenta en su implementación como simulador de conducción). En nuestro caso para dotar de 6 grados de libertad a la plataforma móvil se emplearán 6 servomotores, en vez de actuadores lineales, con los cuales se logrará un movimiento lineal a través de un sistema biela-manivela en cada uno de los servomotores.

3.3.1. Consideraciones teóricas.

En el apartado 2 (Plataformas robóticas), se ha aportado la definición de robot. En este apartado nos centraremos es una visión desde el punto de vista de la cinemática. Por lo que se definirá a continuación el concepto de mecanismo robótico, el cual es un sistema de cuerpos rígidos, denominados miembros o componentes, conectados entre sí por articulaciones. En función de la configuración de los miembros tendremos una cadena cinemática en serie o en paralelo. La cadena cinemática en serie es aquella en la que sus miembros están conectados siguiendo un único camino (se corresponde con los robots serie). La cadena cinemática en paralelo es aquella en la que sus miembros están conectados siguiendo dos o más caminos (esta se corresponde con los robots paralelos).

Un robot interacciona con el medio que le rodea mediante el miembro que se denomina efector final, que es aquel que realizará el movimiento que se desea. En nuestro caso el efector final será la plataforma móvil, por lo tanto, es de vital importancia poder describir la posición y orientación en el espacio del efector final. Para la plataforma Stewart debido a que la plataforma móvil posee seis grados de libertad, esto se realiza fijando los seis valores del vector de posición (X, Y, Z, balanceo, inclinación, guiñada) respecto del punto que se determine como origen de

coordenadas (este conjunto de valores es lo que se conoce como pose o configuración de un cuerpo rígido).

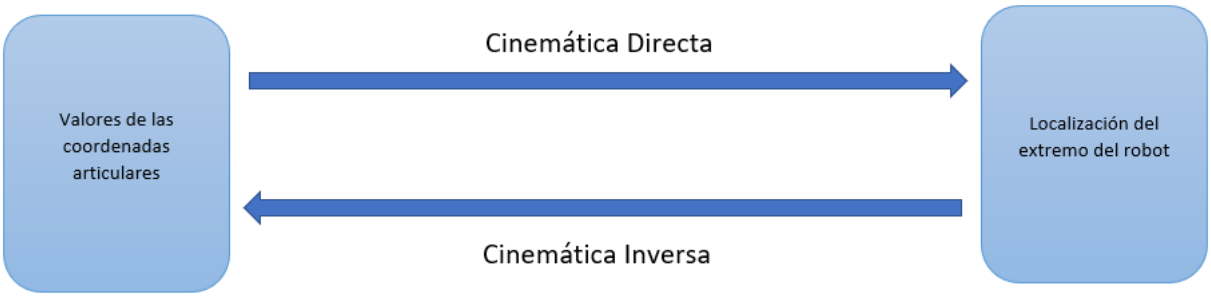
A lo largo de este trabajo, se tendrán en cuenta las siguientes características para el robot paralelo:

- Todos los miembros del robot se aceptarán como solidos rígidos, es decir, no sufrirán deformaciones ante acciones externas, estando en todo momento todos los puntos del miembro a la misma distancia.
- El número de cadenas cinemáticas que tiene el robot paralelo es seis, por lo tanto, el robot tiene 6 grados de libertad.
- Cada una de las cadenas cinemáticas que unen la plataforma fija con la móvil están formadas por dos elementos que forman un sistema biela-manivela, permitiendo el movimiento lineal entre los correspondientes puntos de la plataforma móvil y fija.
- Las articulaciones que unen los servomotores al sistema biela-manivela y este a la plataforma móvil son de tipo esférico. Este tipo de articulaciones permite el giro de un elemento respecto de otro libremente, pero impide el movimiento relativo entre ambos.

3.3.2. Cinemática de la plataforma Stewart.

La cinemática es la rama de la física que describe el movimiento de objetos en el espacio sin considerar las fuerzas que causan dicho movimiento. Por lo tanto, la cinemática aplicada a un robot busca estudiar la posición, velocidad, aceleración y todas aquellas propiedades que evolucionen con el tiempo; y que tengan que ver con la geometría del robot. En particular, el problema principal de un análisis cinemático consiste en establecer la relación entre la posición del efector final y las variables de las articulaciones.

De forma general, existen dos formas de enfrentar los análisis cinemáticos de un robot: la cinemática directa y la cinemática inversa. En esquema de abajo podemos ver claramente cuales son los valores iniciales y cuál es el valor que se obtiene, tanto mediante la cinemática directa como indirecta.



Esquema 1 Cinemática directa e inversa.

Antes de realizar un análisis cinemático de la plataforma es necesario describirlo. Como se ha mencionado anteriormente el hexápodo cuenta con una plataforma móvil sustentada por seis brazos. La plataforma móvil se encuentra unida a los brazos mediante juntas esféricas. Los brazos se encuentran formando un sistema biela-manivela unido por una junta esférica. Los brazos se unen a los servomotores mediante una junta rotacional. Los servomotores se encuentran fijos a la base. Como se observa en la Fig.16., abajo.

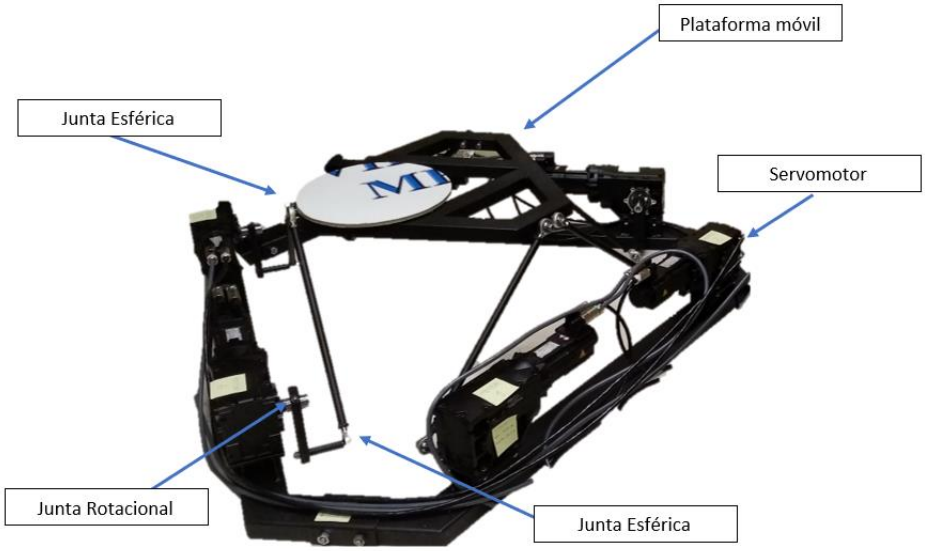


Figura 16 Componentes Plataforma Stewart.

La estructura cinemática de la plataforma puede ser descrita mediante un gráfico de "joints and loops", a modo de resumen, en donde cada caja representa una junta, (R) rotacional y (S) esférica. Las cajas grises representan las juntas donde se localizan los actuadores, por lo que el grafico es el siguiente:

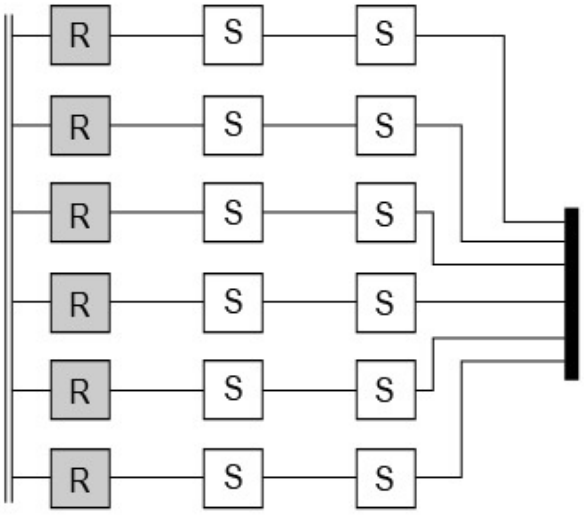


Figura 17 Gráfico de loops and joints de la plataforma Stewart.

3.3.2.1. Cinemática inversa.

De forma general, la cinemática inversa es la técnica que permite determinar el movimiento de las articulaciones del robot (o mecanismo) para lograr que el efector final se situé en una posición concreta, previamente conocida. Esto se realiza a partir de la localización inicial en el espacio de todos los miembros del robot y conociendo la geometría de estos.

Para la plataforma Stewart la resolución de la cinemática inversa consiste en determinar el valor de los ángulos que deben rotar cada uno de los servomotores, dada la posición y orientación de efector final. Esto es así para la geometría de nuestra plataforma, en el caso de emplear actuadores lineales deberíamos determinar las longitudes entre los puntos correspondientes de la plataforma fija y la plataforma móvil, para conseguir el movimiento deseado. La resolución de este problema es de gran importancia para el área de control de posición, ya que el efector final debe seguir una determinada trayectoria. Para lograr esto, es necesario conocer el valor de los ángulos que deben tomar los actuadores que producen el movimiento deseado.

La resolución de la cinemática inversa es un problema complejo para un robot serie, no obstante, en el caso de un robot paralelo este es más sencillo. Muchos de los métodos de resolución de la cinemática inversa se basan en emplear información sobre cómo está construido el robot para obtener las coordenadas de los puntos terminales de las patas respecto al sistema de referencia fijo, debido a que es posible determinar la matriz de transformación del sistema móvil al fijo a partir de la posición y orientación del efector final. Teniendo en cuenta esto último, se pude calcular de manera sencilla la longitud entre los puntos correspondientes de la plataforma móvil y la plataforma fija.

En nuestro caso haremos uso de las relaciones geométricas y trigonométricas de la plataforma. Para esto tomaremos como estado inicial la situación en la que los seis motores se encuentran con la biela en posición horizontal, como se ve en la siguiente figura.

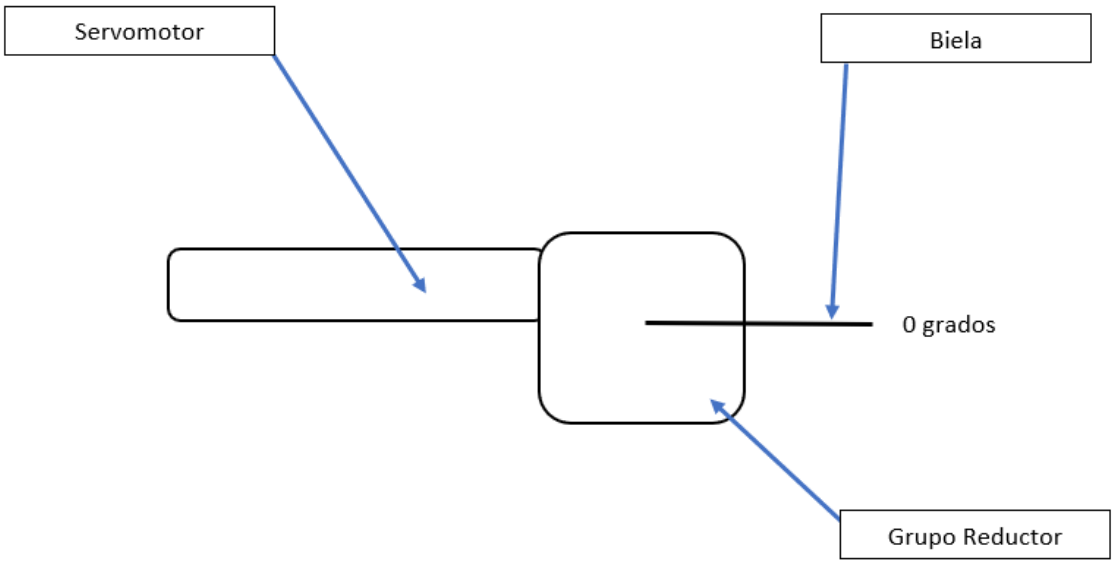


Figura 18 Posición de inicio.

Establecemos nuestro origen de coordenadas en el centro de la plataforma fija, y un segundo origen de coordenadas en el centro de la plataforma móvil. A continuación, establecemos la relación entre los puntos de unión de la plataforma con el centro de coordenadas de esta. Para los desplazamientos en los ejes supondrá un desplazamiento de los puntos de unión igual al del centro de coordenadas.

$$P_{31} = P_4 + \Delta$$

Donde:

- P_{31}, punto de unión del brazo nº1 con la plataforma móvil.
- P_4, punto donde se sitúa el origen de coordenadas de la plataforma móvil respecto de la plataforma fija.
- Δ, variación longitudinal en cualquiera de los 3 ejes

Para las rotaciones sobre los ejes serán diferentes para cada eje. Empezaremos por determinar la relación entre P_4 y P_{31} para una rotación entorno al eje Z.

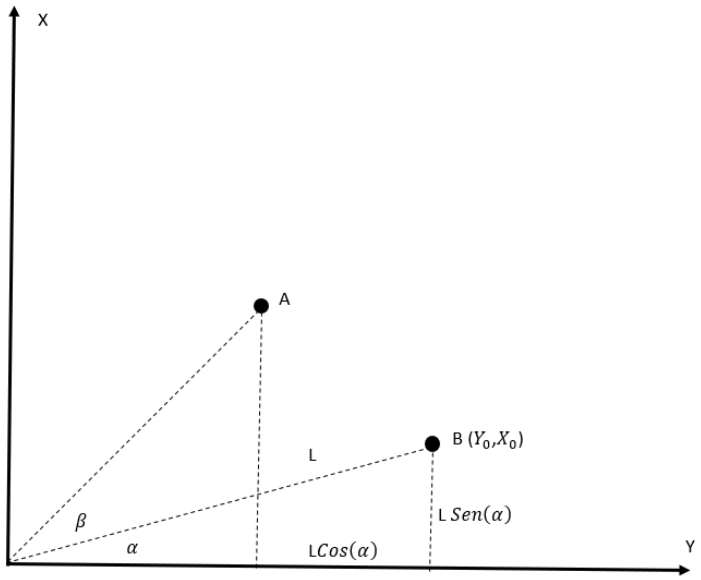


Figura 19 Rotación entorno a Z del pto. B al A.

$$L = \sqrt{{Y_O}^2 + {X_0}^2}$$

$$\theta = \beta + \alpha = \tan^{-1}\left(\frac{X_0}{Y_O}\right)$$

Donde β es el valor de la rotación entorno al eje Z y α es el valor inicial del ángulo. Por lo tanto, la nueva posición de P_{31} será:

$$P_{31}: (Lcos(\theta), Lsen(\theta), z_0) = (\sqrt{{Y_O}^2 + {X_0}^2}cos(\beta + \alpha), \sqrt{{Y_O}^2 + {X_0}^2}sen(\beta + \alpha), z_0)$$

De igual manera para las rotaciones entorno a los ejes X e Y, tenemos que la variación de posición de P_{31}, es la siguiente:

$$P_{31}: (x_0, Lcos(\theta), Lsen(\theta)) = (x_0, \sqrt{{Y_O}^2 + {Z_0}^2}cos(\beta + \alpha), \sqrt{{Y_O}^2 + {Z_0}^2}sen(\beta + \alpha))$$

$$P_{31}: (Lsen(\theta), y_0, Lcos(\theta)) = (\sqrt{{Z_O}^2 + {X_0}^2}sen(\beta + \alpha), y_0, \sqrt{{Y_O}^2 + {Z_0}^2}cos(\beta + \alpha))$$

Estas ecuaciones para determinar la posición de P_{31}, en función del giro entorno a los ejes, también es aplicable al resto de puntos de unión entre la plataforma móvil ($P_{32}, P_{33}, P_{34}, P_{35}, P_{36}$) y los brazos, teniendo diferentes x_0, y_0, z_0, α.

De esta forma conoceremos la distancia entre el punto del eje sobre el que rota la biela y el punto de unión del brazo con la plataforma móvil, para cualquier rotación y/o translación. Ahora determinaremos como influye el ángulo que estable el servomotor, en la distancia entre el punto de unión con la plataforma móvil y el punto de la biela que se sitúa en el eje. Observamos la figura inferior, en la que establecemos un nuevo origen de coordenadas en P_{11}, tenemos un triángulo formado por P_{31}, P_{21}(punto de unión entre la biela y el brazo) y P_{11}(punto de unión de la biela con el eje).

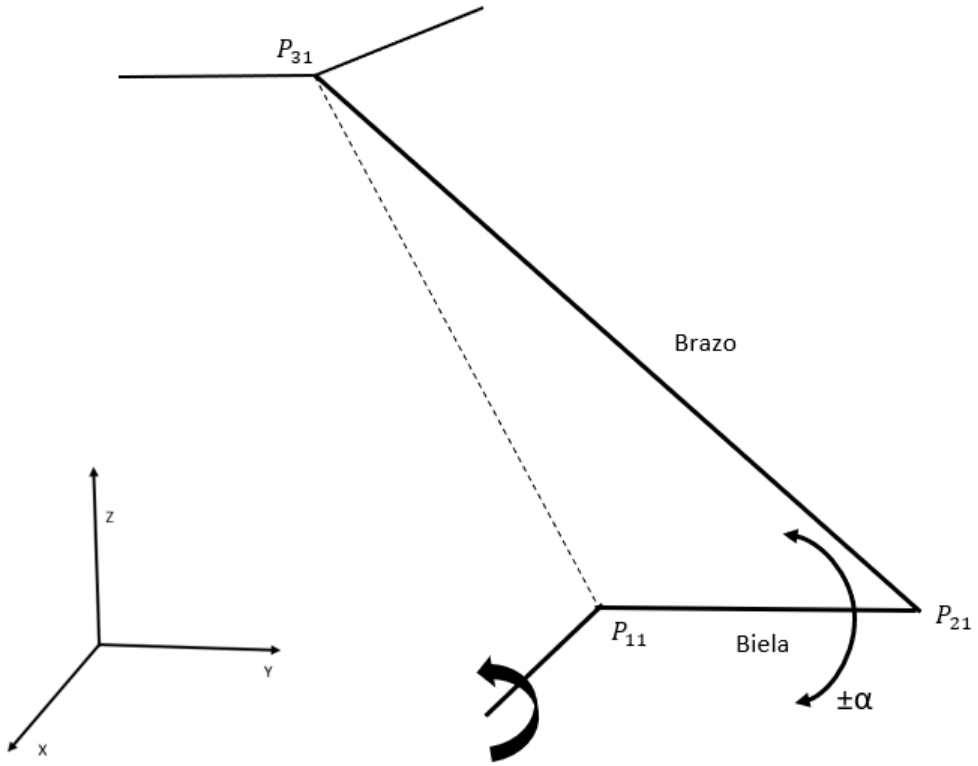


Figura 20 Esquema unión entre plataforma móvil y plataforma fija.

En esta figura tenemos los ángulos α y φ. De forma que α representa el ángulo que realiza el servomotor, tomando como positivo el sentido contrario a las agujas del reloj; y ϕ el ángulo que forma la línea imaginaria que une los puntos y P_{11}, proyectada sobre el plano perpendicular al eje en el punto de unión con la biela, con la vertical. Tenemos, por lo tanto, el siguiente triángulo, sobre este último plano, del que podemos despejar el valor de la distancia entre P_{31} y P_{11}; las dimensiones de los otros dos lados del triángulo, se correspondes con las longitudes de la biela y el brazo, y dependen de la construcción de la plataforma siendo estos valores fijos que podemos medir.

$$|\overline{P_{31}P_{11}}| = D1$$

$D2$ =Longitud Biela

$D3$ =Longitud Brazo

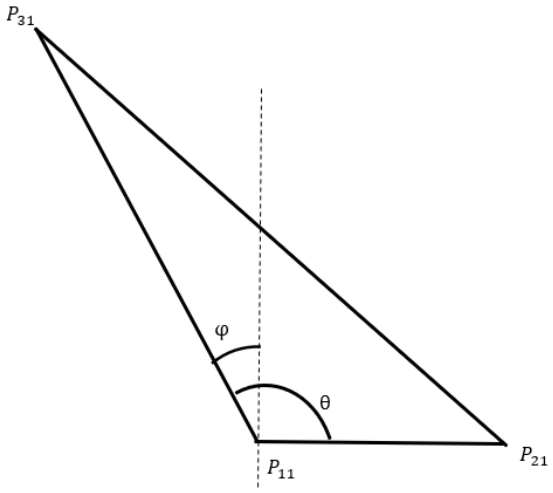


Figura 21 Triángulo formado por Biela-Brazo.

De forma que el ángulo θ, lo podemos despejar con el teorema del coseno. Y φ mediante la relación de la tangente.

$$\theta = \cos^{-1}\left(\frac{D2^2 - D1^2 - D3^2}{2 \cdot D2 \cdot D3}\right)$$

$$\varphi = \tan^{-1}\left(\frac{P_{31Y}}{P_{31Z}}\right)$$

Tenemos finalmente que el ángulo de giro del servomotor será:

$$\varepsilon = -(\theta - \ \varphi - 90)$$

Con esto tenemos el ángulo que debe rotar cada uno de nuestros servomotores para conseguir que la plataforma móvil se posiciones de la forma que deseemos. A la hora de simplemente rotar la plataforma entorno a los ejes no tendremos problemas. No obstante, al tratar de realizar una translación deberemos tener en cuenta que todas las posiciones que atraviese la plataforma estén dentro del espacio de trabajo y que no forme parte de las posiciones singulares, como veremos en el apartado del espacio de trabajo.

3.3.2.2. Cinemática directa.

La cinemática directa es la técnica que permite determinar la localización del efector final, respecto de un sistema de coordenadas de referencia, conocidos los valores de las articulaciones, así como la geometría de todos los miembros del robot. Este planteamiento cinemático es complejo a la hora de ser aplicado a los robots paralelos, no habiendo una solución en forma cerrada. La formulación empleada depende de la configuración del robot paralelo. No obstante, este planteamiento resulta muy útil para robots serie.

En nuestro caso para una plataforma Stewart, el problema cinemático directo consiste en determinar la posición de la plataforma móvil, posición y orientación, conocidos los ángulos de giro de los seis servomotores. Para enfrentar este problema existen diferentes métodos. Un método consiste en generar un polinomio cuya variable es el ángulo que forma la base fija con uno de los planos definido por dos patas que comparten una misma articulación. Este método no ofrece una solución exacta, Dietmaier demostró que se obtienen 40 soluciones [2] y Thomas y col. obtuvieron un polinomio de orden 8 para una geometría 3-3 de la plataforma [3], por lo que es necesario un procedimiento extra para determinar la solución adecuada. Otro método

que ofrece una sola solución consiste en emplear un sistema de ecuaciones no lineales. Mediante el método de Newton-Raphson, no obstante, es necesaria una aproximación inicial. (este método fue propuesto por los siguientes autores Ku [4], Liu y col. [5]). Estos métodos se plantean para el caso de que la plataforma Stewart que posea actuadores lineales, no actuadores rotacionales como es nuestro caso. Otra solución para el problema de la cinemática directa consiste en el uso de sensores, para solucionar o simplificar el problema.

Como hemos visto la solución de la cinemática directa para robots paralelos es un problema de gran complejidad. Para nuestro simulador principalmente necesitamos tener resuelta la cinemática indirecta, pues la orientación y/o localización de la plataforma será definida por el simulador de conducción y deberemos saber cuál debe ser el ángulo que deben proporcionar los servomotores para obtener esa respuesta. Teniendo esto en cuenta en principio no sería necesario el uso de la cinemática directa, no obstante, pueden existir puntos conflictivos en la trayectoria entre dos puntos, ya sea porque no están definidos en el espacio de trabajo o por ser una configuración singular. Por lo tanto, deberíamos verificar que todos los puntos de la trayectoria se encuentran en el espacio de trabajo de nuestra plataforma. Para esto podemos simplificar en gran medida el problema. Pues que conoceremos la posición y orientación aproximados de la plataforma a lo largo de la trayectoria. Teniendo esto presente podemos realizar el siguiente procedimiento: partiendo de encontrar con una simple relación trigonométrica la posición de los puntos finales de la biela (P_{2i}); conociendo la longitud del brazo que une este último punto con la plataforma móvil y sabiendo que ambas uniones (biela-brazo y brazo-plataforma móvil) se realizan mediante una unión esférica, podemos determinar una superficie esférica para cada brazo, con centro en el pto. de unión biela-brazo y de radio la longitud del brazo.

$$(x - P_{2ix})^2 + (y - P_{2iy})^2 + (z - P_{2ix})^2 = D3^2$$

Cada punto de conexión entre el brazo y la plataforma móvil debería situarse sobre la superficie esférica correspondiente. Teniendo en cuenta que conocemos la localización exacta de las superficies esféricas y la localización aproximada de la plataforma móvil, en un instante en el transcurso de la trayectoria, podemos determinar cada una de las distancias entre los puntos de unión y las superficies de la siguiente forma:

$$\varepsilon_i = |\overline{P_{2i}P_{3i}}| - D3$$

Obtenernos 6 valores, ε_i, tomaremos como primer punto de unión aquel que sea de menor valor o 0, siempre por debajo de un ε_{max}, este punto lo uniremos a su superficie esférica en el próximo paso, desplazando toda la plataforma móvil el valor ε_i o 0, correspondiente, en la dirección del brazo. Realizaremos este procedimiento de forma consecutiva con cada uno de los puntos. Si una vez hemos fijado los 6 puntos en ningún momento hemos superado ε_{max}, daremos la posición por buena.

No obstante, las entradas para posicionar la plataforma vienen dadas por los datos sísmicos que representan la onda de un terremoto. Por lo tanto, si el periodo de muestreo de los datos es suficientemente elevado podemos considerar que la señal de entrada de los datos es continua. Haciendo difícil una situación en la que debamos comprobar que la trayectoria se sitúa en el espacio de trabajo de la forma que hemos visto.

3.3.2.3. Criterio de Kutzbach-Grübler.

El criterio de Grübler, o de Kutzbach-Grübler, es una expresión para obtener el grado de movilidad de un mecanismo, es decir, obtener el número de grados de libertad del mecanismo. El criterio consiste simplemente en realizar una diferencia entre los grados de libertad de los eslabones del mecanismo y las restricciones impuestas por los pares cinemáticos. Para mecanismos que se sitúan en el plano la expresión del criterio de Grübler es la siguiente.

$$G = 3(N-1) - 2P_1 - P_2$$

Siendo:

- N: nº de elementos.
- P1: nº de pares clase 1.
- P2: nº de pares clase 2.
- P3: nº de pares clase 3.
- P4: nº de pares clase 4.
- P5: nº de pares clase 5.

En función del número de grados de libertad obtenido podemos clasificar el conjunto de elementos como:

- G<0 Estructura hiperestática.
- G=0 Estructura isostática.
- G=1 Mecanismo desmodrómico: Dada la posición de un elemento se conoce al resto.
- G=2 Mecanismo diferencial.
- G>2 Mecanismo de 'n' grados de libertad.

Para mecanismos espaciales, como es el caso de la plataforma Stewart, también existe una expresión según la cual el criterio de Grübler determina el número de grados de libertad de un mecanismo. No obstante, para el caso de mecanismos espaciales en algunas ocasiones puede ofrecer un resultado erróneo. La expresión es la siguiente.

$$m = \lambda(n - j - 1) + \sum_{i=1}^{j} (fi - If)$$

Donde:

- m: número de grados de libertad del sistema
- λ: grados de libertad del espacio de trabajo del mecanismo.
- n: número de eslabones fijos del mecanismo incluyendo la base fija base móvil (articulaciones).
- j: número de juntas del mecanismo.
- fi: grados de movimiento relativos por junta.
- If: grados de libertad pasivo del sistema.

Entendiéndose por lado libertad pasivo cuando un eslabón binario se encuentra conectado en el mecanismo mediante una combinación de articulaciones, que no sea posible la transmisión de fuerza o torque y por consecuencia en movimiento sobre el eje.

Tomando valores a partir de la plataforma Stewart se tiene:

- λ=6, espacio de trabajo tridimensional.
- n=14, sumando el número de elementos de cada articulación.
- j=18, sumando las juntas esféricas(rotulas) que conectan la manivela y la junta rotacional que transmite el movimiento del servo al sistema.
- fi=3, 3 grados de libertad para las juntas esféricas(rotulas),1 grada de libertad para las juntas rotacionales.
- If=6 grados de libertad pasivos.

Sustituyendo:

$$m = 6(14 - 18 - 1) + \sum_{i=1}^{12} 3 + \sum_{i=1}^{6} 1 - 6 = 6$$

Vemos como el resultado coincide con el resultado esperado que es 6. Por lo que queda demostrado que la plataforma Stewart posee 3 translaciones y 3 rotaciones sobre los ejes de coordenadas.

3.3.2.4. Espacio de trabajo.

El espacio de trabajo o campo de acción es el volumen espacial que puede llegar el efector final, es decir, la plataforma móvil del mecanismo paralelo. Este volumen está determinado por el tamaño, forma y tipo de eslabones que conforman el robot, así como las limitaciones de movimiento impuestas por el sistema de control. Entonces el robot debe elegirse de modo que su espacio el trabajo le permita llegar a todos los puntos necesarios para llevar a cabo su tarea.

Es importante tener en consideración que, aunque el robot pueda acceder a todo el espacio de trabajo no significa que lo pueda hacer con cualquier orientación. Existirá un conjunto de puntos los más alejados y los más cercanos que únicamente se podrán hacer con las orientaciones determinadas mientras que otros puntos admitirán cualquier orientación.

Además de estos límites en la orientación del actuador final en determinadas zonas del espacio de trabajo, existen configuraciones singulares. Estas configuraciones singulares son las posiciones en el espacio a las cuales el robot no pude llegar. La existencia de estas configuraciones singulares genera una reducción del espacio de trabajo del robot. La posición de estas configuraciones depende de las características constructivas del robot, determinar donde se sitúan mediante un análisis hace posible que se lleguen a alcanzar, evitando la perdida de grados de libertad que ello produce. Las diferentes configuraciones singulares del robot pueden ser clasificadas como:

- Singularidades en los límites del espacio de trabajo del robot. Se presenta cuando el extremo del robot está en algún punto del límite de trabajo interior o exterior. En esta situación resulta obvio que el robot no podrá desplazarse en las direcciones que lo alejan de este espacio de trabajo.
- Singularidades en el interior del espacio de trabajo del robot. Ocurre dentro de la zona de trabajo y se producen generalmente por el alineamiento de dos o más ejes de las articulaciones del robot.

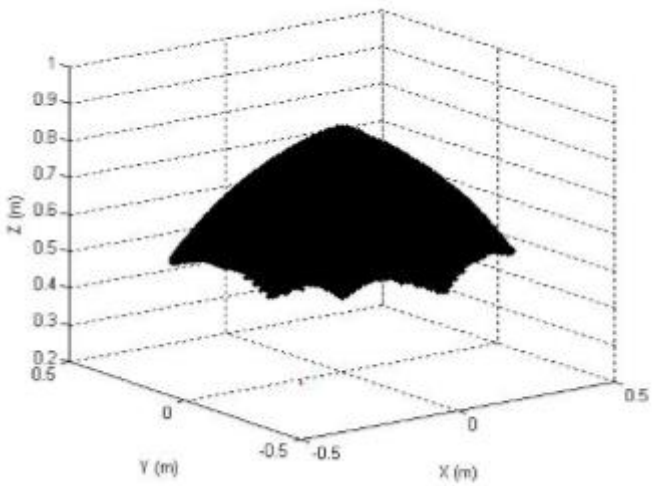


Figura 22 Espacio de trabajo de la plataforma Stewart.

3.3.3. Dinámica de la plataforma Stewart.

El análisis dinámico de los robots paralelos (incluida la plataforma Stewart) es complicado dada la existencia de múltiples cadenas cinemáticas cerradas. Diferentes formulaciones han sido propuestas, para alcanzar una solución adecuada. Entre las diferentes propuestas están la formulación de Newton-Euler, la cual es un algoritmo recursivo que permite obtener un conjunto de ecuaciones recursivas hacia delante de velocidad y aceleración lineal y angular. Otra propuesta la formulación lagrangiana, que a diferencia de la de Newton-Euler que trabaja con magnitudes vectoriales, trabaja con energías cinéticas y potenciales que son magnitudes escalares. Por último, otra formulación propuesta es el principio de trabajos virtuales. La aplicación de estas formulaciones siempre va acompañada de despreciar determinados fenómenos físicos para la simplificación del sistema.

3.3.4. Configuraciones de la plataforma Stewart.

Inicialmente la mayoría de los diseños de la plataforma Stewart se centraron en dos configuraciones. Una con la base y el efector final con forma triangular y los actuadores de dos en dos coincidiendo con los vértices de cada triángulo, denominada como 3-3 plataforma de Stewart. Y otra, con la plataforma móvil de forma triangular y los actuadores coincidiendo de dos en dos en cada uno de los vértices del triángulo, y con una base de forma hexagonal donde se acoplan los actuadores en cada uno de los vértices, denominada 6-3 plataforma de Stewart. No obstante, con estas configuraciones se observó que la coincidencia de las articulaciones esféricas restringía de forma severa la movilidad del efector final. Ante este problema, comenzó a popularizarse otra configuración. Esta configuración posee hexágonos semirregulares, tanto en la plataforma móvil como la base fija, que se conoce como 6-6 plataforma de Stewart.

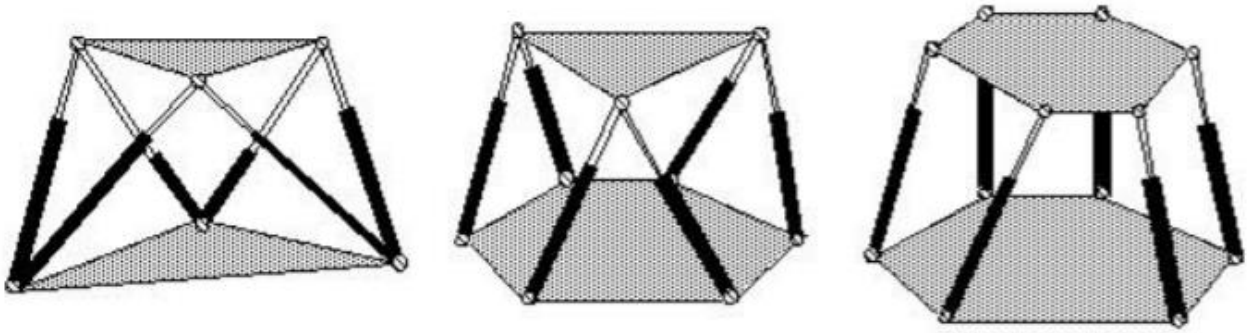


Figura 23 Plataforma Stewart 3-3, 6-3, 6-6.

Fuente:(Silva,2005)

Según sea la estructura de cadenas cinemáticas que unen la base fija con el efector final también podemos tener diferentes configuraciones. Algunas de estas son: la estructura cinemática con articulaciones esféricas a ambos lados de cada actuador, como en la Fig.24., denominada plataforma Stewart 6-sps, donde 6 se refiere al número de grados de libertad del robot y sps es el acrónimo de spherical prismatic spherical. Otra configuración es la que se denomina plataforma de Stewart 6-ups, donde ups es el acrónimo de universal prismatic spherical. Estas dos estructuras son idénticas salvo que la plataforma 6-sps presenta 6 grados de libertad pasivos que permite la rotación de cada cadena cinemática sobre su eje.

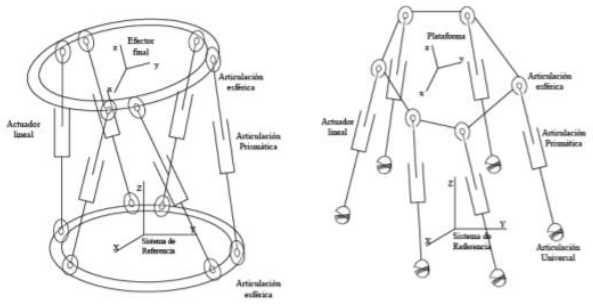


Figura 24 Plataforma Stewart 6-sps(der.) y 6-ups(izq.). Fuente:(Silva,2005)

Para nuestro caso práctico tendremos una configuración de la plataforma Stewart 6-rss, siendo todas las articulaciones esféricas como en la configuración 6-sps. Pero a diferencia de la configuración 6-sps en la nuestra en lugar de actuadores lineales tenemos servomotores, con los cuales se consigue un movimiento rectilíneo mediante un sistema biela-manivela.

3.3.5. Ventajas e inconvenientes.

En este capítulo ya hemos visto la definición de plataforma Stewart, así como la cinemática de esta, en la que hemos visto algunas de las ventajas que esta ofrece. En este apartado nos centraremos en las ventajas e inconvenientes que ofrecen los robots paralelos (plataforma Stewart), frente a los robots en serie.

Generalmente cuando es necesario diseñar un robot para una aplicación de movimiento múltieje, muchos usuarios apilan etapas de movimiento serie, de hecho, este es un buen enfoque para ensambles de solo unos pocos ejes. No obstante, a medida que las aplicaciones se vuelven más complejas, también lo hacen las pilas de etapas equivalentes, y comienza a resultar complejo el control preciso del efector final. Es la imagen inferior podemos apreciar dos sistemas robóticos que proporcionan 6 grados de libertad. Se puede observar a simple vista como el control y estabilidad del robot paralelo (izq.) será mayor que el del robot serie (der.). Puesto que el control se realiza a través de seis etapas consecutivas, sin embargo, en el paralelo los seis actuadores tienen control sobre el actuador final.

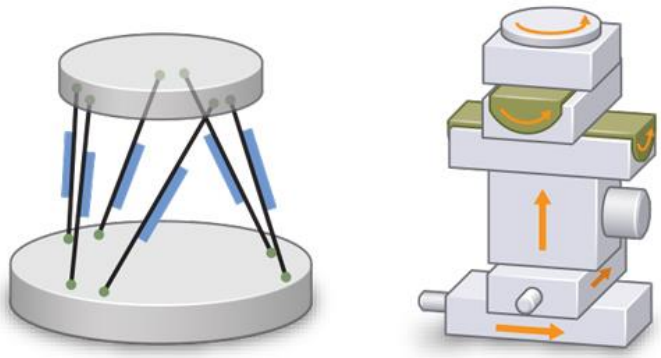
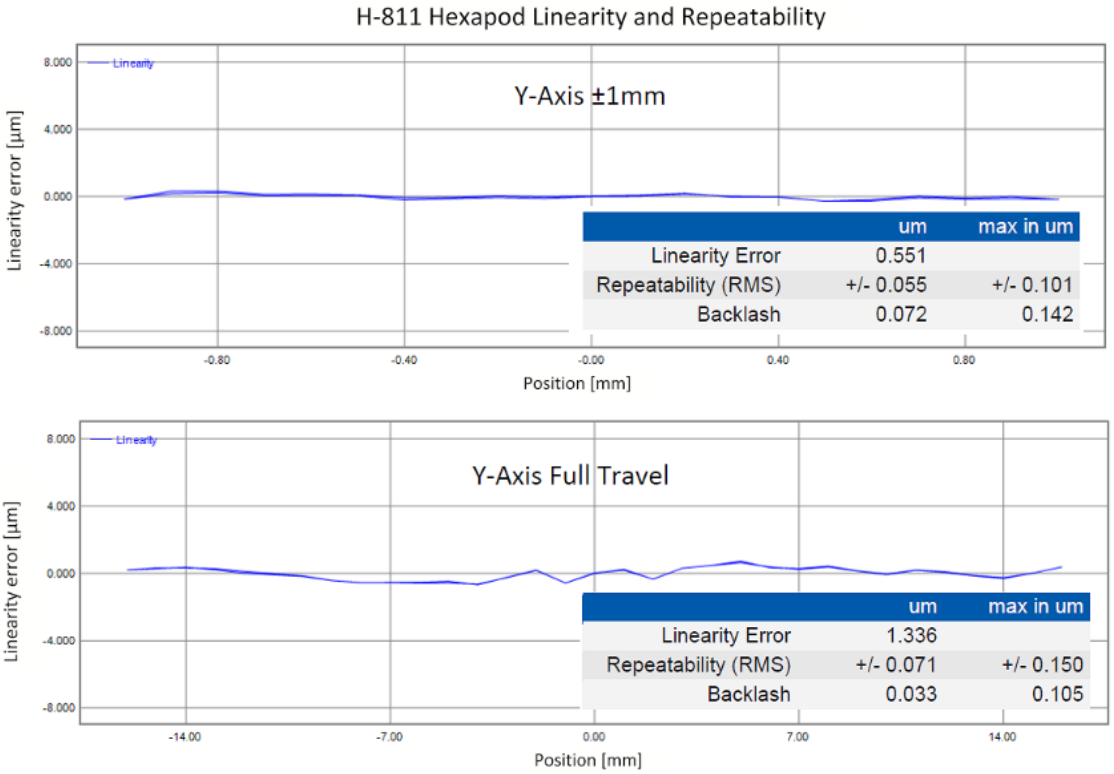


Figura 25 Imagen comparativa entre dos sistemas 6DOF. Sistema paralelo a la izquierda y sistema serie derecha. (Imagen: PI)

A continuación, procederemos a enumerar cuales son las ventajas que ofrecen los robots paralelos.

- Mayor rigidez. Esta mayor rigidez es debido a su estructura más robusta. También debe tenerse en cuenta que, a la hora de situar una carga en el robot, el robot paralelo sufrirá un momento de fuerza menor ante la misma carga que uno serie. En el siguiente grafico podemos observar las desviaciones en el eje Y de una plataforma paralela (modelo: H-811 by PI [6]) para recorridos de 1mm y el máximo admisible.



- Mayor precisión. A causa de su mayor seguridad, presentan una mayor precisión. Haciendo a los robos paralelos óptimos para trabajos de elevada precisión.

- Su arquitectura les permite alcanzar altas velocidades y aceleraciones, lo cual les permite realizar tareas industriales de manera más eficiente.

- Elevada relación carga/potencia. Esto es debido a que los accionadores conectan directamente la base fija del robot al efector final (plataforma móvil). Por lo tanto, los mismo efectores sirven de eventos estructurales de manera simultánea, permitiendo manipular cargas superiores a su propio peso.

- Dinámica más consistente. La etapa inferior de robot serie lleva la masa de todo el conjunto, y así sucesivamente hasta la etapa superior, que lleva la carga de la aplicación. Por lo tanto, el ajuste es un proceso laborioso, eje por eje, con diferentes configuraciones para cada eje, y, en consecuencia, diferente capacidad de respuesta. En contraposición, en un robot paralelo la carga se encuentra repartida entre todos los actuadores que controlan el actuador final, permitiendo lidiar a este tipo de robots con cargas más pesadas con una precisión mayor. Esto también se traduce en menores inercias para los robots paralelos.

- Cableado. Los cables pueden ser un conducto para la vibración que puede afectar la configuración de una aplicación. Incluso la elección de colocar los cables fuera de una plataforma de aislamiento puede influir en la estabilidad y el rendimiento general de una aplicación de manera profunda. A medida que se mueve una etapa, cualquier cable que se arrastre puede contribuir a movimientos parásitos y otros errores, para los robots serie. Los cables rígidos pueden hacerlo incluso si están dispuestos de manera no arrastrante. Los cables pueden romperse y engancharse y aflojarse, lo que contribuye a fallos prematuros que pueden ser difíciles de diagnosticar. Y, en general, estos problemas aumentan con el número de ejes en un sistema serie. Un sistema paralelo no posee estos problemas, ya que todos los actuadores se encuentran unidos a la base fija, siendo fácilmente accesibles y no contando con cables de arrastre.

- Tamaño, peso y fragilidad. Los robots serie son superiores en altura y masa, para una misma aplicación, frente a los robots paralelos. También es muy importante, en los robots en serie, tener en cuenta que las etapas inferiores soportan toda la carga. Por lo tanto, sus rodamientos son vulnerables a la inmovilización y otros daños causados por fuerzas excesivas. Además de provocar daños por golpes en el codo cuando se configura, esto a menudo requiere un desmontaje para el envío, lo que agrega costos y molestias e introduce variabilidad cuando se vuelve a ensamblar.

- Ortogonalidad y errores parásitos. En los robots serie, los ejes apilados interactúan de formas complicadas; por ejemplo, la desviación en el eje X se ve como un movimiento no deseado en los ejes Y Z; la desviación angular de un eje también imparte movimiento en las direcciones de desplazamiento de los otros ejes, con una magnitud proporcional a la distancia al eje en movimiento. Estos efectos se ven aumentados cuanto mayor sea el tamaño del robot serie.

- Fuerza máxima. La fuerza máxima que es capaz de ejercer el robot paralelo será la suma de todos los actuadores en conjunto.

- El punto de pivote, programable por el usuario, permite rotaciones alrededor de cualquier centro en el espacio y múltiples sistemas de coordenadas (la pieza de trabajo y la herramienta se pueden configurar en el software).

A continuación, procederemos a enumerar cuales son las desventajas que ofrecen los robots paralelos.

- Cinemática compleja. La definición de la posición en el espacio del efector final y la construcción del modelo dinámico son más complicados que para el caso serie. La mayoría de las veces esto se resuelve de manera particular para cada configuración de robot.

- Espacio de trabajo reducido. En comparación a los robots serie este suele ser inferior. Además, su cálculo no resulta sencillo, ya que la posición y orientación están fuertemente acopladas.

- Falta de generalidad. Al contrario que ocurre en los robots serie, en los que existe un modelo dinámico general para los mismos, no existe un modelo general dinámico. Esto dificulta el desarrollo de algoritmos de control de carácter general y hace que los robos existentes en la actualidad se controlen de forma semiacoplada.

- Elevado precio. Puesto que su uso esta poco generalizado, centrándose para usos concretos, su precio es elevado.

Hemos visto cuales son las principales ventajas y desventajas que aportan los robots paralelos, en el próximo apartado estudiaremos estas ventajas, para observar en las aplicaciones prácticas como resultan estas ventajas beneficiosas. A modo de resumen debemos tener presentes que las principales ventajas que aportan los robots paralelos son su tamaño más pequeño, mayor rigidez, dinámica, precisión y mayor flexibilidad.

3.3.6. Aplicaciones.

Las aplicaciones para las cuales el empleo de la plataforma Stewart (u otra configuración de robot paralelo) puede resultar ventajosa frente a una configuración serie pueden ser muy variadas, resaltando aquellas en las que se requiera de elevada precisión y/o 6 grados de libertad. Los principales inconvenientes para su utilización son su elevado costo (debemos tener en cuenta que para alcanzar 6 grados de libertad debemos emplear 6 actuadores), y su complejidad, en lo que a su estructura cinemática se refiere, a la hora de solucionar la cinemática directa, para posicionar la plataforma móvil. No obstante, pese a estos inconvenientes, en determinadas aplicaciones como las que se indicaran, pueden resultar ventajosos frente a robots en serie, no solo por la precisión que pueden proporcionar, sino que por su estructura más robusta son menos

propensos a averías frente a robots en serie. Haciendo que la elección de la plataforma Stewart sea la más optima en muchas aplicaciones, como veremos a continuación.

- Plataformas de movimiento 6-DOF para pruebas de calidad de imagen.

Esta es una aplicación con una gran proyección a futuro, puesto que el mercado cada vez requiere de una mayor cantidad de cámaras digitales en móviles, tabletas, ordenadores y en coches. Por ello los requerimientos de calidad de estas son cada vez más elevados. La compañía DxOMark Image Labs realiza pruebas de calidad a cámaras mediante el uso de plataformas 6DOF pudiendo asegurar que sus resultados son repetibles e independientes del operador, gracias a la precisión que se obtiene con las plataformas 6DOF.

Figura 26 Una plataforma Stewart 6DOF simulando el movimiento, como pude ser el temblor de la mano del fotógrafo.

(Imagen: DxOMark Image Labs)

- Robots Hexápodos en la industria automotriz.

Hasta hace relativamente poco tiempo los robots tradicionales (en serie) eran los mayormente empleados en la industria automotriz pues son capaces de realizar trabajos pesados y con rapidez, no obstante, carecen de precisión de posicionamiento. A medida que aumentan los requisitos de precisión en la industria automotriz, los robots hexápodos van ganando presencia en esta industria. Dentro de esta industria pueden desarrollar muchas funciones, desde aseguramiento de la calidad, pruebas y metrología, diseño y evaluación de prototipos, soldadura láser de precisión, hasta en la línea de producción como complemento de los robots industriales tradicionales o para el trabajador en procesos parcialmente automatizados (como se ve en la imagen inf.).

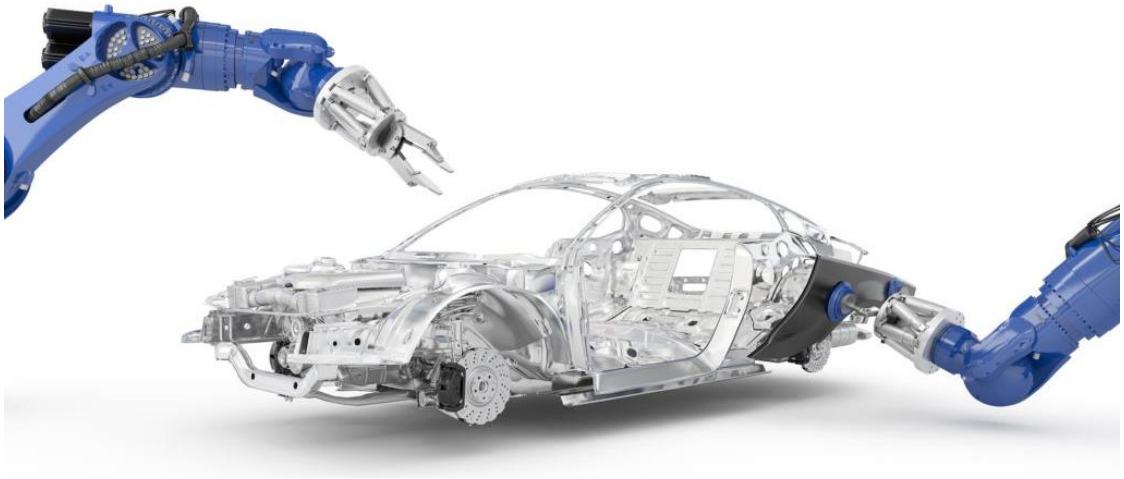


Figura 27 Ejemplo de robots paralelos en industria automotriz.

Hay que señalar que los hexápodos no son nuevos en la industria automotriz; por el contrario, una de las primeras aplicaciones industriales de los robots cinemáticos paralelos fue la prueba de neumáticos bajo diferentes ángulos, desplazamientos y cargas. La primera máquina fue diseñada por el Dr. Eric Gough, que trabajaba para un fabricante de neumáticos en Birmingham, Inglaterra; como se mencionó en el apartado 2.1.1(Contexto histórico de los robots paralelos).

- Estabilización giroscópica de un hexápodo.

Existen muchas aplicaciones relacionadas con la óptica y la metrología que requieren una superficie de trabajo estable en un entorno inestable, ya sea que ese entorno esté vibrando, en un avión, en un barco en el mar o en otro lugar. En condiciones normales, la superficie de trabajo está sujeta a estas perturbaciones ambientales. Las plataformas de movimiento hexápodo proporcionan una buena base para compensar el movimiento en 6 grados de libertad.

En la imagen inferior, podemos ver un par de hexápodos, en una prueba de la compañía PI-USA en la que utilizan un giroscopio para proporcionar retroalimentación a un controlador hexápodo a fin de compensar las perturbaciones de múltiples grados de libertad y mantener una placa superior horizontal durante el proceso.

Figura 28 Estabilización giroscópica

(Imagen: PI)

.

- Robots de alineación.

Los robots paralelos con esta tarea pueden emplearse en diversas aplicaciones. Por ejemplo, para autoposicionar células fotovoltaicas para obtener la mayor cantidad posible. Gracias a la plataforma Stewart, la célula fotovoltaica tiene la capacidad de moverse y rotar sobre sus 3 ejes, manteniendo una posición perpendicular y constante al sol en cualquier momento, lo que incrementa su eficiencia. Otra aplicación es la de mejorar la capacidad de visión que tenemos a través de los diferentes telescopios distribuidos a nivel mundial. Se logra trabajando de forma conjunta con varios telescopios a la vez para lograr imágenes de gran calidad. Para ello, se aúnan los datos en bruto de los diferentes telescopios y luego se procesa una sola imagen; para lograr esto se necesita de una precisión absoluta, es aquí donde entran en juego los robots hexápodos. Los hexápodos son empleados para posicionar con precisión el elemento receptor del telescopio o radio telescopio, consiguiendo así unos datos precisos.

Figura 29 Instalación de hexápodo como sistema de alineación en telescopio ALMA.

- Simuladores.

Los robots paralelos son idóneos para esta aplicación por diversos factores: poseer 6 grados de libertad permite simular con efectividad cualquier movimiento, su espacio de trabajo se adapta perfectamente a esta tarea pues es capad de realizar prácticamente cualquier trayectoria (dentro de su espacio de trabajo), su precisión para reproducir el movimiento a simular, así como su elevada capacidad de carga.

Figura 30 Simulador de la empresa española hi-speed.

- Robot “trepa”.

El robot trepa es un robot paralelo desarrollado por el grupo de robots y máquinas inteligentes de la universidad politécnica de Madrid. Este robot es capaz de trepar por estructuras, implementando un intercambio entre la plataforma fija y la móvil aprovechando la movilidad de la plataforma.

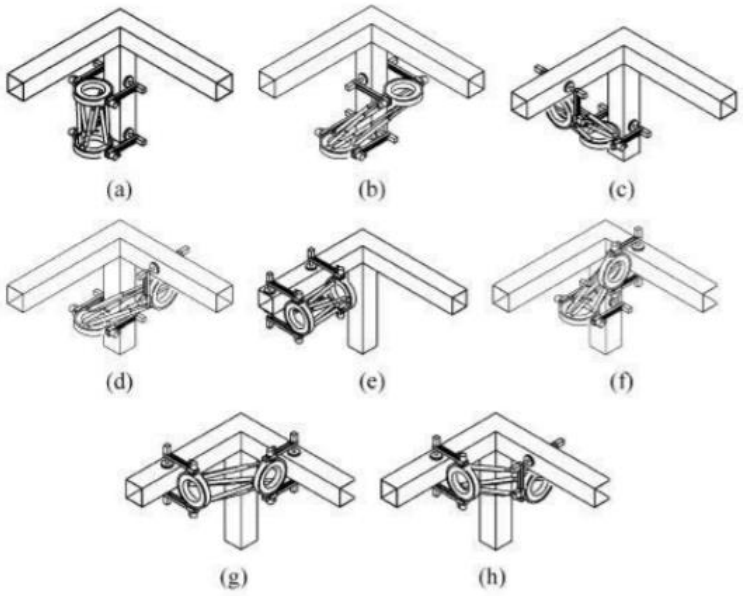


Figura 31Configuraciones que puede tomar el robot trepa para movilizarse.

Fuente:(Madrid, 2007)

3.4. CoDeSys. El control del autómata programable.

Ya hemos visto en profundidad el componente actuador del Simulador, él cual es la plataforma Stewart. Para generar el movimiento controlado de la plataforma, según las entradas que deseemos, entrara en juego el autómata programable.

El comportamiento de un sistema, para nuestro caso la plataforma, está caracterizado por magnitudes representativas, de las que nos interesa su evolución en el tiempo, como pueden ser temperaturas en un sistema térmico, presiones o caudales, y en nuestro caso la posición, velocidad y aceleración del punto final de la plataforma o efector final (end-efector, en inglés). Controlar un sistema implica actuar sobre él, es decir determinar en cada momento los valores

de un conjunto de variables sobre las que se puede actuar y que deben permitir modificar, y por tanto controlar, el comportamiento de el mismo. Este conjunto de variables son las actuaciones, señales de control o entradas del sistema; que en nuestro caso vendrán impuestas por el conjunto de datos que forman la onda sísmica.

En el siguiente esquema se muestra el flujo de trabajo (workflow, en inglés) de todo el simulador, en conjunto. El sistema de control del autómata programable, de acuerdo con los datos entregados por Simulink (la onda sísmica), determinará las acciones de, en nuestro caso, el hexápodo o el sistema de actuadores en un caso general. Esta sería una configuración de control en lazo abierto para nuestro sistema.

Esquema 2 Representación en lazo abierto del simulador.

3.4.1. Señales

Las señales con las cuales trabajan los autómatas programables, en relación con los valores que pueden tomar, son principalmente de dos tipos:

Señales analógicas o continuas: En un rango determinado de valores de la señal se tienen infinitos valores intermedios. Con estas señales se corresponden por ejemplo las magnitudes físicas de un proceso como pueden ser la prisión o la temperatura.

Señales discretas: Estas señales solo pueden tomar un número finito de valores. El caso más típico son las señales todo-nada, denominas señales binarias o booleanas, que solo pueden tener dos valores.

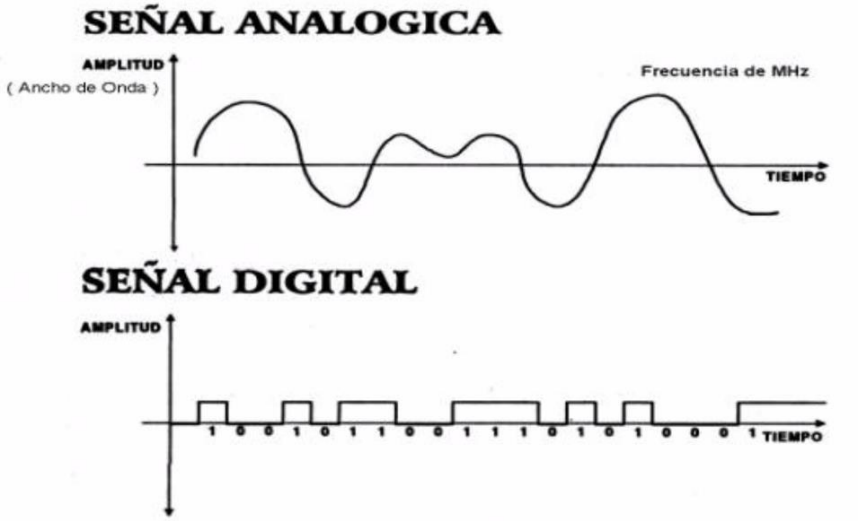


Figura 32 Señales analógica y digital.

Las técnicas de diseño y de implementación para los autómatas programables van a depender del tipo de señales con las cuales trabaje el sistema. En nuestro sistema, el simulador, las señales principales son digitales. Todo el flujo de datos desde Simulink hacia la plataforma se realiza a través de señales digitales.

3.4.2. Entradas al sistema de control.

3.4.2.1. Sensores.

Los sistemas de control (autómatas programables) poseen sensores, que son una parte fundamental del sistema. La función de los sensores es la de transformar una determinada magnitud física de un proceso, (presión, temperatura, posición, inclinación) en otra señal, generalmente eléctrica, la cual será enviada al sistema de control.

Al igual que las señales, los sensores se clasifican en analógicos o discretos en función del tipo de señal con el que operen. Los sensores también son clasificados en función de la magnitud física de medida como pueden ser de nivel de presión, temperatura, proximidad; y en función del principio físico que sustenta su funcionamiento, por ejemplo: sensores ópticos, inductivos, capacitivos, magnéticos…

Figura 33 Acelerómetro triaxial, su funcionamiento se sustenta en variaciones de capacidad de un condensador.

En nuestro caso, en principio no haremos uso de estos puesto que las magnitudes de entrada al sistema de control serán aportadas desde Simulink (a través de un protocolo de comunicación que veremos más adelante), no obstante, sería conveniente su introducción como medida de seguridad ante valores excesivamente elevados que puedan afectar a la integridad de la estructura de la plataforma. Dichos sensores se encargarían de limitar las 3 rotaciones y 3 translaciones que es capaz de realizar la plataforma, de forma que solo se permita un determinado rango de seguridad; así como la velocidad o aceleración de cambio entre una posición angular y otra.

3.4.2.2. Elementos de mando.

Los elementos de mando, junto con los sensores, son otro tipo de dispositivo que es capaz de proveer una señal de entrada al sistema de control. No obstante, la señal que proporcionan estos dispositivos no está determinada por una magnitud física, sino que es el propio usuario el que determina la señal (ordenes). Entre los elementos de mando tenemos dispositivos como pulsadores, interruptores, conmutadores… Estos elementos de mando bien pueden ser dispositivos físicos o bien programados en un sistema grafico tipo SCADA.

SCADA, acrónimo de Supervisory Control and Data Adquisition, es un sistema de monitoreo y control de procesos técnicos mediante ordenador, sirviendo la interfaz de este como elemento de interacción para con el hombre; y que permite controlar y supervisar procesos industriales a distancia en tiempo real.

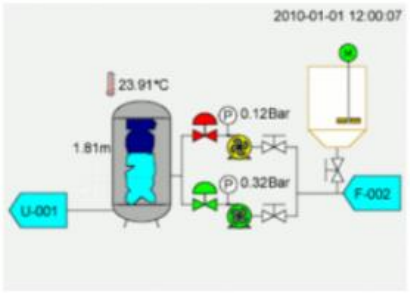


Figura 34 Interfaz de un sistema SCADA.

En nuestro caso concreto, el simulador, el elemento de mando se correspondería con Simulink, pues es desde donde introduciremos la señal que deseamos simular, como hemos visto en el esquema 2.

3.4.3. Salidas del sistema de control. Actuadores.

Los actuadores son dispositivos cuya función es la de suministrar energía a la planta con el objetivo de modificar los valores de las magnitudes que se controlan, a partir de una señal de mando que proviene del sistema del control. En la mayoría de los casos, esto se lleva a cabo mediante una etapa intermedia que se denomina *preaccionador*. Este último elemento se utiliza cuando la salida del sistema de control no tiene suficiente potencia o cuando hay que realizar una conversión de la señal. Este sería el caso de un contactor para controlar la puesta en marcha de un motor o válvula neumática para controlar un cilindro. Atendiendo a la tecnología empleada, los actuadores se clasifican mayormente en: eléctricos, neumáticos o hidráulicos.

Figura 35 Actuador lineal hidráulico.

En nuestro caso, los actuadores son los seis servomotores (Fig.36. abajo) con los cuales controlamos los seis grados de libertad de la plataforma móvil. La unión entre los motores y la plataforma móvil se realiza a través de una configuración biela-manivela (un sistema más simple con el que se conseguirían los mismos resultados seria sustituyendo los motores por actuadores lineales hidráulicos).

Figura 36 Servomotor del simulador.

3.4.4. Implementación de Automatismos Lógicos.

Los automatismos lógicos son sistemas de control en los que las señales con las que se trabaja son de tipo todo-nada. Por tanto, se trata de un sistema que a partir de los valores de entrada binarios debe proporcionar valores a las salidas también binarios. Se diferencian dos tipos de automatismos lógicos:

Combinacionales: los valores de la salida dependen exclusivamente de los valores de las entradas. Por tanto, independientemente de la técnica que se use para la implementación, la lógica del automatismo viene dada por expresiones booleanas de las entradas.

Secuenciales: los valores de salida dependen de las entradas, pero también del estado interno en que se encuentre el sistema. En su implementación se necesitan también elementos de memoria que permitan almacenar el estado en que cada momento se encuentre el automatismo.

En función de la forma de implementar la lógica de un automatismo se distingue:

Lógica cableada: Esta lógica se caracteriza por estar basada en uniones físicas entre los distintos componentes que realizan las operaciones lógicas o de memoria. Dentro de este tipo, lo más habitual es la utilización de contactos y relés con conexiones eléctricas. También hay que señalar que podemos encontrar sistemas de control basados en tecnología neumática, aunque son poco frecuentes.

Lógica programada: El control se realiza mediante la ejecución de un programa en un sistema basado en microprocesador, al que hay que añadir los sistemas de adquisición de datos para conectar con las entradas y salidas de la planta.

Dentro de la lógica programa, los dispositivos más habituales son el PC industrial, los microcontroladores y los autómatas programables. En nuestro caso, emplearemos un autómata programable junto con un PC de uso doméstico desde donde enviaremos los datos.

3.4.5. Definición de autómata programable.

Un autómata programable o PLC(Programmable Logic Controller) es un sistema electrónico programable diseñado para ser utilizado en un entorno industrial, que utiliza una memoria programable para el almacenamiento interno de instrucciones orientadas al usuario, para implantar unas soluciones específicas tales como funciones lógicas, secuencia, temporización, recuento y funciones aritméticas con el fin de controlar mediante entradas y salidas, digitales y analógicas diversos tipos de máquinas o procesos. Esta definición esta dictaminada por la norma IEC-61131, de la cual haremos una breve reseña en el apartado 4.10.

A grandes rasgos podemos decir que un PLC es básicamente un ordenador para una aplicación muy concreta, como es el control de procesos. Esto último determina mayormente el diseño de su hardware y software. Por lo tanto, su diseño debe estar preparado para un ambiente industrial, siendo resistente a ambientes con polvo, vibraciones, golpes… Debe poseer el hardware adecuado para la conexión de señales provenientes de sensores o dispositivos de mando del operario y proporcionar señales a dispositivos de actuación, permitiendo comunicarse con el proceso a controlar y con el usuario.

Una parte fundamental del autómata programable son los sistemas de adquisición de señales y salida. Estos sistemas se encuentran frecuentemente conectados al autoprogramable a través de buses de campo. Estos buses de campo son sistemas independientes del autómata programable cuya función es la transmisión de información, mediante e protocolos de red industriales utilizados para control distribuido en tiempo real. Por lo que los fabricantes proporcionan módulos específicos capaces de gestionar señales provenientes de los distintos buses de campo que se encuentran en el mercado.

En lo referido a la programación de los autómatas programables, estos emplean un lenguaje de programación especifico basado en lógica de contactos, aunque adicionalmente disponen de lenguajes de programación convencionales como puede ser el lenguaje C. La programación de un autómata programable se realiza desde dispositivos externos, ordenadores convencionales, empleando softwares de programación específicos, como es en este caso CoDeSys. La conexión entre el autómata y el PC se realza a través de múltiples sistemas, usualmente USB. Desde el entorno de programación es posible realizar múltiples tareas como son la configuración del autómata, desarrollo de programas usando editores de los distintos lenguajes, transferencia del

programa desde el PC al autómata o en sentido inverso, monitorización del programa y simulación del programa (sin necesidad de transferirlo al autómata programable).

3.4.6. Estructura interna y externa del programable.

3.4.6.1. Estructura externa.

En función del aspecto físico exterior del autómata programable podemos diferenciar entre dos tipos, autómatas programables con estructura compacta y con estructura modular.

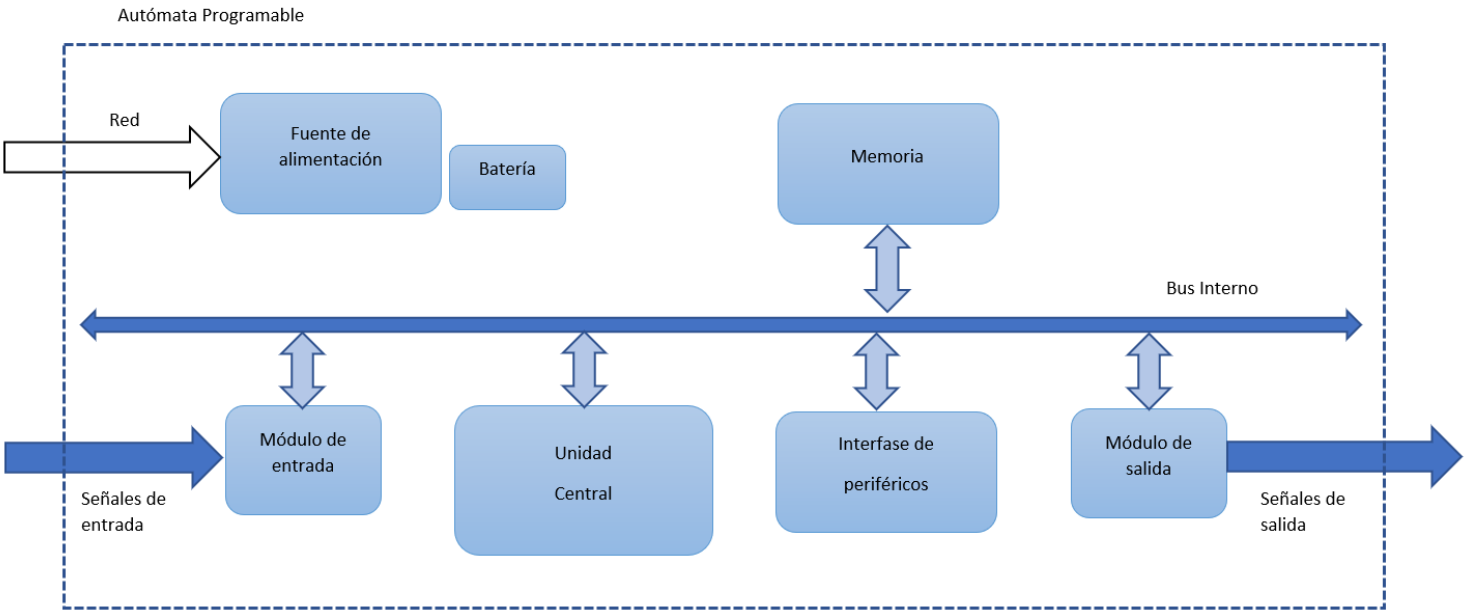
Los autómatas programables con estructura compacta se caracterizan por que todos sus elementos (incluidas conexiones de entrada y de salida) se encuentran en un solo bloque. Son útiles cuando el número de señales cableadas requeridas es pequeño y sin previsiones de ampliación. Estos autómatas programables poseen una carcasa más estanca, y por tanto más adecuada a ambientes industriales, que los autómatas programables modulares.

En los autómatas programables modulares, cada uno de los elementos que componen el autómata es un dispositivo físico distinto, que, en función de las necesidades del sistema de control, se conectan en un sistema de rieles normalizado. Este concepto permite adaptarse de forma fácil a las necesidades de diseño y a posibles ampliaciones. Otra ventaja para tener en cuenta de los autómatas programables modulares es que permiten un funcionamiento parcial ante una avería de un módulo no crítico, así como una rápida reparación mediante una simple sustitución.

Figura 37 Autómata modular de ABB.

3.4.6.2. Estructura interna.

En relación con la estructura interna, un autómata programable no se diferencia en gran medida de cualquier otro ordenador. Compuesto fundamentalmente por una unidad central unida por buses internos a las interfaces de entrada y de salida, periféricos y módulos de memoria.



Esquema 3 Estructura interna Autómata programable.

En la figura superior podemos observar la estructura interna de un autómata programable. La unidad central o CPU es un sistema basado en un microcomputador que se encarga de la ejecución de las instrucciones del programa de usuario, de la gestión de la memoria y de la transferencia de información con los módulos de entrada y salida.

La estructura, utilización y tipo de la memoria dependen del diseño del PLC, aunque de forma general podemos estructurar la memoria del autómata programable en cinco compartimentos.

Sistema operativo o firmware: Es el programa grabado por el fabricante y que realiza el control a bajo nivel de los distintos módulos del autómata programable y de gestionar la ejecución del programa de usuario. Debe estar en una memoria de tipo no volátil, habitualmente de solo lectura tipo ROM.

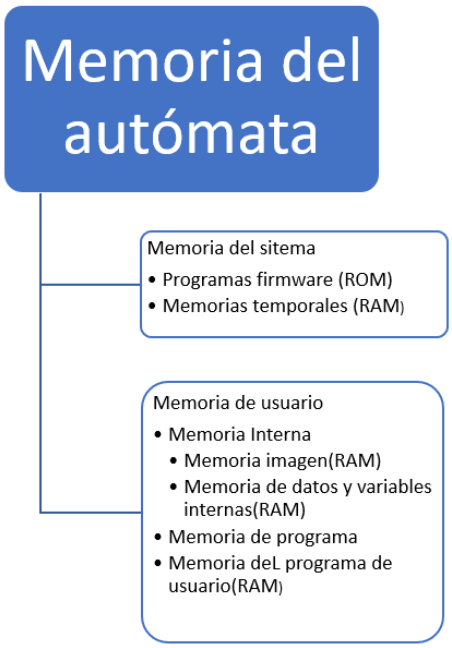
Memoria temporal del sistema: Memoria tipo RAM donde se guardan variables temporales y registros del sistema.

Memoria imagen de Entrada/Salida: Cada una de las señales, tanto de entrada como de salida, conecta físicamente a los módulos de E/S tiene asociada una variable lógica en esta zona de memoria. El programa de usuario accede a esta zona de memoria para leer o escribir las

variables de entrada o salida respectivamente, realizándose el acceso a través de la dirección de memoria de la variable. Es el S.O. el que se encarga de actualizar los valores físicos y lógicos de cada una de las variables de entrada y de salida.

Memoria de datos numéricos y (variables internas): Almacena las variables y datos del programa de usuario. Normalmente de tipo RAM.

Memoria de programa: Es la zona donde se almacena el código escrito por el usuario. Este programa puede estar en memoria RAM, aunque en este caso es imprescindible que tenga un respaldo de batería para evitar la pérdida de información en caso de corte eléctrico. Cada vez es más habitual que el programa de usuario se encuentre en una tarjeta de memoria externa de tipo SD, evitando la perdida de información por corte del suministro eléctrico.



Esquema 4 Estructura de la memoria del Autómata programable.

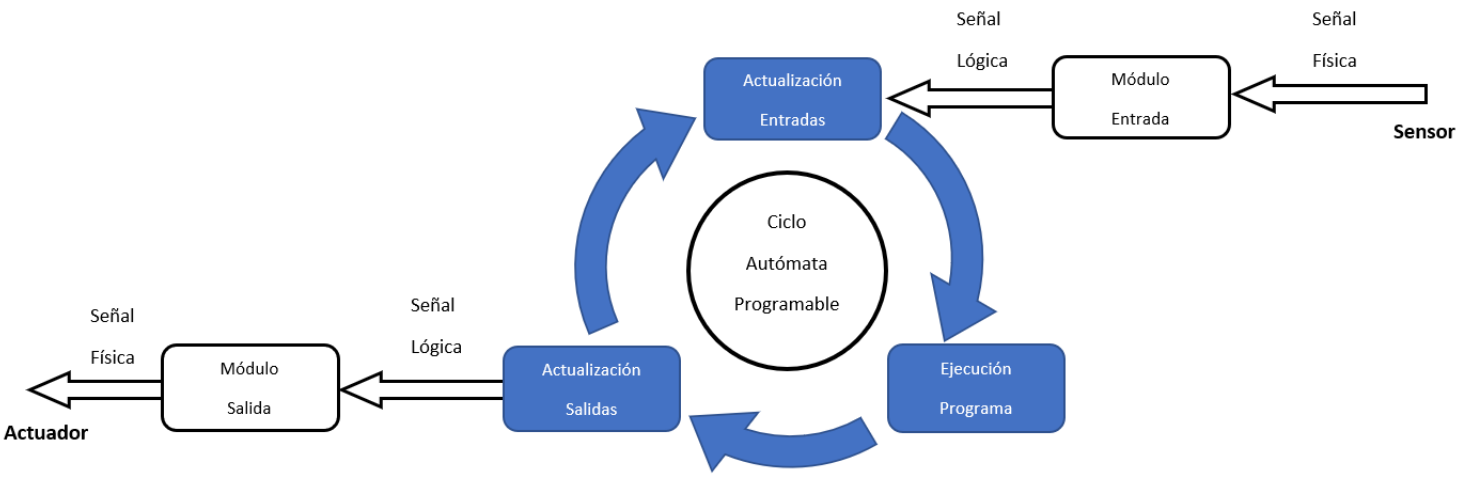
Es impórtate señalar que la cantidad de memoria de la que disponen los autómatas programables suelen rondar 400 kb y 800 kb, siendo muy inferior a un ordenador de uso doméstico.

Los modulos de entrada y de salida son dispositovos que actuan de interfaz entre las señales del proceso, habitualmente en valores normalizados en tension o intensidad, y las variables logicas de la memoria imagen de entrada y salida en el automata programable. Estas variables pueden ser discretas o analogicas.

3.4.7. Funcionamiento de un autómata programable.

El objetivo principal de un autómata programable es controlar un sistema en tiempo real, es decir, a grandes rasgos lo que debe realizar el autómata programable es actualizar continuamente el valor de las entradas que le llegan del proceso y, de acuerdo con el programa de control, actualizar el valor de las salidas de una forma continua. No obstante, la actualización no se realiza de forma continua sino periódica debido a la dinámica de trabajo de un ordenador, pero debe hacerse frecuente como la dinámica del proceso exija.

La ejecución del programa o programas de usuario de un autómata programable es cíclica, por lo que la secuencia de operaciones incluidas en el programa del autómata se va repitiendo continuamente mientras el autómata continúe en funcionamiento. En cada ciclo el PLC realiza las siguientes operaciones, [7] Miguel Ángel Ridao Carlini:



Esquema 5 Funcionamiento Autómata programable.

- Lectura de las señales de entrada: se actualiza el valor el cada una de las variables de la tabla de memoria de entrada con el valor de su correspondiente señal física conectada al módulo de entrada.
- Ejecución del programa de control: se ejecuta el programa de usuario de forma secuencial, es decir, desde la primera a la última línea de código.
- Escritura de las señales de salida: se actualiza el valor de las señales eléctricas del módulo de salida con el valor lógico almacenado en la memoria de salida del PLC.

Teniendo en cuenta el funcionamiento anteriormente explicado, se deben tener en cuenta las siguientes observaciones sobre el modo de funcionamiento de un autómata programable:

La actualización de las salidas y entradas se realiza, como hemos dicho antes, de forma periódica una vez cada ciclo. Por tanto, el tiempo que transcurre entre dos actualizaciones depende directamente de lo que dure el ciclo del autómata programable (este intervalo se denomina Tiempo de Ciclo, Scan Time). Este tiempo de intervalo debe de ser lo suficientemente corto para que se detecten las variaciones que se produzcan en las entradas, esto dependerá por lo tanto de rapidez de variación de la variable que se esté controlando.

De lo explicado anteriormente se deriva que el valor físico de una variable puede cambiar en cualquier momento, pero el valor lógico por el que es representado solo cambia una vez por ciclo. Con lo que en determinados momentos este valor lógico puede estar desactualizado.

Los programas de usuario trabajan exclusivamente con el valor lógico de la memoria de entrada, por tanto, en cada ejecución del programa se trabaja siempre con el mismo valor de entrada, independientemente de que en el transcurso de la ejecución del ciclo haya cambiado el valor físico del sensor.

Al igual que con los valores de entrada, el valor físico de la salida se actualiza solamente al final del ciclo, independientemente de cuantas veces y donde se asigne valor a la variable lógica de salida dentro del programa.

Es muy importante tener presente que el Tiempo de Ciclo dependerá en gran medida del tiempo necesario, por parte del autómata programable, para la ejecución del programa de usuario. Por tanto, hay que tender a programas cortos, lo más secuenciales posible, y evitar dentro de lo posible bucles en los que el programa necesite mucho tiempo de ejecución. Hay que tener presente que el programa debe ejecutarse en un tiempo inferior al Tiempo de Ciclo

máximo que se considere aceptable en cada aplicación. El tiempo de ciclo máximo en nuestro caso, un simulador de ondas sísmicas debe ser igual o inferior al periodo de muestreo con el cual se hallan tomado los datos de la onda sísmica.

Además de las operaciones anteriormente desarrollas, que forman parte del funcionamiento principal, el autómata programable lleva a cabo una serie de operaciones secundarias. En el arranque del autómata se realiza una verificación del hardware y se inicializan las variables del programa. En cada ciclo, el sistema verifica que el programa se está ejecutando en tiempo inferior al establecido como tiempo de ciclo máximo. En caso de que sea superior, el sistema da un aviso o error. Esta tarea la realice un temporizador interno que recibe el nombre de watchdog. También en cada ciclo, el autómata realiza algunas verificaciones relacionadas con las conexiones de entrada, salida y de memoria. Finalmente, en cada ciclo se necesita un tiempo para el intercambio de datos con los periféricos que tenga conectado el autómata.

Teniendo en cuenta todas las operaciones que realiza un autómata programable, el tiempo de ciclo del autómata va a depender principalmente de:

- El código de programa del usuario.
- Número de entradas y salidas.
- Rutinas de chequeo y periféricos.

Modos de funcionamiento del autómata programable.

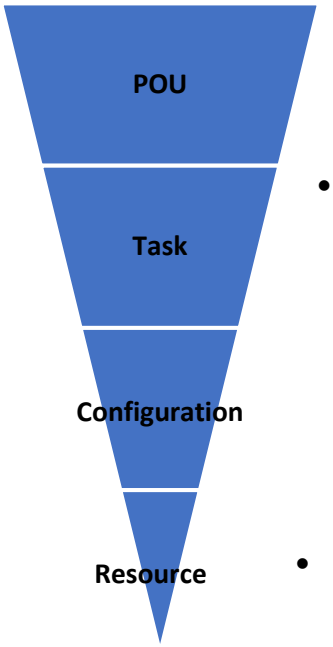
Teniendo en cuenta el funcionamiento general descrito anteriormente un autómata programable puede trabajar en varios modos. Habitualmente el código programa de usuario puede descomponerse en distintos subprogramas y cada uno de ellos se puede ejecutar en un modo distinto. No obstante, los dos modos de funcionamiento más importantes son normal (o cíclico) y periódico.

El modo de funcionamiento normal corresponde con el modo de funcionamiento básico, inmediatamente que se termina un ciclo comienza el siguiente sin solución de continuidad. En CoDeSys este modo de funcionamiento se denomina *Freewheeling*.

En el modo de funcionamiento periódico la ejecución de los ciclos se actualiza de forma periódica como un tiempo fijo entre dos ejecuciones, de modo que dicho tiempo es fijado por el usuario. El tiempo fijado debe ser superior al tiempo necesario por el autómata para ejecutar un ciclo. Este modo de funcionamiento es empleado cuando se controlan variables continuas. De esta manera se fija el tiempo de muestreo del controlador. En CoDeSys este modo de funcionamiento de denomina *cíclico*.

3.4.8. Modelo de software.

En el apartado anterior hemos visto el modo de funcionamiento de la estructura interna del autómata programable y como fluye la información a través de esta. En este apartado nos detendremos en el modelo de software de los autómatas programables, que se encuentra estandarizado según la norma IEC-61131. El modelo de software está basado en una estructura jerárquica, en la que los elementos que la componen desde el nivel más bajo de la jerarquía hasta el más alto son los siguientes.



Esquema 6 Esquema Software.

- POU (Program Organization Unit): Son los elementos básicos de la programación, secciones de código qué pueden estar escritos en cualquiera de los lenguajes que permite la norma.
- Tarea (Task): las POUs estarán asociadas a tareas, que son las que le asignaran las propiedades de ejecución en tiempo real. Por ejemplo, cada tarea especificara si es cíclica, periódica (y el tiempo del periodo) o lanzada por un evento. También se pueden asignar prioridades a las tareas. Todas las POUs que tenga asociada una tarea se ejecutaran con las características de tiempo real especificadas en dicha tarea.
- Recurso (Resource): básicamente consiste en todos los elementos que se definen o ejecutan el único en una única CPU. Incluirá un conjunto de tareas con sus POUs asociadas, variables globales a todas las POUs, etc.

- Configuración (Configuration): Consiste en el conjunto de CPUs que constituyen el sistema de control, que estará formado por un conjunto de recursos. Siendo posible la comunicación entre los distintos recursos dentro de una misma configuración y la definición de variables globales a todos los recursos de la configuración.

Por lo tanto, los bloques básicos de la programación de los autómatas programables lo constituyen las POUs. Una aplicación de usuario, normalmente denominada Proyecto, consta de una o varias POUs. Una aplicación de usuario, normalmente denominada Proyecto consta de una o varias de estas POUs. Las POUs pueden estar escritos en cualquiera de los lenguajes de la norma y pueden llamar a otras POUs con o sin parámetros.

Las POUs pueden ser de tres tipos:

- Funciones (FUN): devuelven el mismo valor si los parámetros de llamada son los mismos.
- Bloques funcionales (FB): mantienen información interna que se utiliza para determinar el valor que devuelven. Como puede ser por ejemplo un contador.
- Programas (PROG): constituyen los programas principales. Son los únicos que tienen acceso a las variables de entrada y de salida.

3.4.9. Lenguajes de programación.

Un lenguaje de programación es un lenguaje formal diseñado para expresar procesos que pueden ser llevados a cabo por maquinas. En el caso de los autómatas programables, los lenguajes de programación aparecieron junto a los mismos, en 1968. Es por esto por lo que para su programación no se emplean lenguajes convencionales como C y Pascal. En su lugar se emplean lenguajes más simples y fáciles de entender. Toda la normativa sobre los lenguajes de programación de los autómatas programables se recoge en la tercera parte de la norma IEC-61131. A continuación, se muestran los lenguajes recogidos en la norma [8].

Lenguaje de contactos (LD), conocido como diagrama de escalera o Ladder es un lenguaje basado en los tradicionales diagramas de relés y contactos, siendo un lenguaje gráfico. Está enfocado a aplicaciones con señales booleanas. Puede soportar la mayoría de los autómatas programables. Dentro de sus características principales se encuentra el uso de barras de alimentación, elementos de enlace y estados; y la posibilidad de utilizar contactos, bobinas y bloques funcionales; así como de evaluar las redes en orden de arriba abajo o de izquierda a derecha.

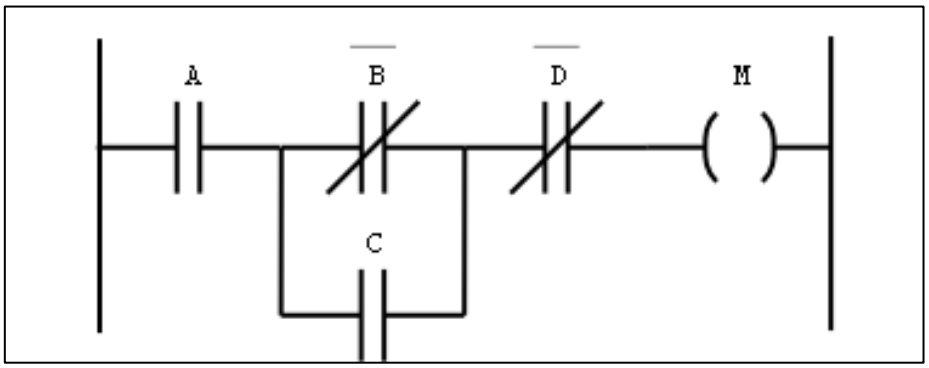


Figura 39 Ejemplo de Lenguaje de contactos.

Lenguaje de Diagramas de Bloques Funcionales (FBD), es también un lenguaje gráfico, basado en la conexión de bloques que representan los distintos operadores o POUs. Estas conexiones representan los parámetros de entrada y salida de los distintos programas o funciones, de forma similar a los diagramas de circuitos electrónicos. Son muy utilizados en la industria de proceso.

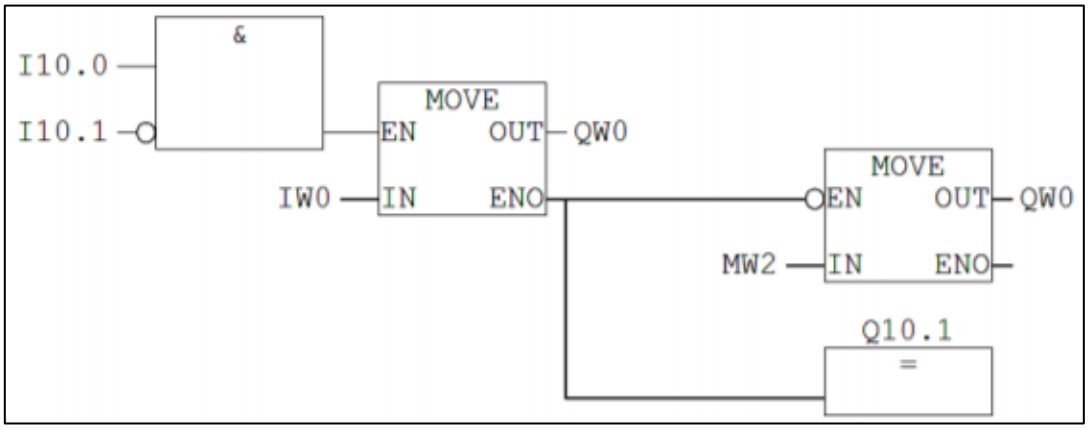


Figura 40 Ejemplo de Diagrama de Bloques Funcionales.

Lenguaje de texto estructurado (ST), es un lenguaje similar a los lenguajes convencionales de programación tipo C. Por lo tanto, es el más indicado para codificar expresiones complejas e instrucciones anidadas. La principal ventaja de este lenguaje es la facilidad para generar bucles mediante comandos (FOR, REPEAT-UNTIL, WHILE DO) y ejecuciones condicionales (IF-THEN-ELSE, CASE).

```
Q 4.0 := I 0.0 AND I 1.1 OR NOT I 0.1
IF Q 4.0 == 1 THEN GOTO M001
ELSE Q 1.0 = NOT Q 4.0;
END_IF;
M001 MW 2= 1+MW 2;
```

Figura 41 Ejemplo de Lenguaje de texto estructurado.

Lenguaje de Lista de Instrucciones (IL), al igual que el anterior se trata de un lenguaje de texto, en concreto es un lenguaje de tipo ensamblador simple. Su uso está muy extendido en Europa y es muy útil a la hora de realizar programas de poca extensión.

```
      A     I     10.0
      AN    I     10.1
      JNB   _001
      L     IW    0
      T     QW    0
      SET
      SAVE
      CLR
_001: A     BR
      =     L     0.0
      AN    L     0.0
      JNB   _002
      L     MW    2
      T     QW    0
_002: NOP   0
      A     L     0.0
      BLD   102
      =     Q     10.1
```

Figura 42 Ejemplo de lista de instrucciones.

Lenguaje de Diagrama de Funciones Secuenciales (SFC), describe gráficamente el comportamiento de un programa de control mediante un grafo de estados y transiciones. Necesita de rutinas escritas en los otros lenguajes para implementar acciones o condiciones lógicas. Se trata de programas que están bien estructurados y cuyos elementos básicos son las etapas, las acciones y las transiciones. De este modo, una secuencia en SFC se compone de una serie de etapas representadas por cajas rectangulares y que se encuentran conectadas entre sí por líneas verticales. Así, cada etapa representa un estado particular del sistema y cada línea vertical a una transición. Estas transiciones están asociadas a una condición "verdadero/falso", dando paso así a la desactivación de la etapa que la precede y activación de la posterior.

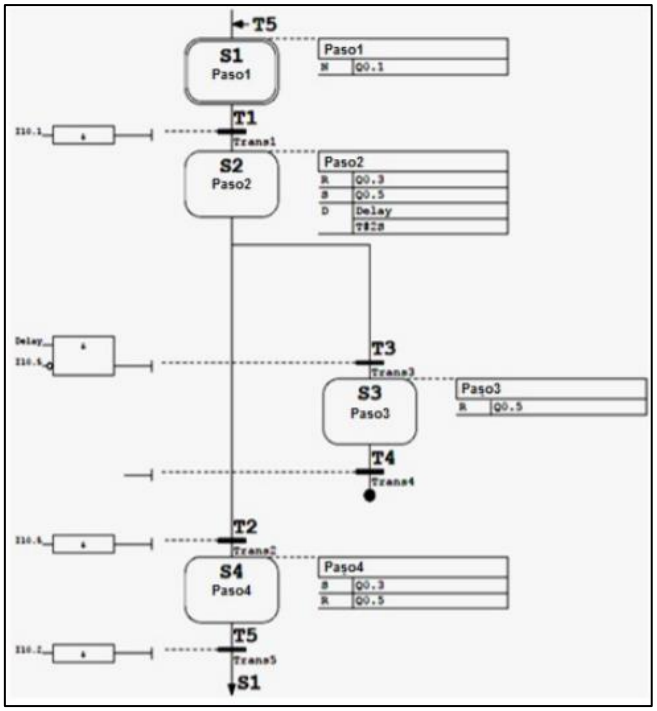


Figura 43 Ejemplo de lenguaje de texto estructurado.

3.4.10. Norma IEC-61131. Estandarización de los autómatas programables.

La norma IEC-61131, anteriormente señalada, es un paso en la estandarización de los autómatas programables y sus periféricos, tanto en la relación al hardware como el software, considerando aspectos en estos dispositivos tales como sus características funcionales, condiciones de servicio, seguridad lenguajes de programación, comunicaciones, etc.

La norma, actualmente, se divide en 8 partes:

1. Información general del Autómata programable.
2. Especificaciones de equipos y ensayos.
3. Lenguaje de programación.
4. Guías para usuarios.
5. Comunicaciones.
6. Seguridad funcional.
7. Programación de control borroso.
8. Guía para la aplicación implementación de lenguajes de programación.

Podemos encontrar más información y futuras modificaciones sobre esta norma en la web de la organización PLCopen: http://www.plcopen.org

3.4.11. CoDeSys.

El autómata programable que emplearemos para el control de la plataforma tendrá como software PLC Designer, el cual está dentro de la norma IEC 61131-3 y su desarrollo está basado en CoDeSys (Control Development System). CoDeSys es una plataforma de software orientado a las tecnologías de automatización industrial. El objetivo principal de CoDeSys es proporcionar a los usuarios un soporte práctico en la implementación de sus tareas de programación de entornos de automatización. Esto se consigue en parte permitiendo a los usuarios emplear el lenguaje de programación que les resulte más cómodo, CoDeSys permite el uso de todos los lenguajes que hemos visto en el apartado 4.9 Lenguajes de programación.

CoDeSys es una plataforma que en los últimos años se ha instaurado como una de las más empleadas por multitud de empresas de automatización: Beckhoff, Festo, EATON, KEBA, IFM, LENZE, Schneider Electric, ABB, ESA. Cuyos softwares de programación están basados en CoDeSys. La plataforma de software presenta múltiples ventajas además de la posibilidad de emplear varios lenguajes de programación.

Otra ventaja es que todo lo necesario a la hora de programar se encuentra en un único interfaz de usuario (como se ve en la Fig.44. abajo): incluyendo extensas funciones para la ingeniería de proyectos, la puesta en servicio de aplicaciones y cambios en la aplicación durante el funcionamiento. A esta interfaz se le pueden añadir módulos adicionales para el desarrollo de determinadas aplicaciones.

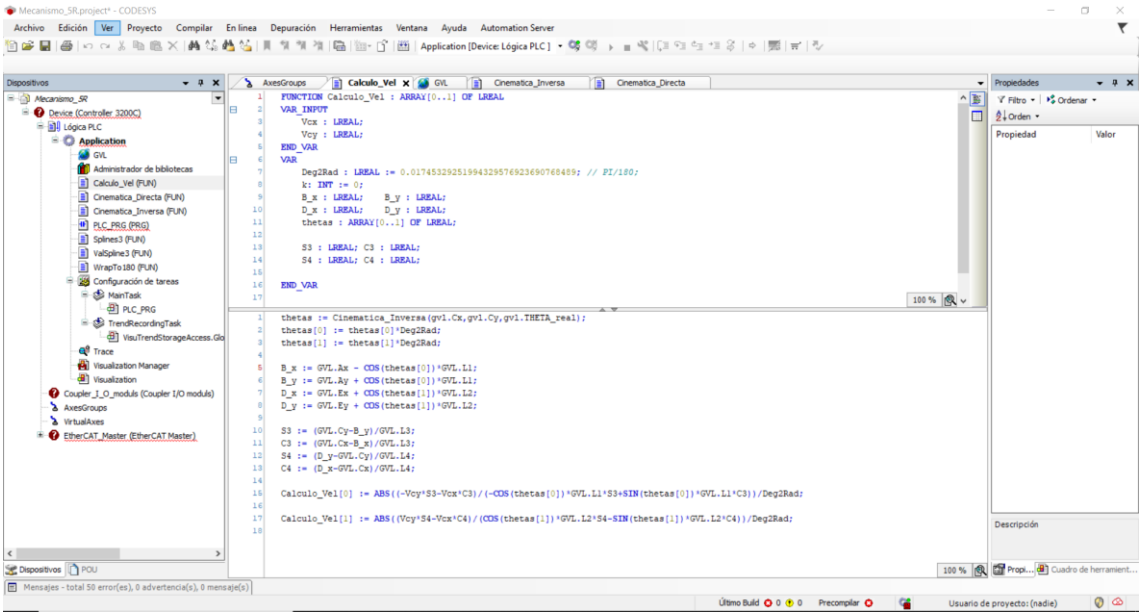


Figura 44 Interfaz de usuario de CoDeSys.

CoDeSys cuenta además con un simulador integrado que permite testear y simular los programas, tanto del autómata programable como de pantallas. Además, mediante CoDeSys Depictor permite realizar simulaciones 3D. También dispone de editor HMI (Human-Machine Interface) para programar interfaces gráficos para pantallas o terminales de operador, permitiendo realizar el programa del autómata y su correspondiente interfaz gráfico en un mismo software.

3.5. Simulink.

En este apartado nos veremos de forma breve las características de Simulink (programa desde el cual enviaremos los datos al PLC).

Simulink es un entorno de programación visual, que funciona sobre el entorno de programación de Matlab. Permite construir y simular modelos de sistemas físicos y sistemas de control mediante diagramas de bloque (como se ve en la imagen inferior). El comportamiento de dichos sistemas se define mediante funciones de transferencia, operaciones matemáticas, elementos de Matlab y señales predefinidas de todo tipo (en nuestro caso una onda sísmica).

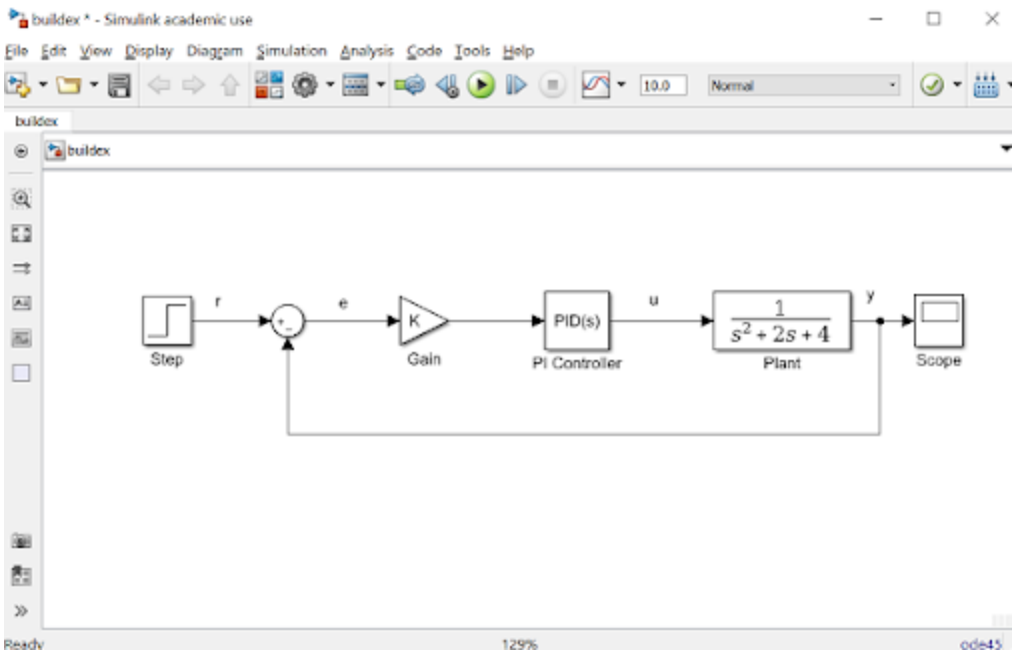


Figura 45 Interfaz de Simulink.

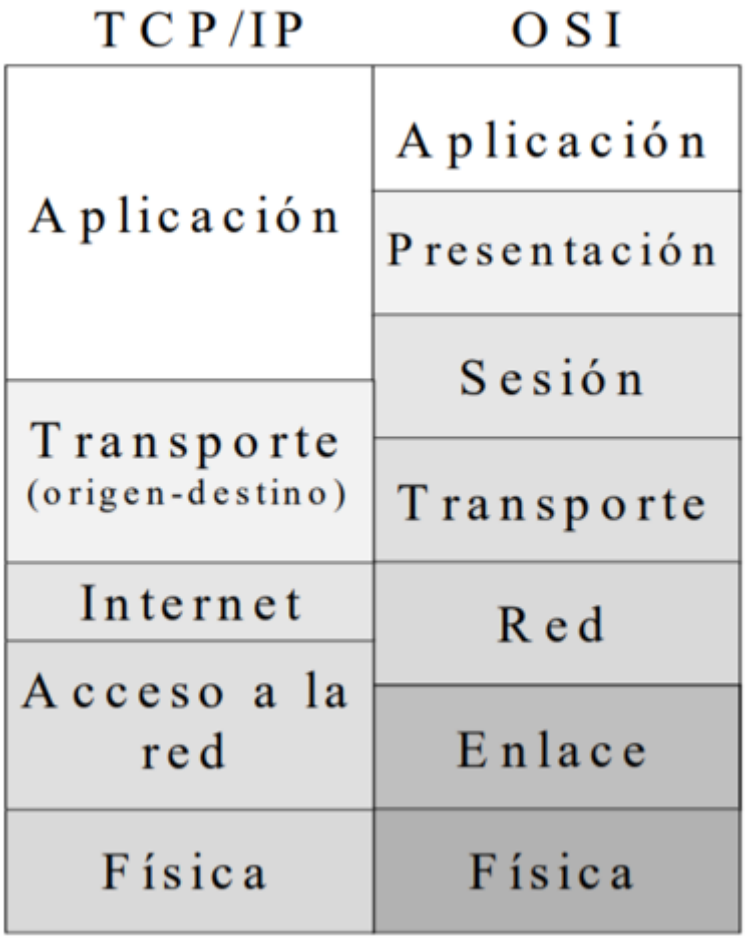
Simulink dispone de una serie de herramientas que facilitan la visualización, análisis y guardado de los resultados de simulación; así como la posibilidad de interactuar con Matlab ya sea para el acceso a datos del workspace de Matlab o para el uso de scripts.

3.6. Protocolo de comunicación. OPC-ua.

Ahora nos centraremos en el protocolo de comunicación que emplearemos para realizar el envío de los datos entre Simulink y CoDeSys.

Un protocolo de comunicación es un sistema de reglas que permite que dos o más entidades de un sistema transmitan bloques de datos (cada uno conocido como Unida de datos de protocolo (PDU)) a través de cualquier tipo de variación de una magnitud física. El protocolo define una serie de reglas, las cuales establecen la sintaxis, la semántica, la sincronización de la comunicación y los posibles métodos de recuperación de errores, propios de cada protocolo.

Para establecer un protocolo de red, los módulos de software del protocolo se interconectan con un modelo implementado en el sistema operativo de la máquina. Dicho modelo establece la funcionalidad de red del sistema operativo. Los modelos más conocidos son el modelo TCP / IP y el modelo OSI (Open System Interconnection), establecen que la comunicación entre ETD (Equipos Terminales de Datos) se divide en niveles o capas de la siguiente forma.



Esquema 7 Capas OSI y TCP/IP.

Para nuestro caso emplearemos el protocolo OPC-ua (Open Platform Communication – unified architecture), que es mantenida por OPC Foundation. Este es un protocolo de comunicación diseñado para comunicar datos de equipos industriales, cuya característica diferenciadora es que puede comunicarse con todas las aplicaciones de una empresa y a través de todas las capas empresariales. Esta versatilidad es debido a que remplaza los protocolos COM y DCOM (específicos de Windows) por otros protocolos abiertos e independientes.

OPC-ua ha sido diseñado para la escalabilidad y es compatible con una amplia gama de dominios de aplicación, que van desde el nivel de campo (por ejemplo, dispositivos para medición o identificación, PLC) hasta el soporte de gestión empresarial. Para lograr estos objetivos de diseño, el estándar OPC-ua proporciona una arquitectura de múltiples capas como se muestra en la siguiente figura:

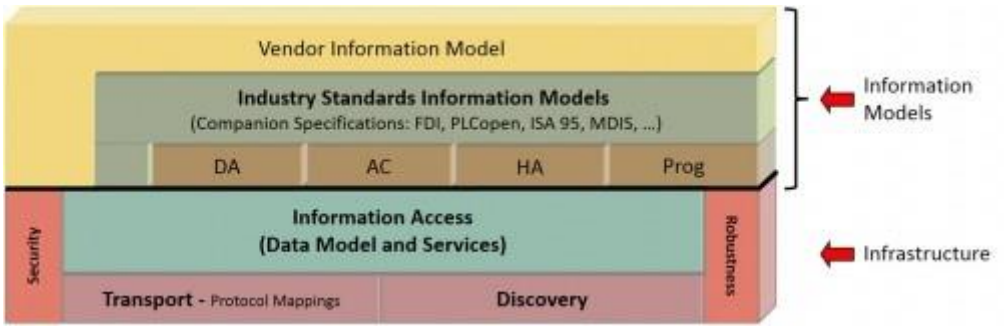


Figura 46 Arquitectura de OPC. (OPC Fundation)

OPC-ua se basa en la siguiente infraestructura:

- Descubrimiento que permite a los clientes encontrar servidores OPC UA, sus protocolos compatibles, políticas de seguridad y otras capacidades.
- Transporte que define mapeos de protocolos que permiten establecer una conexión e intercambiar mensajes bien formados entre aplicaciones OPC UA.
- Acceso a la información, que comprende los medios para exponer modelos de información basados en objetos en un espacio de direcciones y los servicios para acceder a esta información.
- Seguridad y Robustez, que se integran en Transporte y Acceso a la Información.

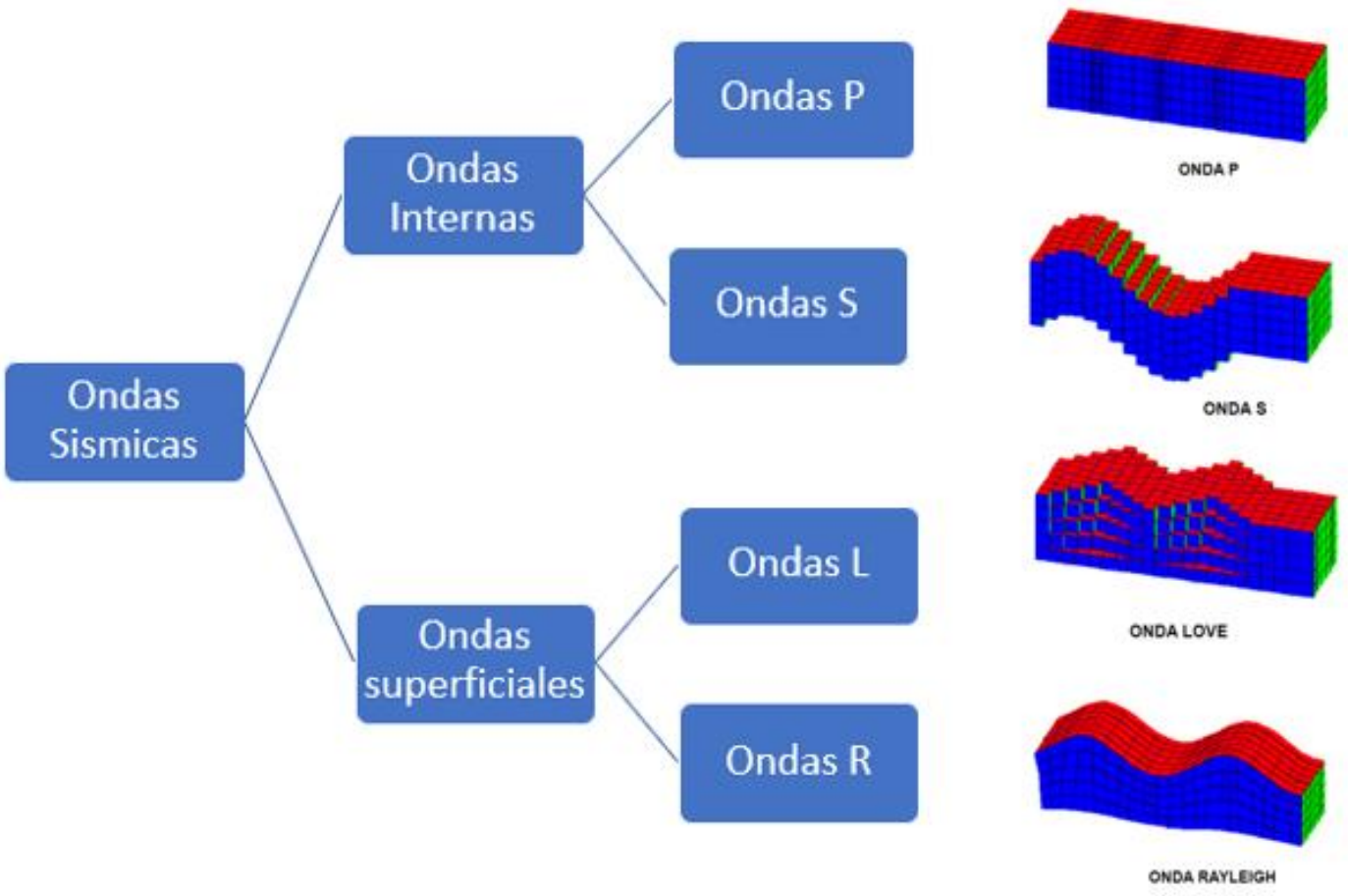
Los modelos de información se superponen en capas sobre esta infraestructura. OPC UA especifica una serie de modelos de información base (DataAccess - DA, Alarms & Condition - AC, y más) que definen objetos de uso común que incluyen alarmas y variables de datos históricos y en tiempo real.

La comunicación entre los elementos en una red OPC-ua se realiza estableciendo roles de maestro/esclavo o cliente/servidor. El maestro tiene la capacidad tanto de leer como de escribir en el esclavo, en las direcciones de memoria compartida.

3.7. Ondas sísmicas.

En este apartado nos detendremos a estudiar el fenómeno que trataremos de replicar mediante el simulador, una onda sísmica (o terremoto).

Una onda sísmica es un tipo de onda elástica, que se propaga desde una fuente (foco o hipocentro) a través de un medio material elástico transportando energía mecánica, esto sucede por la propagación de perturbaciones temporales del campo de tensiones que generan pequeños movimientos en las placas tectónicas. Hay dos tipos de ondas sísmicas: las ondas internas y las superficiales. Las ondas internas viajan a través del interior de la tierra y se subdividen en ondas Primarias(P) y Secundarias(S), la diferencia entre estas dos radica en su forma de propagarse. Las ondas P se propagan de forma longitudinal, de manera que el suelo es comprimido y dilatado alternativamente en la dirección de propagación. Las ondas S se desplazan de forma transversal a la dirección de propagación, estas ondas son las que generan las oscilaciones durante el movimiento sísmico y la que producen la mayor parte de los daños. Las ondas superficiales también se subdividen en dos tipos de ondas: Ondas Love y Rayleigh. Las ondas Love generan un movimiento horizontal de corte en superficie, mientras que las ondas Rayleigh generan un movimiento elíptico retrogrado en el suelo. Además de ondas Love y Rayleigh en sismos de gran intensidad podemos encontrar otro tipo de ondas superficiales denominadas oscilaciones libres, siendo estas vibraciones de la tierra en su totalidad.



Esquema 8 Ondas Sismicas.

La velocidad de propagación de los distintos tipos de ondas es diferente. Siendo sus velocidades aproximadamente de:

- 7 Km/s ondas P
- 4-6 Km/s ondas S
- 2-3 Km/s ondas L
- 90% de la velocidad de las ondas S para las ondas R

Esta diferencia de velocidades provoca que las ondas sean registradas unas detrás de otras, siendo fácil diferenciarlas como vemos en la imagen inferior.

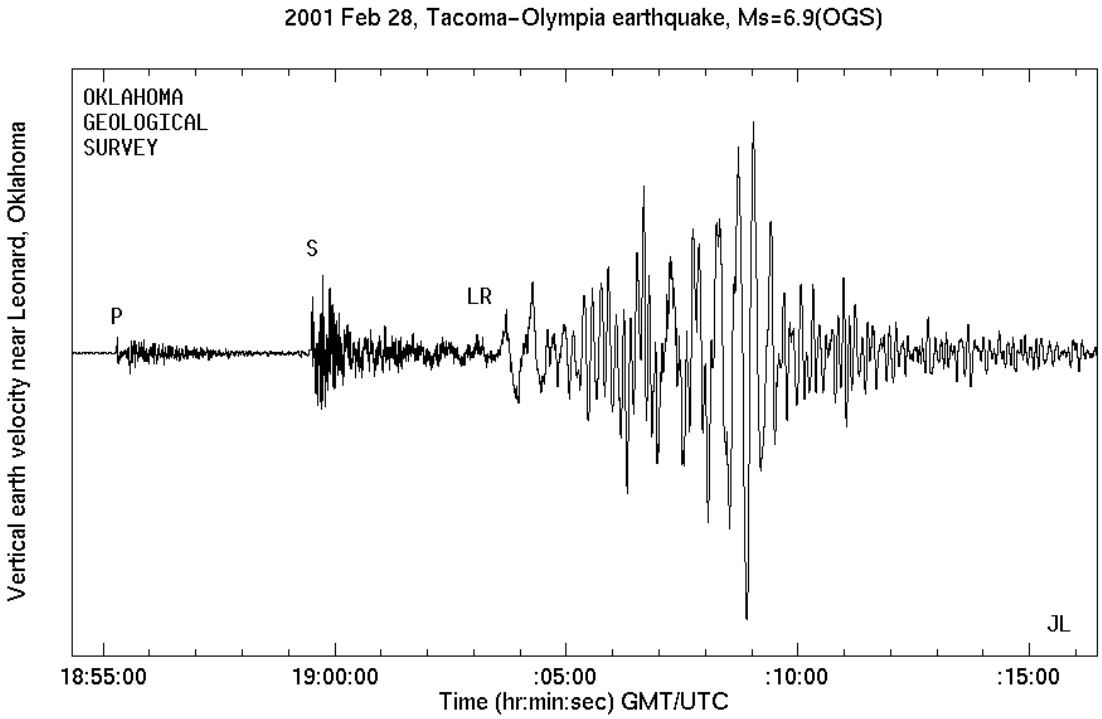


Figura 47 Sismográfica de una Onda sísmica.

En la imagen superior podemos ver el registro que ha realizado un sismógrafo de una onda sísmica (datos similares serán los que usaremos para realizar la simulación). No obstante, arriba hemos visto solamente una de las componentes (la vertical) que tiene un terremoto, es decir, una onda sísmica genera movimiento en las tres direcciones del espacio por lo que podemos dividirla en 3 direcciones espaciales. Los sismómetros realizan mediciones simultaneas en 3 direcciones: arriba-abajo, norte-sur y este-oeste.

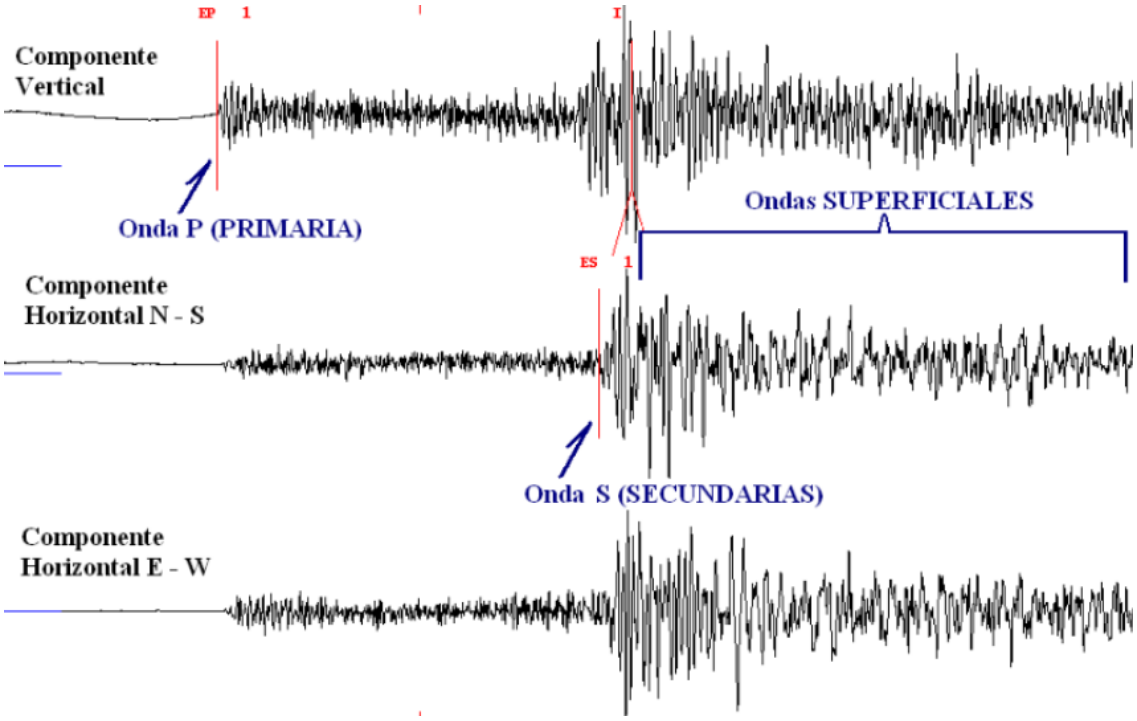


Figura 48 Sismográfica en los 3 ejes espaciales.

3.7.1. Cuantificación de las ondas sísmicas. Escalas.

Para la cuantificación de las ondas sísmicas existen diferentes escalas de magnitud e intensidad. Las escalas de magnitud se basan en la energía que liberan las ondas sísmicas; mientras que las escalas de intensidad se basan en los daños producidos. Por lo tanto, las escalas de magnitud están relacionadas directamente con las ondas sísmicas y las escalas de intensidad con el impacto que provocan en la población, construcciones y naturaleza.

La primera vez que se trataron de catalogar las ondas sísmicas se hizo a través de los efectos observables que estas provocaban. Esta escala de intensidad fue propuesta por Mercalli en 1902 y posteriormente modificada en 1931, pasándose a llamar escala Modificada de Mercalli (MM). No es la única escala de intensidad existente, pero si la más usada. Consta de 12 grados de intensidad, con las características correspondientes a cada grado. La escala MM es una escala subjetiva y no permite la comparación entre diferentes sismos (el daño causado puede depender de otros factores además del propio sismo), además no proporciona información sobre la energía u otra variable del temblor.

Para poder comparar diferentes sismos entre si es necesario catalogarlos a través del registro grafico o numérico que tenemos de ellos, es decir los sismogramas. La escala de magnitud más famosa que se fundamenta en el registro de los sismogramas es la escala de Richter (o escala de magnitud local, M_L). Para esto se toman los valores de la máxima amplitud de la onda sísmica sobre el sismograma (A [mm]) y la diferencia de tiempo entre las ondas P y S (Δt_{S-P} [s]). A través de la siguiente expresión se obtiene la magnitud:

$$M_L = \log_{10} A + 3\log_{10}(8\Delta t_{S-P}) - 2.92$$

Este cálculo para la magnitud de las ondas sísmicas es aplicable a sismos superficiales con una distancia al epicentro menor a 2000 Km. Para sismos de otras características existen otras escalas como la m_b y M_s.

3.7.2. Datos para la simulación.

Los datos necesarios para llevar a cabo la simulación de una onda sísmica los tomaremos de IRIS (Incorporated Research Institutions for Seismology). Es un consorcio de investigación universitario dedicado a explorar el interior de la Tierra a través de la recopilación de datos sismográficos. Entre sus programas se encuentran la mitigación de terremotos y la verificación de un Tratado de Prohibición Completa de Pruebas Nucleares.

Los datos de registro de las ondas sísmicas los podremos encontrar en la siguiente página: http://ds.iris.edu/wilber3/find_event. En esta página tendremos acceso no solo a datos sísmicos de origen natural (o fuente pasiva), también podremos descargar datos de ondas sísmicas debidas a explosivos, experimentos… (fuente activa).

4. Implementación.

En este apartado trataremos como se comunican los diferentes componentes del simulador de ondas sísmicas entre sí, como se transfiere la información entre los diferentes componentes.

A modo de resumen, el simulador de ondas sísmicas está compuesto por una plataforma Stewart, cuyo posicionamiento depende de un autómata programable, la información que usa el autómata para posicionar la plataforma proviene de Simulink. El flujo de la información se realiza en tiempo real mediante el protocolo OPC-ua y de esta forma la plataforma se posiciona de manera que replica la onda sísmica.

4.1. Simulink - Autómata programable.

Para realizar el envío de datos desde Simulink primero deberemos cargar los datos en el workspace de Matlab.

Figura 49 Workspace de Matlab.

Después deberemos crear un vector con el tiempo durante el cual transcuren las ondas. En este caso cada una de las señales posee 12800 datos de muestreo, tomados con un periodo de 0.005. Por lo tanto eltiempo total es:

$$t = nºdatos * periodo = 12800 * 0.005 = 64\ s$$

Asi pués generemos un vector de tiempo desde 0 hasta 64 segundos, con 12800 puntos. Este vector de tiempo lo usaremos para crear un objeto del tipo timeseries con cada una de las 3 señales.

```
Command Window
>> t=linspace(0,64,12800);
>> NS = timeseries(NS_Component',t');
>> EW = timeseries(EW_Component',t');
>> V = timeseries(V_Component',t');
fx >>
```

Figura 50 Generación de Timeseries.

Con esto obtendremos las 3 señales junto con el tiempo:

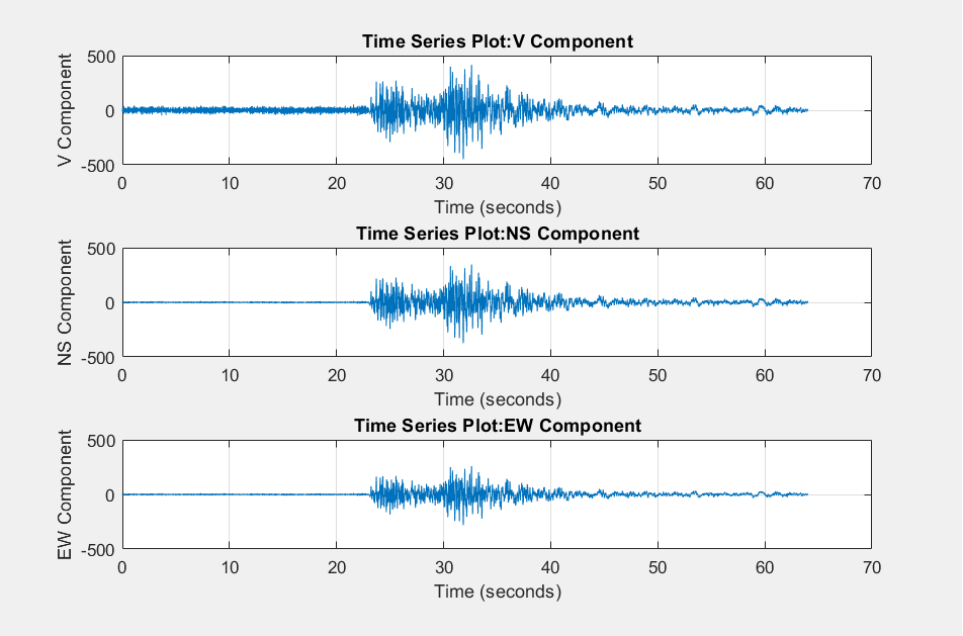


Figura 51 Representación de los datos.

Ahora pasaremos a crear el diagrama de bloques en Simulink, para enviar los datos. Pero primero, debemos establecer estas variables en el programa de control del automata programable. Para ello, en el explorador de proyectos de CoDeSys crearemos un objeto del tipo configuracion de simbolos (o symbol configurtion) y habilitaremos la opcion de compativilidad con características OPC-ua.

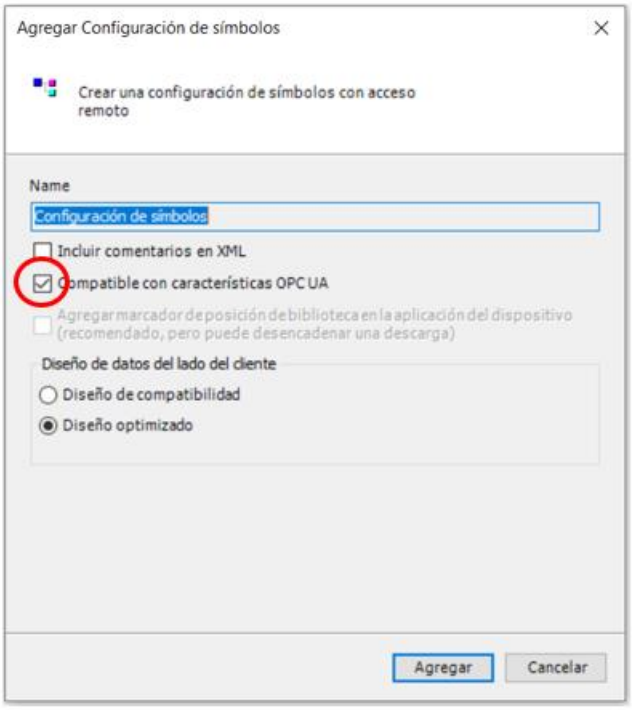


Figura 52 Creación del objeto: Symbol Configuration.

Una vez echo esto deberemos entrar en la opcion de configuracion dentro del objeto y activar el acceso E/S directo, como observamos abajo.

Figura 53 Selección de opción acceso E/S directo.

Echo esto pasaremos a compilar el programa, una vez echo podremos seleccionar de la lista de variables globales (GVL) las que participaran en la conexión OPC-ua. Inicilmente a modo de ensayo conectaremos solamente la componente vertical.

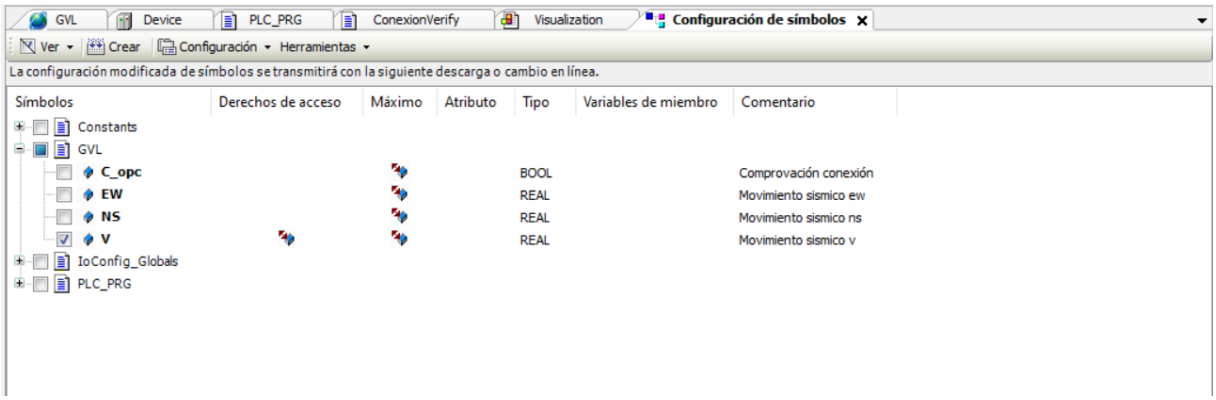


Figura 54 Selección de las variables.

Para poder realizar la conexión mediante OPC-ua el programa debe estar en funcionamiento, por lo que simularemos su funcionamiento con CoDeSys Control Win SysTray (que hara las veces de PLC virtual). Para conectar el programa al PLC virtual, iremos al organizador de proyectos y aremos click izquierdo en Device, en las opciones que nos aparecen selecionamos Actualizar Dispositivo. Se nos desplegara la siguiente ventana y en ella selcionaremos CODESYS Control Win V3 x64.

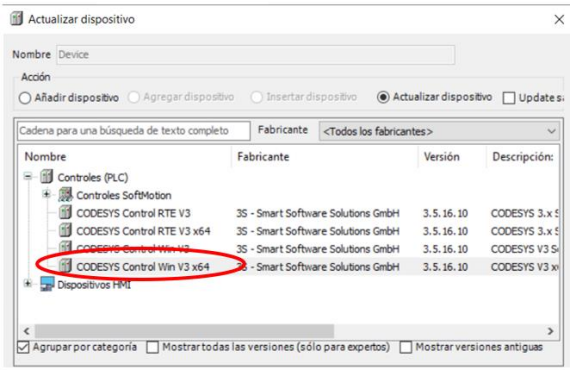


Figura 55 Selección de autómata programable.

Ahora desplegaremos la pestaña Device y en ella examinaremos la red. En ella selecionamos el gateway 1 y nos aparecera el PLC virtual (el cual seleccionaremos).

Figura 56 Selección del simulador de Autómata Programable.

Con todo esto podremos podremos iniciar nuestro programa como si se ejecutase en un PLC real. Al iniciar el programa, nos aparecera en la barra de informacion infeior que el programa se encuentra en ejecuacion. Tambien se abrira una ventana con el HMI que hayamos diseñado para el programa.

Ahora en Simulink realizaremos el diagrama de bloques necesario. Para que Simulink realice la simulacion en tiempo real deberemos emplear el bloque Real-Time Sync, pero ademas sera necesario que instalemos “Real-Time Kernel” en Matlab. Para ecceder a los datos del workspace de Matlab emplearemos el bloque From Workspace, para la configuración del bloque en Data escribimos el nombre de la variable que deseemos leer del workspace, y en SampleTime

introducimos el periodo de muestro (0.005 en este caso). Tambien añadiremos un Scope para poder ver los datos en tiempo real durante la ejecucion.

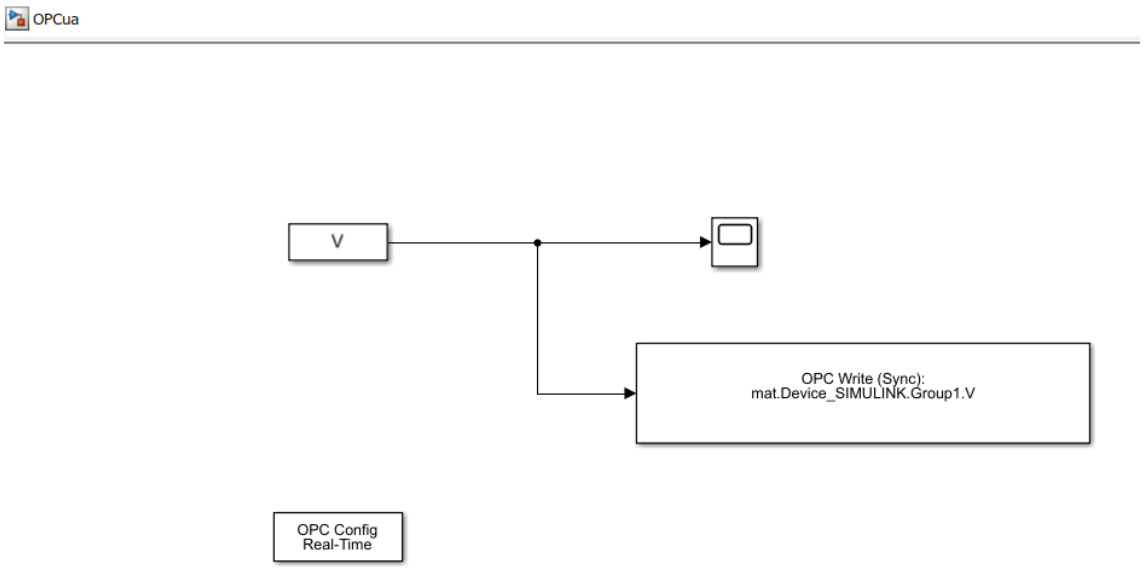


Figura 57 Diagrama de bloques de Simulink.

Tambien haremos uso de los bloques que nos permitiran establecer la conexión OPCua para la cual Simulink tiene una Toolbox dedicada, de esta seleccionaremos los siguientes bloques (que se aprecian en la Fig.75.):

- OPC Configuration Real-Time: nos permitira configurar y seleccionar el servidor OPC al cual nos conectarmos.
- OPC Write(Sync): nos permitira seleccionar las variables del servidor sobre las que deseamos escribir.

Abriremos las opciones del bloque OPC Configuration Real-Time y seleccionaremos Configure OPC Clients, en la pesta que se nos despliega seleccionaremos Add y se nos mostraran los servidores OPC disponibles, seleccionarmeos: Kepware.KEPServerEX.V6. Igualmente abriremos las opciones del bloque OPC Write(Sync) y selaccionaremos Add Items, aquí seleccionaremos la variable V (que previamente deveremos haber configurado en el servidor de KEPServerEX como veremos a continuación).

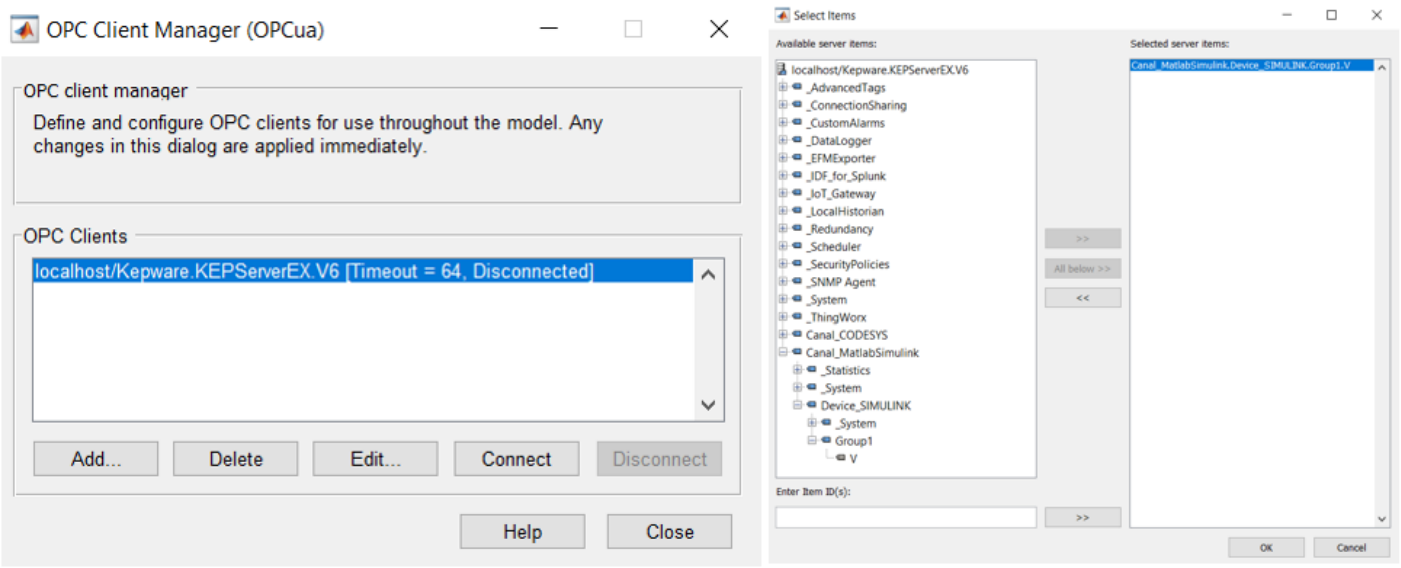


Figura 58 Configuración de los bloques OPC Configuration Real-Time y OPC Write(Sync).

Ahora ya estamos en disposición de conectar Simulink (el maestro) con Codesys (el esclavo), para ello haremos uso del servidor OPC KEPServerEX (desarrollado por Kepware). Este utiliza una estructura cliente-servidor, donde el servidor nos ofrece distintas direcciones de datos a los que el cliente es capaz de acceder para leerlos o escribir nuevos datos.

Figura 59 Logo de KEPServerEX

Creamos dos canales (uno para Codesys y otro para Simulink) dentro de un proyecto de KEPserverEX, dentro de cada canal debemos configurar cada uno de los dispoositivos y seleccionamos las variables que deseamos (en este caso V).

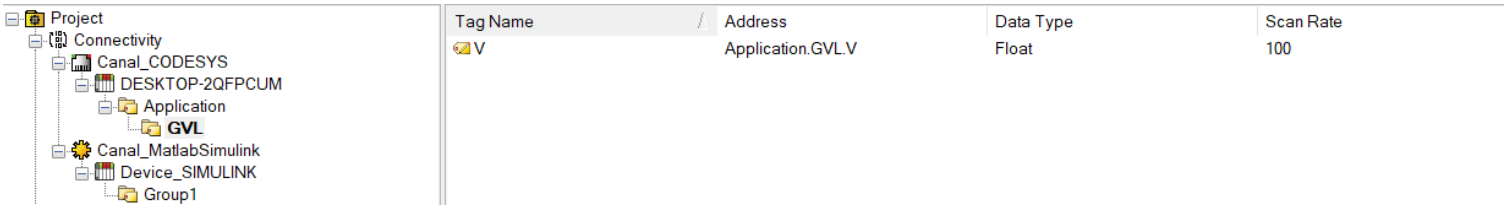


Figura 60 Creación de los canales y dispositivos en KEPServerEX.

El siquiente paso es unir ambas variables, para ello en el arbol de proyecto selecionaremos Advance Tags y elegiremos New Link Tags. De esta forma crearemos un Link Tag entre ambas variables, siendo la variabale de entrada (Input) el valor de simulink y la variable de salida la que se direge a CoDeSys (Output).

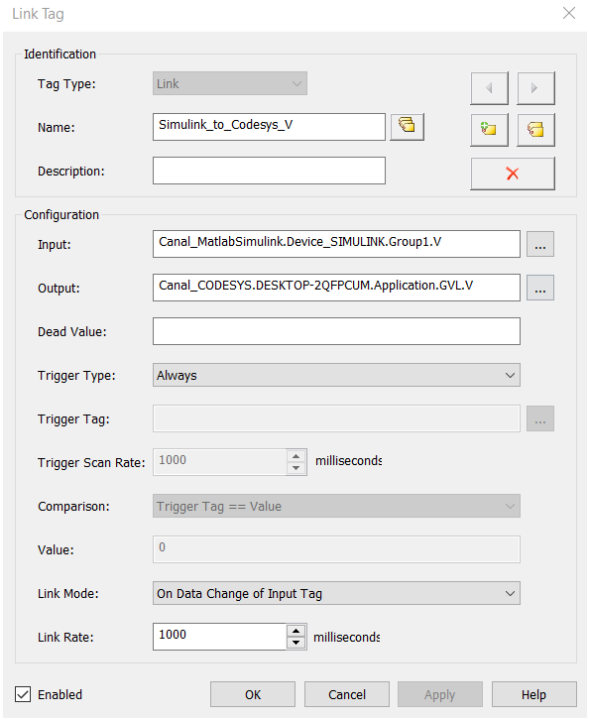


Figura 61 Creación del Link Tag.

Por ultimo, para realizar el envio de datos, debemos de ir a KEPServerEX Settings, pestaña Runtime Process y seleccionamos Interactive como modo de procesamiento. Echo esto estaremos en disposición de realizar la simulacion enviando los datos de una onda sismica desde Simulink a Codesys (o PLCDesigner).

4.2. Autómata programable-Plataforma Stewart.

Una vez la información se encuentre en el autómata programable este se encargará de posicionar la plataforma de acuerdo con la información recibida. Este posicionamiento se realiza por el programa de control que se encuentra en el autómata programable, desarrollado en PLCDesigner. La información que se le proporciona al autómata son las posiciones del actuador final, la plataforma móvil, que es el componente de la plataforma que simula el comportamiento de la onda sísmica. Por lo tanto, se solucionará la cinemática inversa de la plataforma. De esta forma el autómata calculara el valor que debe proporcionar a cada uno de los seis servomotores en cada momento para situar la plataforma de acuerdo con la información recibida a de Simulink.

En nuestro caso el modelo de autómata que emplearemos será el Lenze 3200C, Fig.63.

Figura 62 Autómata programable de Lenze, modelo 3200 C

Este es un autómata de estructura modular que se monta sobre un carril DIN. Posee un procesador Intel Atom. Su sistema operativo es PLCDesigner, el cual está basado en CodeSys. Como se ha mencionado en el 4.10, cumple con la norma IEC 61131-3.

La salida proporcionada por el autómata es una señal digital que contiene la información de cómo debe posicionarse cada motor para dotar a la plataforma del estado cinemático que replique la onda sísmica, pero esta señal en ningún momento hace girar directamente a los servomotores, que como veremos más adelante necesitan una tensión más elevada. Para enviar la señal eléctrica final que llegará a los servomotores se emplea un inversor, cuya función en convertir la señal que le llega del autómata programable a una señal eléctrica de alta tensión que hace girar los servomotores.

Para esta función haremos usos del inversor de Lenze, Servo-Inverter i700, Fig.64.

Datos técnicos

Corriente nominal							
	I [A]	2,5	5	10	16	24	32
Corriente máx. de salida 3 sec							
	I [A]	5	10	20	32	48	64
Tensión de red alimentadores							
	U [V]	3 x 230 ... 480					
Potencia nominal							
	P [kW]	0,75	1,5	4	7,5	11	15
Dimensiones							
Eje individual	Al x An x Pr [mm]	350 x 50 x 260			350 x 100 x 260		
Eje doble	Al x An x Pr [mm]	350 x 50 x 260		350 x 100 x 260			

Fig.63. Servo-Inverter i700

Como vemos en la figura superior el Servo-Inverter i 700 está formado por cuatro módulos. Uno de ellos se encarga del control del resto, mientras que los otros 3 son los encargados de controlar la señal eléctrica de dos servomotores cada uno. Para alimentar a los servomotores se encuentra alimentado por una corriente alterna trifásica a 400/480V, esta será la corriente que regulará Servo-Inverter i700, para controlar los servomotores.

Los servomotores y grupo reductor son los siguientes:

Figura 64 Servomotor y grupo reductor de la plataforma.

Los servomotores son motores eléctricos síncronos, en este tipo de motores la frecuencia de la corriente alterna esta sincronizada con la rotación, y es de esta forma como se lleva a cabo su control por parte del inversor. El modelo del servomotor es el siguiente, Lenze MCS 09D41L. Este servomotor posee una potencia de 1Kw, un momento de fuerza de 2,3 Nm. El grupo reductor como su nombre indica tiene como misión reducir las revoluciones a la salida del eje del servomotor, además desvía 90º el eje de rotación. En este caso es el modelo G50BB, de engranajes cónicos. Finalmente, el eje de salida de estos es donde se acopla la manivela, la cual

en función de su giro entorno a este eje controla el movimiento de la plataforma móvil como hemos visto en la resolución de la cinemática inversa.

5. Resultados.

En este apartado realizaremos la simulación de la conexión OPC-ua entre el PLC virtual y Simulink. A modo de prueba realizaremos primeramente la simulación en uno de los ejes espaciales. Al iniciar la simulacion en Simulink y observar los datos tanto en el Scope de Simulink como en un HMI del programa de CoDeSys, en el vemos una señal lluminica que indica que la conexión se ha establecido de forma exitosa y un instrumento de medida.

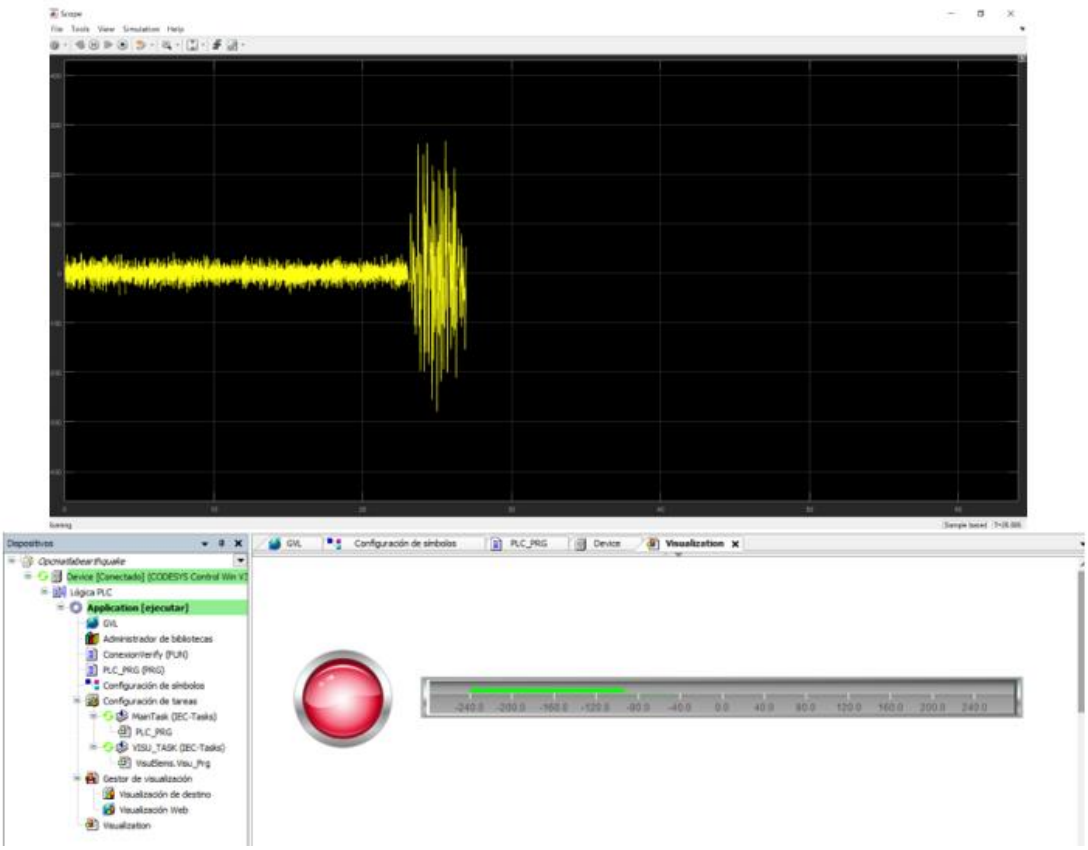


Figura 65 Verificación de la conexión OPCua Simulik-CoDeSys.

Para realizar la simulación de forma fehaciente de la onda sísmica debemos replicarla en las tres direcciones del espacio, por lo tanto, ahora pasaremos a realizar la conexión anteriormente realizada, pero con las tres componentes en este caso. Por lo tanto, rediseñaremos el diagrama de bloques de Simulink de la siguiente manera:

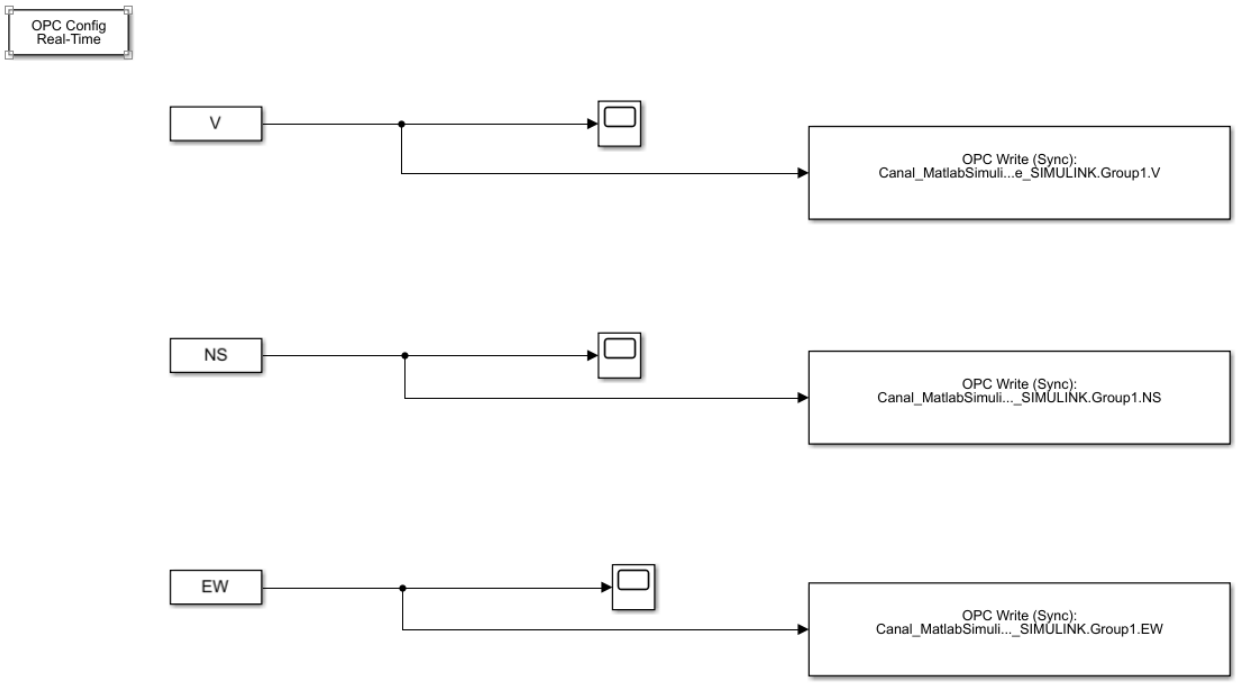


Figura 66 Esquema Simulink 3 componentes.

Para poder seleccionar las variables de KEPServerEX las hemos configurado previamente, tanto para la conexión con Simulink como con CoDeSys. En el programa de CoDeSys introduciremos las dos nuevas variables, de forma que podamos observar la conexión de cada una por separado mediante una señal lumínica.

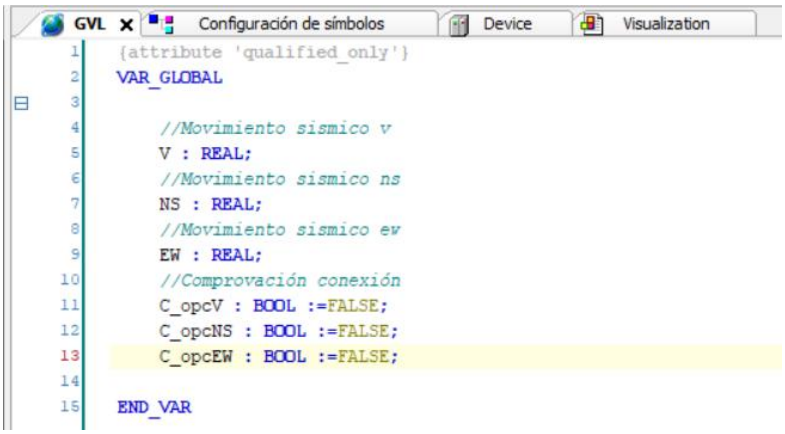


Figura 67 Nombramos las nuevas variables en CoDeSys.

Además de las 3 variables que se correspondes con las componentes de la onda sísmica, declaramos otras 3, para comprobar la conexión con una función de la siguiente forma:

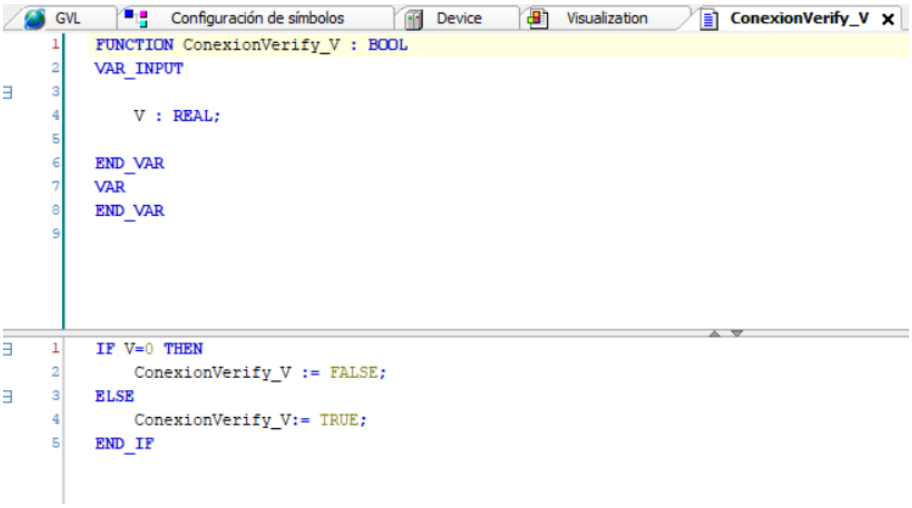


Figura 68 Función de verificación de la conexión.

Generamos una función para comprobar cada componente por separado. Hecho esto volvemos a unir las variables de Simulink con CoDeSys en KEPServerEX, siendo los inputs las variables de Simulink y outputs las variables de CoDeSys. Echo esto podemos pasar a realizar la simulación de la onda sísmica en los 3 ejes.

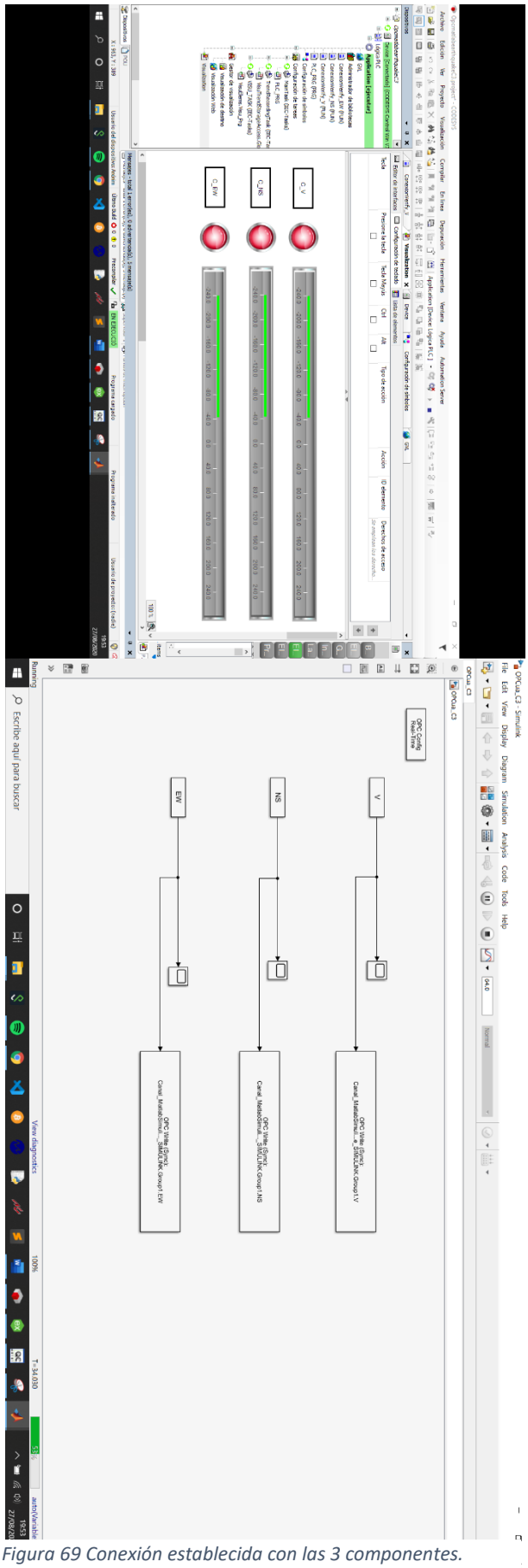


Figura 69 Conexión establecida con las 3 componentes.

En la figura de la página anterior podemos observar como la conexión entre Simulink y CoDeSys es de buena calidad. No obstante, se puede apreciar que el tiempo de ejecución se ve dilatado, respecto a la simulación con una sola variable, debido a la carga de cómputo de las dos variables adicionales para el PC.

6. Conclusiones y líneas futuras.

Hemos visto como se efectúa el envío de datos de una onda sísmica a un PLC virtual mediante una conexión OPC-ua desde Simulink, para simular dicha onda en una plataforma Stewart. Para realizar esta misma conexión sobre un PLC físico bastaría con cargar el programa en dicho PLC e indicar en el servidor de KEPServerEX el IP-address y puerto del autómata. Para la simulación hemos empleado movimiento en los tres ejes del espacio, no obstante, la plataforma Stewart permite giros en los tres ejes del espacio que en esta ocasión no hemos empleado.

Ahí que resaltar la gran versatilidad, pues permite la conexión con cualquier servidor OPC-ua, por ejemplo, Excel permite almacenar datos a través de una conexión OPC, de manera que podríamos tener registrados y almacenados datos de interés del simulador. Así mismo en esta ocasión hemos empleado Simulink para almacenar y enviar los datos, pero es una herramienta mucho mas versátil, con la incluso puede realizarse el control.

En esta ocasión se ha realizado un simulador de ondas sísmicas, no obstante, la plataforma Stewart permite simular todo tipo de fenómenos de forma fehaciente, gracias a su precisión como hemos visto. La primera idea de este trabajo estuvo enfocada en un simulador de conducción, mediante el simulador AssettoCorsa. En este caso como servidor OPC-ua se habría usado el módulo de Python Freeopcua, que junto con la posibilidad que tiene AssettoCorsa de acceder a la memoria compartida desde el propio Python enviaría los datos, al igual que en esta ocasión, al PLC.

7. Bibliografía.

[1] J.-P. MERLET: Parallel robots. Series Editor: G.M.L. GLADWELL, Departamet of Civil Engineering, University of Waterloo, Waterloo, Ontario, Canada N2L 3GI

[2] P. Diermaier: The stewart-gough platform of general geometry can have 40 real postures. En Advances in Robot Kinematics: Analysis and Control, págs. 716, Strobl, Austria, 1998.

[3] N. Rojas, J. Borràs y F. Thomas: A distance-based formulation of the octahedral manipulator kinematics. IFToMM Symposium on Mechanism Design for Robotics, págs. 16, Septiembre 2010.

[4] Kyowa Electronic Instruments CO., LTD: What's a strain gage? introduction to strain gages.

[5] K. Liu, J. M. Fitzgerald y F. L. Lewis: Kinematic analysis of a stewart platform manipulator. IEEE Transactions on Industrial Electronics, 40(2):282284, abril 1993.

[6] pi-usa: https://www.pi-usa.us/en/tech-blog/what-is-the-difference-between-parallel-positioners-and-stacked-serial-kinematics/

[7] Miguel Ángel Ridao Carlini: Introduccion a la programación de autómatas programables usando CoDeSys.

[8] Controladores Industriales inteligentes, Departamento de ingeniería eléctrica, electrónica y de control, UNED

[9] https://opcfoundation.org/about/what-is-opc/

[10] https://es.mathworks.com/discovery/opc-ua.html?s_tid=srchtitle

[11] https://help.codesys.com/api-content/2/codesys/3.5.14.0/en/_cds_runtime_opc_ua_server/

[12] https://www.udc.es/dep/dtcon/estructuras/ETSAC/Investigacion/Terremotos/ondas_s%EDsmicas.htm

[13] https://www.ign.es/web/sis-teoria-general

[14] http://ds.iris.edu/ds/nodes/dmc/data/types/waveform-data/